Multi-objective Collaborative Optimization Algorithm for Air–ground Cooperative Tasks Based on Conflict Resolution

Xuejun Zhang[1], Hao Liu[2], Lele Xue[3], Xiangmin Li[1], Wei Guo[1], Shuangjiang Yu[2], Jingyu Ru[*,3] and Hongli Xu[*,3]

[1] The 54th Research Institute of CETC, Shijiazhuang, Shijiazhuang 050081, China;
[2] School of ISE, Northeastern University, Shenyang 110819, China;
[3] School of RSE, Northeastern University, Shenyang 110819, China.
Correspondence: rujingyu@mail.neu.edu.cn, xuhongli@mail.neu.edu.cn.

Abstract. Using air–ground cooperation to perform tasks, such as detection and search on specific areas, is an essential means to accomplish special tasks. Research on the scheduling and matching of heterogeneous agents for intelligent tasks to meet special task requirements under the condition of satisfying multifaceted party requirements is a popular research topic in the field of task assignment using multiple agents. To overcome the shortcoming of traditional task assignment which only has a single matching dimension and does not consider task conflicts, this paper proposes a novel NSGA-III-based multi-objective task assignment algorithm for heterogeneous agents, namely heterogeneous conflict resolution multitasking optimization(HCRMO), to address the traditional task assignment process. The algorithm comprises a conflict-free minimum solution space and a task-agent conflict resolution module(TCCRM). The algorithm can build a search gene pool to meet the requirements, construct a solution space for each task, and quantify the potential conflict indicators of the task simultaneously to weaken the impact of agent conflicts due to variant search. Finally, the feasibility and effectiveness of the algorithm are verified through simulation.

Keywords: Air–ground cooperation, conflict resolution, multi-tasking optimization, NSGA.

1 Introduction

The execution of tasks is changing toward multidimensionality, specialization, and intelligence with the development of information technology. The air–ground collaborative mission assignment technology with capabilities, such as heterogeneous detection and search, has become a research hotspot for solving related problems due to its practicality, intelligence, and robustness in real complex environments.

Intelligent task assignment algorithms in complex environments have been widely studied by many scholars, and various genetic algorithms, simulated annealing algorithms, and particle swarm algorithms have been widely used in

search and rescue, detection, management, and communication industries [1, 2]. In the traditional task assignment process, part of the research results assumes that the agent only has the ability of a single dimension, and task optimization algorithms are studied in the field of task assignment based on this assumption [3, 4]. Lemaire, Alighanbari, and other scholars have studied the matching of multiple tasks with multiple agents, and these studies are widely used in UAV task assignment [5, 6]. However, most of these studies focus on task assignment investigations of homogeneous agents, not only the studies for heterogeneous agents with multidimensional capabilities is minimal, but most of the existing heterogeneous task allocation methods do not consider the problem of multi-agent cooperation [7]. Thus far, the research on multidimensional heterogeneous agent task assignment algorithms for real environments has been rarely addressed.

The multidimensional heterogeneous agent task assignment algorithm studied in this paper belongs to the type of single-task, multi-robot and time-extended assignment tasks [8], and the research on this type of task will introduce many benefits. First, the heterogeneous characteristics of the system can ensure that the system has excellent air-Cground cooperative operation capability when facing task points in different locations and environments in a real environment task execution scenario [9,10]. Second, considering effectiveness, the intelligent task allocation of the system can reasonably arrange limited agents to avoid agent conflicts between different demands in time and space domains in multi-task situations, thus maximizing the agent allocation efficiency of the system.

Implementing a multidimensional heterogeneous agent task allocation algorithm is difficult despite its numerous benefits. Moreover, this algorithm faces two main challenges as follows. First, different types of intelligence are occasionally required to collaborate to complete the tasks despite the variations in task requirements and usage conditions of real environments. Second, a conflict exists between multiple tasks in the time and space domains. Therefore, designing a set of task assignment strategies is essential to meet the multidimensional requirements using heterogeneous agents [11].

This paper proposes a novel NSGA-III based heterogeneous agent multi-objective task assignment algorithm, namely heterogeneous conflict resolution multitasking optimization (HCRMO) algorithm, to overcome the above challenges. In particular, the contributions of this paper are as follows.

1. An algorithm based on the construction of a conflict-free minimal solution space is proposed. This algorithm can build a search gene pool to satisfy the task requirements, construct a solution space for each task, and improve the optimization efficiency of the agent allocation search.

2. A task-agent allocation conflict configuration module that quantifies the potential conflict indicators of tasks is introduced, thus weakening the impact of agent conflicts from variant search.

3. Numerous simulation experiments on a hypothetical task are conducted in this paper using a mountain map in a real environment, and the performance

is compared with the classical multi-task assignment algorithm to verify the practicality and effectiveness of the algorithm.

2 HCRMO algorithm

2.1 Problem description

Traditional task planning algorithms focus on analyzing the agent allocation problem in ideal situations, ignoring the relationship between task and agent. By default, all agents are homogeneous and only one agent is needed to complete the task. However, agents are usually heterogeneous, which would increase the constraints in task assignment. In air–ground collaborative task assignment, unmanned vehicles are limited by factors such as terrain and landscape and are slow to reach the task location sometimes. And although UAVs can overcome the terrain problem, they cannot perform practical operations such as soil collection like unmanned vehicles. Moreover, in the real environment, the agent performing the task does not always complete the task independently and usually needs multi-agent cooperation for task. For example, int the Table 1, $task_1$ needs to detect, collect, and search; $task_2$ needs to detect, collect, and patrol; $task_3$ needs to collect and search.

Table 1. task requirement vector set

	Detect	**Collect**	**Search**	**Patrol**
$task_1$	1	1	1	0
$task_2$	1	1	0	1
$task_3$	0	1	1	0

Table 2. Agent capability vector set

	Detect	**Collect**	**Search**	**Patrol**
$agent_1$ (vehicle)	1	0	0	1
$agent_2$ (vehicle)	0	1	0	0
$agent_3$ (vehicle)	0	1	0	1
$agent_4$ (UAV)	0	0	1	0
$agent_5$ (vehicle)	1	1	0	0
$agent_6$ (vehicle)	0	1	0	1

In the Table 1, 1 represents that the task needs this ability, and 0 represents that it does not need this ability. Overall, four kinds of abilities are required in task allocation. Performing tasks using six agents is assumed, and the capabilities of each agent are different.

The task demand and agent ability vectors can be summarized. The demand vector of $task_1$ is $task_1 = [1, 1, 1, 0]$. In the Table 2, $agent_1$ ability vector is $agent_1 = [1, 0, 0, 1]$, $agent_1$ & $agent_2$ & $agent_4$ = [1,1,1,1], and agent 1,2,4 can cooperate to complete $task_1$. The actual situation is presented as Fig. **??**.

Fig. **??** shows a hypothetical mountainous area. The tasks and agents in the Fig. **??** correspond to Table 1 and 2. The solid line represents the route for agents to execute tasks directly, and the dotted line represents the $agent_4$ with its speed advantage can arrive at $task_1$ in time to complete it in cooperation with $agent_1$ and $agent_2$ after performing $task_3$ if $task_1$ starts after $task_3$. The agent assignment in Fig. **??** is one of the many assignments that meet all the task requirements.

2.2 Basic setting

This section mainly introduces the formalization of the task assignment problem. The following is the definition of each element in the task allocation process.

Heterogeneous agent: As the subject of task execution, heterogeneous agent is represented as $agent_i(i = 1, 2, \cdots n)$, where n represents the number of agents and the initial position of the agent is (x_i^a, y_i^a), the velocity of the agent is v_i, and the ability vector of agent is $ability_i = [1, 0, 1, \cdots]$. Each ability cannot be replaced by other abilities, and its dimension is the same as the dimension requirement of the task.

Specific task: The specific task is expressed as $task_i(i = 1, 2, \cdots, m)$, where m is the number of tasks, and the position of tasks is (x_i^t, y_i^t). Each task has start and end times, which are respectively expressed as $[start_i, end_i]$. Moreover, each task has its demand, which can be summarized as the demand vector and expressed as $demand_i = [1, 0, 1, \cdots]$.

Optimization objectives: Three optimization objectives are set to evaluate the overall completion of the task.

The first objective is the total driving distance, which indicates the total driving distance of all agents after the completion of all tasks. The fuel consumption is minimal when the total distance is small.

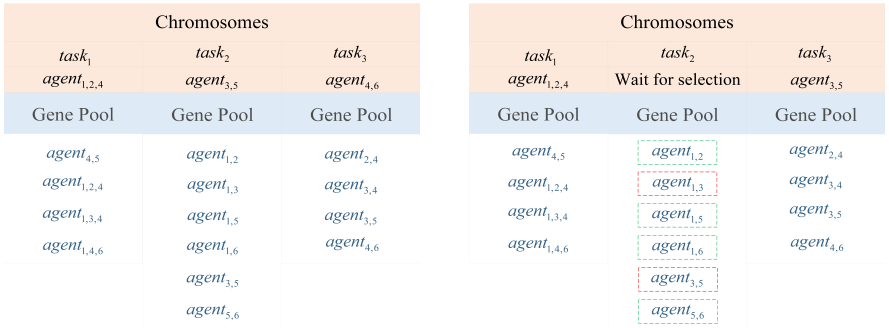
The second optimization objective is the undesirable distance. There are always areas in the map that are undesirable to cross, and these areas are distributed all over the map, sometimes between missions, and may need to be crossed if the total distance is to be reduced.

The third objective is agent utilization, which refers to the number of agents mobilized after completing all tasks; low utilization rate of agents means limited use of agents.

2.3 HCRMO algorithm description

construction of conflict free minimum solution space: The traditional NSGA-III algorithms take the agent as the smallest gene segment and conduct crossover and mutation at the agent level. Achieving the effect of the traditional

algorithm considering multi-machine cooperation for task completion is difficult. One reason for this difficulty is the adoption of unconstrained random crossover and mutation in the crossover and mutation link by the traditional algorithm. The new solution generated by random crossover and mutation considering multi-machine cooperation for task completion does not necessarily meet the task requirements. Therefore, this solution is a failure solution. Such a crossover or mutation not only affects the efficiency of the NSGA-III algorithm but also leads to slow convergence speed and falls into a local optimal solution. This paper proposes the concept of conflict-free minimum agent allocation solution space, which limits the search gene pool, to solve this problem.



(a) Solution space of conflict-free minimum agent allocation

(b) The description of TCCRM

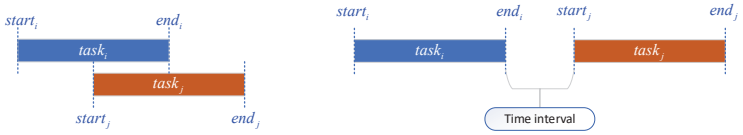
Fig. 1. The description of gene pool and TCCRM

In a chromosome, each task is a part of a specific segment. The *demand* of each task is different. Therefore, the specific segments on the chromosome have different gene requirements. By grouping agents according to task requirements, all implementations of each task can be found, thus enabling the reconstruction of the genetic fragments of each task. All the grouping schemes come together, which is the conflict-free minimum agent allocation solution space (gene pool) $Sov_j^i = \left[s_1^i, s_2^i, s_3^i, \cdots\right]$, where s_j^i represents the jth grouping scheme in the gene pool of $task_i$. The original unconstrained random crossover and mutation can be constrainted by gene pool which constructing for each task. Thus, the new genes generated by each crossover or mutation can meet the requirements of corresponding chromosome segments, and the task requirements can be continuously met. Eliminating the redundancy in agent allocation is necessary to improve the efficiency of agent allocation and reduce unnecessary waste of agents. Correspondingly, the inclusion relationship of gene fragments is excluded in the construction of gene pool, and the gene pool is simplified to avoid agent waste. The construction of conflict-free minimum agent allocation solution space is then completed. Fig. 1(a) shows all the matching relationships between task and agent in Section 2.1. $task_i$ can be completed by agent 4 and 5 or agent 1,

2, and 4. The two schemes cannot meet the task requirements after deleting any agent.

Task agent configuration conflict resolution module: Assuming that an agent can participate in $task_i$ and $task_j$, and a time conflict exists between two tasks for an agent, if the agent participates in $task_i$, then $task_j$ cannot choose the grouping scheme including this agent (whether it is crossover or mutation). This situation is called agent allocation conflict. The traditional cross-mutation method used to solve the agent allocation conflict will produce a chain reaction. That is, the modification of a gene segment on the chromosome will lead to a conflict between this segment of the gene and its other parts. Therefore, the agent allocation scheme is unreasonable, and the chromosome is invalid. This paper proposes a task agent configuration conflict resolution module, which is also called conflict resolution module, to solve this problem.

Temporal constraints can be characterized as hard or soft. Hard temporal constraints cannot be violated. The temporal constraints in this paper are hard constraints, containing only the start time and the end time, and the task cannot be executed beyond the start time. There are two types of temporal relationships between tasks in this paper, namely overlap and interleave, as depicted in Fig. 2.



(a) $task_i$ overlap $task_j$ (b) $task_i$ interleave $task_j$

Fig. 2. All possible relationships between tasks

The time relationship in Fig. 2(a) can be abstracted as 1. At this point, $task_i$ and $task_j$ can not use the same agent. So the conflict adjacency matrix M_{m*m} is established in accordance with the 1. When $M(i,j) = 0$, a time overlap is observed between $task_i$ and $task_j$. when M(i,j)=1, no time overlap is observed.

$$max(start_i, start_j) < min(end_i, end_j) \tag{1}$$

Fig. 2(b) shows that there is a certain time interval when $task_i$ and $task_j$ do not overlap in time. In accordance with 1 we can get 2.

$$max(start_i, start_j) - min(end_i, end_j) = TI \tag{2}$$

In 2, TI means time interval. If $agent_k$ is required to participate in the execution of $task_j$ after the execution of $task_i$, then it needs to satisfy 3.

$$\frac{\sqrt{((x_i^t - x_j^t)^2 + (y_i^t - y_j^t)^2)}}{v_k} \leq TI \tag{3}$$

Here the distance between two tasks is replaced by the Euclidean distance. Since agents are heterogeneous, there are differences in their speed. This is especially evident over UAVs and vehicles. Thus, if $agent_k$ needs to perform $task_j$ after $task_i$ under the condition of meeting the time constraint, the 2 and 3 must be transformed into the 4 from the perspective of the agent.

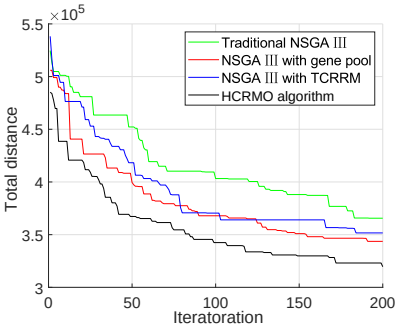
$$min(end_i, end_j) + \frac{\sqrt{((x_i^t - x_j^t)^2 + (y_i^t - y_j^t)^2)}}{v_k} \leq max(start_i, start_j) \tag{4}$$

The unique conflict matrix M_{m*m}^k based on 4 and the conflict matrix M_{m*m} is established for each agent with different speed. When $M_{k*k}^x = 1$, $agent_k$ can arrive at $task_j$ in time after executing $task_i$. When $M_{k*k}^x = 0$ means that there is a time overlap between $task_i$ and $task_j$ or the time interval between $task_i$ and $task_j$ is not enough for $agent_k$ to go from (x_i^t, y_i^t) to (x_j^t, y_j^t). According to the agent-based M_{m*m}^k conflict matrix, combined with conflict-free minimum solution space, conflicting relations between s_j^i belonging to Sov^i and s_z^k belonging to Sov^k can be checked and all s_z^k conflicting with s_j^i are preserved for the crossover and mutation. For example, suppose there is a time interval between $task_2$ and $task_3$. If $agent_3$ cannot complete the distance between $task_2$ and $task_3$ in less time than the time interval, s_2^3 is in conflict with s_2^2 and s_5^2. Therefore, if $task_3$ chooses s_2^3 in crossover and mutation, $task_2$ needs to change its previous choice and reselect the gene that does not contain $agent_3$. Therefore, genes that conflict with s_3^3 can be directly eliminated.

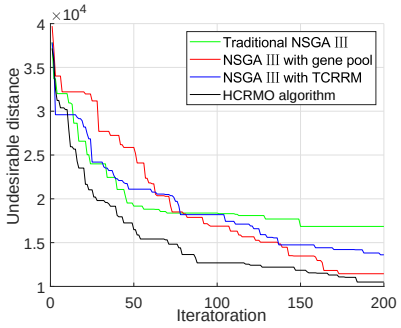
In Fig. 1(b), the red dotted line represents that $task_2$ cannot select this gene and the gene in the green dotted line can be selected. If $task_2$ selects the gene with green dotted line and there is no conflict between the gene selected by $task_2$ and other genes, this solution survives and the crossover or mutation is successful. If the gene reselected by $task_2$ still has conflicts with other genes, repeat the previous session until there are no conflicts.

2.4 Experimental results and analysis

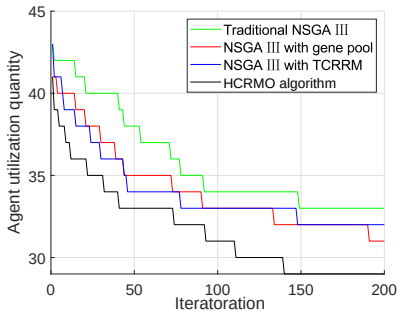
This paper compares the traditional NSGA-III algorithm, the NSGA-III algorithm fused with gene pool, and the NSGA-III algorithm fused with conflict resolution module under the condition of the same initial data to verify the performance of the HCRMO algorithm and its efficiency. First, the agent and task data are randomly generated. The position of each agent is assumed to be in the restricted area to simulate the experimental environment. x^t and y^t are random integers between [0,20000]. The start time of the task $start_i$ is a random integer within [1000,2000), and its corresponding end time end_i is $[start_i, start_i + random[200, 500]]$. The rules of generating the initial position of task and agent are the same. This rule is also observed for the initial circle center position of the undesirable areaand, and the radius of the undesirable area



(a) Total distance at 200 iterations.



(b) Threaten distance at 200 iterations.



(c) Agent utilization quantity at 200 iterations.

Fig. 3. Comparison of the effects of all algorithms

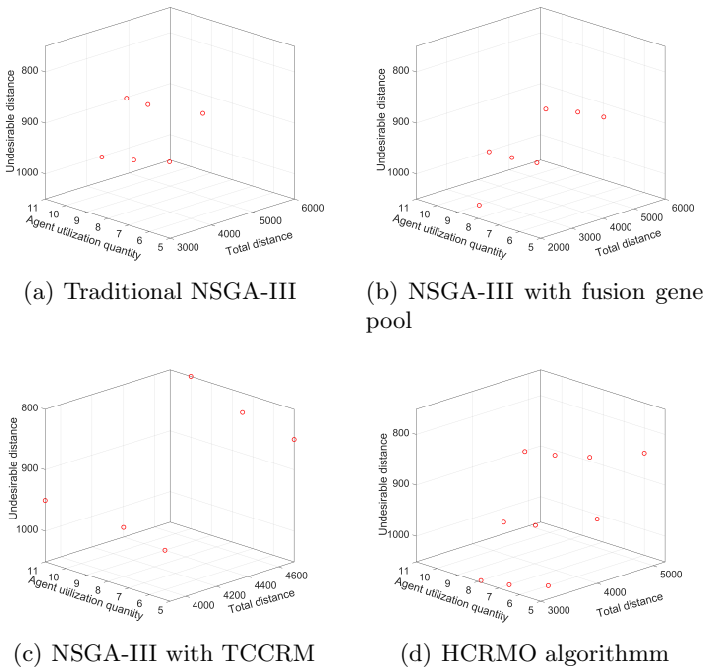
is $random[200, 500]$. The average agent speed is assumed as (5) according to the range of map and the start and end times of task.

$$v = \frac{\sqrt{(max(x) - min(x))^2 + (max(y) - min(y))^2}}{min(start)} \tag{5}$$

x is composed of x^t and x^a. Similarly, y is composed of y^t and y^a. The speed of agents is randomly assigned following the normal distribution with (5) as the mean to ensure that the agents arrive in time for the start of each task. Suppose that the functional constraint part is set to four constraint factors, and the agent has part of the capabilities randomly. The requirements of each task can be met by the cooperation between agents.

Fig. 3 shows the effect of the decline of each optimization objective with the increase of the number of iterations in Experiment 1, which contains 40 tasks and 50 agents. From the Fig. 3, it can be seen that the HCRMO algorithm outperforms the other algorithms on all three optimization objectives, and more importantly, the HCRMO algorithm takes the shortest time, only 463s, the traditional NSGA-III algorithm takes 3027s, NSGA-III with fusion gene pool takes 1904s, and NSGA-III with TCCRM takes 2351s. As a result, the HCRMO al-

gorithm is six times more efficient than the traditional NSGA-III algorithm.



(a) Traditional NSGA-III

(b) NSGA-III with fusion gene pool

(c) NSGA-III with TCCRM

(d) HCRMO algorithmm

Fig. 4. Experimental results of different algorithms

Fig. 4 shows experiment 2, which contains 10 tasks and 20 agents. The traditional NSGA-III takes 135s to find six non-dominated solutions, and the quality of the solutions is poor. The NSGA-III algorithm based on the fusion gene pool takes 108s to find seven non-dominated solutions with good quality. The NSGA-III with conflict resolution module takes 97s to find six non-dominated solutions, but the quality of the solutions is poor. The HCRMO algorithm finds ten non-dominated solutions with good quality.

3 Conclusion

In this paper, a HCRMO algorithm based on NSGA-III that can be widely applied to real scenarios is proposed to solve the heterogeneous generation task assignment problem. HCRMO algorithm fuses the gene pool and conflict resolution modules, which substantially improves the convergence speed and the quality of the pareto solution of the algorithm. To demonstrate the effectiveness of the HCRMO algorithm, we designed two sets of experiments. These two sets of experiments show that the HCRMO algorithm converges with high efficiency regardless of the number of tasks and agents. the HCRMO algorithm provides a new idea for analyzing the task assignment of heterogeneous intelligences.

References

1. A. Alhaqbani, H. Kurdi, and K. Youcef-Toumi, "Fish-inspired task allocation algorithm for multiple unmanned aerial vehicles in search and rescue missions," *Remote Sensing*, vol. 13, no. 1, 2021.
2. L. Huo, J. Zhu, G. Wu, and Z. Li, "A novel simulated annealing based strategy for balanced uav task assignment and path planning," *Sensors*, vol. 20, no. 17, 2020.
3. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.
4. O. Shehory and S. Kraus, "Methods for task allocation via agent coalition formation," *Artificial intelligence*, vol. 101, no. 1-2, pp. 165–200, 1998.
5. T. Lemaire, R. Alami, and S. Lacroix, "A distributed tasks allocation scheme in multi-uav context," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 4. IEEE, 2004, pp. 3622–3627.
6. M. Alighanbari and J. P. How, "Decentralized task assignment for unmanned aerial vehicles," in *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005, pp. 5668–5673.
7. P. Schillinger, M. Bürger, and D. V. Dimarogonas, "Simultaneous task allocation and planning for temporal logic goals in heterogeneous multi-robot systems," *The international journal of robotics research*, vol. 37, no. 7, pp. 818–838, 2018.
8. 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.
9. A. J. Page, T. M. Keane, and T. J. Naughton, "Multi-heuristic dynamic task allocation using genetic algorithms in a heterogeneous distributed system," *Journal of parallel and distributed computing*, vol. 70, no. 7, pp. 758–766, 2010.
10. Y. Zhang and L. E. Parker, "Considering inter-task resource constraints in task allocation," *Autonomous Agents and Multi-Agent Systems*, vol. 26, no. 3, pp. 389–419, 2013.
11. E. Nunes, M. Manner, H. Mitiche, and M. Gini, "A taxonomy for task allocation problems with temporal and ordering constraints," *Robotics and Autonomous Systems*, vol. 90, pp. 55–70, 2017.