引用格式：郑习羽，徐梓毓，王京华. 基于聚类分组的异构多机器人任务分配算法研究[J]. 航空兵器，2022，29(4)：100－109.
Zheng Xiyu，Xu Ziyu，Wang Jinghua. Research on Task Allocation of Heterogeneous Multi-Robot Based on Cluster Grouping Algorithm [J]. Aero Weaponry，2022，29(4)：100－109.(in Chinese)

基于聚类分组的异构多机器人任务分配算法研究

郑习羽[1]，徐梓毓[1]，王京华[1,2*]

(1. 长春理工大学 机电工程学院，长春　130022；
2. 长春理工大学 跨尺度微纳制造教育部重点实验室，长春　130022)

摘　要： 在城市作战环境中，具有时间窗约束的大规模任务目标需要调度多个空地异构机器人处理，为了保证战略执行的快速有效，需要快速计算出可接受的次优任务分配策略。本文提出了一种基于聚类分组的一致性的束算法(C－CBBA)，该分布式算法用于具有时间窗约束的大规模任务分配问题中。算法首先对任务目标点和机器人进行分组，将大规模问题转化为小规模问题，其中分组算法包括三个阶段：(1)利用 K－means 算法按照距离最近原则对任务点进行初步分类；(2)将每个组的任务点组别进行调整，使其不超过每组预分配机器人的载荷上限；(3)利用延迟接受算法(DA)给每个组分配距离最近的机器人。最后改进基于一致性的束算法(CBBA)，对每个子问题进行任务分配问题求解。仿真结果表明，提出的算法具有较高的任务完成度，并且有效减少机器人之间的通信量。

关键词： 聚类分组；多机器人系统；时间窗；任务分配；CBBA 算法；无人系统；作战
中图分类号： TJ810；V279　　**文章编号：** 1673－5048(2022)04－0100－10
文献标识码： A　　**DOI：** 10.12132/ISSN.1673－5048.2021.0249

0　引　　言

诸如无人机、无人车的无人系统(机器人)近年来发展十分迅速，凭借其优秀的自主规划与执行能力，在许多领域得到了广泛应用，包括救援搜索[1]、行星探索[2]、军事行动[3]、生产调度[4]、资源分配[5]等。现代的工程及军事任务变得越来越复杂，往往一个总体任务包含许多个不同类型的子任务，如战场环境中的任务包含探索、打击和毁伤评估等，单个机器人已经无法满足解决大范围内的复杂任务，需要构建多机器人系统协同完成多项不同类型的子任务。

多机器人任务分配(MRTA)问题是多机器人系统协同控制的一个重要研究方向[6]。该问题需要多个机器人在一定约束条件下，规划出无冲突的任务分配策略，并获得全局收益良好的解。这个问题在规定时间内求解是困难的，求解全局最优解需要花费大量的时间和计算量。当分配问题比较复杂时，解集无法穷举出来，几乎不可能求出最优解，所以为了保证算法的效率，通常用近似算法提供一个可接受的次优解。

任务分配方法可以分为集中式和分布式两种。集中式方法需要中央处理器为整个系统生成任务计划，然后将结果传递给各个机器人。蚁群算法[7]、粒子群算法[8]、遗传算法[9]等集中式智能优化算法在解决 MRTA 问题方面都具有较优的解。虽然集中式方法结果全局较优，但是依赖大量的通信负载和计算量。此外，集中式方法的鲁棒性比较差，机器人容易出现单点故障，且对动态变化的通信响应较慢[10]。分布式方法不需要中央处理器，每个机器人在内部生成任务清单，再和周围的机器人通过通信解决冲突。分布式系统能够快速响应外部环境的变化、对通信带宽的依赖性比较小。虽然优化结果不如集中式，但是算法效率和鲁棒性更高，单点故障的影响更小，能够很好地应对野外复杂环境的不确定性[11]。分布式方法主要分为基于博弈论的机制和基于市场的机制，其中博弈论需要相对更长的收敛时间，目前基于市场机制的拍卖算法是广泛应用的高效方法。

Choi 等[12]结合拍卖机制和信息共识机制，提出了基于一致性的束算法(CBBA)，与传统的拍卖算法不同，该算法不需要有中间拍卖商，每个机器人内部具有一致的任务投标规则。CBBA 被证明可以在较短时间内收敛于至少 50% 的纳什均衡解[13]。一些研究在 CBBA 的基础上

收稿日期：2021－12－10
基金项目：露泉创新基金项目(LQ2020－01)
作者简介：郑习羽(1995－)，男，浙江宁波人，硕士研究生。
* 通信作者：王京华(1980－)，男，吉林榆树人，博士。

免费获取电子版

进行了改进，使其适应更复杂的问题。Johnson 等[14]提出了异步解决任务冲突的规则，减少不必要的通信，尽量降低通信负载，实现了 CBBA 的异步通信。Buckman 等[15]考虑到时间敏感和动态出现的约束，提出一种重规划的 CBBA－RP，提高任务快速协调性。基于任务优先级约束，Binetti 等[16]提出了一种分散关键任务分配算法(DCTAA)，将标记的关键任务优先分配给机器人，实现最大化奖励且保证关键任务被全部分配。Ye 等[17]考虑到任务之间的耦合约束，通过引入插入位置可行性处理有冲突的任务规划，并改进了任务规划策略，减少不必要的任务选择计算。一些研究受到 CBBA 的启发，其中性能影响(PI)算法是针对搜救场景开发的，在解决时间约束问题上有良好的表现[18]。但是过快的迭代容易使算法陷入局部最优。Whitbrook 等[19]改进 PI 算法，提出了 PI－softmax 算法，极大提高了算法的任务完成度和任务奖励，但算法的收敛时间在规模较大的人物场景中是难以接受的。

针对大规模的任务场景，上述研究无法获得很好的分配结果。Fu 等[20]将机器人分成若干个组，每组各自生成任务计划，再由每个组的领头机器人传递给其他组，用两层共识规则将大规模问题细分为一个个小规模问题。Jang 等[21]利用博弈论方法，自组织地将机器人按照距离和任务喜好进行任务分配，解决了大规模机器人的分配问题。但是机器人之间的通信量依然没有减少，且算法效率也比较低。

本文针对大规模任务分配问题，在有时间窗的约束下，改进 CBBA 算法，提出一种基于聚类分组的一致性的束算法(C－CBBA)，解决了大规模问题的有限通信问题，在减少通信量的前提下，提高算法效率，保持任务分配结果具有较高的任务完成度和全局任务奖励。

本文的主要贡献为:

(1) 任务点和机器人的聚类分组

基于距离因素，用 K－means 算法对任务点进行分组，然后利用延迟接受(DA)算法将机器人分配给各个分组，减少任务分配算法消耗的通信量，提高算法效率。

(2) 改进任务序列包添加策略

建立未选择任务包，将未被选择的任务尽可能加入任务序列包中，提高任务完成度。

1 问题描述

本文的 MRTA 问题属于单任务单机器人延时分配(ST－SR－TA)问题[22]，即每个任务只需要一个机器人完成，每个机器人一次只能完成一个任务。作战场景如图 1 所示，场景中有异构的无人机和无人车，要求在时间窗的约束下，分别完成多个任务。任务类型分为探索、打击和评估毁伤目标。针对本文研究的任务环境，做如下假设:

假设 1　每个机器人在地图中都是匀速移动的，具有相同的态势感知(SA)，且通信数据不会丢失。

假设 2　地图中没有任何障碍物，机器人和任务之间的距离由欧氏距离或曼哈顿距离表示。

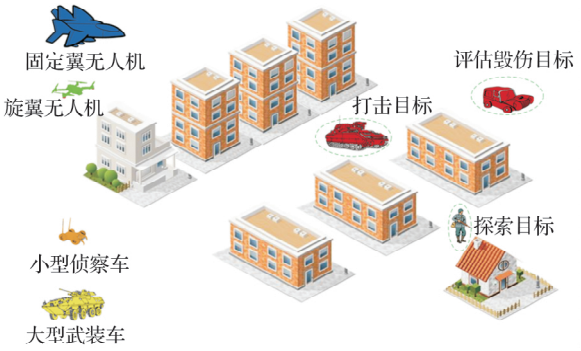

图 1　城市作战场景示意图

Fig. 1　Schematic diagram of urban combat scene

1.1 异构机器人和任务的类型

假设系统一共有 N_R 个异构机器人和 N_T 个不同类型的任务，机器人和任务集合分别为

$$\boldsymbol{R} = \{R_1, R_2, R_3, \cdots, R_i, \cdots, R_{N_R}\} \tag{1}$$

$$\boldsymbol{T} = \{T_1, T_2, T_3, \cdots, T_j, \cdots, T_{N_T}\} \tag{2}$$

本文考虑四种机器人: 固定翼无人机、旋翼无人机、小型侦察车和大型武装车。其中固定翼无人机和大型武装车只能执行打击和毁伤评估任务，旋翼无人机和小型侦察车只能执行探索和毁伤评估任务。规定任务分组的组数为 K。

由于时间窗的约束，机器人 i 执行任务 j 的起始时间必须在规定的时间窗以内，满足:

$$t_j^{\text{start}} \leqslant t_j^i \leqslant t_j^{\text{end}} \tag{3}$$

$$t_j^i + t_j \ll t_j^{\text{end}} \tag{4}$$

1.2 大规模 MRTA 问题目标函数

本文主要的目标是保证算法效率和通信量足够低的基础上，最大化每个分组的任务完成度和全局任务奖励。其目标函数可以表示为

$$\max \sum_{s=1}^{K} \sum_{i=1}^{N_R} \left(\sum_{j=1}^{N_T} x_{ij} \right) \tag{5}$$

$$\max \sum_{s=1}^{K} \sum_{i=1}^{N_R} \left(\sum_{j=1}^{N_T} c_{ij}(\boldsymbol{X}_i, \boldsymbol{P}_i)\, x_{ij} \right) \tag{6}$$

$$\min \sum_{s=1}^{K} f(it) \quad [R^s(R^s - 1)] \tag{7}$$

其约束为

$$\sum_{i=1}^{N_R} x_{ij} \leqslant 1, \ \forall j = 1, 2, \cdots, N_T \tag{8}$$

$$\sum_{j=1}^{N_T} x_{ij} \leqslant L_T^i, \ \forall i = 1, 2, \cdots, N_R \tag{9}$$

$$\sum_{i=1}^{N_R} \sum_{j=1}^{N_T} x_{ij} \leqslant N_{\min} \triangleq \min\left\{N_T, \sum_{i=1}^{N_R} L_T^i\right\} \tag{10}$$

式中: $x_{ij} \in \{0, 1\}$ 为决策变量，数值 1 表示机器人 i 执行任务 j; s 为任务分组的组序号; $\boldsymbol{X}_i \in \{0, 1\}^{N_T}$ 为机器人 i 对任务的可执行能力; $\boldsymbol{P}_i \subseteq (\boldsymbol{T} \cup \{\varnothing\})^{L_T^i}$ 为机器人 i 的任务序列包，储存了机器人 i 计划执行的任务及其排序; $c_{ij}(\boldsymbol{X}_i, \boldsymbol{P}_i) \geqslant 0$ 为执行任务 j 的奖励值，是与 $\boldsymbol{X}_i$ 和 $\boldsymbol{P}_i$ 相关的一个非负函数; $f(it)$ 为每个组收敛时的迭代次数; R^s

为每个组分配的机器人数量。

C – CBBA 算法整体流程框架如图 2 所示。

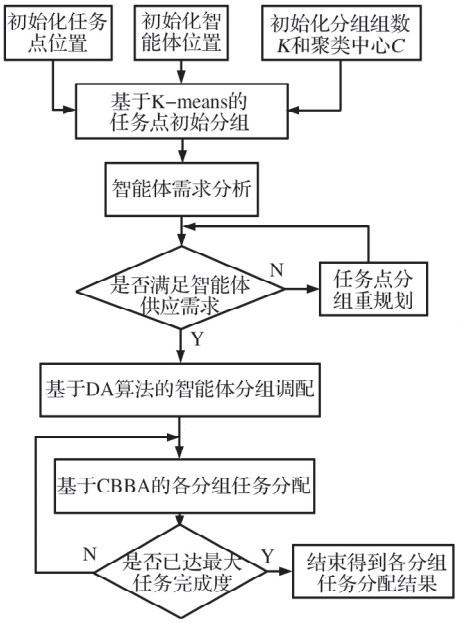

图 2　C – CBBA 算法整体流程框架

Fig. 2　Overall process framework of C – CBBA

2　任务点聚类分组

在三维空间中，既有高空任务，也有地面任务。受限于地面机器人无法执行高空任务，为了保证分配给各组的机器人能够完成任务，在任务分组阶段，将高空任务和地面任务区分开，单独划分出高空任务，其他地面任务再进行分组。即其中一组为高空组，剩余的 $K-1$ 组为地面任务组。

2.1　聚类算法

聚类算法属于无监督学习，通过对任务点坐标值的无标记样本训练，对任务点的分组进行区分。考虑到训练的目标只有任务点的坐标值，本文用 K – means 的方法对任务点进行分组。

K – means 和 K – medoid 是两种最常见的聚类分组算法。K – medoid 对噪声鲁棒性比较好，可以避免“噪声”对聚类结果的影响，但计算速度较慢[23]。K – means 的计算效率更高，针对大样本学习的情况，少数的“噪声”任务点对整体聚类中心的影响不大。所以，K – means 更适应大规模任务点的聚类分组。

2.1.1　预定义聚类中心

针对聚类分组问题，K – means 算法需要事先预定聚类中心，通常随机寻找 K 个任务点作为初始聚类中心。但在分布式系统中会造成不一致的分组，需要预定统一的初始聚类中心。

首先，定义高空任务组的初始聚类中心，假设任务点的出现在三维空间中服从高斯分布，将地图正中心点作为高空组的初始聚类中心，组别为 $s=1$，其坐标向量可以表示为

$$\boldsymbol{C}^{s=1}=\frac{1}{2}[X_{\max},Y_{\max},Z_{\max}] \tag{11}$$

式中：$X_{\max}$，$Y_{\max}$，$Z_{\max}$ 分别为地图极限值。

然后，定义剩余的 $K-1$ 组地面任务组的初始聚类中心。假设地面组的初始全局中心为

$$\boldsymbol{C}^{\text{map}}=\frac{1}{2}[X_{\max},Y_{\max},0] \tag{12}$$

如图 3 所示，以地面中心 $\boldsymbol{C}^{\text{map}}$ 为圆心，各分组聚类中心 $\boldsymbol{C}^{*}$ 围绕其圆心均匀分布。各分组聚类中心三个坐标轴的坐标值分别表示为

$$C_X^s=C_X^{\text{map}}+\frac{X_{\max}}{4}\cdot\sin\left[(s-2)\frac{2\pi}{(K-1)}+\alpha\right] \tag{13}$$

$$C_Y^s=C_Y^{\text{map}}+\frac{Y_{\max}}{4}\cdot\cos\left[(s-2)\frac{2\pi}{(K-1)}+\alpha\right] \tag{14}$$

$$C_Z^s=0 \tag{15}$$

式中：$\alpha=\pi/4$。

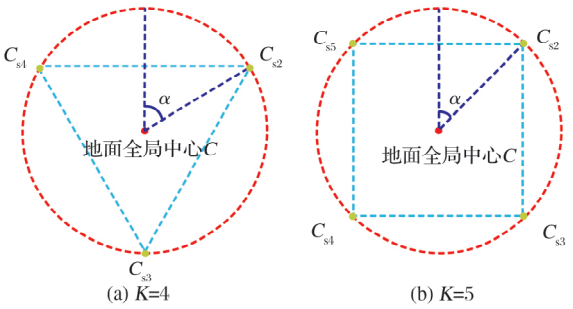

图 3　地面分组聚类中心分布

Fig. 3　Ground grouping cluster center distribution

2.1.2　K – means 聚类

根据任务点坐标的样本集 $\boldsymbol{D}=\{\boldsymbol{d}_1,\boldsymbol{d}_2,\cdots,\boldsymbol{d}_{N_T}\}$，以及初始定义的任务聚类组 $\boldsymbol{S}=\{\boldsymbol{S}_1,\boldsymbol{S}_2,\cdots,\boldsymbol{S}_K\}$，K – means 的目的为最小化每个任务距离所在聚类分组的平方误差：

$$\min E=\sum_{s=1}^{K}\sum_{\boldsymbol{d}\in\boldsymbol{S}_s}\|\boldsymbol{d}-\boldsymbol{\mu}_s\|_2^2 \tag{16}$$

式中：$\boldsymbol{\mu}_s=\frac{1}{|\boldsymbol{S}_s|}\sum_{\boldsymbol{d}\in\boldsymbol{S}_s}\boldsymbol{d}$ 为分组 $\boldsymbol{S}_s$ 的均值向量，即地面分组聚类中心的坐标向量。

算法流程如图 4 所示，分别计算每个任务点距离聚类中心的距离，将任务点加入到距离最近的分组。循环迭代访问直到分组聚类中心不再更新为止。

2.2　任务点分组重规划

K – means 只是对任务分组的初步规划，仅依靠任务点的距离关系进行分组，没有考虑到分配给各组的异构机器人的数量是否满足每个组的任务需求。本文提出了任务分组重规划机制，最小化每个组的机器人数量需求总和，使每个组的任务需求能够满足提供的机器人数量，增大任务完成度。

2.2.1　分组任务类型

在任务点初步分组之后，计算每个组中三种任务的数量，表示为分组需求矩阵 $\boldsymbol{\Omega}$：

$$\boldsymbol{\Omega} = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_K \\ \beta_1 & \beta_2 & \cdots & \beta_K \\ \gamma_1 & \gamma_2 & \cdots & \gamma_K \end{bmatrix} \tag{17}$$

式中：α，β，γ 分别为探索、打击、毁伤评估任务的数量。

算法 1　任务点初始分组算法

输入：任务坐标点样本集 D、聚类分组数 K

输出：任务点分组划分 S

过程：

(1) 按式(11) ~ (15) 计算每个组的初始聚类中心 $\boldsymbol{\mu}$

(2) while

(3) 初始化 $S_s = \varnothing (1 \leqslant s \leqslant K)$

(4) for $j = 1, 2, \cdots, N_{\mathrm{T}}$ do

(5) 计算任务 j 与各分组中心 $\boldsymbol{\mu}_s$ 的距离：$d_{js} = \| \boldsymbol{d}_j - \boldsymbol{\mu}_s \|_2$

(6) 根据距离确定任务 j 的分组 $\xi_j = \mathrm{argmin}_{s \in \{1, 2, \cdots, K\}} d_{js}$

(7) 将任务 j 归入相应的分组：$S_{\xi_j} = S_{\xi_j} \cup \{j\}$；

(8) end for

(9) for $s = 1, 2, \cdots, K$ do

(10) 计算新的分组聚类中心 $\boldsymbol{\mu}_s' = \frac{1}{|S_s|} \sum_{d \in S_s} \boldsymbol{d}$；

(11) if $\boldsymbol{\mu}_s' \neq \boldsymbol{\mu}_s$ then

(12) $\boldsymbol{\mu}_i \leftarrow \boldsymbol{\mu}_i'$

(13) else

(14) 保持当前分组聚类中心不变

(15) end if

(16) end for

(17) until 当前分组聚类中心不再更新

(18) end

图 4　任务点分组算法

Fig. 4　Task point grouping algorithm

2.2.2　机器人供应数量问题模型

假设供应给每个组的旋翼无人机与小型侦察车数量之和为 n_1^s，固定翼无人机与大型武装车数量之和为 n_2^s，多余的任意机器人数量为 n_3^s，则机器人数量分布 $\boldsymbol{\Phi}$ 为

$$\boldsymbol{\Phi} = \begin{bmatrix} n_1^1 & n_1^2 & \cdots & n_1^K \\ n_2^1 & n_2^2 & \cdots & n_2^K \\ n_3^1 & n_3^2 & \cdots & n_3^K \end{bmatrix} \tag{18}$$

因为所有机器人都能执行毁伤评估任务，在机器人执行完探索和打击任务后，仍有空余的载荷去执行毁伤评估任务。当毁伤评估任务较多时，可能仍需要额外多分配几个机器人，n_3^s 表示额外的任意类型机器人数量。

目的是最小化机器人供应数量，目标函数为

$$\mathrm{min}g = \sum_{s=1}^{K} \sum_{h=1}^{3} n_h^s \tag{19}$$

其约束为

$$L_{\mathrm{T}} \cdot n_1^s \geqslant \alpha_s, \ s \in \{1, 2, \cdots, K\} \tag{20}$$

$$L_{\mathrm{T}} \cdot n_2^s \geqslant \beta_s, \ s \in \{1, 2, \cdots, K\} \tag{21}$$

$$L_{\mathrm{T}} \cdot n_3^s - \left[L_{\mathrm{T}} \left(n_1^s + n_2^s \right) - \alpha_s - \beta_s \right] \geqslant \gamma_s \tag{22}$$

$$g \leqslant N_{\mathrm{R}} \tag{23}$$

$$\sum_{s=1}^{K} n_1^s \leqslant n_{\mathrm{e}} \tag{24}$$

$$\sum_{s=1}^{K} n_2^s \leqslant n_{\mathrm{a}} \tag{25}$$

式中：L_T 为每个机器人可执行的最大任务数量；n_{e} 为旋翼无人机与小型侦察车数量之和；n_{a} 为固定翼无人机与大型武装车数量之和。

当机器人数量需求不满足式(23) ~ (25) 时，则需求数量超量，此时需要将某些分组的任务点调整到其他分组，降低机器人数量需求。建立每个分组可接受任务的余量矩阵 $\boldsymbol{\Psi}$：

$$\boldsymbol{\Psi} = \begin{bmatrix} m_1^s = L_{\mathrm{T}} \cdot n_1^s - \alpha_s \\ m_2^s = L_{\mathrm{T}} \cdot n_2^s - \beta_s \\ m_3^s = L_{\mathrm{T}} \cdot n_3^s - \left[L_T \left(n_1^s + n_2^s \right) - \alpha_s - \beta_s \right] - \gamma_s \end{bmatrix} \tag{26}$$

式中：m_1^s，m_2^s，m_3^s 分别为探索、打击、毁伤评估任务的任务余量空间。

任务点分组重规划算法如图 5 所示，先判断机器人数量是否超量，然后确定超量机器人的类型，选择任务余量空间最大的分组，将多余的任务分配到其他有余量的分组。

算法 2　任务点分组重规划

输入：分组需求矩阵 $\boldsymbol{\Omega}$、机器人数量分布 $\boldsymbol{\Phi}$、余量矩阵 $\boldsymbol{\Psi}$、任务聚类分组 $\boldsymbol{S}$

输出：分组需求矩阵 $\boldsymbol{\Omega}^*$、机器人数量分布 $\boldsymbol{\Phi}^*$、余量矩阵 $\boldsymbol{\Psi}^*$、任务聚类分组 $\boldsymbol{S}^*$

过程：

(1) if 式(23) ~ (25) 不成立 then

(2) 寻找需要调整的任务点分组 $s^* = \mathrm{argmax}_{s \in \{1, 2, \cdots, K\}} m^s$

(3) 计算需要调整的任务点数量 $n_{\mathrm{ad}} = L_{\mathrm{T}} - m^{s^*}$

(4) while

(5) for $j = 1 \rightarrow S_s$ 组对应任务类型数量 do

(6) 选择距离最近的转移分组 $\xi_j = \mathrm{argmin}_{s \in \{1, 2, \cdots, K\} \setminus \{s^*\}} d_{js}$

(7) if $d_{js} < d_{\max}$ then

(8) $S_{\xi_i} = S_{\xi_i} \cup \{j\}$，$S_{s^*} = S_{s^*} \setminus \{j\}$

(9) $n_{\mathrm{ad}} = n_{\mathrm{ad}} - 1$

(10) 更新 $\boldsymbol{\Omega}$，$\boldsymbol{\Phi}$，$\boldsymbol{\Psi}$

(11) end if

(12) end for

(13) until $n_{\mathrm{ad}} = 0$ 且 $\boldsymbol{\Omega}$，$\boldsymbol{\Phi}$，$\boldsymbol{\Psi}$，$\boldsymbol{S}$ 没有再更新

(14) end if

(15) end

图 5　任务点分组重规划

Fig. 5　Task point grouping replanning

循环迭代地执行算法 2，直到式(23) ~ (25) 满足约束条件为止，则可以保证每个组的任务理论上可以全部完成。

3　机器人分组调配

假设任务开始时，所有机器人已经分布在地图上，按

照距离最近的原则，将机器人分配到相应的任务分组上。

3.1 机器人分组问题模型

目的是最小化机器人到对应的分组的距离之和，为了简化计算，以机器人和各分组聚类中心的距离作为机器人与各分组的距离，其目标函数为

$$\min d = \sum_{s=1}^{K}\sum_{i=1}^{N_R} d_{is} \cdot u_{is} \tag{27}$$

其约束为

$$\sum_{s=1}^{K} u_{is} \leqslant 1,\ \forall i \in \{1,2,\cdots,N_R\} \tag{28}$$

$$\sum_{i=1}^{N_R} u_{is} \leqslant n^s,\ \forall s \in \{1,2,\cdots,K\} \tag{29}$$

式中：d_{is} 为机器人 i 距离分组 S_s 的距离；$u_{is} \in \{0,1\}$ 为机器人分配组别的决策变量；n^s 为分配给每个组的机器人总数。

3.2 延迟接受(DA)算法

将机器人分配到各任务分组的问题建模为一对多类型的稳定匹配问题，匹配度由机器人到聚类分组中心的距离决定。

初始定义机器人分组表示为

$$\boldsymbol{A} = \{\boldsymbol{A}_1, \boldsymbol{A}_2, \cdots, \boldsymbol{A}_K\} \tag{30}$$

图 6 为机器人分组算法，即延迟接受(DA)。首先每个机器人计算与聚类分组中心的距离 d_{is}，然后每个分组都系统性地从上到下遍历机器人的最近距离分组，选择距离自己最近的若干机器人，若机器人和另外的分组更匹配，则将该机器人让给其他分组。经过多轮迭代后得到一个稳定的匹配结果。

算法 3 机器人分组算法

输入： 机器人集合 $\boldsymbol{R}$、任务聚类分组 $\boldsymbol{S}$、机器人数量分布 $\boldsymbol{\Phi}$

输出： 机器人聚类分组 $\boldsymbol{A}$

过程：

(1) while
(2) for $i = 1 \rightarrow N_R$ do
(3) $\zeta_i = \mathrm{argmin}_{s\in\{1,2,\cdots,K\}} d_{is}$
(4) 将机器人 i 与分组 ζ_i 匹配
(5) if 分组机器人需求数没满 then
(6) $\boldsymbol{A}_{\zeta_i} = \boldsymbol{A}_{\zeta_i} \cup \{i\}$
(7) else if 分组机器人需求数已满 then
(8) 分组 $\boldsymbol{A}_{\zeta_i}$ 删除组内距离最远的机器人 k
(9) $\boldsymbol{A}_{\zeta_i} = \boldsymbol{A}_{\zeta_i} \backslash \{k\}$，$\boldsymbol{A}_{\zeta_i} = \boldsymbol{A}_{\zeta_i} \cup \{i\}$
(10) end if
(11) end for
(12) until 所有机器人都已分配到各聚类分组

图 6 机器人分组算法

Fig. 6 Robot grouping algorithm

4 分组任务分配

利用 CBBA 算法对每个机器人 – 任务分组进行任务分配规划，得到最终的任务规划解。

4.1 基线 CBBA

CBBA 的主要工作是，首先利用贪婪选择算法构建任务序列包 $\boldsymbol{P}_i$，然后使用共识规则建立任务冲突协商机制来解决机器人任务包之间的冲突。

4.1.1 任务包的建立

任务序列包的建立是在每个机器人内部并行运行的，然后通过投标信息解决冲突。投标信息主要包括：

(1) 任务包 $\boldsymbol{b}_i \in (\boldsymbol{T} \cup \{\varnothing\})^{L_T}$，储存了机器人选择的任务。

(2) 任务序列包 $\boldsymbol{P}_i \in (\boldsymbol{T} \cup \{\varnothing\})^{L_T}$，代表按执行顺序排列的任务集合。

(3) 任务的中标报价列表 $\boldsymbol{y}_i \in (R_+)^{N_T}$，其中 $y_{ij} \in \boldsymbol{y}_i (j = 1,2,\cdots,N_T)$ 为执行任务 j 的最高报价，表示每个机器人内部执行该任务 j 的奖励。在共识协商阶段，每个机器人之间对任务 j 进行报价竞争，更新当前最高的任务奖励(最高报价)。

(4) 中标机器人列表 $\boldsymbol{z}_i \in (\boldsymbol{R} \cup \{\varnothing\})^{N_T}$ 为对任务 j 报价最高的机器人 i，表示在共识协商阶段，机器人 i 获取任务 j 的执行权，将其序号 i 储存在中标机器人列表中。

任务序列包构建阶段，机器人 i 利用贪婪算法，每次将当前奖励值最大的任务添加到包中，直到不能再继续添加任务。每添加一个任务，除了将任务序号 j 记录在任务包和任务序列包中，还会在报价列表中记录其任务奖励，作为对任务 j 的竞标报价 c_{ij}，并在中标机器人列表中记录自己的机器人序号 i。

4.1.2 共识协商机制

在冲突协商阶段，机器人利用共识协商策略调节任务冲突，即出价最高的机器人获得任务 j 的执行权，其他落选机器人将包中的任务 j 删除，之后所有机器人更新中标报价列表 $\boldsymbol{y}_i$ 和中标机器人列表 $\boldsymbol{z}_i$，记录当前最高的报价(任务奖励)和中标机器人序号。共识规则如表 1 所示。列表 $\boldsymbol{y}_i$ 和 $\boldsymbol{z}_i$ 的更新操作可以表示为

$$\begin{cases} \text{update 更新：} & y_{ij} = y_{kj},\ z_{ij} = z_{kj} \\ \text{reset 重置：} & y_{ij} = 0,\ z_{ij} = \varnothing \\ \text{leave 保持：} & y_{ij} = y_{ij},\ z_{ij} = z_{ij} \end{cases} \tag{31}$$

式中：update 为中标报价和中标机器人两个列表进行更新，保存当前最高报价和提供报价的机器人序号。Reset 为当前任务进行重置，删除列表相关任务 j 的数据。Leave 为不做任何修改，保持列表原有报价不变。

显然，将大规模机器人和任务进行分组以后，每个机器人内部可选择的任务减少了，同时省略了不同分组机器人之间的多余通信，大大降低了算法中机器人的通信总量。

4.2 任务奖励函数

基线 CBBA 的任务奖励函数是严格基于边际增益函数(DMG)计算的，当任务执行时间超过任务开启时间，奖励函数将趋于 0，导致任务无法选择。在计算任务点奖励时设置固定奖励，使得奖励永远大于成本代价[17]，保证每个任务尽量被选择。目标奖励函数表示为

$$r_i(\boldsymbol{P}_i) = \sum_{j\in P_i} c_{ij} = \sum_{j\in P_i} a_{ij} - p_j \triangleq \sum_{j\in P_i}\left(r_0 + (r_j - r_0)\cdot e^{-\lambda\cdot(t_{ij}-t_j^{start})} - \gamma\cdot d_{ij}\right) \tag{32}$$

$$r_0 \geqslant \gamma\cdot d_{ij} \tag{33}$$

$$d_{ij} = \begin{cases} \sqrt{(X_R - X_j)^2 + (Y_R - Y_j)^2 + (Z_R - Z_j)^2}, & \text{空中机器人} \\ (|X_R - X_j| + |Y_R - Y_j|), & \text{地面机器人} \end{cases} \tag{34}$$

式中：a_{ij} 和 p_j 分别为机器人执行任务的奖励和代价；d_{ij} 为机器人 i 到任务点 j 的距离公式，根据机器人的异构性，空中机器人采用三维欧氏距离，地面机器人采用二维曼哈顿距离；γ 为距离代价常系数；r_0 为任务固定奖励；r_j 为及时完成任务 j 的奖励。本文假设奖励函数是非负的，所以固定奖励不小于代价。

表 1　机器人 i 与机器人 k 的共识协商规则

Table 1　Consensus negotiation rules of robot i and robot k

机器人 k（发送者）的 z_{kj} 值	机器人 i（接收者）的 z_{ij} 值	接收者处理（默认：leave）
k	i	if $y_{kj} > y_{ij}$ → update
	k	update
	$m \notin \{i, k\}$	if $s_{km} > s_{im}$ or $y_{kj} > y_{ij}$ → update
	none	update
i	i	leave
	k	reset
	$m \notin \{i, k\}$	if $s_{km} > s_{im}$ → reset
	none	leave
$m \notin \{i, k\}$	i	if $s_{km} > s_{im}$ and $y_{kj} > y_{ij}$ → update
	k	if $s_{km} > s_{im}$ → update else → reset
	m	if $s_{km} > s_{im}$ → reset if $s_{km} > s_{im}$ and $s_{kn} > s_{in}$ → update
	$n \notin \{i, k, m\}$	if $s_{km} > s_{im}$ and $y_{kj} > y_{ij}$ → update if $s_{kn} > s_{in}$ and $s_{im} > s_{km}$ → reset
	none	if $s_{km} > s_{im}$ → update
none	i	leave
	k	update
	$n \notin \{i, k\}$	if $s_{km} > s_{im}$ → update
	none	leave

4.3　任务选择优化

基线 CBBA 中，机器人在每次迭代过程中选择任务时，总是要从上到下依次遍历所有任务，这样会造成已经竞价失败的任务重新添加到包中，且容易忽略到后续被所有机器人都忽略的任务，使任务完成度较低。

任务序列包构建算法如图 7 所示。在任务选择阶段，改进基线 CBBA 的贪婪算法，构建一个全局任务未选包 $\boldsymbol{b}_i^{no}$，优先对所有机器人都没选择的任务进行访问，然后再从上到下依次访问所有任务，选择报价更高的任务进行任务协商阶段竞价。

算法 4　任务序列包构建算法

输入： 迭代次数 $t-1$ 过程中的 $\boldsymbol{b}_i(t-1)$，$\boldsymbol{P}_i(t-1)$，$\boldsymbol{y}_i(t-1)$，$z_i(t-1)$，$\boldsymbol{t}_i(t-1)$

输出： 迭代次数 t 过程中的 $\boldsymbol{b}_i(t)$，$\boldsymbol{P}_i(t)$，$\boldsymbol{y}_i(t)$，$z_i(t)$，$\boldsymbol{t}_i(t)$

过程：

(1) $\boldsymbol{b}_i(t) = \boldsymbol{b}_i(t-1)$，$\boldsymbol{P}_i(t) = \boldsymbol{P}_i(t-1)$，$\boldsymbol{y}_i(t) = \boldsymbol{y}_i(t-1)$，$z_i(t) = z_i(t-1)$，$\boldsymbol{t}_i(t) = \boldsymbol{t}_i(t-1)$

(2) 构建任务未选包 $\boldsymbol{b}_i^{no}$：$\boldsymbol{b}_i^{no} = \boldsymbol{T} \cap (\boldsymbol{P}_1 \cup \boldsymbol{P}_2 \cup \cdots \cup \boldsymbol{P}_n)$

(3) while $|\boldsymbol{b}_i(t)| \leqslant L_T$ do

(4) $c_{ij}(\boldsymbol{P}_i) = \max_{n\leqslant|\boldsymbol{P}_i|} \boldsymbol{V}_i^{\boldsymbol{P}_i\oplus_n\{j\}} - \boldsymbol{V}_i^{\boldsymbol{P}_i}$

(5) $h_{ij} = \Pi(c_{ij} > y_{ij})$

(6) $J_i = \text{argmax}_j(c_{ij}(\boldsymbol{P}_i) \times h_{ij})$

(7) $n_{i,J_i} = \text{argmax}_j \boldsymbol{V}_i^{\boldsymbol{P}_i\oplus_j\{J_i\}}$

(8) $t_{i,n_{i,J_i}} = \max\left[t_{i,n_{i,J_i-1}} + (\boldsymbol{V}_i^{n_{i,J_i}} - \boldsymbol{V}_i^{n_{i,J_i-1}})/v_i, t_{0,n_{i,J_i}}\right]$

(9) $\boldsymbol{b}_i = \boldsymbol{b}_i \oplus_{end}\{J_i\}$，$\boldsymbol{P}_i = \boldsymbol{P}_i \oplus_{n_{i,J_i}}\{J_i\}$

(10) $\boldsymbol{y}_{i,J_i}(t) = c_{i,J_i}$，$z_{i,J_i}(t) = i$，$\boldsymbol{t}_{i,J_i}(t) = t_{i,n_{i,J_i}}$

(11) end while

图 7　任务序列包构建算法

Fig. 7　Task sequence package construction algorithm

通过改进的任务序列包构建算法，在每次任务访问阶段，都优先搜索还没有被选择的任务，提高任务完成度的同时，也减少了无用任务访问的频次，提高算法效率。通过任务序列包构建和共识协商两个阶段的循环迭代，最终收敛到任务完成度最大的无冲突分配方案。

5　算法仿真实验

为了测试 C-CBBA 的性能，在 Intel(R) Core(TM) i7-10700K CPU @ 3.80 GHz and 16 GB RAM 的 PC 上，用 MATLAB2019a 进行仿真。并和基线 CBBA、具有任务耦合约束的 CBBA(TCC) 进行对比。

5.1　场景参数设置

在 5 km×5 km×1 km 的三维场景下，异构机器人团队在总体时间窗 [0, 5 000 s] 内完成任务的调度。机器人最大任务数量 $L_T = 15$，任务固定奖励 $r_0 = 30$，任务完成奖励 $r_j = 100$，任务价值随时间衰减因子 $\lambda = 0.1$。实验中，模拟了五种机器人-任务点规模，分别为 $10R-100T$，$30R-300T$，$50R-500T$，$80R-800T$，$100R-1\,000T$，任务和机器人分组组数 $K = 5$。每种规模分别进行了 100 次蒙特卡洛随机模拟，每次模拟时算法的迭代次数最多为 100 次，当任务规划迭代次数超过 100，算法将强制停止，输出当前的最优解。此外，为了显示预设分组组数对任务分配结果的影响，对 $30R-300T$ 规模的分配问题，分别模拟了分组组数 K 为 3，4，5，6 四种情况下的实验。

5.2　算法结果展示

为了清晰地展示机器人的任务路线图，选择展示在 $10R-100T$ 规模下 $K=5$ 时的某一次实验结果。实验中最小的规模为 $10R-100T$，其任务执行路线图比较清楚，且在 $K=5$ 的情况下，任务分组的结果比较稀疏，有利于最终结果的观察。相关信息如表 2～3 所示。

表 2　机器人相关信息

Table 2　Robot related information

机器人编号	类型	速度 v/(m/s)	坐标
R_1，R_2，R_3	旋翼无人机	20	随机
R_4，R_5	小型侦察车	15	随机
R_6，R_7，R_8	固定翼无人机	40	随机
R_9，R_{10}	大型武装车	30	随机

表 3　任务目标相关信息

Table 3　Information related to task objectives

任务编号	类型	消耗时间/s	时间窗	坐标
$T_1 \rightarrow T_{33}$	探索	100	随机	随机
$T_{34} \rightarrow T_{66}$	打击	150	随机	随机
$T_{67} \rightarrow T_{100}$	毁伤评估	100	随机	随机

图 8 为任务分配路线图，为了便于观察，忽略了高度，只显示二维平面的路线轨迹。表 4 为机器人分配方案，即机器人执行任务顺序。

其中，算法收敛时间 0.311 s，任务完成度 100%，通信量(通信频次) 150 次，全局任务奖励 4 083.396。

5.3　不同规模的算法性能对比

不同规模下机器人及任务的类型数量分布采用相同的比例构成，任务分组组数都为 $K=5$。表 5 为机器人和任务的类型数量分布情况与编号对应的类型，四种机器人的类型比例分布为 0.3∶0.2∶0.3∶0.2，三种任务的类型数量比例分布为 0.33∶0.33∶0.34。

通过算法在五种不同规模下的仿真，总体性能由算法收敛时间、通信量(通信频次)、任务完成度、全局任务奖励、任务平均花费时间五个方面体现。

其中算法收敛时间的计算从系统开始分组到最终输出任务分配结果结束，运行时间依赖于实验 PC 条件。多机器人之间的通信量依赖于系统通信模型、机器人通信方式、机器人拓扑连接方式、通信损耗等多方面的因素，研究简化了其通信量的计算，用机器人之间的通信频次(冲突解调阶段机器人之间的通信次数)作为通信量的计算依据，两个机器人之间的一次冲突解调为 1 个通信频次。每一次任务包构建和冲突解调的迭代为一轮通讯，一轮通讯中的通信频次依赖于拓扑连接形式，本文按照全连接的形式进行通信，故一轮通讯的通信频次为 $N_s(N_s-1)/2$，N_s 为每个分组的机器人数量。

表 6～10 分别表示不同规模下，五个性能指标的仿真实验结果对比。可见在算法收敛时间、通信量、任务完成度方面，C-CBBA 的性能是最优的，可以解决大规模任务分配问题的最大化算法效率和最小化通信量问题，并能保证大部分任务被机器人选中并执行。

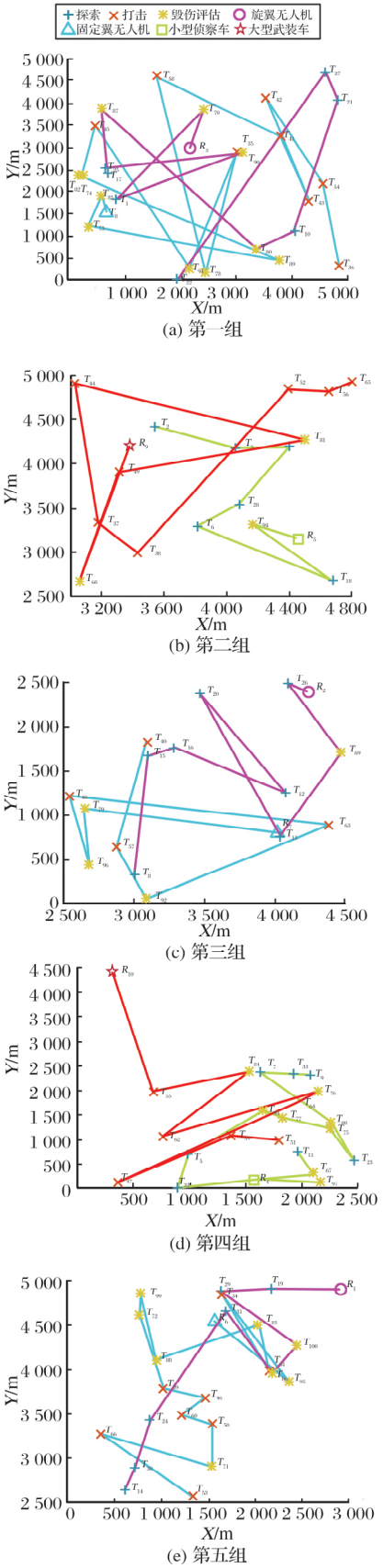

图 8　各聚类分组的机器人任务路线图

Fig. 8　Robot task roadmap of each cluster grouping

(1) 算法收敛时间和通信量方面，预先对任务和机器人进行了分组，每个分组的规模都变小了，任务分配算法需要计算访问的对象缩减，提高了收敛的效率，同时也减少了机器人之间通信的总量，每个机器人只需要和自己组内的机器人进行通信。

(2) 任务完成度方面，在每次任务规划阶段，重点考虑了全局未被选择的任务，所以C－CBBA可以规划出接近100%的任务完成度，而基线CBBA和TCC需要频

表4 不同规模下机器人及任务的类型数量分布

Table 4 Number of type distribution of robots and tasks at different scales

规模	机器人编号(数量)及类型	任务编号及类型
$10R-100T$	$R_1\rightarrow R_3$(3)旋翼UAV	$T_1\rightarrow T_{33}$(33)探索
	$R_4\rightarrow R_5$(2)小型侦察车	$T_{34}\rightarrow T_{66}$(33)打击
	$R_6\rightarrow R_8$(3)固定翼UAV	$T_{67}\rightarrow T_{100}$(34)毁伤
	$R_9\rightarrow R_{10}$(2)大型武装车	评估
$30R-300T$	$R_1\rightarrow R_9$(9)旋翼UAV	$T_1\rightarrow T_{99}$(99)探索
	$R_{10}\rightarrow R_{15}$(6)小型侦察车	$T_{100}\rightarrow T_{198}$(99)打击
	$R_{16}\rightarrow R_{24}$(9)固定翼UAV	$T_{199}\rightarrow T_{300}$(102)毁
	$R_{25}\rightarrow R_{30}$(6)大型武装车	伤评估
$50R-500T$	$R_1\rightarrow R_{15}$(15)旋翼UAV	$T_1\rightarrow T_{165}$(165)探索
	$R_{16}\rightarrow R_{25}$(10)小型侦察车	$T_{166}\rightarrow T_{330}$(165)打击
	$R_{26}\rightarrow R_{40}$(15)固定翼UAV	$T_{331}\rightarrow T_{500}$(170)毁
	$R_{41}\rightarrow R_{50}$(10)大型武装车	伤评估
$80R-800T$	$R_1\rightarrow R_{24}$(24)旋翼UAV	$T_1\rightarrow T_{264}$(264)探索
	$R_{25}\rightarrow R_{40}$(16)小型侦察车	$T_{265}\rightarrow T_{528}$(264)打击
	$R_{41}\rightarrow R_{64}$(24)固定翼UAV	$T_{529}\rightarrow T_{800}$(272)毁
	$R_{65}\rightarrow R_{80}$(16)大型武装车	伤评估
$100R-1\,000T$	$R_1\rightarrow R_{30}$(30)旋翼UAV	$T_1\rightarrow T_{330}$(330)探索
	$R_{31}\rightarrow R_{50}$(20)小型侦察车	$T_{331}\rightarrow T_{660}$(330)打击
	$R_{51}\rightarrow R_{80}$(30)固定翼UAV	$T_{661}\rightarrow T_{1\,000}$(340)毁
	$R_{81}\rightarrow R_{100}$(20)大型武装车	伤评估

表5 机器人任务分配方案

Table 5 Task assignment scheme of robot

机器人	任务序列
R_1	$T_{19}\rightarrow T_{29}\rightarrow T_{100}\rightarrow T_{86}\rightarrow T_{31}\rightarrow T_{24}\rightarrow T_{30}\rightarrow T_{14}$
R_2	$T_{26}\rightarrow T_{69}\rightarrow T_{13}\rightarrow T_{20}\rightarrow T_{12}\rightarrow T_{16}\rightarrow T_{15}\rightarrow T_8$
R_3	$T_{70}\rightarrow T_1\rightarrow T_{90}\rightarrow T_{25}\rightarrow T_{17}\rightarrow T_{87}\rightarrow T_{80}\rightarrow T_{10}\rightarrow T_{21}\rightarrow T_{27}\rightarrow T_{22}$
R_4	$T_{97}\rightarrow T_{11}\rightarrow T_{67}\rightarrow T_{32}\rightarrow T_5\rightarrow T_{93}\rightarrow T_{77}\rightarrow T_{75}\rightarrow T_{23}\rightarrow T_{98}\rightarrow T_7\rightarrow T_{33}\rightarrow T_9$
R_5	$T_{97}\rightarrow T_{18}\rightarrow T_6\rightarrow T_{28}\rightarrow T_4\rightarrow T_3\rightarrow T_2$
R_6	$T_{95}\rightarrow T_{34}\rightarrow T_{61}\rightarrow T_{83}\rightarrow T_{88}\rightarrow T_{99}\rightarrow T_{72}\rightarrow T_{39}\rightarrow T_{46}\rightarrow T_{60}\rightarrow T_{50}\rightarrow T_{71}\rightarrow T_{66}\rightarrow T_{53}$
R_7	$T_{79}\rightarrow T_{96}\rightarrow T_{48}\rightarrow T_{63}\rightarrow T_{92}\rightarrow T_{57}\rightarrow T_{40}$
R_8	$T_{85}\rightarrow T_{73}\rightarrow T_{89}\rightarrow T_{82}\rightarrow T_{74}\rightarrow T_{45}\rightarrow T_{91}\rightarrow T_{35}\rightarrow T_{78}\rightarrow T_{58}\rightarrow T_{41}\rightarrow T_{43}\rightarrow T_{42}\rightarrow T_{54}\rightarrow T_{36}$
R_9	$T_{68}\rightarrow T_{49}\rightarrow T_{81}\rightarrow T_{44}\rightarrow T_{37}\rightarrow T_{38}\rightarrow T_{52}\rightarrow T_{56}\rightarrow T_{65}$
R_{10}	$T_{55}\rightarrow T_{84}\rightarrow T_{62}\rightarrow T_{76}\rightarrow T_{64}\rightarrow T_{47}\rightarrow T_{59}\rightarrow T_{51}$

表6 三种算法的收敛时间

Table 6 Convergence time of three algorithms

规模	算法收敛时间/s		
	C－CBBA	CBBA	TCC
$10R-100T$	0.174	0.289	0.761
$30R-300T$	3.981	6.704	17.010
$50R-500T$	17.012	30.328	73.920
$80R-800T$	66.115	125.701	294.999
$100R-1\,000T$	131.965	237.270	523.723

表7 三种算法的通信量(通信频次)

Table 7 Traffic of three algorithms (communication frequency)

规模	通信量		
	C－CBBA	CBBA	TCC
$10R-100T$	220.000	1 211.400	1 389.600
$30R-300T$	6 448.180	29 849.700	32 712.000
$50R-500T$	27 848.960	134 064.000	143 717.000
$80R-800T$	105 985.500	538 021.600	566 145.600
$100R-1\,000T$	203 181.140	999 900.000	999 900.000

表8 三种算法的任务完成度

Table 8 Task completion rate of three algorithms

规模	任务完成度/%		
	C－CBBA	CBBA	TCC
$10R-100T$	98.99	29.12	84.55
$30R-300T$	99.99	32.98	91.30
$50R-500T$	100.00	34.61	93.21
$80R-800T$	100.00	35.89	94.66
$100R-1\,000T$	100.00	36.46	95.18

表9 三种算法的全局任务奖励

Table 9 Global task reward of three algorithms

规模	全局奖励		
	C－CBBA	CBBA	TCC
$10R-100T$	3 696.224	2 683.141	4 005.383
$30R-300T$	12 997.682	9 140.278	13 664.166
$50R-500T$	22 853.105	16 016.801	23 832.670
$80R-800T$	37 952.158	26 745.065	39 521.090
$100R-1\,000T$	48 081.749	33 983.197	50 094.391

表10 三种算法的任务平均用时

Table 10 Average task time of three algorithms

规模	任务平均用时/s		
	C－CBBA	CBBA	TCC
$10R-100T$	197.442	219.754	185.696
$30R-300T$	177.248	197.275	169.440
$50R-500T$	169.598	189.255	163.459
$80R-800T$	163.964	183.764	158.878
$100R-1\,000T$	161.837	181.124	156.833

繁地删除任务重新规划。三种算法的任务奖励都是按照边际效益递减的规律进行计算的，为了贴合实际任务执行时，开始时间越晚，收益越低的情况，引入了边际效益递减因子 λ。因此，算法对时间的增加比较敏感，CBBA 算法的任务奖励会随着时间的增加快速降低，没有任务固定奖励，收益将小于 0，导致任务无法选择。而 TCC 虽然也引入了固定奖励，但是算法执行时依然要频繁地删除任务，也会导致任务完成度不高。C - CBBA 算法在引入固定奖励的基础上，还加入了未选任务优先访问机制，尽量将全局未选择的任务加入任务序列包中。

(3) 全局任务奖励和平均任务花费时间方面，由于聚类分组本身会缩小分配解集，可能会将最优的解排除，所以 C - CBBA 的总奖励并不是最高的，说明分配策略的奖励不是最高的。其原因为 C - CBBA 的任务完成度高，机器人需要执行的任务多，任务平均花费时间相应增加，在时间窗约束下，部分任务的执行时间无法在任务开启时立刻执行，导致任务奖励不能达到理论最大值。

5.4 不同分组规模的仿真

不同的分组组数 K 对算法也有影响，对 $30R-300T$ 规模的机器人 - 任务组进行仿真，分别将其划分为 3，4，5，6 组。

不同组数的算法性能如表 11 所示。在不同分组组数下，随着组数的增加，算法收敛时间和通信量明显降低，任务完成度和全局任务奖励变小。这是因为分组越细，默认忽略的分配解集越多，增加算法收敛速度，降低了解的质量。所以在不同场合，应该根据不同的需求进行取舍。

表 11 不同组数的算法性能

Table 11 Algorithm performance of different groups

性能类型	$K=3$	$K=4$	$K=5$	$K=6$
算法时间/s	8.717	5.445	4.017	3.240
通信量	8 175.160	6 928.140	6 535.100	6 418.720
任务完成度/%	100.00	100.00	99.99	99.85
全局任务奖励	13 110.514	13 027.407	13 001.639	12 866.740
任务平均时间	177.254	177.070	176.713	177.116

结合前两种仿真情况，C - CBBA 在收敛时间、通信量、任务完成度方面具有优势，算法的设定条件是任务的位置和类型预先已知，机器人提前布置在任务环境中，且机器人和任务的类型数量是提前设定好的，这意味着在任务分配前的机器人分组调配必须是有解的，否则将造成算法失效。本文适用于具有时间窗约束的大型任务场景下的大规模任务分配问题中，最大化任务完成度及最小化的收敛时间的情况。表 6 显示，从规模 $80R-800T$ 开始，算法收敛时间超过了 60 s，针对超大规模的任务分配问题，C - CBBA 算法只能适用于静态任务，不再适用于动态任务。针对规模小于 $50R-100T$ 的问题，C - CBBA 算法具有较好的适应能力。

6 结 论

本文针对具有时间窗约束的大规模 MRTA 问题，提出了基于聚类分组的一致性的束算法(C - CBBA)，解决算法效率和降低通信量的问题。首先利用 K - means 算法对任务点按照距离分组，并用任务点重规划机制最小化所需机器人资源，然后用 DA 算法将机器人分配到各聚类分组，最后改进 CBBA 算法，求解划分好的若干小规模问题。仿真实验结果表明，该算法能够在较短的时间内，用较少的通信量达到可接受的任务完成度和全局任务奖励。但是在性能检验方面，通信量受系统通信模型、机器人拓扑连接方式等多种因素影响，本文对通信量的计算比较理想化，未来研究还应考虑实际情况，对通信进一步细化研究。而且聚类分组可能会忽略全局最优解，在更大范围内，机器人并不能获得所有其他机器人的信息。未来将考虑机器人之间的具体通信模型，并改进任务和机器人分组机制，在最小化资源需求的基础上，最大化全局任务奖励。

参考文献:

[1] Tang J, Zhu K J, Guo H X, et al. Using Auction-Based Task Allocation Scheme for Simulation Optimization of Search and Rescue in Disaster Relief [J]. Simulation Modelling Practice and Theory, 2018, 82: 132 - 146.

[2] Sun C H, Wang X C, Qiu H X, et al. Game Theoretic Self-Organization in Multi-Satellite Distributed Task Allocation [J]. Aerospace Science and Technology, 2021, 112: 106650.

[3] Chen H X, Nan Y, Yang Y. Multi-UAV Reconnaissance Task Assignment for Heterogeneous Targets Based on Modified Symbiotic Organisms Search Algorithm [J]. Sensors, 2019, 19(3): 734.

[4] Huang X M, Yu R, Ye D D, et al. Efficient Workload Allocation and User-Centric Utility Maximization for Task Scheduling in Collaborative Vehicular Edge Computing [J]. IEEE Transactions on Vehicular Technology, 2021, 70(4): 3773 - 3787.

[5] Lin K, Li Y H, Zhang Q, et al. AI-Driven Collaborative Resource Allocation for Task Execution in 6G-Enabled Massive IoT [J]. IEEE Internet of Things Journal, 2021, 8(7): 5264 - 5273.

[6] Seenu N, Kuppan Chetty R M, Ramya M M, et al. Review on State-of-the-Art Dynamic Task Allocation Strategies for Multiple-Robot Systems [J]. Industrial Robot: the International Journal of Robotics Research and Application, 2020, 47(6): 929 - 942.

[7] Cao R Y, Li S C, Ji Y H, et al. Task Assignment of Multiple Agricultural Machinery Cooperation Based on Improved Ant Colony Algorithm [J]. Computers and Electronics in Agriculture, 2021, 182: 105993.

[8] Geng N, Chen Z T, Nguyen Q A, et al. Particle Swarm Optimization Algorithm for the Optimization of Rescue Task Allocation with Uncertain Time Constraints [J]. Complex & Intelligent Systems, 2021, 7(2): 873 - 890.

[9] Han S, Fan C C, Li X B, et al. A Modified Genetic Algorithm for Task Assignment of Heterogeneous Unmanned Aerial Vehicle System [J]. Measurement and Control, 2021, 54(5/6): 994 - 1014.

[10] Kim K S, Kim H Y, Choi H L. A Bid-Based Grouping Method for Communication-Efficient Decentralized Multi-UAV Task Allocation [J]. International Journal of Aeronautical and Space Sciences, 2020, 21(1): 290 - 302.

[11] Zhang Y Z，Feng W C，Shi G Q，et al. UAV Swarm Mission Planning in Dynamic Environment Using Consensus-Based Bundle Algorithm [J]. Sensors (Basel，Switzerland)，2020，20(8)：2307.

[12] Choi H L，Brunet L，How J P. Consensus-Based Decentralized Auctions for Robust Task Allocation [J]. IEEE Transactions on Robotics，2009，25(4)：912 - 926.

[13] Choi H L，Kim K S，Johnson L B，et al. Potential Game-Theoretic Analysis of a Market-Based Decentralized Task Allocation Algorithm [C] // Distributed Autonomous Robotic Systems，2016.

[14] Johnson L B，Ponda S S，Choi H L，et al. Asynchronous Decentralized Task Allocation for Dynamic Environments [C] // AIAA Infotech@ Aerospace Conference，2011.

[15] Buckman N，Choi H L，How J P. Partial Replanning for Decentralized Dynamic Task Allocation [C] // AIAA Scitech Forum，2019.

[16] Binetti G，Naso D，Turchiano B. Decentralized Task Allocation for Heterogeneous Agent Systems with Constraints on Agent Capacity and Critical Tasks [C] // IEEE International Conference on Robotics and Biomimetics，2012：1627 - 1632.

[17] Ye F，Chen J，Sun Q，et al. Decentralized Task Allocation for Heterogeneous Multi-UAV System with Task Coupling Constraints [J]. The Journal of Supercomputing，2021，77(1)：111 - 132.

[18] Zhao W Q，Meng Q G，Chung P W H. A Heuristic Distributed Task Allocation Method for Multivehicle Multitask Problems and Its Application to Search and Rescue Scenario [J]. IEEE Transactions on Cybernetics，2016，46(4)：902 - 915.

[19] Whitbrook A，Meng Q G，Chung P W H. Reliable，Distributed Scheduling and Rescheduling for Time-Critical，Multiagent Systems [J]. IEEE Transactions on Automation Science and Engineering，2017，15(2)：732 - 747.

[20] Fu X W，Feng P，Li B，et al. A Two-Layer Task Assignment Algorithm for UAV Swarm Based on Feature Weight Clustering [J]. International Journal of Aerospace Engineering，2019(5)：1 - 12.

[21] Jang I，Shin H S，Tsourdos A. Anonymous Hedonic Game for Task Allocation in a Large-Scale Multiple Agent System [J]. IEEE Transactions on Robotics，2018，34(6)：1534 - 1548.

[22] Gerkey B P，Matarić M J. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems [J]. The International Journal of Robotics Research，2004，23(9)：939 - 954.

[23] Pinheiro D N，Aloise D，Blanchard S J. Convex Fuzzy K-Medoids Clustering [J]. Fuzzy Sets and Systems，2020，389：66 - 92.

Research on Task Allocation of Heterogeneous Multi-Robot Based on Cluster Grouping Algorithm

Zheng Xiyu[1]，Xu Ziyu[1]，Wang Jinghua[1，2*]

(1. College of Mechanical and Electric Engineering，Changchun University of Science and Technology，Changchun 130022，China；2. Ministry of Education Key Laboratory for Cross-Scale Micro and Nano Manufacturing，Changchun University of Science and Technology，Changchun 130022，China)

Abstract: In the urban combat environments，multiple heterogeneous air-to-ground robots need to process large-scale task targets with time window constraints. In order to ensure fast and effective strategic execution，an acceptable suboptimal task allocation strategy needs to be quickly calculated. This paper proposes a cluster grouping consensus-based bundle algorithm (C-CBBA) for large-scale task assignment problems with time window constraints. Firstly，the task target points and robots are grouped，and the large-scale problems are tranformed into small-scale problems. The grouping algorithm includes three stages：the K-Means algorithm is used to preliminarily classify the task points according to the principle of nearest distance，the task point group of each group is adjusted so that it does not exceed the upper load limit of each pre-assigned robot group，and the delay acceptance algorithm (DA) is used to assign the nearest robot to each group. Finally，the improved consensus-based bundle algorithm (CBBA) is used to solve each sub-problem. Simulation results show that the proposed algorithm has high task completion and effectively reduces communication between robots.

Key words: cluster grouping；multi-robot system；time window；task allocation；CBBA algorithm；unmanned system；combat