A Memetic Algorithm for the Task Allocation Problem on Multi-robot Multi-point Dynamic Aggregation Missions

Guanqiang Gao[1], Yi Mei[2], Bin Xin[1], Ya-Hui Jia[2] and Will Browne[2]
1. School of Automation
Beijing Institute of Technology
Beijing, China
{3120170426, brucebin}@bit.edu.cn
2. School of Engineering and Computer Science
Victoria University of Wellington
Wellington, New Zealand
{yi.mei, jiayahui, will.browne}@ecs.vuw.ac.nz

***Abstract*—Multi-Point Dynamic Aggregation (MPDA) is a novel task model to determine task allocation for a multi-robot system. In an MPDA scenario, several robots with different abilities aim to complete a set of tasks cooperatively. The demand of each task is time varying. It increases over time at a certain rate (e.g. the bush fire in Australia). When a robot executes a task, the demand of the task decreases at another certain rate, depending on the robot's ability. In this paper, the objective is to design a task plan for minimising the maximal completed time of all tasks. But coupling cooperative and time-varying characteristics of MPDA brings great challenges to modelling, decoding, and optimisation. In this paper, a multi-permutation encoding is used to represent every robot's visiting sequence of tasks, and an implicit decoding strategy with heuristic rules is designed to simplify the problem from a hybrid variable optimisation to a multi-permutation optimisation. Memetic algorithms for the task allocation of MPDA with two local search methods are designed: equality one-step local search with a better exploration ability and elite multi-step local search with a better exploitation ability. Computational experiments show that the proposed decoding method leads to a better performance given the same computational time budget. Experimental results also show that the proposed memetic algorithms outperform the state-of-the-art method in solving the task planning problems of MPDA.**

***Index Terms*—Multi-robot system, task allocation, memetic algorithm, multi-point dynamic aggregation mission**

I. Introduction

The task allocation problem is a key cooperation issue for multi-robot systems, which determines how to assign limited resources to achieve a mission most efficiently [1], [2]. Although research on the multi-robot task allocation (MRTA) problem has achieved promising results, research considering the time-variance and complex dependence among robots is insufficient. These two characteristics widely exist in real applications of multi-robot systems, but they also bring many difficulties for modelling and planning. A novel problem, named Multi-point Dynamic Aggregation (MPDA), as a kind of MRTA was introduced to involve these two characteristics by Xin et al [3]. In an MPDA scenario, a number of task points are located in different places and their demands change over time. Multiple robots are aggregated to these tasks and execute the tasks cooperatively to satisfy the demands of all the task points. By planning the robots' routes and scheduling their process time, the goal of MPDA is to allocate tasks to robots most efficiently. MPDA can be used to describe a wide range of time-varying coordination tasks, such as human rescue and disaster mitigation [4], [5]. For instance, when MPDA is used to model a fire-fighting mission, the growth of forest fires over time can be described as the tendency of tasks' demands.

An MPDA problem resembles traditional optimisation problems such as the vehicle routing problems (VRP) and resource scheduling problem [6], [7]. The task allocation in MPDA needs to determine paths of robots visiting all tasks, which is similar to the route planning in VRP. From the perspective of VRP, the initial positions of robots in MPDA can be regarded as deposits' positions in the multi-deposit VRP, tasks can be regarded as customers. The task allocation in MPDA also needs to determine when robots arrive at and depart from tasks, which is similar to the planning of arrival time and release time in the scheduling problems. From the perspective of the scheduling problems, robots can be regarded as machines in the parallel machine scheduling, tasks can be regarded as jobs. Thus, the task allocation problem in MPDA is a NP-hard problem, which can not be solved by deterministic algorithms for large-scale instances.

Although MPDA has some similarities with VRPs and scheduling problems, it still has some essential differences from these traditional optimisation problems. Firstly, VRP

This work was supported in part by the National Outstanding Youth Talents Support Program 61822304, the National Natural Science Foundation of China under Grant 61673058, the Beijing Advanced Innovation Center for Intelligent Robots and Systems, the Peng Cheng Laboratory, the NSFC-Zhejiang Joint Fund for the Integration of Industrialization and Informatization under Grant U1609214, the National Key R&D Program of China (2018YFB1308000). Corresponding author: Bin Xin (brucebin@bit.edu.cn)

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

does not allow multiple vehicles to serve a customer at the same time and the scheduling problem does not allow multiple machines to process a job at the same time [8]. The MPDA problem allows robots to perform one task cooperatively and visit the same task several times. The characteristic of time-variance in VRP is embodied by the shifting route time. Usually, the serving time and demand of customers are fixed. In MPDA, the serving time of a task is variable, and it is affected by the abilities of robots aggregating at it. Thus, previous algorithms proposed for solving the VRP and scheduling problems can not be applied to MPDA directly due to its cooperative and time-varying characteristics.

New encoding/decoding and optimisation methods have been proposed for solving MPDA [9]–[11]. Xin proposed an Estimation of Distribution Algorithm (EDA) using K-means clustering and multi-modal Gaussian distribution to solve the task planning problem [10]. A multi-permutation representation and a corresponding decoding method by recording arrival events are also proposed in [10]. The decoding method designs a heuristic rule which requires that a completed task can not be visited by a robot. Apparently, according to the triangle inequality, the task performance of a robot arriving at a completed task must be worse than the task performance of a robot that goes to the next task directly without staying at the completed task for a while. Although this rule can improve the allocation results slightly, it will make the recorded event time unordered during the decoding process. When the unordered situation has happened, the decoding process [10] needs to restart to keep events in order. This restart strategy leads to a great increase of the time complexity of the decoding process. Thus, a more efficient decoding method needs to be designed. Furthermore, evolutionary Gaussian models used in [10] can not describe the cooperation and dependences among robots explicitly.

Memetic algorithms (MAs) with individual learning strategies, which maintain balance between exploration and exploitation, have been successfully employed to solve VRP and scheduling problems [12], [13]. This is because parts of the solution space can be clustered together which need to be exploited and links between clusters/new clusters need to be explored. Considering the difficulties caused by coordination and time-varying characteristics, defects of the algorithm proposed by Xin [3] and merits of MAs, MAs for solving the task allocation in MPDA with two local search methods are designed in this paper. Specifically:

- The multi-permutation decoding process that distinguishes arrival and departure events is proposed in this paper. Compared with Xin's decoding method [10], the proposed decoding method has a lower computation cost.
- Two local search strategies with a classical swap operator on permutations have been designed in this paper. The first one is called equality one-step local search (OLS) which has an improved exploration ability. Every individual has a certain probability to participate in OLS. The other one is called elite multi-step local search (MLS) which has an improved exploitation ability. Only the elite individual will participate in the local search process.

The rest of this paper is organised as follows. Section II introduces MPDA and important related work. Section III describes the representation scheme and decoding method in detail. Afterwards, two MAs with different local search strategies are proposed in Section IV. Section V presents the computational experiments, which include empirical comparison with other algorithms and the demonstration of the efficacy of operators. Finally, conclusions are drawn in Section VI.

II. Background

In this section, the task allocation problem of MPDA is formulated. Then, a brief review of the most relevant approaches that have been used to tackle MPDA is introduced.

A. Problem description

A number of tasks and robots are distributed in a planar space and assumed as points in the configuration space with no obstacles, which can be defined by a graph $G(\mathscr{V}, \mathscr{E})$. $\mathscr{V}$ is the set of vertices and $\mathscr{E}$ is the set of edges. The vertex set $\mathscr{V} = \{v_1, v_2, \ldots, v_m, v_{m+1}, \ldots, v_{m+n}\}$ contains m tasks and n robots, the edge set $\mathscr{E}$ describes distances among robots and tasks. The travel time matrix $\mathcal{C} = (c(x, y))$ of rob_i is related to the edge length and the robot's speed. Speeds of all robots are assumed as 1 to simplify the description in this paper. Thus, the positive travel time $c(x, y)$ is equal to e_{xy} in $\mathscr{E}$ for all robots.

Each task in the MPDA has a demand characteristic. In this paper, the demand of each task is assumed to follow an exponential law, which is shown as (1) and (2).

$$s_j(t + \Delta t) = \begin{cases} 0, & if\ s_j(t) \leq \sigma, \\ s_j(t)e^{\lambda_j(t)\Delta t}, & otherwise \end{cases} \tag{1}$$

where σ represents the demand threshold of a task, which is set to 0.1 in this paper. The task is denoted by $task_j$. When $s_j \leq \sigma$, $task_j$ is considered as a completed task and Tc_j is the completed time when s_j is first equal to σ.

$$\lambda_j(t) = \alpha_j - \sum\nolimits_{i=1}^{n} v(i, j, t)\beta_i \tag{2}$$

where $v(i, j, t)$ represents a binary value. $v(i, j, t)$ is equal to 1 when rob_i is performing $task_j$ at time t. Otherwise, $v(i, j, t)$ is equal to 0. The current change rate λ_j of $task_j$ influences the state of $task_j$ directly. λ_j is determined by the initial increment rate of $task_j$ and abilities of robots aggregating at $task_j$.

An optimisation problem that minimises the maximal completed time of MPDA can be established as follows.

$$\mathcal{SP}^* = \arg\min \max_{j \in \{1,\ldots,m\}} Tc_j(\mathbf{SP}) \tag{3}$$

where $\max_{j\in\{1,\ldots,m\}} Tc_j$ represents the makespan of configuration space $\{\mathcal{R}, \mathcal{T}\}$ in schedule plan $\mathbf{SP}$.

B. Related Works

MPDA has been discussed in the literature and approaches to MPDA have been developed. A receding horizon path planning method with a distributed decision mechanism was proposed to solve the motion planning problem in MPDA [3]. MPDA is used to formulate the multi-robot motion planning problem in a cooperative multi-area coverage mission in [14]. These distributed algorithms can obtain a schedule plan in a short time, but the schedule plan is not of high quality. A hybrid algorithm that combines the Differential Evolution and EDA approaches is proposed in [11]. Lu [9] proposed a multi-model EDA employing node histogram models and edge histogram models in probability modelling to be used to solve the routing problem of MPDA. An EDA using K-means clustering and multi-modal Gaussian distribution is used to solve the task planning problem [10]. The dependence and cooperation relationships are so complex that evolutionary models in these algorithms can not match schedule plans accurately. Thus, the evolutionary process expressed in this paper will pay more attention to the relationships of permutations. In addition, the aforementioned EDA methods did not introduce individual learning strategies, so the exploitation ability of their algorithm is not good. To gain more exploitation ability, this paper designs two MAs considering exploration and exploitation to solve the MPDA problem.

III. Representation Scheme and Decoding Method

A. Representation scheme

The representation of solutions in evolutionary algorithms has a great influence on the design of evolutionary operators and algorithmic efficiency. In order to describe the whole action plan, which is denoted by $\mathbf{SP}$, an event tuple structure is introduced, which is shown as follows.

$$event_k = [R_k, M_k, T_k, C_k] \tag{4}$$

while the first position of $event_k$ records robot information, the second position records the task information, the third position T_k represents the time when $event_k$ begins to occur, and the boolean value C_k in the tuple represents the event type which is either arrival or departure events. $C_k = 1$ means that a robot arrives at a task. Otherwise, $C_k = 0$ denotes that a robot departs from a task. Thus, $\mathbf{SP}$ can be represented by a list of event tuples in chronological order.

$\mathbf{SP}$ contains continuous and discrete variables and its length is not fixed. Thus, an explicit encoding method of $\mathbf{SP}$ is very difficult to design. In addition, as MPDA is a constraint optimisation problem, there are so many infeasible solutions in the solution space leading to a discrete landscape. An implicit representation scheme was introduced by previous research [9], [11]. Similar to the permutation scheme representation in VRPs, a fixed-length solution without any continuous values is expressed as n lists with m integral elements as shown in (5).

$$X = \begin{bmatrix} \pi_{1,[1]} & \pi_{1,[2]} & \cdots & \pi_{1,[m]} \\ \pi_{2,[1]} & \pi_{2,[2]} & \cdots & \pi_{2,[m]} \\ \vdots & \vdots & \vdots & \vdots \\ \pi_{n,[1]} & \pi_{n,[2]} & \cdots & \pi_{n,[m]} \end{bmatrix} \tag{5}$$

where each row in X represents the task-performing sequences of rob_i and it is a permutation of $\mathcal{T}$. For example, if ith row is $[1, 3, 2]$, rob_i will intend to visit $task_1$ $task_3$ and $task_2$ in sequence.

B. Decoding method

The proposed decoding algorithm is shown as Algorithm 1. As the formulation in (4), $event$ contains information about arrival or departure time of a robot. In the proposed decoding strategy, two types of robots' status are distinguished as $OnRoad$ and $OnTask$ corresponding to two different values of C. If a robot's status is $OnRoad$, it becomes ready to arrive at a task to execute. Otherwise, a robot will depart from a task. The decoding process adopts the event trigger mechanism. Heuristic rules, which decrease the problem's dimension and increase the performance of a solution, added into the decoding process are shown as follows.

- **Rule 1**: When a robot reaches a task to execute, the robot will stay at the task until it has been completed.
- **Rule 2**: When a robot completes a task, it will travel to the next uncompleted task without any waiting time.
- **Rule 3**: A robot can only visit the same task at most once.
- **Rule 4**: $\mathbf{SP}$ is generated by adding $event$ iteratively during the decoding process. The function shown in Algorithm 2 is used to determine $event$.
- **Rule 5**: A robot is allowed to visit a completed task.

According to **Rules 1-3**, the arrival and departure time of a robot can be determined without giving a certain time for it to occur. The problem is simplified from a discrete and continuous mixed optimisation problem to a multi-permutation optimisation problem. As we know that the demands of tasks change over time, it is very important to ensure the time sequence of the events in the decoding process. **Rule 4** guarantees $event$ is added into $\mathbf{SP}$ in chronological order. Algorithm 2 compares the time of the events, which are about to happen, to find the minimal T. The $event$ corresponding to the minimal T will be added into $\mathbf{SP}$. If a robot's status is $onRoad$, its arrival time will be compared. Otherwise, the departure time will be compared.

The Function *FINDEVENT* always compares predicted events. Thus, there will be a situation a robot arrives at a completed task. The previous decoding method [10] optimised this situation during the decoding process, which makes the robot go to the next uncompleted task of the robot's sequence directly as shown in Fig. 1. This strategy can enhance the performance of a schedule plan, but it will mix-up $\mathbf{SP}$. For example, a solution is represented as (6), where the predicted events of visiting $task_1$ by rob_1 and rob_2 will be compared

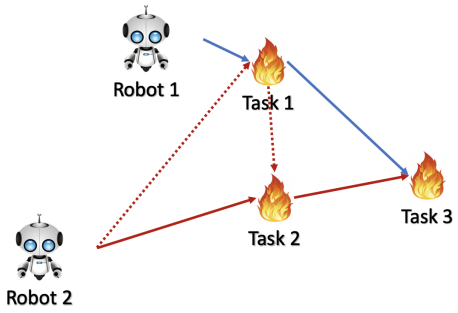

Fig. 1. An example to describe the triangle optimisation strategy. The red dots line represents the origin routing of rob_2 and the red solid line represents the optimized routing of rob_2.

firstly. Assume that the event time when rob_1 arrives at and departs from $task_1$ is 0.2, 0.4, the event time of rob_2 arriving at $task_1$ is 0.5.

$$X = \begin{bmatrix} 1 & 3 & 2 \\ 1 & 2 & 3 \end{bmatrix} \tag{6}$$

In the previous decoding method [10], rob_2 will abandon $task_1$ instead of going to $task_2$ directly. Assume the predicted time of rob_2 arriving at $task_2$ is 0.1. Thus, there will be a situation where the time of rob_2 arriving at $task_2$ is before the time of rob_1 arriving at $task_1$. In order to guarantee the decoding process is in chronological order, the decoding process needs to be restarted. Although this strategy optimises the performance of a schedule plan, it brings a large computational cost. Especially for large-scale instances, where the out-of-order situations generally exist, the decoding process will be restarted many times. In the heuristic decoding rule proposed by this paper, rob_2 will still go to the completed $task_1$ to ensure the decoding process is in order.

The decoding algorithm contains two processes: initialisation and loop. In the initialisation process, **SP** is set as an empty set. Every robot's status will be set as $onRoad$ and its arrival time is calculated by the travel time matrix $\mathcal{C}$. For the loop process, it will be terminated when all the tasks have been completed. During each iteration of loop process, the function *FINDEVENT* finds $event$, which is added to the whole schedule plan **SP**. When an event added to **SP** has been determined, the status of several robots needs to be updated. When $event$ is an arrival event, the current demand and rate of the related task are calculated and the corresponding robot's status is set as $onTask$. If the current rate $\lambda(t)$ of a visiting task is less than 0, which can be completed in a finite time, the completed time of visiting task and departure time of all coordinated robots that are executing the task are updated. If the current rate $\lambda(t)$ is greater or equal to 0, the departure time of all coordinated robots is equal to ∞. When $event$ is a departure event, the related task has been completed and the corresponding robot's status is set as $onRoad$. The decoding algorithm finds the next uncompleted task, which is denoted by $task_x$ in the corresponding robot's visiting task sequence and predicts the arrival time by $\mathcal{C}$. It is worth mentioning that this part is different from the **Rule 5** as it will not mix up **SP**. Thus, the triangle inequality can be used to enhance a representation scheme's performance. At the end of the decoding process, the maximal completed time and **SP** can be obtained.

Algorithm 1 Decoding method

Require: $\mathcal{R}$, $\mathcal{T}$ and X
Ensure: **SP**
Initialisation Process :
SP $= []$
for all rob_i **do**
  $arrT_i$ = $\mathcal{C}(\pi_{i,[0]}, i+m)$
  The status of rob_i is $onRoad$
end for
Initialise states of m tasks.
Loop Process:
while all tasks have not been completed **do**
  $event = [R, M, T, C]$ = *FINDEVENT*
  Add $[R, M, T, C]$ to **SP**
  if $C == aEvent$ **then**
    rob_R's status $= OnTask$
    if $task_M$ has been completed **then**
      $depT_R = T$
    else
      Calculate the current demand and rate of $task_M$.
      if $task_M$ is able to be completed **then**
        Predict cT_M and $depT_R$
        Update coordination robots' status.
      else
        $depT_R = \infty$
      end if
    end if
  end if
  if $C == dEvent$ **then**
    rob_R's status $= OnRoad$, $cT_M = T$
    Find the next uncompleted task $task_x$ in π_i.
    $arrT_i = T + \mathcal{C}(M, task_x)$
  end if
end while
return **SP**

In terms of worst-case time complexity, the previous decoding method is $O(m^2 \times n^2(n+m))$, while the proposed decoding method is $O(m \times n(n+m))$, which is much smaller than that of the previous decoding method.

IV. Memetic algorithms for the task allocation problem in MPDA

Two MAs with different individual learning strategies are proposed to solve the task allocation problem in MPDA. The algorithm begins with N_{pop} individuals to evolve. Then, the crossover operator is used to produce 2 offspring for $opsize$ rounds. An intermediate population pop_t is constructed by the current population and offspring generated by the crossover operator. Individuals in pop_t will participate in the local search

Algorithm 2 Function of finding $event$

$T = \infty$
for i **to** n **do**
  if rob_i is $onRoad$ **then**
    if $arrT_i < T$ **then**
      $event = [rob_i, \pi_{i,[c]}, arrT_i, 1]$
    end if
  end if
  if rob_i is $onTask$ **then**
    if $depT_i < T$ **then**
      $event = [rob_i, \pi_{i,[c]}, depT_i, 0]$
    end if
  end if
end for
return $event$

Algorithm 3 Procedures of memetic algorithm

Generate an initial population.
while The stop criterion is not met **do**
  Set an intermediate population $pop_t = pop$.
  for $x = 1 \rightarrow opsize$ **do**
    Randomly select different solutions X_a and X_b from pop.
    Apply the crossover operator to X_a and X_b to generate offspring S.
    $pop_t = pop_t \cup S$
  end for
  Do the local search process.
  Sort the individuals in pop_t by fitness.
  Set pop = the best N_{pop} individuals in pop_t.
end while

process for enhancing the population's performance. After the local search process, the N_{pop} best individuals of pop_t will be selected to the next generation. The entire procedure is described in Algorithm 3 and operators are discussed in the following sections.

1) Crossover: In the crossover procedure, two parent individuals denoted as X_a and X_b are randomly selected to produce offspring. A solution X_a can be divided into n parts as $[\pi_{a,1}, \ldots, \pi_{a,n},]$. $\pi_{a,i}$ and $\pi_{b,i}$ are permutations of tasks, which are similar to solution representations in VRP. Each permutation in parent individuals is executed by a partially matched crossover [15] shown in Fig. 2. The partially matched crossover can guarantee child permutations feasible. Then, all permutations are combined to produce two offspring S and add them to pop_t.

2) Local search: A local search procedure is designed to enhance an individual's quality by exploiting its neighbourhood. Thus, learning takes place in every generation of the proposed MAs. The swap operator, which is a traditional local search operator for a discrete optimisation problem, is used to produce its neighbour individuals [16]. One or two permutations of an original individual are randomly selected. The swap operator is applied to selected permutations to

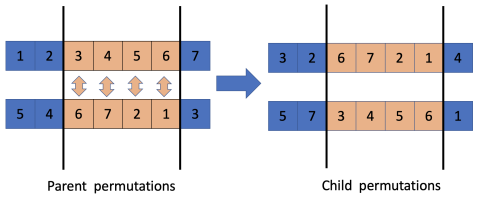

Fig. 2. Partially matched crossover.

Algorithm 4 Procedures of probability local search

for all $X_t \in pop_t$ **do**
  Sample a random number r from the uniform distribution between 0 and 1;
  if $r < P_{ls}$ **then**
    Apply local search to X_t to generate X_{ls}
    if X_{ls} is better than X_t **then**
      $X_t = X_{ls}$
    end if
  end if
end for

produce neighbouring individuals. Two local search strategies named OLS and MLS are designed in this paper, which are shown in Algorithm 4 and 5 respectively. In the OLS strategy, each individual in pop_t will explore its neighbourhood by the same probability P_{ls}. During the local search process, 10 neighbouring individuals of an origin individual are generated by the swap method. If the best performance neighbouring individual X_{ls} is better than the original individual, it will replace the original one. In the MLS strategy, the best performance individual X_e in pop_t is selected to exploit its 10 neighbouring individuals by a certain probability P_{ls}. If a better individual is found, it will replace the best performing individual and continue to participate in the subsequent local search process until no better individuals can be found by the MLS strategy. It is noticed that OLS can maintain a better diversity of the population, but it has a slower convergence speed. On the other hand, the MLS has a strong convergence speed, but it may fall into the local optima.

Algorithm 5 Procedures of elite local search

Sample a random number r from the uniform distribution between 0 and 1;
if $r < P_{ls}$ **then**
  while True do
    Apply local search to X_e to generate X_{ls}
    if X_{ls} is better than X_e **then**
      $X_e = X_{ls}$
    else
      break;
    end if
  end while
end if

3) Survivor Selection: The survivor selection scheme is a ranking technique. The best individual has a rank 1, and the worst individual has a rank $|pop_t|$. The top best N_{pop} individuals of pop_t propagate to the next generation. This method gives populations a hard selective pressure to converge to a good solution and includes an elitist strategy.

V. Computation Experiments

To evaluate the efficacy of the proposed decoding method, local search strategies and MAs, two experiments have been carried out. During the first part, we mainly focus on the effect of the proposed decoding method and the local search strategies. After that, the performance of the proposed algorithm compared with a state-of-the-art algorithm is investigated. According to [16], [17], the parameters of the proposed algorithm are set as follows: $N_{pop} = n \times m$, $P_{ls} = 0.2$, $opsize = 2$. As different algorithms have different numbers of fitness evaluation (NFE) in one generation, in order to keep the fairness of comparison, the stopping criteria of following test algorithms are set as that NFE $= n \times m \times 700$. All the values in following figures are an average of 30 independent runs.

Since there is no standard benchmark instance for testing MPDA problem, we designed a set of benchmark instances according to the method which generates benchmarks for the capacitated VRP [18]. All robots and task are located at 100 $\times$ 100 planar configuration space. Positions of tasks (PT) and positions of robots (PR) are classified into five alternative types: Random (R), Clustered (CL), Random-Clustered (RC), Eccentric (E) and Central (CE). The abilities of robots and the initial rates of tasks can be classified as Unitary (U); small values, large coefficient of variation (SVLCV); large values, large coefficient of variation (LVLCV); small values, small coefficient of variation (SVSCV); small values, large coefficient of variation (LVSCV); depending on quadrant (Q). The main characteristics of eight benchmarks are shown in Table I.

TABLE I
Main Characteristics of Eight Benchmarks.

No.	n	m	PR	PT	β	α
1	5	4	RC	RC	SVLCV	SVSCV
2	5	5	E	R	SVSCV	LVSCV
4	8	8	CL	CL	SVLCV	U
3	8	8	E	R	U	Q
5	11	11	R	CL	SVLCV	Q
6	17	23	RC	CL	LVLCV	LVSCV
7	20	20	CL	R	Q	LVSCV
8	20	18	R	E	Q	SVLCV

A. Parameter Sensitivity Analysis

This part analyses the sensitivity of MA-OLS and MA-MLS on their different parameters in order to find an appropriate parameter for the following experiments. $opsize$ is an intrinsic parameter. We will analyse its sensitivity by running experiments on two typical instances. One is a small-scale No.3 instance, the other is a large-scale No.7 instance. Fig. 3 shows

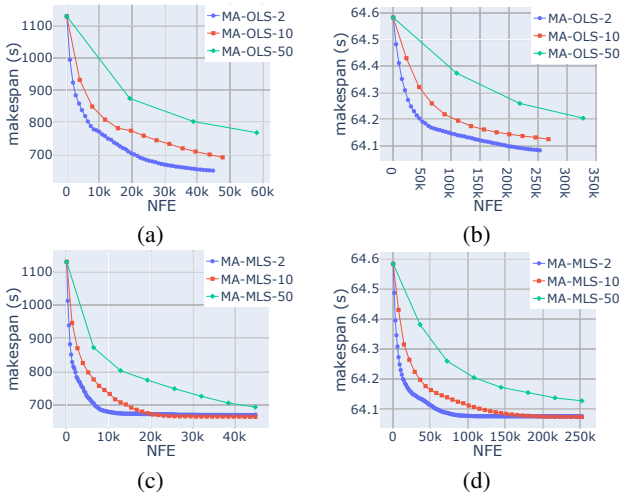

Fig. 3. Evolutionary progress of MA-OLS and MA-MLS with different $opsize$ for the selected problem instances. (a) MA-OLS on No. 3 (b) MA-OLS on No. 7 (c) MA-MLS on No. 3 (d) MA-MLS on No. 7

convergence curves under different $opsize$, which is set as 2,10 and 50 separately.

$opsize = 2$ will be adopted in following experiments, because it get a fast convergence speed and a stable convergence result. From Fig. 3, it is noticed that the smaller $opsize$ can get a stronger convergence speed than the greater $opsize$ in the MAs. MA-MLS with $opsize = 2$ and $opsize = 10$ both can converge to stable states, and convergence results approximates to each other. Thus, different $opsize$ do not make a significant difference in the later phase of the evolutionary process.

B. Comparison between MA-OLS and MA-MLS

We here analyse the progress of MA-OLS and MA-MLS to verify the efficacy of the designed local search processes. The GA method that the local search process is replaced by a mutation process is also compared as the baseline.The mutation process exchanges two tasks randomly in permutations [19]. The probability of mutation in GA is set as 0.2, which is same as P_{ls}. For brevity, we consider No.2, 4, 6 and 8 problem instances. Fig. 4 shows curves of the average minimum fitness in a population during the evolutionary process of these three algorithms.

From Fig. 4(a) and Fig. 4(b), it can be observed that MA-OLS achieved the minimal fitness among the three algorithms at the end of evolution on small-scale instances. As to the large-scale instances, Fig. 4(c) and Fig. 4(d) both show that MA-OLS can maintain a steady convergence speed during the whole optimization process, however, on the No. 6 instances, the convergence speed is not fast enough to get a better solution than GA and MA-MLS.

Contrarily, MA-MLS outperforms GA and MA-OLS on large-scale instances. On small-scale instances, its performance is not satisfactory. MA-MLS has the fastest convergence speed among these three algorithms on all instances in the former phase of evolution and the convergence progress

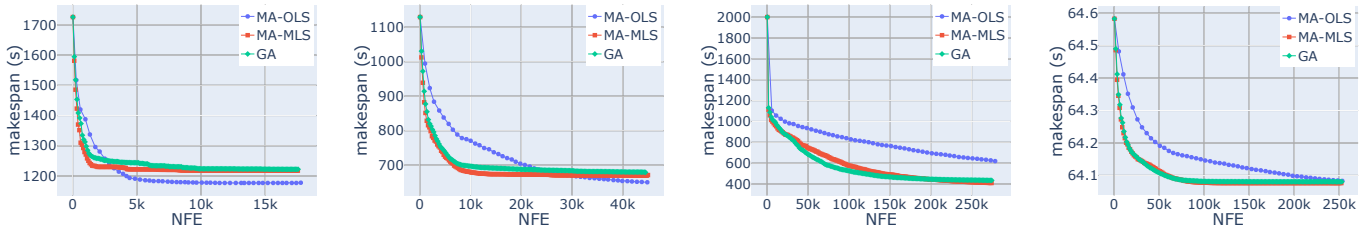

Fig. 4. Evolutionary progress of MA-OLS, MA-MLS and GA for the selected problem instances. (a) No. 2. (b) No.4 (c) No.6 (d) No.8

slows down in the later phase of evolution. The reason is that the elite individual that has been trapped into local optima has a very small opportunity for improvement by the swap operator, and it will also lead other individuals to the same local optimum..

The performance of GA is between the proposed two MAs and the evolutionary process of GA stops earliest. It indicates that the traditional mutation process of GA has difficulty to handle the premature convergence. According to the aforementioned results, we can conclude that MA-OLS is suggested to use with a sufficient NFE and MA-MLS is suggested to adopt when an acceptable solution is required in a short time.

C. Comparison of Decoding Methods

The performances of the two MAs with different decoding methods are analysed in order to verify the efficacy of the proposed decoding method. No.2, 4, 6 and 8 problem instances are also considered for brevity. In Fig. 5, the legend whose suffix is TD represents that the decoding method proposed by [10] is used and the abscissa shows the running time.

From Fig. 5, it can be observed that the decoding method proposed in this paper has a lower time cost than the previous method in the same NFE. In addition, the MAs using the proposed decoding method converge faster than the MAs using the previous decoding method. When given the same computational time budget, the MAs using the proposed decoding method perform better.

Besides these observations, the Wilcoxon rank-sum test is used to further analyse the results. The statistical results of the best individual in each run demonstrate that obvious dominance relationships of MAs with two decoding methods do not exist in the same NFE on No.2, 4, 8 instances. On No.6 instance, MAs with the previous decoding method can get a better solution with a higher computation cost in the later phase of evolution.

From the perspective of individual learning strategies, the previous decoding method can be regarded as a multi-step local search process using the triangle inequality's knowledge. The local search process will be carried on each individual during the evolutionary process. Actually, there are some worse individuals that do not need the local search process in each generation. Thus, the previous decoding method will cause a waste of computing resources. In summary, the decoding method proposed by this paper more efficient than the previous one due to its lower computation complexity.

D. Comparison with EDA

Here, the performances of MA-OLS, MA-MLS, and EDA are shown in order to verify the efficacy of proposed algorithms. The EDA counts the sequence number that the task appears in the robot's task planning order and updates a probabilistic model by some elite individuals. Parameters are set as same as [10], except that the stop criterion is set to NFE $= n \times m \times 700$. We use the best and average fitness of Hall-of-fame individuals of 30 independent runs as the comparison metric, which are shown as Table II. An additional row at the bottom of this table represents the number of instances on which the novel algorithm has achieved the better result than the EDA methods under the Wilcoxon rank-sum test. The best mean values are highlighted in bold for all instances.

From Table II, it is noticed that the number of best performance instances of EDA is 1, which is less than the other two MAs. When the characteristic of an instance does not distinguish indexes of tasks or robots, the method of counting the task index in a robot's action order does not work. For example, robots in the No.3 instance have the same initial positions and abilities. In this condition, the task index for one robot's action order is not important. It is important to figure out the coordination relationships among robots. When the scale of instances becomes large, the modelling method of EDA also can not describe coordination relationships among robots. Also, the fact that MA-OLS has a better performance on small-scale instances is testified again.

In addition, the standard deviation values of MA-OLS are small, which indicates that the performance of MA-OLS is stable. MA-MLS finds better solutions on more instances compared with the other two algorithms. It represents that MA-MLS has a stronger exploitation ability.

VI. Conclusion

This paper designs MAs for solving the task allocation problem efficiently in an MPDA scenario. In the proposed algorithms, a new decoding method and two individual learning strategies are designed. The decoding method, which allows robots to visit a completed task, is proposed to decrease the computation cost. When the same computational time budget is given, the proposed decoding method has a better performance during the searching process. Two individual learning

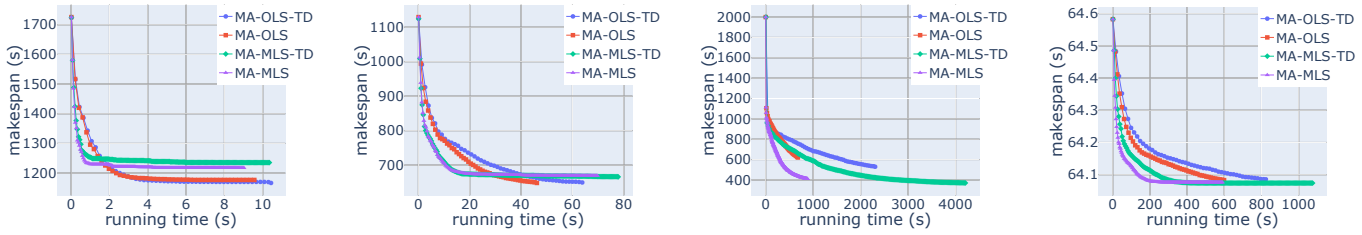

Fig. 5. Evolutionary progress of MAs with different decoding methods for the selected problem instances. (a) No. 2. (b) No.4 (c) No.6 (d) No.8

TABLE II
COMPUTATIONAL RESULTS WITH EDA OF THE BENCHMARK SET.

instance		EDA	MA-OLS	MA-MLS
1	mean	**1.052E+2**	1.059E+2(-)	1.067E+2(-)
	std	0.000E-10	1.503E+0	1.841E+0
	best	1.052E+2	1.052E+2	1.052E+2
2	mean	1.258E+3	**1.177E+3(+)**	1.218E+3(+)
	std	3.038E+1	3.493E+1	4.417E+1
	best	1.198E+3	1.135E+3	1.135E+3
3	mean	8.403E+2	**6.495E+2(+)**	6.694E+2(+)
	std	3.786E+1	2.595E+1	2.220E+1
	best	7.539E+2	6.222E+2	6.222E+2
4	mean	5.854E+2	**4.537E+2(+)**	4.652E+2(+)
	std	3.408E+1	1.398E+1	1.661E+1
	best	5.148E+2	4.401E+2	4.424E+2
5	mean	3.883E+2	3.044E+2(+)	**3.028E+2(+)**
	std	9.604E+0	1.233E+1	1.762E+1
	best	3.665E+2	2.698E+2	2.649E+2
6	mean	8.489E+2	6.237E+2(+)	**4.107E+2(+)**
	std	1.084E+2	3.066E+1	2.858E+1
	best	6.888E+2	5.540E+2	3.540E+2
7	mean	6.444E+1	6.407E+1(+)	**6.408E+1(+)**
	std	9.778E-2	7.378E-3	8.371E-3
	best	6.427E+1	6.407E+1	6.407E+1
8	mean	2.795E+2	2.344E+2(+)	**1.809E+2(+)**
	std	1.406E+1	1.016E+1	1.170E+1
	best	2.422E+2	2.016E+2	1.621E+2
(+)/(≈)/(-)			7/0/1	7/0/1

strategies are designed in this paper. The OLS has a better exploration ability and the MLS has a stronger convergence ability. The designed individual learning strategies are validated comparing experiments with GA. Finally, experiments show that MAs combined with the proposed decoding method and individual learning strategies outperform EDA, which is a state-of-the-art algorithm for solving the task allocation problem in MPDA scenarios.

In the future, the proposed MAs can be improved by combining the two local search methods to get a better balance between exploration and exploitation.

REFERENCES

[1] A. Khamis, A. Hussein, and A. Elmogy, "Multi-robot task allocation: A review of the state-of-the-art," in *Cooperative Robots and Sensor Networks 2015*. Springer, 2015, pp. 31–51.

[2] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.

[3] B. Xin, Y.-G. Zhu, Y.-L. Ding, and G.-Q. Gao, "Coordinated motion planning of multiple robots in multi-point dynamic aggregation task," in *Proceedings of the 12th IEEE International Conference on Control and Automation*. IEEE, 2016, pp. 933–938.

[4] G.-Q. Gao and B. Xin, "A-stc: auction-based spanning tree coverage algorithm formotion planning of cooperative robots," *Frontiers of Information Technology & Electronic Engineering*, vol. 20, no. 1, pp. 18–31, 2019.

[5] S. Zhang, D. Martinez, and J.-B. Masson, "Multi-robot searching with sparse binary cues and limited space perception," *Frontiers in Robotics and AI*, vol. 2, p. 12, 2015.

[6] S. Karakatič and V. Podgorelec, "A survey of genetic algorithms for solving multi depot vehicle routing problem," *Applied Soft Computing*, vol. 27, pp. 519–532, 2015.

[7] A. Hamzadayi and G. Yildiz, "Modeling and solving static m identical parallel machines scheduling problem with a common server and sequence dependent setup times," *Computers & Industrial Engineering*, vol. 106, pp. 287–298, 2017.

[8] S. Singh and I. Chana, "A survey on resource scheduling in cloud computing: Issues and challenges," *Journal of grid computing*, vol. 14, no. 2, pp. 217–264, 2016.

[9] S. Lu, B. Xin, L. Dou, and L. Wang, "A multi-model estimation of distribution algorithm for agent routing problem in multi-point dynamic task," in *2018 37th Chinese Control Conference (CCC)*. IEEE, 2018, pp. 2468–2473.

[10] B. Xin, S. Liu, Z. Peng, and G. Gao, "An estimation of distribution algorithm for multi-robot multi-point dynamic aggregation problem," in *Proceedings of the 2018 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2018, pp. 775–780.

[11] R. Hao, J. Zhang, B. Xin, C. Chen, and L. Dou, "A hybrid differential evolution and estimation of distribution algorithm for the multi-point dynamic aggregation problem," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 2018, pp. 251–252.

[12] T. Vidal, T. G. Crainic, M. Gendreau, and C. Prins, "Implicit depot assignments and rotations in vehicle routing heuristics," *European Journal of Operational Research*, vol. 237, no. 1, pp. 15–28, 2014.

[13] X. Chen, P. Zhang, G. Du, and F. Li, "Ant colony optimization based memetic algorithm to solve bi-objective multiple traveling salesmen problem for multi-robot systems," *IEEE Access*, vol. 6, pp. 21 745–21 757, 2018.

[14] B. Xin, G.-Q. Gao, Y.-L. Ding, Y.-G. Zhu, and H. Fang, "Distributed multi-robot motion planning for cooperative multi-area coverage," in *2017 13th IEEE International Conference on Control & Automation (ICCA)*. IEEE, 2017, pp. 361–366.

[15] P. Kora and S. R. Kalva, "Improved bat algorithm for the detection of myocardial infarction," *SpringerPlus*, vol. 4, no. 1, p. 666, 2015.

[16] K. Tang, Y. Mei, and X. Yao, "Memetic algorithm with extended neighborhood search for capacitated arc routing problems," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 1151–1166, 2009.

[17] Z. Ursani, D. Essam, D. Cornforth, and R. Stocker, "Localized genetic algorithm for vehicle routing problem with time windows," *Applied Soft Computing*, vol. 11, no. 8, pp. 5375–5390, 2011.

[18] E. Uchoa, D. Pecin, A. Pessoa, M. Poggi, T. Vidal, and A. Subramanian, "New benchmark instances for the capacitated vehicle routing problem," *European Journal of Operational Research*, vol. 257, no. 3, pp. 845–858, 2017.

[19] A. Garcia-Najera and J. A. Bullinaria, "An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows," *Computers & Operations Research*, vol. 38, no. 1, pp. 287–300, 2011.