Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

A-STC: auction-based spanning tree coverage algorithm for motion planning of cooperative robots*

Guan-qiang GAO[1,2,3], Bin XIN‡[1,2,3]

[1]*School of Automation, Beijing Institute of Technology, Beijing 100081, China*
[2]*Key Laboratory of Intelligent Control and Decision of Complex Systems, Beijing 100081, China*
[3]*Beijing Advanced Innovation Center for Intelligent Robots and Systems, Beijing 100081, China*
E-mail: 3120170426@bit.edu.cn; brucebin@bit.edu.cn

Received Sept. 11, 2018; Revision accepted Nov. 27, 2018; Crosschecked Jan. 8, 2019

Abstract: The multi-robot coverage motion planning (MCMP) problem in which every reachable area must be covered is common in multi-robot systems. To deal with the MCMP problem, we propose an efficient, complete, and off-line algorithm, named the "auction-based spanning tree coverage (A-STC)" algorithm. First, the configuration space is divided into mega cells whose size is twice the minimum coverage range of a robot. Based on connection relationships among mega cells, a graph structure can be obtained. A robot that circumnavigates a spanning tree of the graph can generate a coverage trajectory. Then, the proposed algorithm adopts an auction mechanism to construct one spanning tree for each robot. In this mechanism, an auctioneer robot chooses a suitable vertex of the graph as an auction item from neighboring vertexes of its spanning tree by heuristic rules. A bidder robot submits a proper bid to the auctioneer according to the auction vertexes' relationships with the spanning tree of the robot and the estimated length of its trajectory. The estimated length is calculated based on vertexes and edges in the spanning tree. The bidder with the highest bid is selected as a winner to reduce the makespan of the coverage task. After auction processes, acceptable coverage trajectories can be planned rapidly. Computational experiments validate the effectiveness of the proposed MCMP algorithm and the method for estimating trajectory lengths. The proposed algorithm is also compared with the state-of-the-art algorithms. The comparative results show that the A-STC algorithm has apparent advantages in terms of the running time and the makespan for large crowded configuration spaces.

Key words: Coverage motion planning; Multi-robot system; Auction algorithm; Spanning tree coverage algorithm
https://doi.org/10.1631/FITEE.1800551 **CLC number:** TP182

1 Introduction

Motion coverage is an important real-world application for mobile robots, such as floor cleaning, harvesting, and surveillance by robots (Di Franco and Buttazzo, 2016; Yehoshua et al., 2016; Xin et al., 2017; Li et al., 2018). In these applications, the coverage motion planning (CMP) problem is a key planning issue (An et al., 2017; Radmanesh et al., 2018). It is a typical complex problem that requires robots to plan trajectories to cover all points of interest while avoiding conflicts. At the same time, the CMP problem can be seen as a variant of the traveling salesman problem (TSP) (Galceran and Carreras, 2013). Therefore, the CMP problem is a non-deterministic polynomial (NP) hard problem in which the computation

‡ Corresponding author
* Project supported by the National Natural Science Foundation of China (Nos. 61822304, 61673058, and 61621063), the Project of Major International (Regional) Joint Research Program NSFC (No. 61720106011), and the NSFC–Zhejiang Joint Fund for the Integration of Industrialization and Informationization (No. U1609214)
ORCID: Bin XIN, http://orcid.org/0000-0001-9989-0418

time increases quickly with the dimensionality of the problem. In recent years, more researchers have concentrated on the multi-robot coverage motion planning (MCMP) problem (Gautam et al., 2015; Chakraborty et al., 2017; Fang et al., 2017). Multiple robots can partition a workload and cover target areas cooperatively. Although it is more complex than a single-robot coverage problem, multi-robot coverage has several potential advantages in terms of robustness and task performance.

Choset (2001) classified CMP algorithms as complete or heuristic algorithms based on whether the algorithms can plan paths to cover all reachable vertices. In addition, the CMP algorithm can be classified as off-line CMP algorithms if robots know the global environment information in advance, and online CMP algorithms otherwise. Khan et al. (2017) summarized environmental CMP algorithm modeling methods as classical exact cellular decomposition methods, grid-based methods, etc.

Rekleitis et al. (2008) proposed an algorithm which adopts the exact cellular decomposition, called "boustrophedon decomposition," to solve the MCMP problem. In this algorithm, the robots play two roles: one is an explorer that covers the border, and the other covers the remaining areas. Karapetyan et al. (2017) presented a greedy approach that extends an exact cellular decomposition by heuristics. Azpúrua et al. (2018) proposed an algorithm that segments the configuration space into hexagonal cells and allocates groups of robots into different clusters of coverage cells.

Grid-based methods approximately decompose the configuration space into several minimum cells. Each minimum cell is a square whose size is the same as that of the robot's associated coverage range. Gabriely and Rimon (2002) first proposed the spiral spanning tree coverage (STC) algorithm belonging to the class of grid-based methods. The STC algorithm divides the environment into mega grid cells whose size is twice that of the minimum cell to obtain an undirected graph structure. Then the STC algorithm can quickly obtain a Hamiltonian cycle of the graph as a coverage path for a single robot. Gabriely and Rimon (2003) provided the full STC algorithm, which can deal with situations where mega cells partially contain some obstacles. Hazon and Kaminka (2008) proposed a robust and efficient algorithm based on the STC algorithm for the MCMP problem. Using this algorithm, which constructs a spanning tree for all robots, every robot circumnavigates a part of the spanning tree to generate multiple coverage trajectories. According to the robots' initial positions, Kapoutsis et al. (2017) designed an MCMP algorithm, called "DARP," which divides areas equally to plan coverage trajectories. Kapanoglu et al. (2012) proposed heuristic coverage motion planning with a grid decomposition that adopts a pattern-based genetic algorithm (GA) to solve the MCMP problem.

In the MCMP problem, a key issue is how to allocate coverage task areas to robots. A good task allocation needs to balance each robot's workload and avoid conflicts among robots. The auction algorithm (Dias et al., 2006; Khamis et al., 2015) is a common market-based mechanism for task allocation used in multi-robot systems. In auction processes, robots compete for each sub-task and try to minimize a global function. Tang et al. (2018) used an auction-based task allocation for a multi-robot search-and-rescue task. Elango et al. (2011) provided a balancing task allocation algorithm using auction mechanisms that concentrate on distance and workload. Auction-based approaches can be divided into two types: centralized and distributed approaches (Kong et al., 2016).

The heuristic pattern-based GA can handle all cases (Kapanoglu et al., 2012). Nevertheless, it has high computation costs. The complete DARP algorithm can obtain the optimal solution in some cases (Kapoutsis et al., 2017); unfortunately, it cannot deal with some situations, such as a disconnected configuration space or a configuration space where mega cells are occupied partially by obstacles.

To design a complete MCMP algorithm which can deal with all cases, we propose a novel auction-based spanning tree coverage (A-STC) algorithm. The configuration space is modeled as a graph structure by the STC decomposition method. Each robot's spanning tree is managed by an auction mechanism to plan multiple coverage trajectories. An auctioneer robot selects a suitable vertex of the graph as an auction item using heuristic rules. A bidder robot will submit a proper bid to the auctioneer according to the auction vertex's connection relationships with its spanning tree and the estimated length of its trajectory. The estimated length is calculated based on vertexes and edges in the spanning tree,

which provides an upper bound to approximate the real path length. After iterative auction processes, complete coverage trajectories that avoid obstacles and conflicts among robots can be generated. Contributions compared with previous research are as follows:

1. The proposed auction strategies guarantee that each robot's spanning tree is connected, and that the workload is balanced for all robots.

2. The proposed MCMP algorithm is complete and can handle situations where mega cells are partially occupied by obstacles.

3. Acceptable makespans and trajectories can be obtained in a short computation time using the A-STC algorithm.

2 Problem definition and preliminaries

2.1 Problem definition

In the MCMP problem (Hazon and Kaminka, 2008; Di Franco and Buttazzo, 2016), n homogeneous robots are distributed at different locations in the configuration space, which can be decomposed into multiple cells. The size of a cell is equal to that of a robot's associated coverage range D, and each cell is numbered by its center's coordinate. In this grid-based decomposition method, two types of cells are distinguished. One is a free cell that robots can cover; the other is a cell that is occupied by obstacles. Assume that robots with a uniform speed v_{el} can move in four base directions (north, south, east, and west), and the time needed for a robot to change its direction is short enough to be neglected. rob_i denotes the robot with index i ($i \in \mathbb{R}^+$). $v_i(t) = (x_i(t), y_i(t))$ denotes the position of rob_i at moment t. Motion trajectory L_i of rob_i can be represented by a sequence of consecutive cells $[v_{b,i,1}, v_{b,i,2}, \ldots, v_{b,i,p}]$. $v_{b,i,q}$, which corresponds to a cell's center coordinate, denotes the position of rob_i at the q^{th} sequence of L_i. $\{L_i\}$ denotes the set of all cells without repetition in L_i.

The global mission is that robots must cover all reachable points in target areas while avoiding collisions with other robots and obstacles. The objective is to minimize the maximum completion time of multiple robots, which is the makespan of the coverage task. After that, the MCMP problem can be formulated as the following minimization problem:

$$\begin{aligned} L^* &= \arg\min_{L} \max_{i\in\{1,2,\ldots,n\}} T_i \\ \text{s.t. } &\{L_1\}\cup\{L_2\}\cup\ldots\cup\{L_n\} = G_f, \\ &\forall i \in \{1,2,\ldots,n\},\ \{L_i\}\cap G_{\mathrm{ob}} = \varnothing, \\ &\forall i \in \{1,2,\ldots,n\},\ L_i \text{ is continuous}, \\ &\forall i,j \in \{1,2,\ldots,n\},\ i\neq j,\ v_{b,i,q}\cap v_{b,j,q} = \varnothing, \end{aligned} \tag{1}$$

where $L = \{L_1, L_2, \ldots, L_n\}$ represents the set of robot motion trajectories, L^* represents the optimal set of robot trajectories for the MCMP problem, T_i represents the completion time of rob_i moving in trajectory L_i, $\max_{i\in\{1,2,\ldots,n\}} T_i$ represents the makespan of the coverage task, G_f represents the set of all free cells in the configuration space, and G_{ob} represents the set of all cells that contain obstacles. The first constraint means that all reachable minimum cells must be covered. The second constraint means that robots must avoid obstacles in the configuration space. The third constraint means that trajectories are realizable in the real world. The last constraint means that robots do not conflict with each other.

Additionally, the distance between any two vertexes $v_a = (x_a, y_a)$ and $v_b = (x_b, y_b)$ can be calculated as follows:

$$d(v_a, v_b) = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}. \tag{2}$$

2.2 Spiral spanning tree coverage algorithm for a single robot

Fig. 1 shows the single-robot off-line STC algorithm (Gabriely and Rimon, 2002, 2003), which is an approximate cellular-decomposition CMP method. In this study, the STC algorithm is used to plan each robot's trajectory based on its spanning tree. In the STC algorithm, the configuration space is divided into mega square cells of size $2D$. Based on these mega cells, an undirected graph structure $G_s = (V_s, E_s)$ can be obtained. When a mega cell is occupied totally by obstacles or free cells, the corresponding vertex of G_s is set individually as a free or obstacle vertex. The free vertex can connect neighbor vertexes in four directions, whereas the obstacle vertex cannot connect any neighbors. When a mega cell in the configuration space is partially occupied by obstacles, the connection relationships between the corresponding STC vertexes and neighbors of the vertex will change. The changed connection

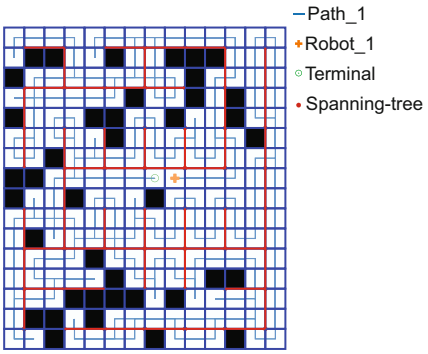

Fig. 1 Coverage path planning using the spiral spanning tree coverage (STC) algorithm

relationships can be divided into four types (Fig. 2). When an STC mega cell contains only a single obstacle, its corresponding vertex can connect neighbors in four directions, in the same way as a free vertex does. A double-obstacle vertex with two obstacles in the same side belongs to the second type. The double-obstacle vertex can extend in all directions except the direction of obstacles. The third type is a triple-obstacle vertex, which can connect vertexes of G_s in only two directions. In the last type, an STC mega cell is partially occupied by two obstacles in different sides. The mega cell will be decomposed into two triple-obstacle vertexes of G_s, which are adjacent to neighbor cells in two directions.

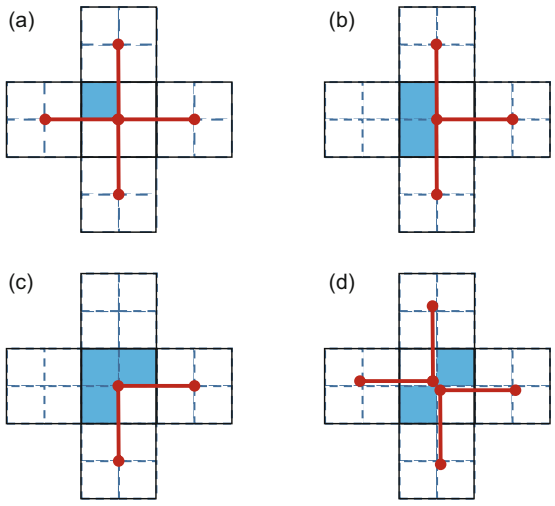

Fig. 2 Connection relationships of a single-obstacle vertex (a), a vertex with double obstacles in the same side (b), a triple-obstacle vertex (c), and double vertexes (d) with neighbors

For $G_s = (V_s, E_s)$, $V_s = \{v_{s,1}, v_{s,2}, \ldots, v_{s,m}\}$ is the set of all reachable vertexes in G_s, and $E_s = \{e_{s,1}, e_{s,2}, \ldots, e_{s,p}\} \subseteq V_s \times V_s$ is the set of edges that connect two adjacent vertexes. V_s contains all reachable minimum cells. A spanning tree of G_s can be generated easily by spanning-tree algorithms, such as the Dijkstra algorithm and Prim algorithm (Yehoshua et al., 2016). Then a coverage trajectory can be planned where the robot moves along the spanning tree until it returns to the start position. When mega cells in the configuration space are totally occupied by obstacles or free cells, the STC planning algorithm can obtain the Hamiltonian cycle that minimizes the coverage time.

2.3 Problem analysis

Completion time T_i of rob_i is directly proportional to v_{el} and the number of cells in L_i. In view of the STC algorithm, a coverage motion trajectory L_i can be generated based on the spanning tree in the STC graph G_s. $\mathrm{Tr}_{s,i}$ represents the spanning tree of rob_i. $\mathrm{Tr} = \{\mathrm{Tr}_{s,1}, \mathrm{Tr}_{s,2}, \ldots, \mathrm{Tr}_{s,n}\}$, which is composed by all spanning trees of robots, can correspond to a solution L in the MCMP problem. The cardinality of $\mathrm{Tr}_{s,i}$ is defined by the number of cells in the corresponding L_i. Meanwhile, the relationship between T_i and the cardinality of $\mathrm{Tr}_{s,i}$ is

$$T_i = \frac{|\mathrm{Tr}_{s,i}| \cdot D}{\mathrm{vel}} = \frac{|L_i| \cdot D}{\mathrm{vel}}, \tag{3}$$

where $|\mathrm{Tr}_{s,i}|$ represents the cardinality of $\mathrm{Tr}_{s,i}$, and $|L_i|$ represents the number of cells in L_i.

To approximate the optimal solution L^*, constraints are set as

$$\mathrm{Tr}_{s,i} \cap \mathrm{Tr}_{s,j} = \varnothing,\ \forall i, j \in \{1, 2, \ldots, n\},\ i \neq j, \tag{4}$$

$$\mathrm{Tr}_1 \cup \mathrm{Tr}_2 \cup \ldots \cup \mathrm{Tr}_n = V_s, \tag{5}$$

$$|\mathrm{Tr}_{s,1}| \approx |\mathrm{Tr}_{s,2}| \approx \ldots \approx |\mathrm{Tr}_{s,n}|, \tag{6}$$

$$\mathrm{Tr}_{s,i} \text{ is connected},\ \forall i \in \{1, 2, \ldots, n\}, \tag{7}$$

$$v_{s,i}(0) \in \mathrm{Tr}_{s,i},\ \forall i \in \{1, 2, \ldots, n\}. \tag{8}$$

Constraint (4) makes sure that each STC vertex belongs to only one robot's spanning tree, and the robots' paths have no conflicts. Constraint (5) highlights that every free minimum cell will be covered. As shown in constraint (6), the makespan of the coverage task will decrease when the maximum cardinality difference among all robots' spanning trees decreases. Conditions (7) and (8) guarantee that the STC algorithm can plan coverage paths for multiple

robots. $v_{s,i}(0)$ represents an STC vertex corresponding to the initial position of rob_i. However, it is not always possible to satisfy all these criteria in complex environments.

According to the above analyses, how to generate one spanning tree for each robot is a key issue for the MCMP problem. In the next section, an auction mechanism is designed with several heuristic rules and strategies to manage the spanning tree of each robot.

3 Auction-based spanning tree coverage algorithm

The A-STC algorithm is an off-line MCMP algorithm as shown in Algorithm 1, applicable only for environments that can be decomposed into undirected graphs. First, the STC graph G_s of the configuration space is constructed. Each robot's spanning tree $\text{Tr}_{s,i}$ is a null set, denoted by $\varnothing$. Then, an STC vertex $v_{s,i}(0)$ corresponding to the initial position of rob_i is added to $\text{Tr}_{s,i}$. In iterative auction processes, all robots participate to solve the MCMP problem. A fixed number of iterations "maxIter" is set as a termination condition for the auction processes. The other termination condition is that all robots are in sleep mode. A robot in sleep mode cannot organize an auction. The sleep mode is designed to reduce invalid auctions. At the k^{th} auction, an auctioneer must be selected first. An auctioneer chooses an STC vertex (denoted by v_a) from the STC graph. Every robot can be a participant which submits a bid to the auctioneer. After all bids are received, the auctioneer determines the winner robot (denoted by $\text{rob}_{\text{winner}}$) from all bidder robots. Robots will be ready for the $(k+1)^{\text{th}}$ auction after they update their relevant statuses and spanning trees. When iterative auction processes are terminated, each robot will obtain its own spanning tree $\text{Tr}_{s,i}$. Using the STC algorithm, robots circumnavigate their spanning trees to plan coverage trajectories.

Based on different vertexes and edges in spanning tree $\text{Tr}_{s,i}$ of rob_i, trajectories with different lengths will be planned. Meanwhile, a trajectory that covers p free target cells is planned by the STC algorithm in time $O(p)$. To reduce the computation time of the MCMP problem, a heuristic method for estimating the length of rob_i's trajectory (Fig. 3) is designed in this study. In this heuristic estimation method, the estimated length of a trajectory is equal to the sum of the estimated cost of each STC vertex

Algorithm 1 Auction-based spanning tree coverage (A-STC) algorithm

Require: Configuration space and maxIter.
Ensure: $L = \{L_1, L_2, \ldots, L_n\}$.
According to the configuration space, construct G_s;
$\forall i \in \{1, 2, \ldots, n\}$, add $v_{s,i}(0)$ to $\text{Tr}_{s,i}$;
$k = 0$;
for $k <$ maxIter **do**
  Select an auctioneer;
  Auctioneer determines an auctioneer vertex;
  if $\forall i \in \{1, 2, \ldots, n\}$, rob_i is in the sleep mode **then**
    terminate iterations;
  end if
  if auctioneer is in the sleep mode **then**
    continue;
  end if
  Bidders provide bids;
  Auctioneer determines the winner robot;
  Bidders update own statuses and spanning trees;
  $k = k + 1$;
end for
$\forall i \in \{1, 2, \ldots, n\}$, plan L_i based on $\text{Tr}_{s,i}$ by the STC algorithm.

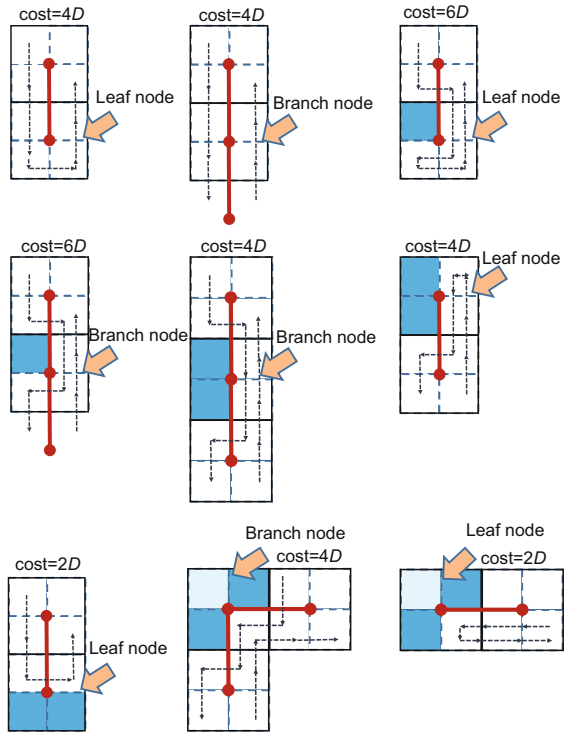

Fig. 3 Estimated costs of different vertexes with different edges in the spanning tree

in the spanning tree. Two categories of vertexes in the spanning tree are distinguished. One is a vertex whose estimated cost keeps the same value denoted by $v_{s,s}$; the other vertex's estimated cost (denoted by $v_{s,c}$) changes with its connection relationships to neighbors. Vertexes with unchanged estimated costs contain free vertexes and single-obstacle vertexes. Their estimated costs are set as $4D$ and $6D$, respectively. According to different degrees in a spanning tree, nodes are classified as leaf and branch nodes. The estimated value of a vertex with double obstacles is set as $2D$, provided that it is a leaf node and that its adjacent edge is perpendicular to the side of obstacles. In other situations, the estimated value of a double-obstacle vertex is $4D$. If a vertex of G_s with triple obstacles is a leaf node, its estimated cost is set as $2D$. Otherwise, the estimated cost of a triple-obstacle STC vertex is $4D$.

3.1 Auction process

The auctioneer is a role that selects a vertex for bidding, originates an auction, and determines the winner robot. First, a neighbor vertex set of $\mathrm{Tr}_{s,i}$ is defined as

$$V^*_{s,i}=\{v_s\in G_s|\exists v_{s,i}\in \mathrm{Tr}_{s,i},\ e(v_s,v_{s,i})\in E_s,\ v_s\notin \mathrm{Tr}_{s,i}\}, \tag{9}$$

where $V^*_{s,i}$ represents the set of all reachable vertexes adjacent to vertexes of spanning tree $\mathrm{Tr}_{s,i}$ except those belonging to $\mathrm{Tr}_{s,i}$.

If rob_i wants to be an auctioneer, it will select a mega vertex as an item for bidding from $V^*_{s,i}$. Thus, when $V^*_{s,i}=\varnothing$, rob_i, which cannot be an auctioneer, is set in sleep mode. This strategy for selecting a bidding vertex is used to keep each spanning tree's connectedness. $V^*_{s,i}$ contains three parts. The first part represents a set of vertexes that are assigned to other robots, and is denoted by $V^*_{s,i,\mathrm{as}}=V^*_{s,i}\cap\sum_{i=1}^{n}\mathrm{Tr}_{s,i}$. The second part represents a set of vertexes with unchanged estimated costs that have not been assigned, and is denoted by $V^*_{s,i,\mathrm{us}}$. The last part represents a set of unassigned vertexes whose estimated costs change with relationships of connections, and is denoted by $V^*_{s,i,\mathrm{uc}}$. Because the size of vertexes in $V^*_{s,i}$ is usually not equal to 1, several heuristic rules are designed to select a suitable vertex v_a from $V^*_{s,i}$.

Rule 1 The robot rob_i prefers to select v_a from the set of unassigned vertexes. At the same time, vertexes in $V^*_{s,i,\mathrm{us}}$ have higher priorities to be an auction item than vertexes in $V^*_{s,i,\mathrm{uc}}$. When vertexes belong to the same type, to determine the most suitable STC vertex in $V^*_{s,i,\mathrm{us}}\cup V^*_{s,i,\mathrm{uc}}$, we use

$$P_{\mathrm{su}}(i,j)=\sum_{\substack{p=1\\p\neq i}}^{n}\sum_{v_s\in\mathrm{Tr}_{s,i}}d(v^*_{i,j},v_s), \tag{10}$$

where $P_{\mathrm{su}}(i,j)$ is a priority cost that represents the sum of the distances from vertex $v^*_{i,j}$ to all current vertexes assigned to robots except rob_i. Robot rob_i chooses the element that corresponds to the maximum $P_{\mathrm{su}}(i,j)$ as an auction vertex v_a.

Rule 2 When $V^*_{s,i,\mathrm{us}}=V^*_{s,i,\mathrm{uc}}=\varnothing$, an auctioneer robot chooses a candidate v_a from $V^*_{s,i,\mathrm{as}}$. If a graph constructed by $\overline{V^*}_{i,j}=\{x\in\mathrm{Tr}_{s,i}|x\neq v^*_{i,j}\}$ is disconnected, $v^*_{i,j}$ cannot be selected as an auction item. In addition, the priorities of vertexes in $V^*_{s,i,\mathrm{as}}$ can be described as

$$P_{\mathrm{sa}}(i,j)=C_p,\ v^*_{i,j},\in\mathrm{Tr}_{s,i}, \tag{11}$$

where $P_{\mathrm{sa}}(i,j)$ means the estimated coverage length of the robot whose spanning tree $\mathrm{Tr}_{s,i}$ contains vertex $v^*_{i,j}$. The vertex corresponding to the maximum value of $P_{\mathrm{as}}(i,j)$ will be chosen as an auction item. If the maximum value of $P_{\mathrm{as}}(i,j)$ is not greater than the auctioneer's estimated length, the auctioneer will also be set in sleep mode. In this mode, the robot cannot organize any auctions. Nevertheless, a robot in sleep mode will awake when the maximum value of $P_{\mathrm{as}}(i,j)$, which is calculated by Eq. (11), is greater than its estimated path length.

Rule 3 When the maximum values obtained by Rule 2 are not unique, the robot will choose the vertex nearest to its spanning tree $\mathrm{Tr}_{s,i}$ as the auction item:

$$P_{\mathrm{ss}}(i,j)=\frac{1}{\sum\limits_{v_p\in\mathrm{Tr}_{s,i}}d(v^*_{i,j},v_p)}, \tag{12}$$

where $P_{\mathrm{ss}}(i,j)$ represents the reciprocal of the sum of distances from the candidate vertex $v^*_{i,j}$ to all vertexes in $\mathrm{Tr}_{s,i}$. The robot will choose the vertex that has the minimum sum of the distances for an auction process.

Remark 1 Rule 1 is designed to auction all vertexes rapidly so that the spanning tree of each robot can be generated rapidly. Vertexes that have lower estimated costs in their leaf node conditions prefer to become leaf nodes in spanning trees. Rules 2 and 3 are used to choose a vertex that has a potential ability to decrease the makespan of the coverage task.

According to Rules 2 and 3, an auction process for a single robot is shown in Algorithm 2. In lines 5 and 14 of Algorithm 2, the auctioneer robot will choose a robot with the highest bid as $\text{rob}_{\text{winner}}$ to decrease the makespan of the coverage task. The results are broadcasted to make the information concordant at the end of the auction process.

Algorithm 2 Auction process of rob_i

Require: $V^*_{s,i}, \text{Tr}_{s,1}, \text{Tr}_{s,2}, \ldots, \text{Tr}_{s,n}$

1: **if** $V^*_{s,i} \neq \varnothing$ **then**
2: **if** $V^*_{s,i,\text{us}} \neq \varnothing$ or $V^*_{s,i,\text{uc}} \neq \varnothing$ **then**
3: Obtain v_a according to Rule 1;
4: Collect all bids from the other robots;
5: Select the highest bidder as $\text{rob}_{\text{winner}}$;
6: Broadcast the results of this auction;
7: **else**
8: Calculate the status of rob_i;
9: **if** rob_i is in the sleep mode **then**
10: rob_i cannot organize an auction;
11: **end if**
12: **if** rob_i is awake **then**
13: Obtain v_a according to Rules 2 and 3;
14: Collect all bids from the other robots;
15: Select the highest bidder as $\text{rob}_{\text{winner}}$;
16: Broadcast the results of this auction;
17: **end if**
18: **end if**
19: **else**
20: Set rob_i in the sleep mode;
21: **end if**

3.2 Bid process

The strategies for bidding and determining the winner robot cooperatively keep the continuity of the motion trajectories and the balance of each robot's workload. In the A-STC algorithm, every robot estimates the length of its trajectory based on the above estimation method. The estimated cost of rob_i's path is denoted by C_i. When the bidder robot receives the STC vertex v_a offered by the auctioneer, it will bid for this coverage sub-task. First, if $v_a = v_{s,i}(0)$, the bid of rob_i is infinite to meet the demand $v_{s,i}(0) \in \text{Tr}_{s,i}$. To satisfy constraint (7), the bid offered by rob_i is zero when $v_a \notin \text{Tr}_{s,i} \cup V^*_{s,i}$. Otherwise, if $v_a \in \text{Tr}_{s,i}$, the bid offered by rob_i at the k^{th} iteration auction is the reciprocal of $C_i(k)$. When $v_a \in V^*_{s,i}$, edges e_s which connect v_a and a vertex of $\text{Tr}_{s,i}$ are not always unique. Different connection relationships can cause different trajectory lengths. Thus, to determine how to connect v_a and $\text{Tr}_{s,i}$, a minimization problem is formulated as

$$\begin{aligned} e^*_s &= \arg \min_{e_s} \acute{C}_i \\ \text{s.t. } e_s &=< v_a, v_s >, \\ v_s &\in \text{Tr}_{s,i}, \\ e_s &\in E_s, \end{aligned} \tag{13}$$

where $\acute{C}_i$ represents an estimated cost based on a new spanning tree composed of $\text{Tr}_{s,i}$ and v_a with edge e_s, e^*_s represents the optimal solution, and C^*_i represents the estimated cost of the optimal edge.

Because the number of items in e_s is up to 4, the minimum $\acute{C}_i$ can be calculated using the exhaustion method. $\acute{C}_i$ is equal to C_i plus the increment of the estimated cost caused by v_a and v_s. Because v_a is a leaf node, the increment is calculated first in terms of its type and connection relationships with neighbors. If the degree of v_s is equal to 1 and its estimated cost is $2D$, $2D$ will also be added to the increment of estimated cost.

According to the above bid strategies, the bid process of a single robot is shown in Algorithm 3. If rob_i wins an auction vertex that is not in $\text{Tr}_{s,i}$, it adds v_a and the optimal edge e^*_s to $\text{Tr}_{s,i}$. When rob_i loses an auction vertex, it deletes v_a and its corresponding edge from $\text{Tr}_{s,i}$. If the deletion makes the spanning tree disconnected, $\text{Tr}_{s,i}$ will be reconstructed to make it connected again. Vertexes with unchanged estimated costs have higher priorities than vertexes with changed estimated costs in the reconstruction. At the same time, rob_i updates its $V^*_{s,i}$ and C_i to get ready for another bid process.

3.3 Algorithm analysis

Lemma 1 The A-STC algorithm generates n trajectories that jointly cover every cell accessible from the starting position of each robot.

Proof Because the STC algorithm can circumnavigate the spanning tree to obtain a closed path that covers all the cells, the completeness problem can be decomposed into two problems. One problem is to prove that spanning trees of all robots can occupy all vertexes; the other problem is to prove the connectedness of spanning trees. Assuming that v_c is a reachable vertex that has not been assigned to robots, a traversal algorithm can obtain a connected graph G_c composed by unassigned vertexes.

Algorithm 3 Bidding process of rob_i

Require: $V^*_{s,i}, \text{Tr}_{s,i}, v_a, C_i$

Receive v_a from the auctioneer;
if $v_a \in \text{Tr}_{s,i} \cup V^*_{s,i}$ **then**
  if $v_a = P_i(0)$ **then**
    bid $= \infty$ (infinity);
  else
    if $v_a \in V^*_{s,i}$ **then**
      Calculate $\mathcal{C}^*_{i,a}$ and e^*_s;
      bid $= \frac{1}{\mathcal{C}^*_{i,a}}$;
    else
      bid $= \frac{1}{C_i}$;
    end if
  end if
else
  bid $= 0$;
end if
Submit bid to the auctioneer;
Receive the auction results from the auctioneer;
if $\text{rob}_{\text{winner}} \neq \text{rob}_i$ **and** $v_a \in \text{Tr}_{s,i}$ **then**
  Delete v_a and its corresponding edge from $\text{Tr}_{s,i}$, and reconstruct $\text{Tr}_{s,i}$;
  Update $V^*_{s,i}$ and C_i;
end if
if $\text{rob}_{\text{winner}} = \text{rob}_i$ **and** $v_a \in V^*_{s,i}$ **then**
  Add v_a and the optimal edge e^*_s into $\text{Tr}_{s,i}$;
  Update $V^*_{s,i}$ and C_i;
end if

According to Rule 1, if G_c and a robot's spanning tree are connected, v_c will be auctioned and added to a spanning tree after several iterations. An auctioneer robot selects a neighbor vertex as an auction vertex, and a bidder robot bids for a neighbor vertex or a vertex in its spanning tree. Thus, the connectedness of each spanning tree can be ensured.

Lemma 2 Denote the number of free minimum cells in the configuration space by x. Define cells that share at least one point with the grid boundary as boundary cells. Denote the number of boundary cells by y. The total length of trajectories that are planned by the A-STC algorithm is less than or equal to $(x + y)D$.

Proof When a path is planned by the single-robot STC algorithm, its length is less than or equal to $(x + y)D$. The A-STC algorithm partitions a whole spanning tree into n spanning trees.

To promote the generality of the A-STC algorithm, the method for selecting an auctioneer robot is not specified. For example, each robot can become an auctioneer in a particular order, or a robot which has the minimum C_i will be selected as an auctioneer. These two methods for selecting an auctioneer robot are both effective for the A-STC algorithm.

In iterative auction processes (Fig. 4), vertexes with unchanged estimated costs are assigned rapidly. At the same time, robots' spanning trees tend to remain far away from each other. When the auctioneer robot receives multiple bids ($0 < \text{bid} < \infty$) from different robots, the estimated lengths of bidder robots' trajectories can get close to each other to meet constraint (6). The difference in estimated lengths can decrease if an auction vertex has been assigned. When an auction vertex is not assigned to any robots, the minimum estimated length of robot spanning trees that are adjacent to the auction vertex will increase. The coverage trajectories will not conflict with each other because an STC vertex is assigned to only one robot.

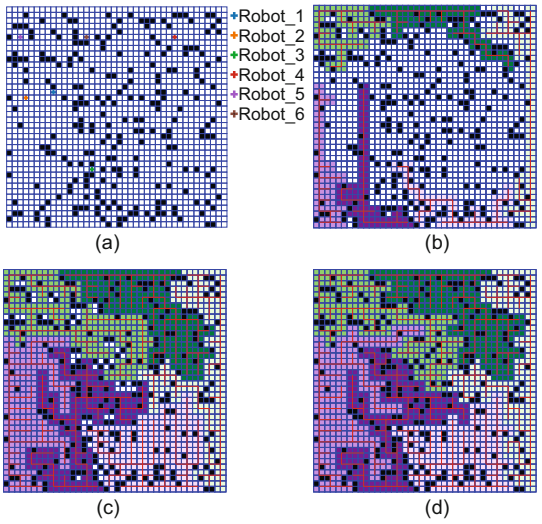

Fig. 4 Progressions of spanning trees with $k = 0$ (a), $k = 150$ (b), $k = 400$ (c), and in the termination condition (d) (References to color refer to the online version of this figure)

The A-STC algorithm can determine the paths that cover all minimum cells until all STC vertexes have been assigned. Therefore, the maximum number of iterations, maxIter, must ensure that all vertexes can be auctioned. When maxIter is set as too large a number, the proposed sleep mode can end the A-STC algorithm to reduce invalid iterations.

The proposed A-STC algorithm is used to solve the MCMP problem of n robots. Table 1 shows the computational complexities of the processes in the A-STC algorithm.

Table 1 Computational complexities of the processes in the auction-based spanning tree coverage (A-STC) algorithm

Step	Process	Computational complexity
1	Initialization	$O(\|G_f\|)$
2	Auctioneer selection	$O(n)$
3	Auction process	$O(\|G_f\|^2)$
4	Bidding process	$O(\|G_f\|)$
5	Trajectories generation	$O(\|G_f\|)$

The worst-case time complexity of the A-STC algorithm can be approximately expressed by

$$O(\text{maxIter} \cdot (|G_f|^2 + n)). \tag{14}$$

4 Computational experiments and analysis

This section is devoted to the performance investigation of the proposed algorithm. To simplify descriptions, the velocity of the robot is assumed to be D in the whole section. A low boundary (LB) of an MCMP problem's solution is defined as the minimum number of reachable cells that are covered by each robot. The MCMP problem, which is an NP-hard problem, cannot easily determine the optimal solution when the instance size is large. LB is used in this section to evaluate a solution's proximity to the optimal solution. First, the effectiveness of the proposed MCMP method and estimation method is validated in different configuration spaces. Second, to illustrate the generality of the A-STC algorithm, different methods for selecting an auctioneer robot are compared and discussed. Finally, the A-STC algorithm is separately compared with the GA and DARP algorithms (Kapanoglu et al., 2012; Kapoutsis et al., 2017). GA can handle all configuration spaces that are modeled by the grid decomposition method. Nevertheless, the computation cost of GA cannot be endured when the environment size and the number of robots become large. The A-STC algorithm is compared mainly with GA in terms of the performance of small-scale MCMP problems. The DARP algorithm, with low computation costs, can handle the MCMP problem only in special configuration spaces where all mega cells are totally occupied by free cells or obstacles. The A-STC algorithm is also compared with the DARP algorithm in these special configuration spaces. It should be highlighted that separate comparison experiments were designed to illustrate different abilities of the A-STC algorithm. The purpose of GA comparison experiments is to verify the ability of the A-STC algorithm to approximate the optimal solution in common configuration spaces. The comparison experiments with the DARP algorithm focused mainly on validating the low computation cost of the A-STC algorithm. A PC with Intel® Xeon® E5 @2.60 GHz and 32 GB RAM was used for all computational experiments. The A-STC algorithm compiled by C++ and the DARP algorithm, written in Java, were provided by Kapoutsis et al. (2017).

4.1 Experiment 1: validation effectiveness

Two situations with different environments and robot positions are shown in Figs. 5 and 6. The configuration spaces with random obstacles have 40×40 unit cells. Figs. 5b and 6b highlight the spanning trees of all robots that are generated by the auction mechanisms. Final trajectories of robots covering all reachable areas are described in Figs. 5c and 6c. Accordingly, the adaptability of the proposed algorithm to the configuration spaces and the number of robots is verified.

A trajectory length is proportional directly to the time spent by a robot that moves along the trajectory. At the termination condition, the estimated length of rob_i is denoted by $C_{t,i}$. The corresponding estimated completion time can be calculated by the estimated path length. Thus, to evaluate the effectiveness of the proposed method for estimating the trajectory length, a bias ratio of time is denoted by Br as

$$\text{Br} = \frac{\max\limits_{i\in\{1,2,\ldots,n\}}\{C_{t,i}/D\} - \max\limits_{i\in\{1,2,\ldots,n\}} T_i}{\max\limits_{i\in\{1,2,\ldots,n\}} T_i}, \tag{15}$$

where $\max\limits_{i\in\{1,2,\ldots,n\}}\{C_{t,i}/D\}$ represents the estimated makespan which is calculated by the proposed estimation method, and $\max\limits_{i\in\{1,2,\ldots,n\}} T_i$ represents the real makespan which is planned by the A-STC algorithm. In the computational experiments, the above two configuration spaces were used as test environments. A total of 200 experiments were carried out for different numbers of robots. Fig. 7 shows the relationships between the mean bias ratio of 200 experiments and the number of robots. The number of robots was up to ten. The bias ratio did not change

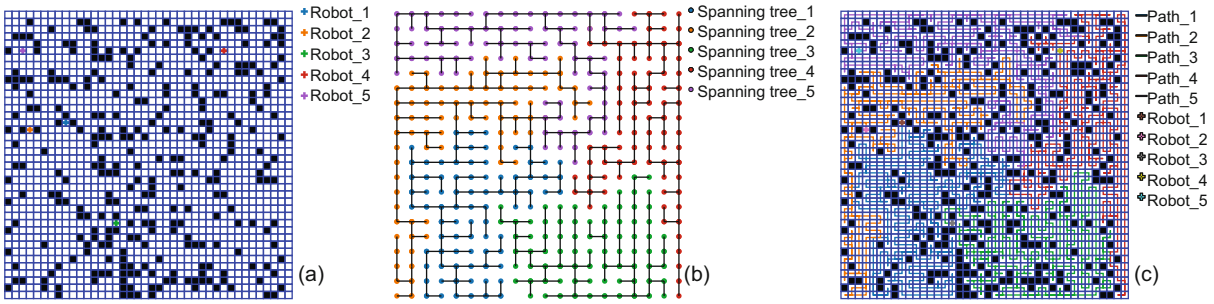

Fig. 5 Initial positions (a), spanning trees (b), and trajectories (c) of robots in configuration space I (References to color refer to the online version of this figure)

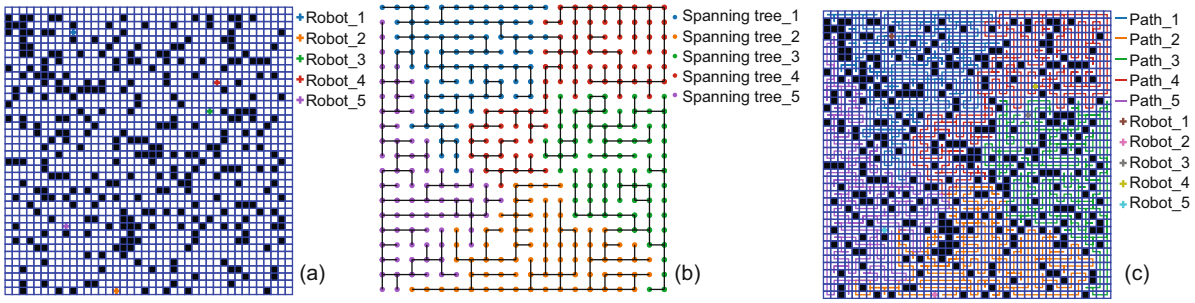

Fig. 6 Initial positions (a), spanning trees (b), and trajectories (c) of robots in configuration space II (References to color refer to the online version of this figure)

apparently with environments and the numbers of robots. The maximum bias ratio was less than 0.08. Therefore, the proposed estimation method is effective, and can be used in auction processes.

4.2 Experiment 2: sensitivity analysis of decision orders

The method for selecting an auctioneer in the A-STC algorithm is equivalent to that for determining a decision order. Different decision orders will cause different solution qualities in multi-robot systems. Thus, computational experiments were carried out

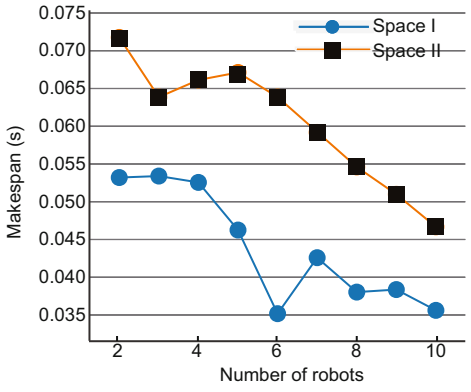

Fig. 7 Bias ratio versus the number of robots

to validate the influences of decision orders on the algorithm proposed in this paper. In the experiments, the maximum number of robots was 10, the above configuration spaces were adopted, and 200 different decision orders were tested for each different number of robots. In each decision order, every robot had the same chance to be an auctioneer. According to the statistical results (Fig. 8), the makespan of the coverage task planned by the proposed algorithm was insensitive to the decision order. The makespans of 200 different decision orders were very close to that planned by the A-STC algorithm in the sequential decision. Fig. 8 also shows a considerable decrease in the makespan of the coverage task as the number of robots increased. The effectiveness of the A-STC algorithm is highlighted in Fig. 8.

4.3 Experiment 3: comparison with the genetic algorithm

To test the A-STC algorithm's performance in different environments, some typical experimental environments were applied. As shown in Fig. 9, the experimental environments contained free, outdoor, bar maze, office, living room, and circular maze (Gautam et al., 2015). Coverage rate Cr and re-

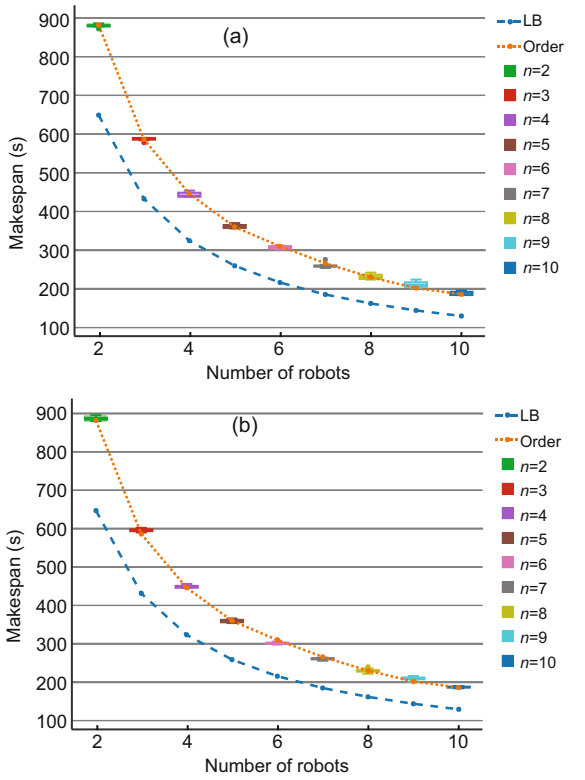

Fig. 8 Statistical results with respect to different decision orders in configuration spaces I (a) and II (b) (References to color refer to the online version of this figure)

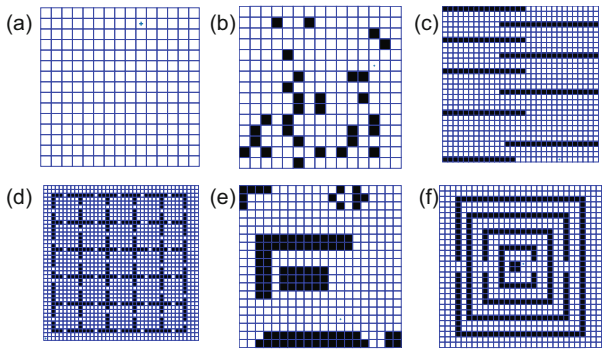

Fig. 9 Configuration spaces of the free (a), outdoor (b), office (c), living room (d), bar maze (e), and circular maze (f) environments

peated coverage Rc are introduced in this study to investigate the algorithms' performances:

$$\mathrm{Cr} = \frac{|\{L_1\} \cup \{L_2\} \cup \ldots \cup \{L_n\}|}{|G_f|}, \tag{16}$$

$$\mathrm{Rc} = \frac{\sum_{i=1}^{n} |L_i| - |\{L_1\} \cup \{L_2\} \cup \ldots \cup \{L_n\}|}{\sum_{i=1}^{n} |L_i|}, \tag{17}$$

where $|\{L_1\} \cup \{L_2\} \cup \ldots \cup \{L_n\}|$ represents the number of all covered cells without repetition, $|G_f|$ represents the number of all reachable cells, and $\sum_{i=1}^{n} |L_i|$ represents the total number of cells in all robots' trajectories including repeated covered cells. The makespan and repeated coverage obtained by the A-STC algorithm are shown in Table 2. In the A-STC algorithm, robots with the minimum estimated cost will be selected as the auctioneer. In the free and living room environments, the makespan planned by the A-STC algorithm is very close to LB. The repeated coverage of the A-STC algorithm increases slowly as environments become complex. In the outdoor environment, the makespan of the A-STC algorithm becomes apparently large due to the repeated coverage. No matter how complex environments are, the makespan decreases as the number of robots becomes large. GA's parameters were set the same as the parameters adopted by Kapanoglu et al. (2012). Table 2 shows the experimental results in terms of mean and standard deviation of final solutions' capabilities within 10 independent runs obtained by GA for the selected test instances. The computation time of GA is very long. For example, the computation for the outdoor environment with 40×40 cells requires eight hours. Thus, the comparative experiments focused on small environments. Cr in the A-STC algorithm is always equal to one because of the completeness of the algorithm. For all instances, the A-STC algorithm shows better performance than GA in terms of the coverage rate. The makespan of GA is not a real makespan when the corresponding coverage rate is not equal to 1. The makespan is equal to the time when all robots are stuck. Nevertheless, the A-STC algorithm is significantly better than GA when environments are complex or the number of robots becomes large.

4.4 Experiment 4: comparison with the DARP algorithm

The parameters of the A-STC and DARP algorithms in the comparison experiments were set as follows: The size of the configuration spaces ranged from 16×16 to 100×100 units. The number of robots varied from 2 to 80. Robots and obstacles were distributed randomly in the configuration space. All mega cells in the configuration space were totally occupied by obstacles or free cells. In this situation,

Table 2 Experimental results of the A-STC and genetic algorithms on the 15 instances

Type	Number of robots	Space size	Lower bound (s)	A-STC algorithm		Genetic algorithm		
				Makespan (s)	Repeated coverage	Makespan (s)	Coverage rate	Repeated coverage
Free	2	20×20	2.000×10^{2}	2.000×10^{2}	0×10^{0}	2.000×10^{2} (0×10^{0})	1.000×10^{0} (0×10^{0})	1.000×10^{0} (0×10^{0})
	6	40×40	2.667×10^{2}	2.680×10^{2}	0×10^{0}	2.725×10^{2} (1.55×10^{0})	1.000×10^{0} (0×10^{0})	1.531×10^{-2} (5.343×10^{-3})
	8	40×40	2.000×10^{2}	2.040×10^{2}	0×10^{0}	2.094×10^{2} (2.629×10^{0})	9.998×10^{-1} (4.167×10^{-4})	3.28×10^{-2} (1.02×10^{-2})
Outdoor	4	30×30	1.845×10^{2}	2.440×10^{2}	2.683×10^{-1}	2.434×10^{2} (8.542×10^{0})	9.749×10^{-1} (9.44×10^{-3})	2.432×10^{-1} (1.686×10^{-2})
	6	40×40	2.127×10^{2}	2.980×10^{2}	3.668×10^{-1}	3.211×10^{2} (1.421×10^{1})	9.725×10^{-1} (9.649×10^{-3})	3.076×10^{-1} (1.754×10^{-2})
	8	40×40	1.595×10^{2}	2.340×10^{2}	4.013×10^{-1}	2.558×10^{2} (1.529×10^{1})	9.758×10^{-1} (7.526×10^{-3})	3.078×10^{-1} (1.465×10^{-2})
Bar maze	3	30×30	2.480×10^{2}	2.760×10^{2}	9.812×10^{-2}	2.747×10^{2} (7.76×10^{0})	9.972×10^{-1} (4.235×10^{-3})	5.424×10^{-2} (1.975×10^{-2})
	6	30×30	1.240×10^{2}	1.380×10^{2}	7.527×10^{-2}	1.375×10^{2} (2.62×10^{0})	1.000×10^{0} (0×10^{0})	4.610×10^{-2} (1.014×10^{-2})
Office	4	40×40	3.178×10^{2}	3.760×10^{2}	1.660×10^{-1}	3.845×10^{2} (1.459×10^{1})	9.826×10^{-1} (1.210×10^{-2})	1.622×10^{-1} (1.768×10^{-2})
	6	40×40	2.118×10^{2}	2.480×10^{2}	1.660×10^{-1}	2.737×10^{2} (7.508×10^{0})	9.880×10^{-1} (5.147×10^{-3})	1.952×10^{-1} (1.110×10^{-2})
	8	40×40	1.589×10^{2}	1.880×10^{2}	1.715×10^{-1}	2.118×10^{2} (6.383×10^{0})	9.878×10^{-1} (7.700×10^{-3})	2.002×10^{-1} (2.153×10^{-2})
Living room	2	20×20	1.515×10^{2}	1.620×10^{2}	6.271×10^{-2}	1.625×10^{2} (1.819×10^{0})	9.948×10^{-1} (3.590×10^{-3})	6.957×10^{-2} (1.193×10^{-2})
	4	20×20	7.575×10^{1}	8.400×10^{1}	7.591×10^{-2}	8.593×10^{1} (6.892×10^{0})	9.967×10^{-1} (4.667×10^{-3})	8.216×10^{-2} (1.680×10^{-2})
Circular maze	2	30×30	3.290×10^{2}	3.920×10^{2}	1.626×10^{-1}	3.428×10^{2} (3.007×10^{1})	9.500×10^{-1} (5.475×10^{-2})	6.637×10^{-2} (3.046×10^{-2})
	8	30×30	8.225×10^{1}	1.120×10^{2}	1.748×10^{-1}	1.094×10^{2} (7.47×10^{0})	9.968×10^{-1} (4.265×10^{-3})	1.074×10^{-1} (5.016×10^{-2})

the trajectory generated by the STC algorithm did not have any repeated coverage, and in some cases, the optimal makespan of the MCMP problem was equal to LB. In the A-STC algorithm, each robot became an auctioneer in sequence. The maximum number of iterations of the auction-based STC algorithm was set to be $0.375|G_f|$. To make the DARP algorithm iterate sufficiently, the maximum number of iterations of the DARP algorithm was set to be 80 000, which is larger than $0.375|G_f|$.

The results obtained during computational experiments are presented in Table 3. The sizes of the tested instances are classified as small, medium, and large scales. In the small- and medium-scale instances without obstacles, the A-STC and DARP algorithms both had a good ability to approximate the optimal solution. The A-STC algorithm can obtain an acceptable solution rapidly for large-scale instances without obstacles. In contrast, the DARP algorithm, which has a great ability of global optimization, can obtain a better makespan of the coverage task in more computation time. In some situations, the DARP algorithm cannot divide the target areas. The termination condition of the DARP algorithm is when the makespan of the planned solution is equal to LB. In the large-scale instance, LB is not always close to the optimal value; thus, the DARP algorithm is hard to converge. One termination condition for the A-STC algorithm is that all robots are in sleep mode, and the A-STC algorithm can obtain coverage trajectories in all tested situations. In summary, the A-STC algorithm is complete for every instance, and it can find nearly optimal solutions in less time to obtain paths covering all reachable areas.

Table 3 Experimental results of the A-STC and DARP algorithms on the 16 instances

Instance size	Number of robots	Space size	Number of obstacles	Lower bound (s)	A-STC algorithm		DARP algorithm	
					Makespan (s)	Running time (s)	Makespan (s)	Running time (s)
Small	2	20×20	0	2.000×10^{2}	2.000×10^{2}	4.000×10^{-3}	2.000×10^{2}	5.000×10^{-3}
	7	30×30	0	1.285×10^{2}	1.320×10^{2}	1.800×10^{-2}	1.320×10^{2}	1.997×10^{-1}
	5	16×16	20	4.720×10^{1}	4.800×10^{1}	1.000×10^{-3}	4.800×10^{1}	1.700×10^{-2}
	7	30×40	164	1.480×10^{2}	1.760×10^{2}	4.200×10^{-2}	*	*
Medium	10	64×44	0	2.816×10^{2}	2.840×10^{2}	2.500×10^{-1}	2.840×10^{2}	5.004×10^{-1}
	15	64×64	0	2.730×10^{2}	2.760×10^{2}	1.100×10^{-1}	2.760×10^{2}	1.981×10^{0}
	20	64×64	0	2.048×10^{2}	2.080×10^{2}	2.220×10^{-1}	2.080×10^{2}	6.767×10^{0}
	10	64×48	320	2.752×10^{2}	2.800×10^{2}	8.200×10^{-2}	2.760×10^{2}	7.275×10^{-1}
	15	64×64	640	2.304×10^{2}	2.440×10^{2}	6.130×10^{-1}	2.320×10^{2}	1.151×10^{1}
	20	64×64	1272	1.412×10^{2}	1.640×10^{2}	2.860×10^{-1}	1.480×10^{2}	6.583×10^{0}
Large	40	80×80	0	1.600×10^{2}	2.040×10^{2}	1.016×10^{0}	1.600×10^{2}	1.342×10^{3}
	60	100×100	0	1.667×10^{2}	1.840×10^{2}	1.737×10^{0}	1.680×10^{2}	5.947×10^{2}
	80	100×100	0	1.250×10^{2}	1.440×10^{2}	1.126×10^{0}	*	*
	40	80×80	1204	1.299×10^{2}	1.520×10^{2}	5.750×10^{-1}	*	*
	60	100×100	1612	1.398×10^{2}	1.760×10^{2}	3.956×10^{0}	*	*
	80	100×100	1612	1.049×10^{2}	1.200×10^{2}	7.130×10^{-1}	*	*

$*$ means that the algorithm cannot return a result within the limited number of iterations

5 Conclusions and future work

In this paper, we have described the MCMP problem. An A-STC algorithm that contains an auction mechanism was proposed to manage multiple spanning trees. In the proposed algorithm, the spanning tree of every robot can be generated rapidly. The A-STC algorithm ensures the connectedness of each spanning tree, and tries to balance the workload among robots. The validity and adaptability of the A-STC algorithm to different environments and robot positions were verified by computational experiments. Comparative experiments with the GA and DARP algorithms showed that the A-STC algorithm has advantages in large complex configuration spaces and in terms of running time. In the future, we will investigate the MCMP problem with different robot coverage ranges. The different coverage ranges will mean that robots cannot auction vertexes in the same graph.

References

An V, Qu ZH, Roberts R, 2017. A rainbow coverage path planning for a patrolling mobile robot with circular sensing range. *IEEE Trans Syst Man Cybern Syst*, 48(8):1238-1254. https://doi.org/10.1109/TSMC.2017.2662623

Azpúrua H, Freitas GM, Macharet DG, et al., 2018. Multi-robot coverage path planning using hexagonal segmentation for geophysical surveys. *Robotica*, 36(8):1144-1166. https://doi.org/10.1017/S0263574718000292

Chakraborty A, Misra S, Sharma R, et al., 2017. Observability conditions for switching sensing topology for cooperative localization. *Unmann Syst*, 5(3):141-157. https://doi.org/10.1142/S2301385017400039

Choset H, 2001. Coverage for robotics—a survey of recent results. *Ann Math Artif Intell*, 31(1-4):113-126. https://doi.org/10.1023/A:1016639210559

Dias MB, Zlot R, Kalra N, et al., 2006. Market-based multirobot coordination: a survey and analysis. *Proc IEEE*, 94(7):1257-1270. https://doi.org/10.1109/JPROC.2006.876939

Di Franco C, Buttazzo G, 2016. Coverage path planning for UAVs photogrammetry with energy and resolution constraints. *J Intell Robot Syst*, 83(3-4):445-462. https://doi.org/10.1007/s10846-016-0348-x

Elango M, Nachiappan S, Tiwari MK, 2011. Balancing task allocation in multi-robot systems using K-means clustering and auction based mechanisms. *Expert Syst Appl*, 38(6):6486-6491. https://doi.org/10.1016/j.eswa.2010.11.097

Fang H, Lu SL, Chen J, et al., 2017. Coalition formation based on a task-oriented collaborative ability vector. *Front Inform Technol Electron Eng*, 18(1):139-148. https://doi.org/10.1631/FITEE.1601608

Gabriely Y, Rimon E, 2002. Spiral-STC: an on-line coverage algorithm of grid environments by a mobile robot. Proc IEEE Int Conf on Robotics and Automation, p.954-960. https://doi.org/10.1109/ROBOT.2002.1013479

Gabriely Y, Rimon E, 2003. Competitive on-line coverage of grid environments by a mobile robot. *Comput Geom*, 24(3):197-224. https://doi.org/10.1016/S0925-7721(02)00110-4

Galceran E, Carreras M, 2013. A survey on coverage path planning for robotics. *Robot Auton Syst*, 61(12):1258-1276. https://doi.org/10.1016/j.robot.2013.09.004

Gautam A, Murthy JK, Kumar G, et al., 2015. Cluster, allocate, cover: an efficient approach for multi-robot coverage. Proc IEEE Int Conf on Systems, Man, and Cybernetics, p.197-203.
https://doi.org/10.1109/SMC.2015.47

Hazon N, Kaminka GA, 2008. On redundancy, efficiency, and robustness in coverage for multiple robots. *Robot Auton Syst*, 56(12):1102-1114.
https://doi.org/10.1016/j.robot.2008.01.006

Kapanoglu M, Alikalfa M, Ozkan M, et al., 2012. A pattern-based genetic algorithm for multi-robot coverage path planning minimizing completion time. *J Intell Manuf*, 23(4):1035-1045.
https://doi.org/10.1007/s10845-010-0404-5

Kapoutsis AC, Chatzichristofis SA, Kosmatopoulos EB, 2017. DARP: divide areas algorithm for optimal multi-robot coverage path planning. *J Intell Robot Syst*, 86(3-4):663-680.
https://doi.org/10.1007/s10846-016-0461-x

Karapetyan N, Benson K, McKinney C, et al., 2017. Efficient multi-robot coverage of a known environment. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.1846-1852.
https://doi.org/10.1109/IROS.2017.8206000

Khamis A, Hussein A, Elmogy A, 2015. Multi-robot task allocation: a review of the state-of-the-art. In: Koubâa A, Martínez-de Dios J (Eds.), Cooperative Robots and Sensor Networks 2015. Springer, Cham.
https://doi.org/10.1007/978-3-319-18299-5_2

Khan A, Noreen I, Habib Z, 2017. On complete coverage path planning algorithms for non-holonomic mobile robots: survey and challenges. *J Inform Sci Eng*, 33(1):101-121.
https://doi.org/10.6688/JISE.2017.33.1.7

Kong Y, Zhang MJ, Ye DY, 2016. A group task allocation strategy in open and dynamic grid environments. In: Fukuta N, Ito T, Zhang M, et al., (Eds.), Recent Advances in Agent-Based Complex Automated Negotiation. Springer, Cham.
https://doi.org/10.1007/978-3-319-30307-9_8

Li GS, Chou WS, Yin F, 2018. Multi-robot coordinated exploration of indoor environments using semantic information. *Sci China Inform Sci*, 61(7):79201.
https://doi.org/10.1007/s11432-017-9336-x

Radmanesh M, Kumar M, Guentert PH, et al., 2018. Overview of path-planning and obstacle avoidance algorithms for UAVs: a comparative study. *Unmann Syst*, 6(2):95-118.
https://doi.org/10.1142/S2301385018400022

Rekleitis I, New AP, Rankin ES, et al., 2008. Efficient boustrophedon multi-robot coverage: an algorithmic approach. *Ann Math Artif Intell*, 52(2-4):109-142.
https://doi.org/10.1007/s10472-009-9120-2

Tang J, Zhu KJ, Guo HX, et al., 2018. Using auction-based task allocation scheme for simulation optimization of search and rescue in disaster relief. *Simul Model Pract Theor*, 82:132-146.
https://doi.org/10.1016/j.simpat.2017.12.014

Xin B, Gao GQ, Ding YL, et al., 2017. Distributed multi-robot motion planning for cooperative multi-area coverage. Proc 13th IEEE Int Conf on Control & Automation, p.361-366.
https://doi.org/10.1109/ICCA.2017.8003087

Yehoshua R, Agmon N, Kaminka GA, 2016. Robotic adversarial coverage of known environments. *Int J Robot Res*, 35(12):1419-1444.
https://doi.org/10.1177/0278364915625785