arXiv:2601.00580v1 [cs.RO] 2 Jan 2026

优先级感知多机器人覆盖路径规划

Kanghoon Lee、Hyeonjun Kim、Jiachen Li、Jinkyoo Park

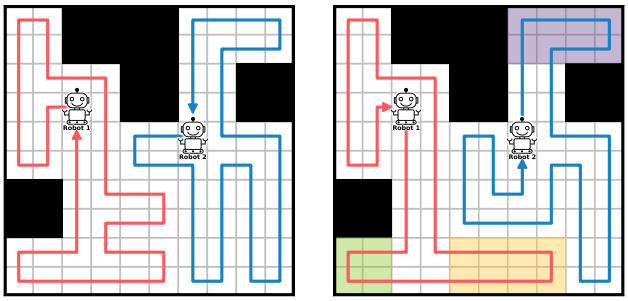
摘要—多机器人系统广泛用于需要在大型环境中进行高效协同的覆盖任务。在多机器人覆盖路径规划（MCPP）中，目标通常是通过生成互不重叠的路径实现全区域覆盖，同时最小化最大完工时间。然而，现有大多数方法假设各区域具有相同的重要性，因此在某些区域需要更快响应的场景中效果有限。我们提出了优先级感知多机器人覆盖路径规划（PA-MCPP）问题：将环境的一部分指定为带有相应权重的优先区域。其目标是按照字典序，最小化区域覆盖的总优先级加权延迟和整体最大完工时间。为此，我们提出了一种可扩展的两阶段框架，包括：（1）结合局部搜索的贪心区域分配，以及基于生成树的路径规划；（2）由Steiner树引导的剩余覆盖。涵盖多种场景的实验表明，与标准MCPP基线方法相比，我们的方法显著降低了优先级加权延迟，同时保持了具有竞争力的最大完工时间。敏感性分析进一步表明，该方法能够随机器人数量增加而良好扩展，并且可以通过调整优先级权重有效控制区域覆盖行为。

I. 引言

多机器人系统日益广泛地用于处理超出单个机器人能力范围的复杂时空任务 [1], [2], [3], [4]。并行运行能够加快任务完成速度，并使系统可扩展至规模庞大、结构复杂的环境。这类系统应用于多个领域，包括搜索与救援 [5], [6], 监视 [7], [8],监控 [9],仓储物流 [10], 运动/占用预测 [11], [12], [13], 以及物体运输 [14]。然而，要充分发挥多机器人系统的潜力，必须解决具有挑战性的协调问题，尤其是在任务目标涉及空间覆盖时 [15]。

此类结构化任务中的一个重要类别涉及覆盖，其中要求机器人协同访问或观测已知环境中的每个区域 [16]。在多机器人覆盖路径规划中，主要目标是通过生成协调且不重叠的路径，使机器人能够高效覆盖整个区域，从而最小化总体最大完工时间 [17], [18], [19]。在某些场景中，还会考虑减少转弯次数，以提升运动平滑性或能源效率 [20], [21]。尽管多机器人覆盖路径规划已得到广泛研究，但大多数方法都假设所有区域具有相同的重要性。当某些区域需要更快关注时，例如危险区域、关键资产或时间敏感的检查地点，这一假设便不再适用。这些区域的覆盖延迟

K. Lee 任职于韩国大田的韩国科学技术院（KAIST）。leehoon@kaist.ac.kr。H. Kim 任职于韩国首尔的韩国军事学院（KMA）。hyunjoon0605@kma.ac.kr。J. Li 任职于美国加利福尼亚州的加州大学河滨分校（UCR）。jiachen.li@ucr.edu。J. Park 任职于韩国科学技术院（KAIST）和 Omelet。jinkyoo.park@kaist.ac.kr。



(a) 多机器人覆盖路径规划　(b) 优先级感知多机器人覆盖路径规划

图 1. **多机器人覆盖路径规划与优先级感知多机器人覆盖路径规划的比较。**标准MCPP通过在机器人之间分配覆盖任务来最小化整体最大完工时间，而优先级感知多机器人覆盖路径规划还会考虑空间优先级，鼓励尽早完成高优先级区域的覆盖。右图中以绿色、黄色和紫色阴影区域标示优先区域。

可能会降低任务性能或安全性，凸显了在路径规划中纳入空间优先级的必要性。

为解决均匀覆盖的局限性，我们定义了优先级感知多机器人覆盖路径规划（PA-MCPP）问题，其中某些区域被赋予优先级权重，以反映其紧迫程度或价值。目标是规划机器人路径，在覆盖整个环境的同时，确保尽可能早地访问高优先级区域，这反映了安全关键任务中严格遵从优先级至关重要的情形 [22],[23]。具体而言，我们按字典序最小化两个指标：**(1)** 优先级加权延迟，即完全覆盖每个区域所需的时间乘以该区域的优先级权重；以及 **(2)** 最大完工时间，即最后一台机器人完成其覆盖任务的时间，如图 1 所示。由于现有多机器人覆盖路径规划方法缺乏先最小化延迟、再重新平衡剩余覆盖任务的机制，因此这一字典序目标使直接应用这些方法颇具挑战性。该目标源于实际需求：即使最大完工时间略有增加，也要优先覆盖关键区域。与加权和方法不同，字典序目标强制执行这种严格的优先级结构，确保优先完成覆盖。

为解决 PA-MCPP问题，我们提出了一个与字典序目标相一致的两阶段框架。第一阶段采用贪心启发式，将优先区域分配给机器人，以最小化优先级加权延迟，随后进行局部搜索优化。接着，每个机器人通过按区域构建并遍历生成树，依次覆盖分配区域，从而生成自身轨迹。第二阶段则在剩余区域上构建斯坦纳树，以确保全面覆盖，并根据先前的工作量，采用负载均衡方案将其分配给各机器人。最终路径整合了两个阶段的结果，既能尽早覆盖关键区域，也能高效完成整体覆盖。

总而言之，我们的主要贡献如下：

- 我们对 PA-MCPP问题进行了形式化，并设定了字典序目标：优先尽早覆盖关键区域，同时确保环境得到完整覆盖。

- 我们提出了一种可扩展的两阶段框架，将贪心区域分配与优先区域的局部搜索优化相结合，并通过 Steiner 树引导的剩余覆盖实现工作负载均衡。

- 我们在多种布局上验证了该方法。与多机器人覆盖路径规划基线方法相比，该方法显著降低了优先级加权延迟，同时保持了有竞争力的最大完工时间。

II. 相关工作

A. 多机器人覆盖路径规划（*MCPP*）

多机器人覆盖路径规划旨在为多个机器人生成协同路径，使其完整覆盖环境，同时尽量缩短最大完工时间并避免冗余。分治策略从早期的去中心化分区方法 [24]到近期的 DARP [25] 和 GM-VPC[26], ，均先将环境划分为均衡且互不重叠的区域，再计算覆盖路径。生成树覆盖（STC） [27], 是一种单机器人方法，它基于生成树生成连续的覆盖路径。其多机器人扩展方法包括 MSTC [16], 、MFC [28], 、AWSTC [17], 、MSTC*[18], 和 MIP-SRH [19]，通过构建生成树或森林来生成重叠较少的覆盖路径。为处理大规模机器人数量及实际的冲突消解问题，LS-MCPP [15] 结合局部搜索与路径冲突消解，以避免机器人间碰撞。TMC [20] 和 TMSTC* [21] 等转弯最小化方法可减少转弯，使运动更加平稳；另有研究探讨了基于局部搜索的单机器人优化方法 [29]。对于部分已知或动态环境，ConCPP [30] 等在线方法会在执行过程中逐步生成路径。以往研究侧重于工作负载均衡、避碰和运动效率，但假设所有区域的重要性相同。相比之下，我们提出的 PA-MCPP 在更高层次上针对区域分配进行局部搜索，并结合字典序目标，优先覆盖高权重区域。

B. 具有区域优先级的任务规划

具有区域优先级的任务规划适用于某些区域需要比其他区域更早得到关注的场景，而均匀覆盖方法无法处理这种情况。旅行维修工问题（TRP，[31]）及其加权变体和多仓库变体等经典模型，明确以最小化目标点到达时间的加权和为目标，确保高优先级位置能够得到及时服务 [32], [33]。除这些空间规划模型外，持续监测研究还通过信息衰减对时变优先级进行建模 [34], [35]。这些思想也延伸到了覆盖和巡检任务中，优先级地图或区域权重会引导智能体在任务执行过程中优先前往更有价值的区域 [22], [23], [36]。多机器人信息路径规划同样强调重新访问过时区域，以维持感知质量 [37]。类似地，在海上搜救（SAR）中，任务通常涉及在高概率区域内搜寻目标点根据漂移预测和环境约束，力求减少路径重叠和行驶时间 [5], [6]。尽管这些研究提升了对优先区域的响应能力，但无法确保完整覆盖。我们的方法通过字典序目标，同时确保全面覆盖和优先级。

III. 问题表述

我们考虑一个定义在二维网格环境上的 PA-MCPP 问题。该环境建模为无向图 $G = (V,\ E)$，其中每个节点 $v \in V$表示网格中的一个离散单元格。每个顶点 v 都被赋予一个遍历成本 $c(v)$。如果两个节点在水平方向或垂直方向上相邻，则它们之间存在一条边 $e \in E$ 。为便于基于STC的路径生成，我们引入一个由 G导出的超图表示 $H = (V_h,\ E_h)$。具体而言，将 V 中的每个 2×2 网格单元块收缩为 V_h中的一个超顶点；如果两个超顶点各自包含的网格单元中至少有一对相邻，则在 E_h 中连接它们的超边。该环境包含一组 n区域 $Z = \{Z_j\}_{j=1}^{n}$，其中每个区域 $Z_j \subseteq V$都关联一个正的优先级权重 $w_j > 0$，适用于所有 $j \in \{1, \ldots, n\}$。假设区域内的所有节点彼此连通。令 $R = \{r_i\}_{i=1}^{k} \subseteq V$ 表示 k 个机器人的初始位置，机器人按机器人集合 $I = \{1, 2, \ldots, k\}$编入索引。那么，一个 PA-MCPP 实例可表示为元组 $(G,\ Z,\ R,\ I)$。

对于每个机器人 $i \in I$，路径定义为节点序列 $\pi_i = (v_1,\ v_2,\ \ldots\ldots,\ v_{|\pi_i|})$，该路径的起点和终点均为机器人的初始位置，即 $v_1 = v_{|\pi_i|} = r_i$；并且每对连续节点 $(v_{j-1},\ v_j)$ 都是 E 中的一条边，对所有 $j = 2,\ 3,\ \ldots\ldots,\ |\pi_i|$ 均成立。令 $\pi_i^{(t)} = \{v_1, v_2, \ldots, v_t\}$ 表示机器人 i 截至时间步 $t \le |\pi_i|$ 访问过的节点集合。区域 Z_j 的覆盖时间记为 T_j，即至少一个机器人访问完该区域所有节点的最早时间步：

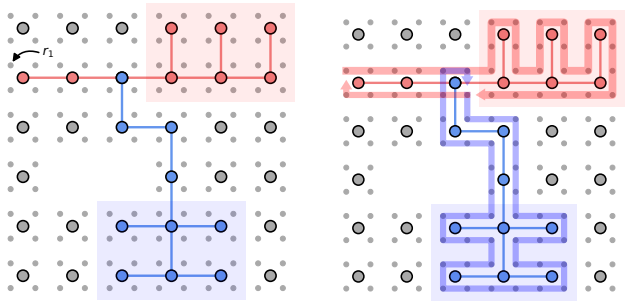
$$T_j = \min \left\{ t \mid Z_j \subseteq \bigcup_{i \in I} \pi_i^{(t)} \right\}, \quad \forall j \in \{1, \ldots, n\}. \tag{1}$$

PA-MCPP 的目标是找到一组机器人路径，使字典序目标函数最优；该目标函数按优先级排列多个准则，如果一个解在更高优先级的目标上表现更好，则该解优于另一个解：

$$\operatorname*{lex\,min}_{\{\pi_i\}_{i \in I}} \quad \sum_{j=1}^{n} w_j T_j,\ \max_{i \in I} |\pi_i| \tag{2}$$

$$\text{s.t.} \quad \bigcup_{i \in I} \pi_i = V. \tag{3}$$

第一个目标是最小化区域覆盖的优先级加权延迟总和。第二个目标是最小化最大完工时间，其定义为所有机器人中的最大路径长度。虽然加权和形式可以表达类似的权衡，但在不同环境中难以一致地设定其权重。如果不存在优先区域，PA-MCPP问题就会简化为标准MCPP [15]，为保持一致性，我们沿用其符号和结构。



(a) 分区树构建　　(b) 顺序树遍历

图2. **单机器人路径规划策略概览**。红色和蓝色阴影区域分别表示第一个和第二个分配区域（Z_1，Z_2），以及它们对应的生成树（T_1，T_2）。

IV. 方法

我们提出一种两阶段方法来解决PA-MCPP问题，该方法明确基于STC范式。整体方法与字典序目标一致：第一阶段最小化区域覆盖的优先级加权延迟总和，第二阶段确保完整覆盖环境，如图3所示。构建多个覆盖树与最小最大树覆盖问题 [38], 相关，但我们构建这些树时优先考虑优先区域，然后生成覆盖整片区域的树。我们首先在IV-A节介绍按给定顺序覆盖多个分配区域的单机器人路径生成方法，随后在IV-B节将其扩展到多机器人情形，并采用贪心和局部搜索进行区域分配。最后，在IV-C节中，通过构建剩余覆盖树并将其与优先级轨迹拼接，覆盖剩余区域。为便于说明，我们用符号 u 表示用于路径规划的底层图中的顶点；在本文中，它是超图 $H = (V_h,\ E_h)$中的一个超顶点 $u \in V_h$ 。超顶点的成本是其四个组成网格单元成本的平均值，而超边的成本是其所连接的两个超顶点成本的平均值 [19]。

A. 多分区单机器人路径规划

如图2所示，我们介绍单个机器人按照预定义顺序覆盖多个区域的路径规划过程。机器人从初始位置出发，依次访问各个区域。为了覆盖一个区域，它首先通过最短路径从当前位置连接到目标分区中最近的点，然后构建一棵覆盖整个区域的生成树。对于每个区域，这条路径与生成树将合并为一棵树。构建这些树后，机器人按顺序遍历它们，以生成一条完整路径。当一个区域被完全覆盖后，它会直接移动到下一个区域，从而确保各区域之间覆盖有序且连续。

分区树构建：算法1概述了构建一系列生成树的流程，使机器人能够从其初始位置 r出发，按顺序覆盖一组分配区域$\mathbf{Z} = [Z_j]_{j=1}^{m}$。 [第1-2]行 算法首先初始化一个空列表以存储生成的树，并将锚点节点设为起始位置。 [行 对于序列中的每个区域，机器人

算法1：分区树构建

输入： 起始节点 u^{start}，
分配区域列表 $\mathbf{Z} = [Z_j]_{j=1}^{m}$
输出： 生成树列表 $\mathbf{T} = [T_j]_{j=1}^{m}$

$\mathbf{T} \leftarrow [\,]$
$u^{\text{锚点}} \leftarrow u^{\text{开始}}$
对每个 $Z_j \in \mathbf{Z}$ **执行**
　$u^{\text{entry}} \leftarrow$求最小值 $_{u \in Z_j}$ 距离(u, u^{anchor})
　$\widetilde{T} \leftarrow$ 从 u^{anchor} 到 u^{entry}的最短路径
　$\widehat{T} \leftarrow$ 求 Z_j的最小生成树
　$\mathbf{T} \leftarrow \mathbf{T} \parallel [\widetilde{T} \cup \widehat{T}]$
　$u^{\text{锚点}} \leftarrow \text{argmin}_{u \in \widetilde{T}} \textsc{Dist}(u, u^{\text{entry}})$
返回 T

算法 2：顺序树遍历（STT）

输入： 分配区域列表 $\mathbf{Z} = [Z_j]_{j=1}^{m}$，
生成树列表 $\mathbf{T} = [T_j]_{j=1}^{m}$
输出： 完全覆盖路径 π

$\hat{\pi} \leftarrow$树遍历$(\mathbf{T}[0])$
如果 $|\mathbf{Z}| = 1$ **则**
　返回 $\hat{\pi}$
否则
　$\pi \leftarrow [\,]$
　$C \leftarrow \emptyset$
　对每个 $v \in \hat{\pi}$ **执行**
　　如果 $C =$**Z**$[0]$ **，则**
　　　$\pi \leftarrow$ STT$(\mathbf{Z}[1:], \mathbf{T}[1:])$
　　　$\pi \leftarrow \pi \parallel \hat{\pi}$
　　else
　　　$\pi \leftarrow \pi \parallel [v]$
　　如果 $v \in$**Z**$[0]$ **，则**
　　　$C \leftarrow C \cup \{v\}$
返回 π

通过使用迪杰斯特拉算法计算从锚点到所有节点的最短距离，找出区域 Z_j 中距离当前锚点v^{anchor} 最近的节点 [39]，并将该节点选为入口点 v^{entry}。 [行 随后将从锚点到入口点的最短路径存储为 T_i。[行 然后，使用克鲁斯卡尔算法 [40], 在Z_j 上构建内部覆盖树 T_i ，得到一棵最小成本生成树。 [行 将 7] T_i 和 T_i 合并为一棵树，然后将其追加到列表**T**中。 [行 将锚点节点更新为连接路径 T_j 上距离入口点 v^{entry}最近的节点，使机器人能够高效地开始向下一个区域移动。 [行 最终输出是一个有序的生成树列表，每棵树对应一个分配区域，如图2a所示。

顺序树遍历：构建好按区域划分的生成树列表**T**后，机器人通过遍历这些树生成完整覆盖路径，具体如算法 2 所示。 [行 1]遍历从第一棵树开始，采用深度优先策略，遵循基于 STC 的覆盖路径规划问题中的标准流程。 [行 2–3] 如果只有一个区域，则直接返回遍历路径 $\hat{\pi}$ 。[行 5–6] 否则，算法初始化覆盖缓冲区 C ，并迭代地将已访问节点追加到最终

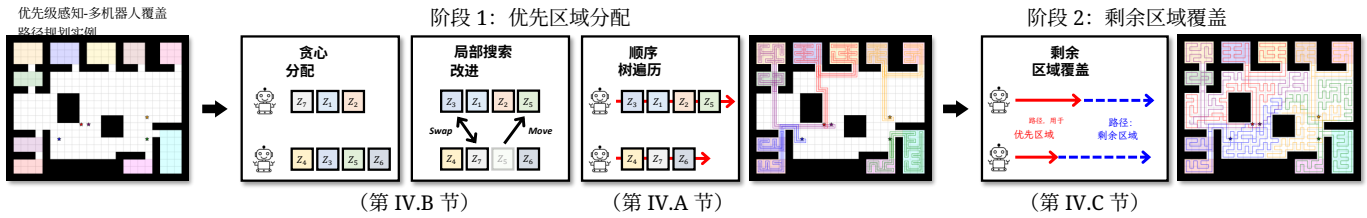


图 3. **PA-MCPP 算法概述**。该实例包含带有优先区域及其对应权重的地图。阶段 1 采用贪心分配将优先区域分配给机器人，再通过局部搜索优化分配结果，随后规划各区域内的遍历序列。阶段 2 覆盖剩余区域以确保完整覆盖，并根据先前的分配结果平衡工作负载，从而最小化最大完工时间。

路径 π中。 [行 8–10] 当前区域中的所有节点访问完毕后，算法会递归调用自身，处理剩余区域及其对应的生成树。返回的子路径 $\tilde{\pi}$ 已在预先构建的树中包含通往下一区域的连接；跳过当前节点 v可避免重复访问，并确保平滑衔接。[行 12] 否则，节点会像往常一样追加到路径中。 [行 13–14] 在整个过程中，缓冲区 C 会持续更新，以监控当前区域内的覆盖进度。 [行 15] 最终输出为生成树 **T**的完全覆盖路径π ，如图 2b 所示。

B. 多台机器人优先区域分配 *bots*

此步骤的目标是将区域及其访问顺序分配给多台机器人，以最小化公式（2）中定义的优先级加权延迟目标函数。我们将其表述为多仓库 k-TRP 的一种变体，其中每台机器人的初始位置充当一个仓库，每个区域则作为一个待访问目标 [33]。目标是最小化所有机器人访问各区域的优先级加权到达时间之和。然而，正如先前研究[33]所指出的，对于大规模场景，通过整数规划（IP）求得精确最优解在计算上难以实现。为此，我们采用一种结合贪心分配与局部搜索优化的实用方法。

贪心区域分配：算法 3 概述了将有序区域分配给多台机器人的流程，以最小化总优先级加权延迟。 [第 1-3] 行：算法首先为每台机器人初始化一个空序列$\mathbf{Z}_i$ 和遍历时间 T_i ，并将所有区域标记为未分配。 [第 4-5] 行：对于每个区域，使用克鲁斯卡尔算法计算区域内部的行进成本 $c(Z_j)$。 [第 6-10] 行：通过 Dijkstra 算法计算每个机器人-区域对的仓库到区域遍历成本 $c(i,\ Z_j)$，同时将区域间遍历成本 $c(Z_j,\ Z_{j'})$ 乐观地设为两个区域中任意节点对之间的最小距离。 [第 11-28] 行：随后，贪心分配循环迭代选择使增量优先级加权延迟最小的机器人-区域对（$i,\ Z_j$），同时考虑行进成本和区域内部成本。如果机器人尚未被分配过区域，则使用从其仓库出发的成本；否则，考虑从其上一个已分配区域出发的成本。将选定区域 Z_{j^*} 追加到机器人 i^* 的序列$\mathbf{Z}_i$中，并更新其遍历时间 T_{i^*}。将选定区域从未分配集合 $\mathcal{U}$中移除。此过程持续进行，直到所有区域均已分配。 [行29] 最终输出为所有机器人的有序区域分配$\mathbf{Z}_i$

算法 3： 贪心区域分配

输入： 机器人集合 I 及其初始位置 $\{r_i\}_{i\in I}$，区域 $Z=\{Z_j\}_{j=1}^{n}$，优先级权重 $\{w_j\}_{j=1}^{n}$
输出： 有序区域分配 $\{\mathbf{Z}_i\}_{i\in I}$

1 $\mathcal{U}\leftarrow Z$ // 未分配区域
2 **对每个** $i\in I$ **执行**
3 $\quad\mathbf{Z}_i\leftarrow()$; $T_i\leftarrow 0$
// 计算遍历成本
4 **对每个** $Z_j\in Z$ **执行**
5 $\quad c(Z_j)\leftarrow$ Z_j 的最小生成树成本
6 **对每个** $i\in I,\ Z_j\in Z$ **执行**
7 $\quad c(i,\ Z_j)\leftarrow$ 最小值$_{v\in Z_j}$ Dist$(r_i,\ v)$
8 **对每个** $Z_j,\ Z_{j'}\in Z$ 满足 $j<j'$ **执行**
9 $\quad c(Z_j,\ Z_{j'})\leftarrow$ 最小值$_{v\in Z_j,\ v'\in Z_{j'}}$ Dist$(v,\ v')$
10 $\quad c(Z_{j'},\ Z_j)\leftarrow c(Z_j,\ Z_{j'})$
// 贪心分配循环
11 **当** $\mathcal{U}\neq\emptyset$ **时执行**
12 $\quad\Delta_{\min}\leftarrow+\infty$
13 $\quad$**对每个** $Z_j\in\mathcal{U}$ **执行**
14 $\quad\quad$**对每个** $i\in I$ **执行**
15 $\quad\quad\quad$**if** $\mathbf{Z}_i=\emptyset$ **那么**
16 $\quad\quad\quad\quad\Delta\leftarrow w_j\cdot(T_i+c(i,\ Z_j)+c(Z_j))$
17 $\quad\quad\quad$**else**
18 $\quad\quad\quad\quad Z_{\text{end}}\leftarrow$ $\mathbf{Z}_i$ 中的最后一个区域
19 $\quad\quad\quad\quad\Delta\leftarrow w_j\cdot(T_i+c(Z_{\text{end}},\ Z_j)+c(Z_j))$
20 $\quad\quad\quad$**如果** $\Delta<\Delta_{\min}$，**那么**
21 $\quad\quad\quad\quad\Delta_{\min}\leftarrow\Delta$; $(i^*,\ Z_{j^*})\leftarrow(i,\ Z_j)$
22 $\quad\mathbf{Z}_{i^*}\leftarrow\mathbf{Z}_{i^*}\|[Z_{j^*}]$
23 $\quad$**如果** $|\mathbf{Z}_{i^*}|=1$ **则**
24 $\quad\quad T_{i^*}\leftarrow T_{i^*}+c(i^*,Z_{j^*})+c(Z_{j^*})$
25 $\quad$**else**
26 $\quad\quad Z_{\text{prev}}\leftarrow$ 上一个区域位于$\mathbf{Z}_{i^*}$
27 $\quad\quad T_{i^*}\leftarrow T_{i^*}+c(Z_{\text{prev}},Z_{j^*})+c(Z_{j^*})$
28 $\quad\mathcal{U}\leftarrow\mathcal{U}\setminus\{Z_{j^*}\}$
29 **返回** $\{\mathbf{Z}_i\}_{i\in I}$

机器人，其作为后续单机器人路径规划的输入，详见 IV-A节。

局部搜索优化：获得有序区域序列 $\{\mathbf{Z}_i\}_{i\in I}$ 的初始分配后，我们使用局部搜索优化分配，以进一步降低总优先级加权延迟。在每次迭代中，我们通过应用以下两种扰动之一来定义局部邻域：

1) **移动算子**：随机选择一个区域 Z_j，将其从某个机器人的序列中移出i，并将其插入另一个随机机器人的序列中的随机位置 i'。

2) **交换算子**：随机选择两个区域 Z_j 和 $Z_{j'}$

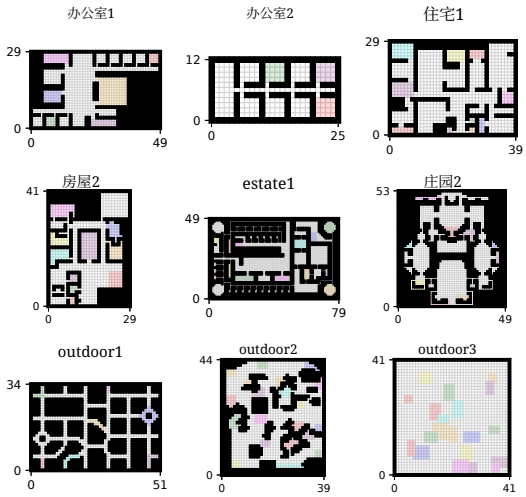


图 4. **PA-MCPP 地图实例**。黑色、白色和彩色方块分别表示障碍物、普通地形和优先区域。不同颜色表示不同区域。

这些区域可能分配给不同的机器人，并直接交换它们的分配关系和相应位置。

每次修改后，我们都会重新计算新分配方案的总目标函数值。如果新配置降低了成本，就接受该配置；否则将其舍弃。该过程通过迭代探索当前解附近的邻域，逐步改进解。

C. 使用斯坦纳树进行剩余区域覆盖

每台机器人通过遍历在超图$H = (V_h, E_h)$中构建的树 $\{\mathbf{T}_i\}_{i\in I}$ 完成优先区域覆盖后，我们确定已覆盖超顶点的集合。该集合表示为$\mathcal{V}_{\text{covered}} = \bigcup_{i\in I} V_h(\mathbf{T}_i)$，其中 $V_h(\mathbf{T}_i)$表示包含在机器人i的覆盖树中的超顶点。剩余未覆盖区域为 $\mathcal{V}_{\text{res}} = V_h \setminus \mathcal{V}_{\text{covered}}$。为高效覆盖 $\mathcal{V}_{\text{res}}$，我们在超图 H上构建斯坦纳树 T_S ，其终端集合为剩余超顶点的集合 $\mathcal{V}_{\text{res}}$。斯坦纳树是终端集合上的最小生成树；如果纳入其他非终端顶点能够降低总连接成本，则也可以将这些顶点纳入其中。

然后，我们在 T_S 上生成一条遍历路径，并使用MSTC* [18]将其分配给各机器人。分配的目标是最小化$\max_{i\in I}\left(C_i^{(1)}+C_i^{(2)}\right)$，其中 $C_i^{(1)}$ 是第一阶段的成本， $C_i^{(2)}$ 是分配后的剩余成本。这种极小极大分配直接考虑每台机器人的前期工作量，从而确保工作负载均衡并将最大完工时间降至最低。最终，将剩余路径接在优先级区域路径之后，形成每台机器人的最终路径。

V. 实验

在以下实验中，我们旨在验证四个关键假设。（1）我们提出的方法能否有效解决PA-MCPP问题？（2）我们的方法能否随着机器人数量的增加而良好扩展？（3）通过调整分配给区域的优先级权重，我们能否控制所提出方法的行为？（4）我们的贪心区域分配结合局部搜索，能否生成足够高质量的分配方案？需要注意的是，在所有实验中，我们均使用

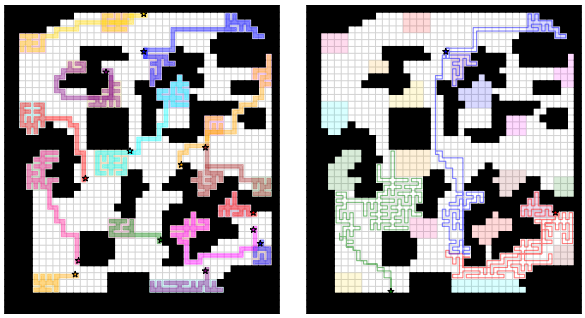


图5. **区域分配与机器人覆盖路径的可视化**。左：匹配的优先区域及其对应的生成树和分配路径。右：机器人1（红色）、机器人3（绿色）和机器人15（蓝色）的完全覆盖路径，展示了先覆盖优先区域，再覆盖剩余区域。

随机的初始机器人位置，并在互不重叠的常规地形上进行10次试验，以确保结果的鲁棒性和公平性，实验方案遵循先前工作的实验流程 [15]。所有实验均在一台配备AMD Ryzen Threadripper 3970X 32核处理器的Linux工作站上进行。

A. 性能比较

本小节评估所提出的 PA-MCPP 是否能降低优先级加权区域覆盖延迟，同时保持具有竞争力的最大完工时间。作为比较，我们选取了几种成熟的多机器人覆盖算法作为基线方法：MFC [28] 和 MSTC* [18]。尽管这些方法并非专为优先级目标设计，但可作为有用的最大完工时间参照。我们通过对它们生成的路径进行后处理，计算区域覆盖延迟。纳入这些基线方法有助于验证，我们的方法在优化区域覆盖延迟的同时，不会显著损害最大完工时间。所有基线方法的实现均基于公开可用的代码，以确保公平性和可复现性[1]。

我们在九种不同的地图布局上评估所提出的方法，这些布局旨在代表广泛的场景，包括庄园、住宅、办公室和户外环境，如图4所示。每张地图配置了不同数量的机器人和区域，如表 I 所示。所有区域的优先级权重均设为 1。地图实例的第三行中，顶点权重从均匀分布 $\mathcal{U}(0.8,1.2)$ 中随机抽样，以引入变化，并更真实地反映非均匀环境。表 I 展示了不同地图实例上的基线性能比较。在所有实例中，与 MSTC* 相比，PA-MCPP 的延迟平均降低了 $62.5 \pm 14.4\%$，同时最大完工时间保持相近，仅增加 $9.7 \pm 9.4\%$。总体而言，PA-MCPP 实现了预期的权衡：大幅缩短完成高优先级区域所需的延迟时间，同时仅适度增加最大完工时间。

图5展示了优先级感知覆盖分配以及部分机器人的最终路径。在图5a中，每个优先区域都被合理分配给机器人，以尽量减少行程延迟；随后，通过依次遍历相应的生成树来生成覆盖路径。

[1]https://github.com/reso1/MSTC 星标

表 I 不同地图实例上的基线性能比较

实例	地图规模	# of Zone	# of 机器人	加权	区域覆盖延迟 (↓) MFC	MSTC*	PA-MCPP	最大完工时间 (↓) MFC	MSTC*	PA-MCPP
办公室1	49×29	6	10	✗	431.1 ± 26.0	443.9 ± 50.5	283.7 ± 5.5	121.7 ± 10.2	96.7 ± 3.9	127.9 ± 5.2
办公室2	25×12	3	3	✗	104.2 ± 13.1	109.5 ± 17.5	65.7 ± 4.8	64.8 ± 8.8	47.7 ± 1.9	54.1 ± 3.2
住宅1	39×29	10	5	✗	1182.3 ± 152.9	1138.3 ± 226.8	339.9 ± 15.0	226.5 ± 19.8	183.1 ± 3.6	193.2 ± 3.9
房屋2	29×41	10	10	✗	1065.8 ± 145.8	1066.2 ± 85.4	455.5 ± 17.1	213.6 ± 13.6	157.8 ± 3.3	168.5 ± 5.4
estate1	79×49	30	5	✗	7623.7 ± 834.9	7480.0 ± 772.8	1964.3 ± 85.6	566.5 ± 57.1	433.1 ± 14.1	508.4 ± 13.5
庄园2	49×53	10	10	✗	1026.4 ± 157.0	995.1 ± 102.3	248.6 ± 17.6	200.6 ± 27.1	150.8 ± 5.5	153.7 ± 6.9
outdoor1	51×34	7	10	✓	443.9 ± 52.8	446.7 ± 46.0	158.9 ± 6.3	125.3 ± 9.7	88.9 ± 3.5	89.6 ± 3.7
户外2	39×44	20	15	✓	1793.9 ± 75.0	1698.7 ± 111.6	558.2 ± 22.6	171.5 ± 10.8	128.2 ± 3.5	134.4 ± 4.6
户外3	41×41	20	20	✓	7982.4 ± 1002.8	7690.2 ± 391.4	1616.2 ± 35.3	661.3 ± 79.7	545.1 ± 4.4	570.1 ± 5.5
相较于 MSTC 的平均提升* (%)					-1.1 ± 3.3	-	62.5 ± 14.4	-31.2 ± 6.0	-	-9.7 ± 9.4

图5b展示了三个具有代表性的机器人的完整覆盖轨迹：机器人1（红色）、机器人3（绿色）和机器人15（蓝色）。机器人3未被分配任何优先区域，因此会立即前往其剩余覆盖区域。机器人1只有一个优先区域，该区域紧邻其起始位置；它会先覆盖该区域，然后继续前往剩余区域。机器人15被分配了两个优先区域，并依次完成对它们的覆盖，然后转向剩余覆盖区域。这些结果表明，我们的区域分配策略不仅考虑了距离，还兼顾了每个机器人的初始位置和优先区域的大小，从而减少行程开销，并使高优先级区域得以提前完成，同时不会显著影响整体覆盖效率。

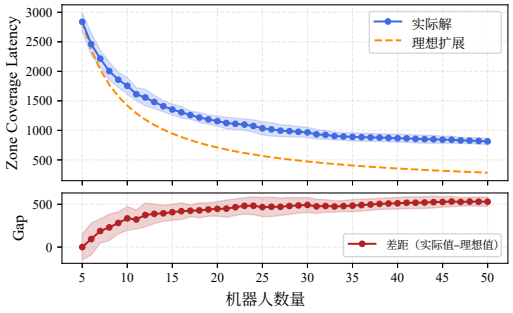
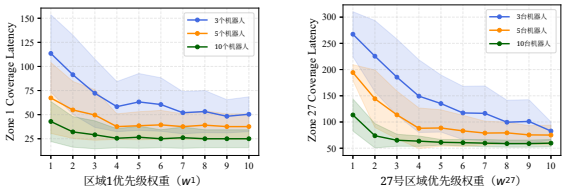


图 6. **机器人数量的敏感性分析**。上图：区域覆盖延迟与机器人数量的关系，以及理想扩展基线。下图：实际解与理想基线之间的差距。



(a) 1号区域（最小区域）　(b) 27号区域（最大区域）

图 7. **优先级权重敏感性分析**。不同优先级权重和机器人数量下的区域覆盖延迟。

B. 敏感性分析

在本小节中，我们验证该方法能否随着机器人数量的增加而有效扩展，以及能否通过调整区域的优先级权重来控制其行为。除非另有说明，所有实验均在estate1上进行，默认配置为5个机器人。

图6展示了区域覆盖与延迟与机器人数量。我们观察到，随着机器人数量增加，总体延迟会降低，这表明我们的算法能够有效利用额外的机器人来提高覆盖速度。然而，将性能与理想扩展基线（计算方式为$y^5_{\text{sol}} \times {}^{5}/_{\text{机器人数量}}$，其中 y^5_{sol} 表示实际解在五台机器人下的区域覆盖延迟）进行比较时，我们注意到，机器人越多，性能差距越大。该理想扩展假设能够实现完美并行化，且不会产生额外的协调开销，这在实践中很难做到。差距不断扩大表明，尽管我们的方法在更大规模下仍然有效，但由于机器人协调和区域分配的复杂性增加，其效率略有下降。

图7评估了提高优先级权重能否加快区域覆盖。我们针对地图中面积最小的区域（1号区域）和面积最大的区域（27号区域），使用不同数量的机器人进行了敏感性分析。对于1号区域，我们观察到，随着其优先级权重增加，覆盖时间迅速缩短，但很快趋于稳定，表明超过某一权重后，收益递减。相比之下，对于面积大得多的27号区域，趋于稳定的过程更为缓慢，而且提高优先级权重带来的收益更大程度上取决于机器人数量。此外，这些结果表明，通过调整优先级权重，我们的方法能够灵活控制区域覆盖行为，从而验证了该算法能够适应用户指定的偏好。

C. 区域分配策略评估

在本小节中，我们验证所提出的贪心区域分配结合局部搜索，是否能够为多台机器人生成高质量的分配方案。我们在 estate1 布局上进行实验，使用五台机器人，与第V-B节相同。图8左侧的图展示了不同初始化方法和算子选择策略下，区域覆盖延迟随局部搜索迭代次数的变化。我们将贪心初始化与随机初始化进行比较，并评估两种算子选择方案：余弦调度和静态调度。在余弦调度中，我们将周期设为总迭代次数的 10%，使每个算子的选择概率在每个周期内于 1.0 和 0.0 之间平滑振荡。在静态调度中，移动或交换算子均以相同概率随机选取。贪心初始化始终能提供比随机初始化更好的起点。随着

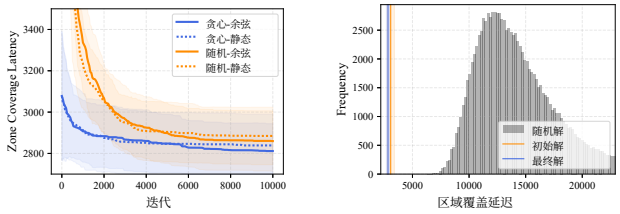


图8. **区域分配优化的结果**。左：不同初始化和算子选择策略下，区域覆盖延迟随局部搜索迭代次数的变化。右：随机分配的区域覆盖延迟分布，竖线标示初始贪心解以及局部搜索后的最终解。

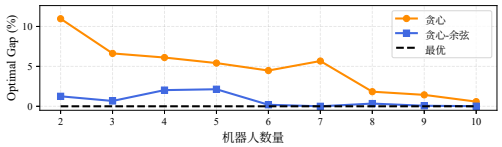


图9. **与区域分配最优IP解的比较**。初始贪心解、局部搜索后的最终解以及outdoor1上最优IP基线方法的区域覆盖延迟最优性差距随机器人数量的变化。

局部搜索迭代次数增加，解会进一步改进并最终收敛。然而，贪心初始化始终比随机初始化产生更好的最终解。此外，对于两种初始化方法，余弦调度都比静态调度取得更好的结果。

图8中的右图展示了随机分配的区域覆盖延迟分布，说明随机方案与我们的方案之间存在显著的性能差距。竖线表示初始贪心分配以及局部搜索优化后最终解的延迟。这些结果证明了将强初始分配与局部搜索相结合的有效性，也支持了我们的方法能够找到高质量区域分配。

图9将所提方案与最优IP基线方法 [33]进行了比较。结果表明，将贪心初始化与局部搜索相结合可实现接近最优的性能，证明了所提出的算法的有效性。随着机器人数量增加，最优性差距缩小，因为每个机器人负责的区域更少。在这种情况下，简单的贪心分配几乎达到最优，因为大多数机器人被分配到最近的区域。由于求解IP公式的计算复杂度，该比较仅在区域数量相对较少的地图（例如outdoor1）上进行。

D. 机器人工作负载均衡评估

为了定量评估工作负载均衡情况，我们计算分配给各个机器人的路径长度的最大值与均值之比（MMR），其定义为最大个体工作负载与所有机器人平均工作负载的比值，数值越接近 1 表示均衡性越好。表II报告了不同

团队规模。在所有案例中，PA-MCPP 的 MMR 值与 MSTC* 相当，差异（Δ）处于统计波动范围内。对于规模更大的团队（例如 20 个机器人），方差有所增加（0.06±0.23），这表明有些试验

表II 工作负载均衡的最大值与均值之比比较

机器人数量	最大值与均值之比（↓）		
	MSTC*	PAMCPP	差值 (Δ)
5	1.01 ± 0.01	1.01 ± 0.0	-0.01 ± 0.01
10	1.04 ± 0.04	1.05 ± 0.03	0.01 ± 0.05
15	1.19 ± 0.13	1.20 ± 0.08	0.01 ± 0.15
20	1.21 ± 0.15	1.27 ± 0.18	0.06 ± 0.23

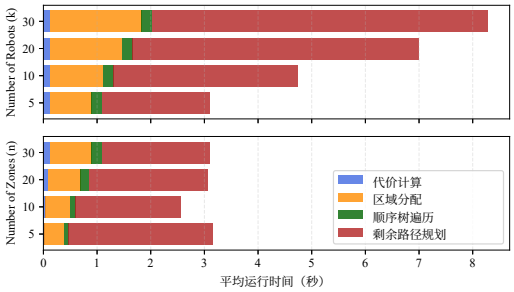


图 10. **按算法阶段划分的平均运行时间分解**。上：区域数量固定为 30 时，不同机器人数量 k 下的平均执行时间。下：机器人数量固定为 5 时，不同区域数量 n 下的平均执行时间。

呈现出的工作负载均衡程度较低。这种波动源于 MSTC* 的固有特性：随着机器人数量增加，要保持工作负载完全均衡会变得越来越困难。与这一基线行为相比，我们的集成仅带来了微小的额外不均衡。总体而言，PA-MCPP 在实现主要性能提升的同时，保持了与 MSTC* 相当的工作负载均衡。

E. 计算时间分析

图 10 展示了所提出的算法在四个主要计算阶段中的平均运行时间分解。在所有测试配置中，剩余路径规划占据了计算时间的主要部分，而且随着机器人数量增加，其占比更加显著。成本计算、区域分配和顺序树遍历的运行时间则随着 k 或 n 的增加而逐渐增长，这反映了额外的成对计算和分配复杂度。不过，剩余路径规划的耗时不会随着 n 单调变化。当 n 较小时（例如 $n = 5$），初始覆盖后未覆盖节点的数量相对较多，因此剩余路径规划步骤所需的处理量远高于 $n = 10$ 或 20 时的情况。对于较大的 n （20–30），这种影响趋于饱和，规划时间也大致相当。总体而言，机器人数量对计算时间的影响大于区域数量，主要原因是分支因子较大，且分配与遍历的复杂度更高。

六、结论与未来工作

本文提出了 PA-MCPP，这是一种兼顾优先区域重要性与整体覆盖效率的两阶段框架。该方法将贪心区域分配与局部搜索优化相结合，随后覆盖剩余区域以确保完整性。实验表明，该方法能够有效降低优先级加权延迟，同时保持具有竞争力的最大完工时间，并通过敏感性分析展示了其可扩展性

和可控性。然而，当前框架的一个主要局限是局部更新锚点策略。由于该策略采用短视设计，可能导致区域连接并非最优；此外，区域与机器人之间的一对一分配，也可能妨碍其在包含少量大型优先区域的环境中的可扩展性。未来，我们计划探索更灵活的分配方式，允许多台机器人共同负责同一区域。

参考文献

[1] X. Zhang, H. Qin, F. Wang, Y. Dong, and J. Li, “Lamma-p：使用语言模型驱动的PDDL规划器实现可泛化的多智能体长时域任务分配与规划”，载于*2025*年*IEEE*国际机器人与自动化会议（*ICRA*），IEEE，2025年。[2] H. Kim, K. Lee, J. Park, J. Li, and J. Park, “基于人类隐式偏好的策略微调，用于无人水面艇集群的多智能体强化学习”，载于*2025*年*IEEE/RSJ*智能机器人与系统国际会议（*IROS*），IEEE，2025年，第18 653–18 659页。[3] X. Gao, R. Xu, J. Li, Z. Wang, Z. Fan, and Z. Tu, “STAMP：可扩展、任务与模型无关的协同感知”，载于第十三届国际学习表征会议，2025年。[4] J. Li, X. Liu, B. Li, R. Xu, J. Li, H. Yu, and Z. Tu, “CoMamba：借助状态空间模型实现实时协同感知”，载于*IEEE/RSJ*智能机器人与系统国际会议（*IROS*），2025年。[5] B. Ai, M. Jia, H. Xu, J. Xu, Z. Wen, B. Li, and D. Zhang, “利用强化学习进行海上搜救覆盖路径规划”，海洋工程，第241卷，第110 098页，2021年。[6] J. Wu, L. Cheng, S. Chu, and Y. Song, “基于深度强化学习的落水人员海上搜救自主覆盖路径规划算法”，海洋工程，第291卷，第116 403页，2024年。[7] S. Velhal, S. Sundaram, and N. Sundararajan, “用于周界防御的分散式多机器人时空多任务分配方法”，*IEEE*机器人汇刊，第38卷，3096，2022 5，3085，第–页。[8] S. Bajaj, S. D. Bopardikar, E. Torng, A. Von Moll, and D. W. Casbeer, “锥形环境中的多车辆周界防御”，*IEEE*机器人汇刊，第40卷，第1439–1456页，2024年。[9] A. B. Asghar, S. Sundaram, and S. L. Smith, “多机器人持续监测：在充电约束下最小化延迟和机器人数量”，*IEEE*机器人汇刊，2024年。[10] A. Bolu and ¨O. Korc¸ak, “多机器人智能仓库的自适应任务规划”，*IEEE Access*，第9卷，第27 346–27 358页，2021年。[11] Z. Wang, Y. Wang, Z. Wu, H. Ma, Z. Li, H. Qiu, and J. Li, “CMP：利用多智能体通信进行协同运动预测”，*IEEE*机器人与自动化快报，2025年。[12] L. Wang, M.-A. Lavoie, S. Papais, B. Nisar, Y. Chen, W. Ding, B. Ivanovic, H. Shao, A. Abuduweili, E. Cook, 等, “迈向可部署且可泛化自主性的运动预测趋势：回顾与展望”，*arXiv*预印本 *arXiv:2505.09074*，2025年。[13] Y. Wang, X. Huang, X. Sun, M. Yan, S. Xing, Z. Tu, and J. Li, “UniOcc：自动驾驶占用预测与预报的统一基准”，载于*IEEE/CVF*国际计算机视觉会议论文集（*ICCV*），IEEE，2025年。[14] Z. Wang, S. Singh, M. Pavone, and M. Schwager, “无需对等通信的多架四旋翼无人机三维协同物体运输”，载于*2018*年*IEEE*国际机器人与自动化会议（*ICRA*），IEEE，2018年，第1064–1071页。[15] J. Tang, Z. Mao, and H. Ma, “带路径冲突消解的网格大规模多机器人覆盖路径规划”，*IEEE*机器人汇刊，2025年。[16] N. Hazon and G. A. Kaminka, “多机器人覆盖中的冗余、效率与鲁棒性”，载于*2005*年*IEEE*国际机器人与自动化会议论文集，IEEE，2005年，第735–741页。[17] W. Dong, S. Liu, Y. Ding, X. Sheng, and X. Zhu, “面向分散式飞行机器人的人工加权生成树覆盖算法”，*IEEE*自动化科学与工程汇刊，第17卷，第4期，第1689–1698页，2020年。[18] J. Tang, C. Sun, and X. Zhang, “MSTC*：物理约束下的多机器人覆盖路径规划”，载于*2021*年*IEEE*国际机器人与自动化会议（*ICRA*），IEEE，2021年，第2518–2524页。

[19] J. Tang and H. Ma，“用于时间最优多机器人覆盖路径规划的混合整数规划与高效启发式算法”，《IEEE机器人与自动化快报》，第8卷，第10期，第6491–6498页，2023年。[20] I. Vandermeulen, R. Groß, and A. Kolling，“转弯次数最少的多机器人覆盖”，载于2019年国际机器人与自动化会议（ICRA），IEEE，2019年，第1014–1020页。[21] J. Lu, B. Zeng, J. Tang, T. L. Lam, and J. Wen，“Tmstc*：一种用于减少多机器人覆盖转弯次数的路径规划算法”，《IEEE机器人与自动化快报》，第8卷，5282，2023 8，5275，第–页。[22] T. Kusnur, S. Mukherjee, D. M. Saxena, T. Fukami, T. Koyama, O. Salzman, and M. Likhachev，“用于持续多无人机覆盖并实现全局冲突消解的规划框架”，载于《野外与服务机器人：第12届国际会议论文集》，施普林格，2021年，第459–474页。[23] H. Song, J. Yu, J. Qiu, Z. Sun, K. Lang, Q. Luo, Y. Shen, and Y. Wang，“有限续航条件下的多无人机灾害环境覆盖规划”，载于2022年国际机器人与自动化会议（ICRA），IEEE，2022年，第10 760–10 766页。[24] M. Jager and B. Nebel，“面向协作清洁机器人的动态分散式区域划分”，载于2002年IEEE国际机器人与自动化会议论文集，IEEE，第4卷，2002年，第3577–3582页。[25] A. C. Kapoutsis, S. A. Chatzichristofis, and E. B. Kosmatopoulos，“DARP：用于最优多机器人覆盖路径规划的区域划分算法”，《智能与机器人系统杂志》，第86卷，第3期，第663–680页，2017年。[26] V. G. Nair and K. Guruprasad，“GM-VPC：一种利用广义Voronoi划分实现已知空间多机器人覆盖的算法”，Robotica，第38卷，第5期，第845–860页，2020年。[27] Y. Gabriely and E. Rimon，“移动机器人对连续区域进行基于生成树的覆盖”，《数学与人工智能年刊》，第31卷，第1期，第77–98页，2001年。[28] X. Zheng, S. Jain, S. Koenig, and D. Kempe，“多机器人森林覆盖”，载于2005年IEEE/RSJ智能机器人与系统国际会议，IEEE，2005年，第3852–3857页。[29] D. Krupke，“考虑转弯成本的近似最优覆盖路径规划”，载于2024年算法工程与实验研讨会论文集（ALENEX），工业与应用数学学会，2024年，第118–132页。[30] R. Mitra and I. Saha，“在线并发多机器人覆盖路径规划”，arXiv预印本 arXiv:2403.10460，2024年。[31] F. Afrati, S. Cosmadakis, C. H. Papadimitriou, G. Papageorgiou, and N. Papakostantinou，“旅行维修工问题的复杂性”，《RAIRO理论信息学与应用》，第20卷，第1期，第79–87页，1986年。[32] A. Blum, P. Chalasani, D. Coppersmith, B. Pulleyblank, P. Raghavan, and M. Sudan，“最小延迟问题”，载于第二十六届ACM计算理论年会论文集，1994年，第163–171页。[33] M. E. Bruni, S. Khodaparasti, I. Mart´ınez-Salazar, and S. Nucamendi-´Guill´en，“多仓库k旅行维修工问题”，《优化快报》，第16卷，第9期，第2681–2709页，2022年。[34] X. Lan and M. Schwager，“高斯随机场中感知机器人的周期性持续监测轨迹规划”，载于2013年IEEE国际机器人与自动化会议，IEEE，2013年，第2415–2420页。[35] R. Han, Y. Wen, L. Bai, J. Liu, and J. Choi，“面向智能交通系统的信息年龄感知无人机部署”，《IEEE智能交通系统汇刊》，第23卷，第3期，第2705–2715页，2021年。[36] S. Poudel and S. Moh，“面向高效且负载均衡多无人机作业的优先级感知任务分配与路径规划”，《车辆通信》，第42卷，第100 633页，2023年。[37] A. Singh, A. Krause, C. Guestrin, W. Kaiser, and M. Batalin，“多机器人信息丰富路径的高效规划”，载于第二十届国际人工智能联合会议论文集，系列IJCAI’07，美国加利福尼亚州旧金山，2007年，第2204–2211页。[38] G. Even, N. Garg, J. K¨onemann, R. Ravi, and A. Sinha，“图的极大极小树覆盖”，《运筹学快报》，第32卷，第4期，第309–315页，2004年。[39] E. W. Dijkstra，“关于图的两个相关问题的札记”，载于《Edsger Wybe Dijkstra：生平、工作与遗产》，2022年，第287–290页。[40] J. B. Kruskal，“关于图的最短生成子树与旅行商问题”，《美国数学学会会刊》，第7卷，第1期，第48–50页，1956年。