**武汉大学**

**高级算法课程报告**

**无人机配送路径规划问题**

专 业 名 称 ：网络空间安全

课 程 名 称 ：高级算法

学 生 学 号 ：2023202210166

学 生 姓 名 ：陶镝霄

二○二四年六月

### 1. 问题描述

无人机可以快速解决最后10公里的配送。在一个区域中，共有j个配送中心，任意一个配送中心有用户所需要的商品，其数量无限，同时任一配送中心的无人机数量无限。该区域同时有k个卸货点（无人机只需要将货物放到相应的卸货点即可），假设每个卸货点会随机生成订单，一个订单只有一个商品，但这些订单有优先级别，分为三个优先级别（用户下订单时，会选择优先级别，优先级别高的付费高）：

* 一般：3小时内配送到即可；
* 较紧急：1.5小时内配送到；
* 紧急：0.5小时内配送到。

我们将时间离散化，也就是每隔t分钟，所有的卸货点会生成订单（0-m个订单），同时每隔t分钟，系统要做成决策，包括：

1. 哪些配送中心出动多少无人机完成哪些订单；

2. 每个无人机的路径规划，即先完成那个订单，再完成哪个订单，...，最后返回原来的配送中心；

注意：系统做决策时，可以不对当前的某些订单进行配送，因为当前某些订单可能紧急程度不高，可以累积后和后面的订单一起配送。

目标：一段时间内（如一天），所有无人机的总配送路径最短

约束条件：满足订单的优先级别要求

假设条件：

1. 无人机一次最多只能携带n个物品；

2. 无人机一次飞行最远路程为20公里（无人机送完货后需要返回配送点）；

3. 无人机的速度为60公里/小时；

4. 配送中心的无人机数量无限；

5. 任意一个配送中心都能满足用户的订货需求；

### 2. 数学建模

#### 2.1 符号说明

1. 表示无人机启动成本
2. 表示无人机单位距离的飞行成本
3. 表示无人机晚于配送时间的单位时间惩罚成本
4. M表示配送中心数量
5. N表示配送中心和卸货点数量，n表示节点集合：（配送中心：1，2，···，J；卸货点：J+1，J+2，···，N）；
6. 表示无人机k到达配送中心i的时间（）
7. K表示每次决策需使用的无人机数，m表示无人机集合（m=1，2，3，···，K）；
8. D表示无人机最大航程；
9. 表示到的距离，
10. 表示第j个配送中心的t优先级订单的数量
11. 表示无人机的最大载重
12. 决策变量：

#### 2.2 数学模型

目标函数：

约束：

1. 配送中心约束：所有无人机均由配送中心出发，完成配送任务后返回配送中心
2. 卸货点流量平衡：进出无人机数相等
3. 重量约束：每辆无人机的装载量不能超过其最大携带限制：
4. 无人机飞行距离约束：每辆无人机配送距离不超过最大航程：
5. 所需无人机数约束：
6. 飞行时间约束：
7. 时间约束：到达当前节点的时间=到达前一个节点的时间+前一个节点到当前节点的飞行时间
8. 时间惩罚：无人机晚于规定时间到达会产生时间惩罚：

### 3. 算法设计

#### 3.1 算法流程

对于本问题，可拆分为两个阶段进行：1. 首先将有订单的卸货点分配给配送中心，转换成单个配送中心问题；2. 在对每一个配送中心的无人机配送路径进行路线优化。

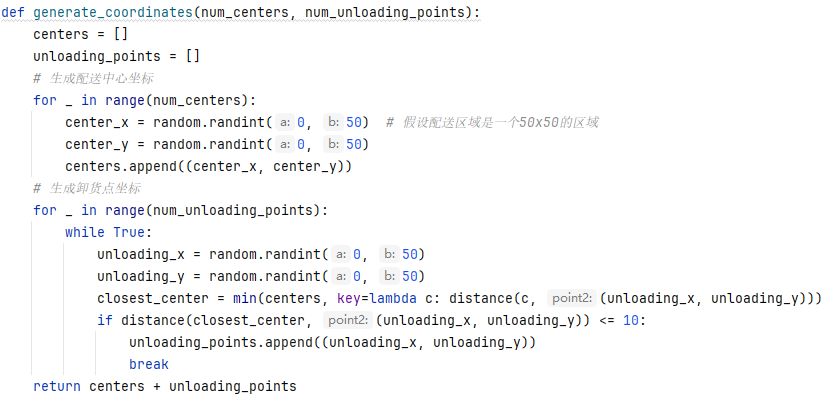
针对第一个问题，可以将卸货点分配给最近的配送中心负责配送订单。而第二个问题，则使用粒子群算法混合遗传算法解决。

粒子的编码以卸货点（编号为1，2，3…）为自然数编码（不包含配送中心0），生成一个序列。在解码时，以载重和飞行距离约束下采用贪婪策略为卸货点分配无人机（即依次将卸货点分配给无人机，直到无人机无法再装下下一个卸货点的订单需求）。

采用基于遗传算法交叉算子进行粒子群的更新，采用顺序交叉算子，对惯性因子w、自我认知因子c1、社会认知因子c2则以w/(w+c1+c2)，c1/(w+c1+c2)，c2/(w+c1+c2)的概率接受粒子本身、当前最优解、全局最优解交叉的父代之一（即按概率选择其中一个作为父代，不加权）。

#### 3.2 算法实现

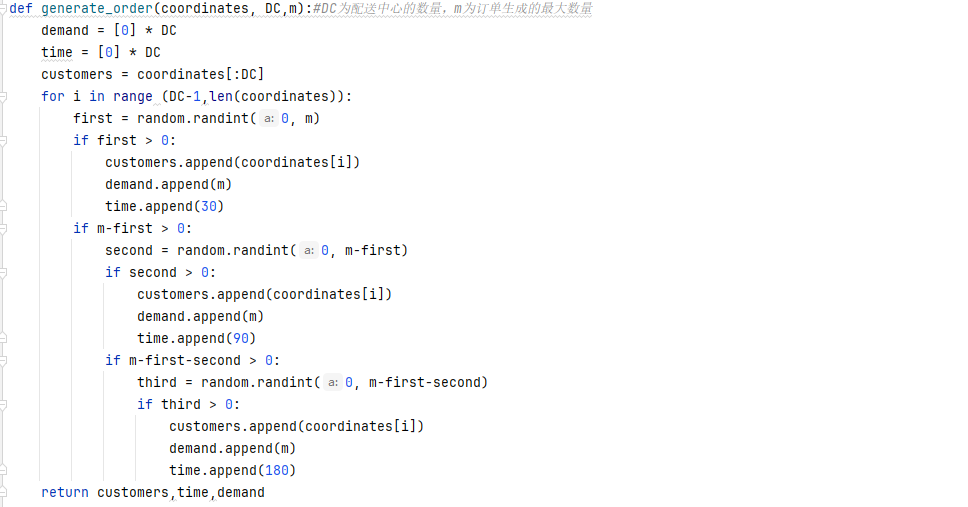
##### 3.2.1 生成配送中心与卸货点坐标



随机生成m个配送中心，对于每个配送中心，随机生成一个x坐标和一个y坐标，这些坐标的取值范围都在0到50之间（假设配送区域是一个50x50的正方形区域）。

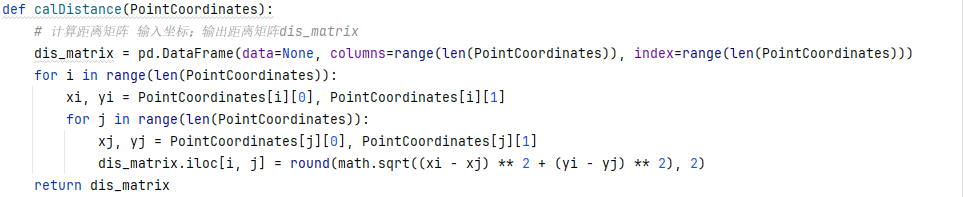
随机生成n-m个卸货点，对于每个卸货点，随机生成一个x坐标和一个y坐标，这些坐标的取值范围都在0到50之间（假设配送区域是一个50x50的正方形区域），同时检查每个卸货点与其最近配送中心的距离是否小于等于10。小于等于10表明该卸货点合法，反之不合法，重新生成。

##### 3.2.2 生成订单

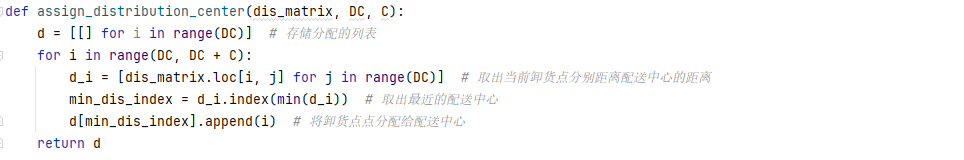


为每一个卸货点生成3个级别的订单，订单总量不超过m。并且将同等级同卸货点的订单进行合并。

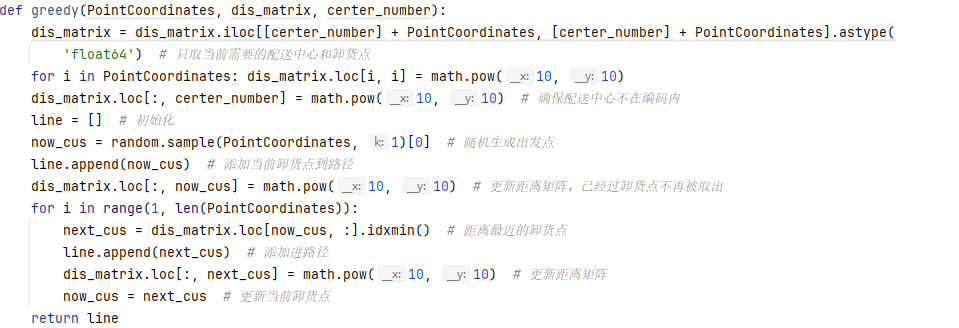
##### 3.2.3 计算点的距离矩阵



##### 3.2.4 将卸货点分配给各配送中心

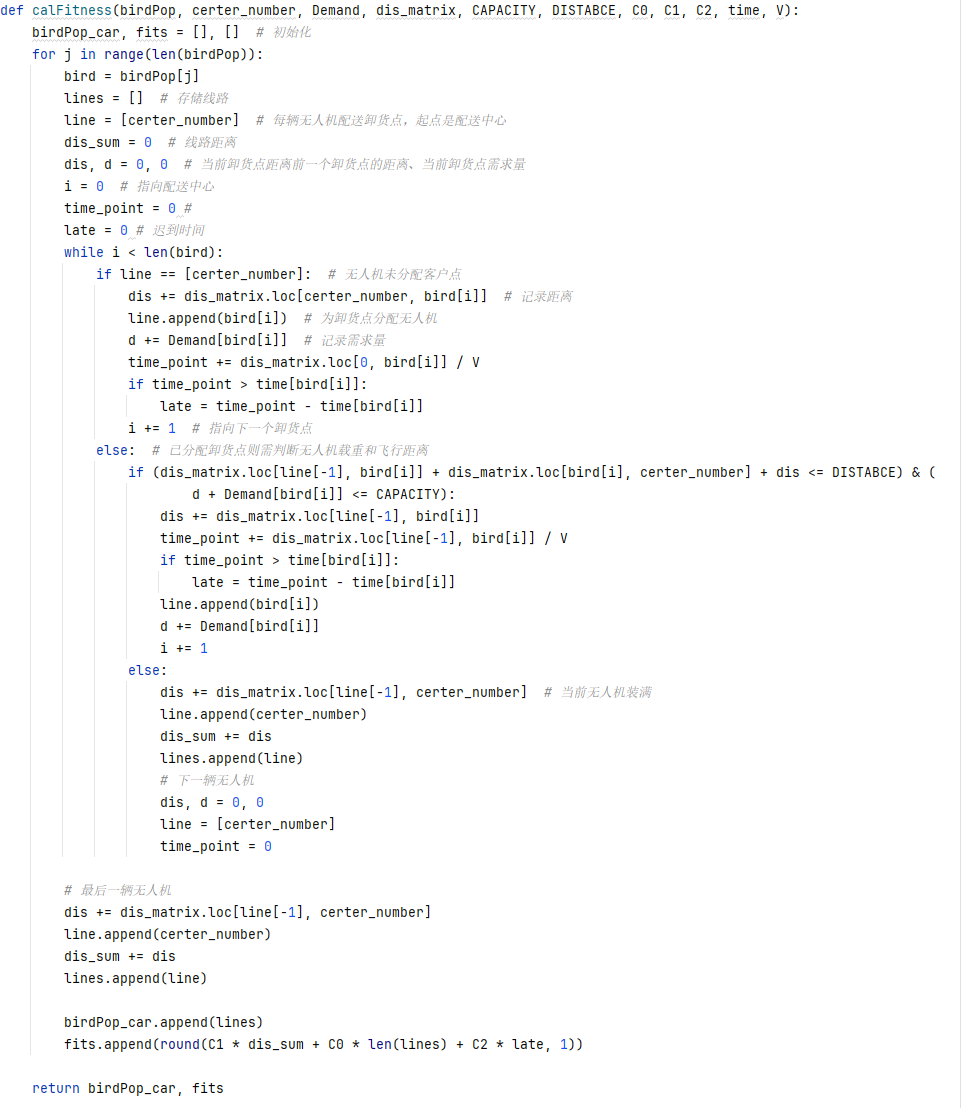


##### 3.2.5 构造初始解



首先，随机选择一个卸货点作为出发点，随后通过查找距离矩阵，找到距离当前节点最近的下一个卸货点。将经过的点在距离矩阵dis\_matrix中的对应列的值设置为一个较大的数，表示这些点已经被访问过，不再被取出。直至遍历完所有的卸货点。

##### 3.2.6 适应函数



1. 初始化路径列表 birdPop 和适应度列表 fits。

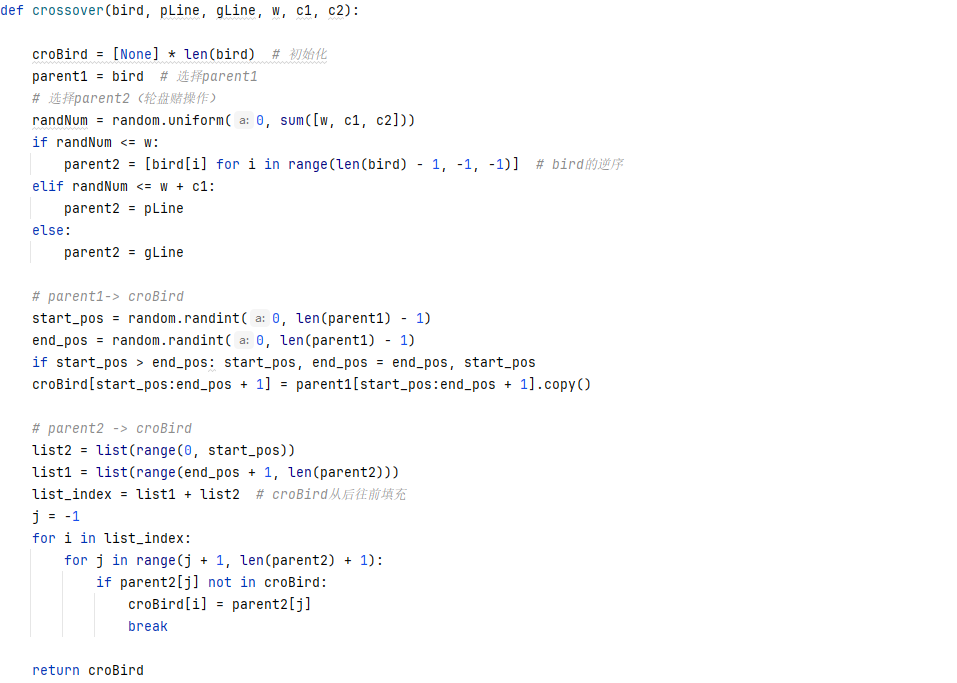
2. 对于每条路径：

1. 初始化线路列表 lines，用于存储分配多辆无人机后的线路；当前线路 line，起点是配送中心；线路距离 dis\_sum；初始化当前卸货点距离前一个卸货点的距离 dis 和当前卸货点订单需求量 d；当前时间点 time\_point 和迟到时长 late。
2. 遍历路径bird的节点，进行以下循环：
   1. 如果线路 line 中只包含配送中心，表示无人机未分配卸货点：
      * 计算当前配送点到配送中心的距离，并累加到距离 dis 上。
      * 将当前卸货点bird[i] 加入线路 line。
      * 累加当卸货点 bird[i] 的需求量到 d 上。
      * 根据无人机的速度计算时间点，并与卸货点的时间窗口进行比较，如果超过时间窗口则计算迟到时间 late。
      * 遍历下一个卸货点。
   2. 若线路line已经分配了卸货点，需要判断无人机载重和飞行距离是否满足条件：

* 判断当前卸货点 bird[i] 到最后一个卸货点的距离加上当前卸货点到配送中心的距离（确保返航）再加上距离 dis 是否小于等于无人机最大行驶距离，并且当前卸货点的需求量加上 d 是否小于等于无人机最大载重。
  + - 如果满足条件，将当前卸货点 bird[i] 加入线路 line。更新距离 dis 和需求量 d。根据无人机的速度计算时间点，并与卸货点的时间窗口进行比较，如果超过时间窗口则计算迟到时间 late。遍历下一个卸货点。
    - 如果不满足条件，表示当前无人机装满，需要将配送中心加入新的线路 line，并将距离 dis 加到线路距离 dis\_sum 上。并将前一条线路 line 添加到线路列表 lines 中。重置距离 dis 和需求量 d 为 0。重置线路 line 为只包含配送中心。重置时间点 time\_point 为 0。
  1. 对于最后一辆无人机：
* 将最后一个卸货点到配送中心的距离加到距离 dis 上。
* 将配送中心加入线路 line。
* 将距离 dis 加到线路距离dis\_sum上。
* 将线路 line 添加到线路列表 lines 中。
  1. 将分无人机后的线路列表 lines 添加到 birdPop 中。
  2. 计算适应度值 fit，采用以下公式：C1 \* dis\_sum + C0 \* len(lines) + C2 \* late，其中 C1、C0、C2 是权重参数，用于平衡距离、线路数量和迟到时间的重要性。并将适应度值 fit 添加到适应度列表 fits 中。

1. 返回分无人机后的路径列表 birdPop和适应度列表 fits。

##### 3.2.7 交叉操作更新粒子群



1. 初始化交叉后的粒子列表 croBird，长度与输入粒子列表 bird 相同。
2. 选择 bird 作为第一个父代 parent1。
3. 根据概率选择第二个父代 parent2：
4. 生成一个随机数 randNum，范围在 0 到惯性因子、自我认知因子和社会认知因子之和之间。
5. 如果 randNum小于等于惯性因子 w，则选择将 bird 逆序作为 parent2。
6. 如果 randNum大于惯性因子w, 小于等于惯性因子加上自我认知因子 w + c1，则选择当前最优解pLine 作为 parent2。
7. 否则，选择全局最优解gLine 作为 parent2。
8. 将 parent1 中的一段路径随机复制到 croBird 中：
   * 生成两个随机数 start\_pos 和 end\_pos，范围在 0 到 parent1 的长度之间。
   * 如果 start\_pos 大于 end\_pos，则交换两个值。
   * 将 parent1 中从 start\_pos 到 end\_pos 的路径段复制到 croBird 对应的位置上。
9. 将 parent2 中剩余的路径填充到 croBird 中：
   * 根据交叉操作的规则，将 parent2 中未出现在 croBird 中的路径按照逆序插入到 croBird 的空缺位置中。
10. 返回交叉后的粒子列表 croBird。

##### 3.2.8 主函数



1. 定义无人机参数，包括最大容量（CAPACITY）、最大行驶距离（DISTANCE）、C0、C1、C2和速度（V）。
2. 定义粒子群参数，包括粒子数量（birdNum）、惯性因子（w）、自我认知因子（c1）和社会认知因子（c2）。
3. 定义其他参数，包括迭代次数（iterMax）、配送中心个数（DC）、卸货点数量（C）、每次配送的最大载重量（m）和时间间隔（t）。
4. 生成配送中心和卸货点的坐标。
5. 在总时间不超过1440分钟的条件下进行循环。
6. 在每次循环中，重新设置粒子群参数和其他参数。
7. 生成订单，包括卸货点的坐标、到达时间和需求量。
8. 计算城市间距离矩阵。
9. 将卸货点分配给配送中心。
10. 对每个配送中心进行迭代优化。

a. 使用贪婪算法构造初始解。

b. 计算种群适应度，包括分配无人机和计算适应度值。

c. 初始化全局最优值（gBest）、当前最优值（pBest）以及全局最优解（gLine）和当前最优解（pLine）。

d. 进行迭代优化，包括交叉操作和更新最优值和最优解。

e. 记录每个配送中心的最优路径顺序和最优值。

1. 更新总时间。

#### 3.3 运行结果

##### 3.3.1 参数设置

无人机参数

* CAPACITY = 100 *# 无人机最大容量*
* DISTABCE = 20 *# 无人机最大行驶距离*
* C0 = 5 *# 无人机启动惩成本*
* C1 = 1 *#无人机单位距离的飞行成本*
* C2 = 10 *#无人机晚于配送时间的单位时间惩罚成本*
* V = 60 *# 无人机的速度*

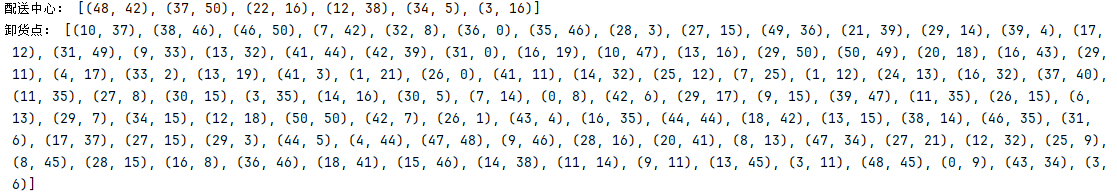
其他参数

* bestfit = [] *# 记录每代最优值*
* DC = 6 *# 配送中心个数*
* C = 100 *# 卸货点数量*
* time\_total = 0
* m = 10 *#每个卸货点单次生成的订单量的最大值*
* t = 180 *# 生成订单的时间间隔*

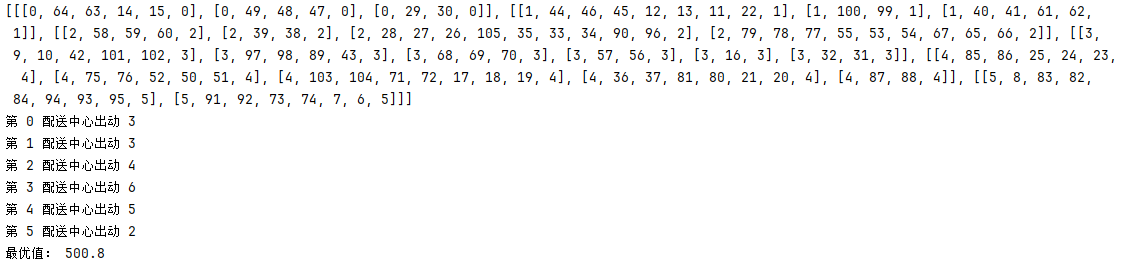
粒子群参数

* birdNum = 30 *# 粒子数量*
* w = 0.2 *# 惯性因子*
* c1 = 0.4 *# 自我认知因子*
* c2 = 0.4 *# 社会认知因子*
* pBest, pLine = 0, [] *# 当前最优值、当前最优解，（自我认知部分）*
* gBest, gLine = 0, [] *# 全局最优值、全局最优解，（社会认知部分）*

##### 3.3.2 区域节点坐标



##### 3.3.3 单次订单决策



##### 3.3.4 最终结果

