**目 录**

[1问题表述 1](#_Toc74772803)

[2问题分析 1](#_Toc74772804)

[3解题思路 2](#_Toc74772805)

[3.1基于Voronoi划分的任务预分配 2](#_Toc74772806)

[3.2任务重分配 3](#_Toc74772807)

[3.3勘探路径的求解 3](#_Toc74772808)

[3.4重分配变动元素数量求解 4](#_Toc74772809)

[3.5子图不连通情况分析 5](#_Toc74772810)

# 1问题表述

1.拓扑地图模型

本文首先对侦察地图进行构建，用侦察区域的集合，其中为第个侦察节点，为任务终点。因此，可以用如下无权无向图来表示侦察地图,其中表示无向图的边，表示和相连。

2.航空器模型

本文用表示3个航空器的集合，使用表示航空器在时刻的位置，其中，表明航空器在每一时刻都在任一侦察节点上。

定义航空器的路径，则航空器的勘探总路程为。为定义航空器的任务集合，这里首先引入表示时刻未被侦察节点组成的集合，其中表示初始时刻，满足如下更新方式：

 (公式1)

则，勘探总时间为。

基于上述定义，本次勘探任务可表述为以下优化问题：

 (公式2)

其中为比例系数。

# 2问题分析

考虑到题目要求侦察区域必须覆盖全部区域，且耗时最长的航空飞行器和耗时最短的航空飞行器时间之差不能超过耗时最短航空飞行器用时的30%。基于此，对侦察地图的节点数和连通性做如下分析。

1.侦察节点数分析

现设三架航空器飞行时间分别为且，则有，即，但由于，故若则有，即侦察区域节点总数的取值范围如下。

2.侦察地图连通性分析

由于三架航空器要侦察完所有区域，且要求回到同一目标终点，故该侦察地图一定是连通的，即从任一侦察节点出发总能到达其余所有侦察节点。

# 3解题思路

第1节中的优化问题(公式2)是一个典型的NP问题，难以在多项式时间内找到最优解。因此，本文的目标是在与(公式2)相同的约束条件下，寻找一个次优解题策略。

考虑到题目要求终点必须是集合中的最后一个节点，即航空器在探测过程中不能经过。因此，本文在进行任务预分配与重分配的过程中使用了忽略侦察节点和与其相连接的边的子图，直至。虽然忽略终点的子图在一定程度上可能破坏原图的连通性，但本文将先按照子图连通的情况处理，并在3.5节给出子图不连通情况下的处理策略。

## 3.1基于Voronoi划分的任务预分配

Voronoi划分原理：由几何原理知，两个节点的分界线为该两点之间的垂直平分线，可将全平面分为两个半平面。当节点数大于两个时，全平面被多条垂直平分线划分为多个凸多边形区域，每个区域中有且仅含一个节点，这种由多点将平面划分的图称为泰森多边形，又名Voronoi图。

本文将Voronoi图理论应用于拓扑图结构，代表了节点间的一种邻近关系，首先定义距离为从到的最短路径。故可根据每个航空器的初始节点将航空器的预分配任务空间划分为3个子区域。

 (公式3)

进一步定义航空器的预分配任务区域如下：

 (公式4)

其中表示区域和的公共边，即，由此可得对于不同的航空器，各航空器的预分配任务区域不重叠。

## 3.2任务重分配

由于3.1节的预分配过程没有考虑勘探时间的约束，因此可能会存在不满足约束的情况，故仍需考虑对任务区域的重分配。

考虑到在重分配过程中任一都有可能存在元素的变动，现假设变动的元素数量为（其中，表示添加元素，表示删除元素，且，其求解将在3.4节给出），由此来确定当时，需要附加节点的集合。

定义航空器需要重新分配的侦察节点集合如下：

 (公式5)

则航空器预分配剩余侦察节点集合，满足。则有所有需要重新分配的侦察节点的集合，可得如下：

 (公式6)

其中满足。

## 3.3勘探路径的求解

航空器的路径主要包含三部分，一部分是由预分配剩余侦察节点集合生成的，一部分是根据附加节点的集合生成的，最后一部分是根据和生成的。其求解步骤如下：

* 以为起点，对集合中元素在图中的位置采用最小生成树（算法伪代码如表1）的方式进行遍历，得到第一部分路径，设其终点为。

表1：最小生成树算法伪代码

|  |
| --- |
| **MST-PRIM** |
| **For** each          **While**    **For** each  **If**    **Decrease-Key** |

* 以为起点，依次选取附加节点集合中距离最近的点作为终点，并根据航空器的当前位置通过Dijkstra算法（算法伪代码如表2）寻找最短路径，生成路径，设其终点为。

表2：Dijkstra算法伪代码

|  |
| --- |
| **Dijkstra** |
| Initialize a set  Initialize a priority queuecontaining  **For**        **While**    **Addto**  **For** each  where  **If** |

* 以为起点，为终点，通过Dijkstra算法寻找最短路径，生成路径。

**注释1**：采用最小生成树在搜索时优先搜索距离起点最近的点，因此在设置合理时，能保证各航空器的路径不会出现交叉的情况。

## 3.4重分配变动元素数量求解

首先对预分配的任务区域进行排序，这里设，其中下标分别代表集合中元素的个数最大、中等和最小航空器的索引。则有求解过程如下：

1. 任取且满足如下约束：

 (公式7)

1. 利用初步生成任务集合和路径。
2. 判断和是否满足如下约束，若不满足则返回第1步进行重新选择，若满足则保存其结果。

 (公式8)

4. 对上述保存的结果进行对比，选择最优的任务集合、路径和其对应的。

## 3.5子图不连通情况分析

本文在上述的处理中仅考虑了子图连通的情况，现对子图不连通的情况进行分析。子图不连通时有以下两种情况：

1. 子图分为两个部分

该情况若有解，则只可能是一部分有一架航空器，一部分有两架航空器。则只有一架航空器的部分必须搜索完其所有节点，即按最小生成树的方式对路径进行规划。另一部分则可以按照3.1~3.4节的方式进行求解。并在最后分别用Dijkstra算法规划出到终点的路径即可。

2. 子图分为三个部分

该情况若有解，则只可能是每一部分均只有一架航空器，则该情况下每架航空器只需要搜索完自己部分的节点即可，按照最小生成树的方式进行路径规划，并在最后分别用dijkstra算法规划出到终点的路径即可。