# 2020 年湖南省研究生数学建模竞赛

# 竞赛承诺书

我们仔细阅读了湖南省研究生数学建模竞赛的竞赛规则。

我们完全明白，在竞赛开始后参赛队员不能以任何方式（包括电话、电子邮件、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛规则的，如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们郑重承诺，严格遵守竞赛规则，以保证竞赛的公正、公平性。如有违反竞赛规则的行为，我们将受到严肃处理。

我们授权湖南省研究生数学建模竞赛组委会，可将我们的论文以任何形式进行公开展示（包括进行网上公示，在书籍、期刊和其他媒体进行正式或非正式发表等）。

我们参赛选择的题号是（从组委会提供的试题中选择一项填写）： B

我们的参赛报名号为（如果组委会设置报名号的话）： 202018001010

所属学校（请填写完整的全名）： 国防科技大学

参赛队员（打印并签名）：1. 刘心溥

2. 李铭典

3. 刘月玲

指导教师或指导教师组负责人 （打印并签名）：

日期： 2020 年 11 月 30 日

---

评阅编号（由组委会评阅前进行编号）：

评阅编号：
（由组委会填写）

# 2020年湖南省高校第五届研究生数学建模竞赛

# 编 号 专 用 页

评阅记录：

| | | | | | | |
|---|---|---|---|---|---|---|
| 评阅人 | | | | | | |
| 备注 | | | | | | |

**（请勿改动此页内容和格式。此编号专用页仅供评阅使用。）**

# 湖南省第五届研究生数学建模竞赛

## 题目　基于多目标优化模型的定向越野比赛路线设计研究


**摘　　要：**

针对定向越野比赛中的不同要求，本文依据给定的数据和信息建立了不同约束条件和优化目标下的基于分层序列法的多目标优化模型，求解了定向越野路径规划中的三个问题，具体路线规划见**附件4**（相关可视化结果和代码见附件文件夹），它们的解决步骤、算法设计及结果如下：

针对问题一，以参赛人数最多为主目标函数，以总长度方差最小、总难度方差最小、平均重复度最小这四个重要性依次递减的指标为次优目标函数，在时间约束、检查点承载容量约束等条件下建立了多目标优化模型。首先根据区域形状、起点位置以及检查点的分布确定了终点、必经点1和必经点2位置分别为**(1445,327)**、**(462,416)**、**(1137,1082)**。采取分步的方法解决了非必经点可重复的问题：第一步通过设计的基于多旅行商的插入必经点路径规划遗传算法获得非必经点不可重复情况下，路线数分别为2、3、4时的路线规划解集；第二步将第一步的可行解组合叠加，得到路线数大于4时，非必经点可重复情况下的可行解。设置各路线起跑间隔，使用次优指标逐步筛选得到最终解。问题一的最多可参赛人数为296，路线数为14。

针对问题二，将爬高量作为第二优化目标，并调整算法，将爬高量方差作为基于多旅行商的插入必经点路径规划遗传算法中的适值函数。求解三维坐标下的距离，并确定了终点、必经点1和必经点2位置分别为**(1485,682,1)**、**(802,321,0)**、**(754,1108,7)**。最多参赛选手数量为328，路线数为16。

针对问题三，以最少的路线数量为主目标函数，最短的竞赛总时长为第二优化目标调整优化模型，适当调整算法。随后确定了终点、必经点1和必经点2位置分别为**(1746,427,1)**、**(598,473,4)**、**(1290,1273,13)**。根据选手的速度对选手进行分类，使得同一路线上有各种速度成分的选手，速度快的先出发，速度慢的后出发，得到最短路线数为8，最短竞赛总时长为2h45min。

最后，本文对模型的优缺点和敏感性进行了分析评价，发现该多目标优化模型自变量较多，对自变量的敏感性较强。利用路线的总长度方差、总难度方差、平均重复度、总爬高量方差对满足主优化目标和约束条件下的最优解进行性能评估，结果如下表所示，可见本文所得的路线规划方案能较好地适应定向越野比赛中的多个优化目标和约束条件。

| | 总长度方差/$km^2$ | 总难度方差/$min^2$ | 平均重复度 | 总爬高量方差/$m^2$ |
|---|---|---|---|---|
| 问题一 | 0.3 | 3.3 | 21% | 0 |
| 问题二 | 0.5 | 3.1 | 15% | 3.1 |
| 问题三 | 1.4 | 6.3 | 8% | 7.8 |



**关键词：**分层序列法；多目标优化模型；基于多旅行商的插入必经点路径规划遗传算法；迭代优化；定向越野

# 1 问题重述

## 1.1 问题背景

定向越野是一项借助地形图和指北针，按规定的顺序独立地打卡若干个检查点，并以最短的时间完成任务的一项户外运动，从竞技比赛到团建活动，已受到越来越多的人们的喜爱。而如何根据比赛的目标、参赛人员情况、场地限制等进行比赛线路的灵活设计关乎比赛的趣味性、公平性和参与度[1]。

## 1.2 问题的要求

（1） 一次比赛安排若干条从起点到终点的比赛路线，所有参赛选手须按照所选路线中检查点的顺序依次打卡，每一条比赛路线可安排一个或多个参赛选手，选择同一条路线的参赛选手按照一定的时间间隔错时出发，不同路线的首发参赛选手同时出发；

（2） 所有路线的起点和终点相同，终点设置在难度较低的检查点，所有路线有两个相同的必经检查点且每条路线检查点数量不低于 20 个；

（3） 不同路线的长度、总打卡难度应尽可能均匀，不同路线之间应尽可能避免包含较多重复路段，也应避免参赛选手在完成任务过程中距离过近；

（4） 同一个检查点在一分钟内被访问的次数应控制到不超过 5。

## 1.3 待解决的问题

（1） 根据附件 1 考虑在二维平面中的路线设计问题，期望本次活动中的参赛选手完成任务的平均时间不低于 2 小时，竞赛的总完成时间最长不超过 3 个小时，尽可能多参赛选手能够加入到此次竞赛中。请为本次定向越野比赛设计比赛路线、路线的终点、必经检查点，并给出总参赛选手数量、各条路线的参赛选手数量、检查点、总长度及总难度。

（2） 根据附件 2 考虑在三维平面中的路线设计问题，要求每条比赛路线的爬高量尽可能均匀。请重新考虑问题（1），给出总参赛选手数量、各条路线的参赛选手数量、检查点、总长度、总难度及爬高量。

（3） 考虑在三维平面中的路线设计问题，附件 3 给出了不同参赛选手的速度数据，且此次活动有 120 名参赛选手参赛，期望参赛选手平均完成时间不低于 2 小时，比赛总完成时间不超过 3.5 小时。请为此次活动设计最佳比赛路线，给出各条路线的参赛选手数量、检查点、总长度、总难度及爬高量，使得使用的路线尽可能少，完成的时间尽可能短。

# 2 模型假设与条件

（1） 假设每一条路线的不同选手起跑间隔是固定的，不同线路参赛人员的出发可以不同，便于简化计算；

（2） 假设所有选手全程能保持匀速，不会出现加快或减慢的现象；

（3） 假设任何检查点的难度不会随选手的“跟跑”或是交流现象发生变化；

（4） 假设每条路线的检查点数相同，给予选手相对公平的参赛感受并方便算法设计；

（5） 假设定向越野时天气晴朗，人的有效目视距离为 400 米，超出该范围后不能跟跑前一队伍，则同一条线路选手间的出发间隔为 $\min\{\text{目视距离/速度}, \max\{\text{节点转换时间}\}\}$，其中，节点转换时间为节点间路径的转移时间与下一个节点难度时间之和。

# 3 问题分析

## 3.1 问题一分析

定向越野路线设计中，有若干约束条件下，有多个优化目标[2]，用多目标模型进行描述并转化为具体实现算法是解决问题的关键。

问题要求各路线的长度、难度尽可能均匀，重复度尽可能小，于是可以使用总长度方差、总难度方差和平均重复度这些指标来量化衡量这些要求。长度、难度不难理解，重复度我们定义为两条路线中相同的路段数在两条路线路段总数的比值，计算所有两两路线的重复度后取平均，得到平均重复度。

## 3.2 问题二分析

相比问题一，问题二将各检查点的坐标变为了三维坐标，并增加了爬高量相近这一优化目标，于是本文将对模型进行部分调整，在求解时改变距离的计算方式，并将爬高量作为第二优化目标。

## 3.3 问题三分析

相比问题二，问题三考虑了不同选手的体能素质，给出了各选手的速度数据和参赛的选手总数，并扩大竞赛总时长上限至 3.5 小时。问题三的主要优化目标发生了改变，要求路线数量最少，竞赛总时长尽量短。

本文按照参赛人员的速度进行分类，由于每一条道路的长度、难度和爬高量等参数应当尽量相近，不同路径的情况几乎相同，而参赛人员的速度不尽相同，在比赛的过程中可能会出现同一条路线中选手相互赶超，“追赶”的现象。因此，本文中为了避免这种现象，将选手进行分类，使得同一组中的有各种速度成分的选手，在比赛的过程中速度快的先出发，速度慢的后出发。

对问题三，本文将对问题二下的模型进行调整，令最少的路线数量为主目标函数，最短的竞赛总时长为第二优化目标，求解时仍采取问题一的方法分步求解，对路线数进行迭代，设置各路线起跑间隔，获取参与选手达到 120 人，满足竞赛总时长、时间约束的最少路线数。

# 4 问题一的模型建立与求解

## 4.1 基于分层序列法的多目标优化模型

分层序列法的基本思想是将多目标优化问题中的多个目标函数分清主次，按其重要程度逐一排除，然后依次对各个目标函数求最优解，后一目标应在前一目标最优解的集合域内寻优[3]。

设$\Phi_1$为候选检查点编号的集合，$\Phi_1=\{1,2,\ldots \mathrm{N}\}$，由附件一数据易知，$\mathrm{N}=98$，即共有 98 个元素。令$i$、$j$为任一候选检查点，即$i$、$j$为集合内任一元素，且$i\neq j$。设$i$、$j$的二维坐标分别为$(x_i,y_i)$、$(x_j,y_j)$。令所有线路的已知起点为$j_b$，两个未知的必经点分别为$j^*$、$j^{**}$，未知的终点为$j_e$。假设总共有$\mathrm{M}$条线路。针对该问题的多个优化目标，定义如下决策变量、目标函数和约束条件。

决策变量：

$$y_{mi}=\begin{cases}1, \text{线路}m\text{有检查点}i\\ 0, \text{线路}m\text{无检查点}i\end{cases} \tag{4-1}$$

$$x_{m,ij}=\begin{cases}1,\text{线路}m\text{有}i\to j\text{的直连线段}\\0,\ \text{线路}m\text{无}i\to j\text{的直连线段}\end{cases} \tag{4-2}$$

决策矩阵：

$$X_m=\begin{bmatrix}0 & \cdots & x_{m,1\mathrm{N}}\\ \vdots & 0 & \vdots\\ x_{m,\mathrm{N}1} & \cdots & 0\end{bmatrix}_{\mathrm{N}\times\mathrm{N}} \tag{4-3}$$

约束条件：

由于线路 $m$ 从检查点 $i$ 至多出发一次，线路 $m$ 到达检查点 $j$ 至多一次，所以

$$0\le\sum_{j=1}^{\mathrm{N}}x_{m.ij}\le 1 \tag{4-4}$$

$$0\le\sum_{i=1}^{\mathrm{N}}x_{m.ij}\le 1 \tag{4-5}$$

由于每条线路的检查点不低于 20 个，所以

$$\forall \mathrm{m}\in[1,\mathrm{M}],\mathrm{s.t.}\mathrm{S}_m=\sum_{i=1}^{\mathrm{N}}y_{mi}\ge 20 \tag{4-6}$$

设向量 $\vec{q}_m=X_m\cdot\begin{bmatrix}1\\ \vdots\\ i\\ \vdots\\ \mathrm{N}\end{bmatrix}_{\mathrm{N}\times 1}$，则向量 $\vec{q}_m$ 表示线路 $m$ 中包含的检查点及终点编号，将一个 $1\times\mathrm{S}_m$ 的全零向量 $\vec{p}_m$ 的第一个元素置换为起点 $j_b$，再将 $\vec{p}_m$ 的第二个元素根据索引值 $j_b$ 进行置换，迭代直至将 $\mathrm{S}_m$ 个元素置换完成。譬如，假设

$$\vec{q}_m=\begin{bmatrix}3 & 5 & 6 & 7 & 1 & 4 & 10 & \cdots & 0 & \cdots & 28\end{bmatrix}$$

$$j_b=2$$

操作过程如图 1 所示。

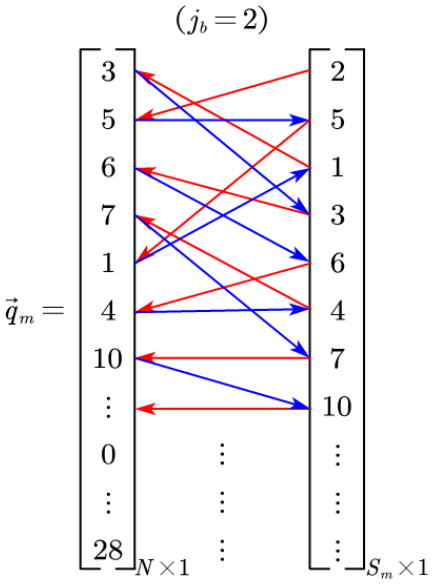


图 1 $\vec{p}_m$ 向量迭代置换示意图

得到表示线路 $m$ 依次访问的检查点的向量 $\vec{p}_m$，

$$\vec{p}_m=\begin{bmatrix} j_b \\ \vdots \\ j^* \\ \vdots \\ j^{**} \\ \vdots \\ j_e \end{bmatrix}_{\mathrm{S}_m\times1}=\begin{bmatrix} p_m(1) \\ \vdots \\ p_m(s) \\ \vdots \\ p_m(\mathrm{S}_m) \end{bmatrix}_{\mathrm{S}_m\times1} \tag{4- 7}$$

$p_m(s)$ 表示线路 $m$ 的第 $s$ 个检查点，$m\in[1,\mathrm{M}]$，$s\in[1,\mathrm{S}_m]$，并令 $p_m(s)=j$。

由于各路线起点、终点、必经点相同，则

$$p_1(1)=p_m(1)=\cdots=p_{\mathrm{M}}(1) \tag{4- 8}$$

$$p_1(\mathrm{S}_1)=p_m(\mathrm{S}_m)=\cdots=p_{\mathrm{M}}(\mathrm{S}_{\mathrm{M}}) \tag{4- 9}$$

$$\begin{cases} y_{mj^*}=1 \\ \sum_{m=1}^{\mathrm{M}} y_{mj^*}=\mathrm{M} \end{cases} \tag{4- 10}$$

$$\begin{cases} y_{mj^{**}}=1 \\ \sum_{m=1}^{\mathrm{M}} y_{mj^{**}}=\mathrm{M} \end{cases} \tag{4- 11}$$

设 $d_{ij}$ 为检查点 $i$ 到检查点 $j$ 的二维距离（m），则

$$d_{ij}=\sqrt{(x_i-x_j)^2+(y_i-y_j)^2} \tag{4-12}$$

所以路线 $m$ 的总长度为

$$D_m=\sum_{i=1}^{\mathrm{N}}\sum_{j=1}^{\mathrm{N}}x_{m,ij}d_{ij} \tag{4-13}$$

设 $w_i$ 为检查点 $i$ 的难度（min），路线 $m$ 的总难度为

$$W_m=\sum_{i=1}^{\mathrm{N}}y_{mi}w_i \tag{4-14}$$

定义两条路线的重复度是重复的路段数在两条路线的路段数之和中的占比，并设线路 $m$ 和线路 $m'$ 的重复度为 $u_{mm'}$，所以

$$u_{mm'}=1-\frac{\left\|X_m-X_{m'}\right\|_{m_1}}{\mathrm{S}_m+\mathrm{S}_{m'}} \tag{4-15}$$

设线路 $m$ 的第 $r$ 个选手到达第 $s$ 个检查点的时刻为 $T^{(s)}{}_{mr}$（h），线路 $m$ 安排 $\mathrm{R}_m$ 个选手，出发时间间隔为 $\Delta_m$，则 $T^{(s)}{}_{mr}$ 可由下式计算得到

$$\begin{cases} T_{m,r}{}^{(s)}=T_{m,r}{}^{(s-1)}+\dfrac{d_{p_m(s-1)p_m(s)}}{1000\times v}+\dfrac{w_{p_m(s-1)}}{60} \\ T_{m,r}{}^{(1)}=T_{m,r-1}{}^{(1)}+\Delta_m \quad ,\ r\in[1,\mathrm{R}_m] \\ T_{1,1}{}^{(1)}=\cdots=T_{m,1}{}^{(1)}=\cdots T_{\mathrm{M},1}{}^{(1)}=0 \end{cases} \tag{4-16}$$

由于选手平均完成任务时间大于 2 小时且竞赛总时长小于 3 小时这一时间约束，且各选手的速度相同，设速度为 $v$（Km/h）,则

$$\begin{cases} \displaystyle\sum_{m=1}^{\mathrm{M}}\frac{\displaystyle\sum_{i=1}^{\mathrm{N}}\sum_{j=1}^{\mathrm{N}}x_{m,ij}d_{ij}}{1000\times v}+\frac{\displaystyle\sum_{i=1}^{\mathrm{N}}y_{mi}w_i}{60}}{\mathrm{M}}\geq 2 \\ \max(T^{(\mathrm{S}_m)}{}_{m,\mathrm{R}_m})\leq 3, m\in[1,\mathrm{M}] \end{cases} \tag{4-17}$$

设路线 $m$ 的第 $r$ 个选手到达检查点 $j$ 的时刻为 $t^j{}_{m,r}$（h），由于线路 $m$ 有检查点 $j$ 时，$j=p_m(s)$，所以令

$$t^j{}_{m,r}=\begin{cases} \infty, y_{mj}=0 \text{或} r>\mathrm{R}_m \\ t^{p_m(s)}{}_{m,r}=T^{(s)}{}_{m,r}, y_{mj}=1 \text{且} r\in[1,\mathrm{R}_m] \end{cases} \tag{4-18}$$

所以对于检查点 $j$，被访问的时刻矩阵 $V^j$ 为

$$V^j=\begin{bmatrix} t_{1,1} & \cdots & t_{1,r} & \cdots & t_{1,\max(\mathrm{R}_m)} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ t_{m,1} & \cdots & t_{m,r} & \cdots & t_{m,\max(\mathrm{R}_m)} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ t_{\mathrm{M},1} & \cdots & t_{\mathrm{M},r} & \cdots & t_{\mathrm{M},\max(\mathrm{R}_m)} \end{bmatrix}_{\mathrm{M}\times\max(\mathrm{R}_m)} \tag{4- 19}$$

设集合$\Phi_{V^j}$包含$V^j$中的任意 6 个元素，由于每一个检查点在不超过 1 分钟的时间段内被访问的次数小于等于 5 这一承载容量约束，则

(4- 20)

目标函数：

$$C=\max(\sum_{m=1}^{\mathrm{M}}\mathrm{R}_m) \tag{4- 21}$$

$$\sigma_D=\min\left(\frac{\sum_{m=1}^{M}(D_m-\bar{D})^2}{M}\right) \tag{4- 22}$$

$$\sigma_W=\min\left(\frac{\sum_{m=1}^{M}(W_m-\bar{W})^2}{M}\right) \tag{4- 23}$$

$$U=\min\left(\frac{\sum_{m=2}^{\mathrm{M}}\sum_{m'=1}^{m-1}\left(u_{mm'}\right)}{C_{\mathrm{M}}^2}\right) \tag{4- 24}$$

为使尽可能多的选手参与该比赛，并使各路线的总长度、总难度尽可能相近，重复度尽可能小，本文相应建立了目标函数$C$、$\sigma_D$、$\sigma_W$、$R$，它们的重要程度依次减小，我们在$C$的最优解集中寻找$\sigma_D$的最优解集、在$\sigma_D$最优解集中寻找$\sigma_W$的最优解集，依次下去。目标函数$C$表示能参与该比赛的选手最大数；目标函数$\sigma_D$表示各路线的长度的最小方差，本文使用总长度的方差来衡量路线长度的相近程度，各路线总长度的方差越小，代表各路线的总长度越相近；目标函数$\sigma_W$表示各路线的难度的最小方差，各路线总难度的方差越小，代表各路线的总难度越相近；目标函数$U$表示总重复度，描述 M 条路线中，重复路段数占总路段数的平均比率，平均比率越高，说明各路线重合得越多，越容易导致参赛选手“跟跑”和相互交流现象。

## 4.2 分步优化组合迭代算法设计

### 4.2.1 算法流程

分步优化组合迭代算法首先分析场地、起点等因素确定终点及必经检查点，对可参赛人数最多这一主要优化目标采取分步优化组合迭代方法，第一部分得到非必经点不可重复条件下不同路线数时的路线规划设计；第二部分将第一步的可行解组合叠加，筛选优化得到非必经点可重复条件下的最大可参赛人数，如图 2 所示。

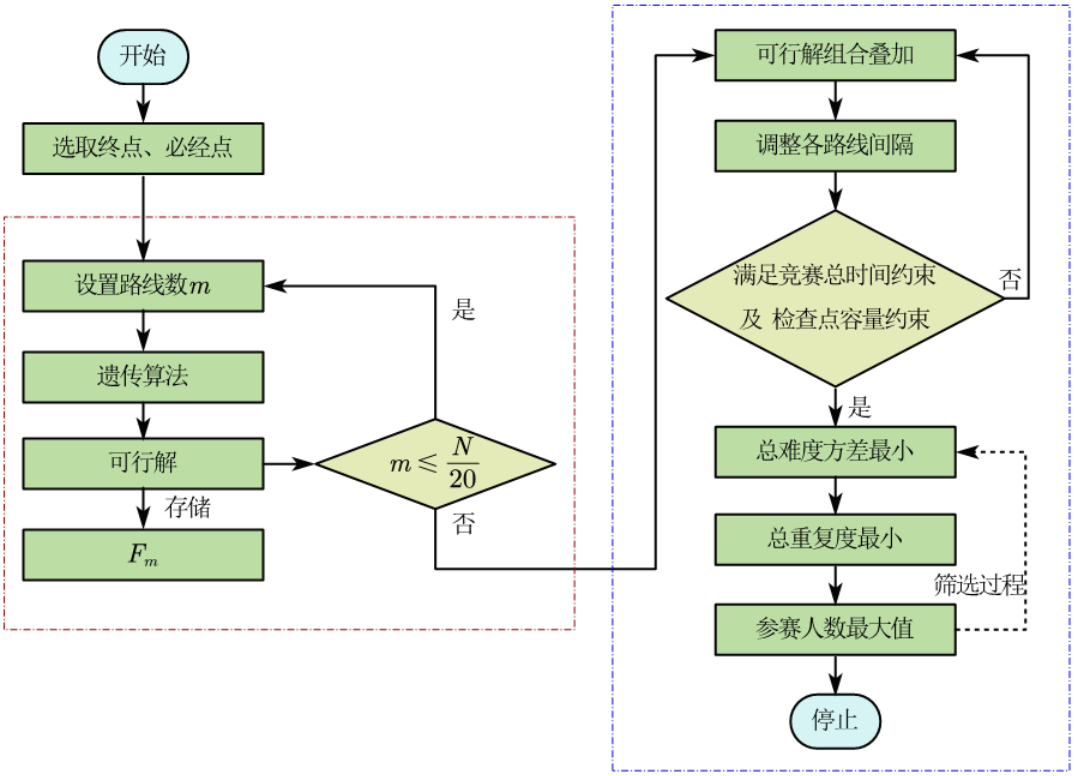


图 2 分步优化组合迭代算法流程图

在第一部分中，第一步固定路线数，利用基于多旅行商的插入必经点路径规划遗传算法得到满足约束条件的路线规划，第二步迭代路线数，从而得到有限个非必经点不可重复条件下的可行解。在第二部分中，第一步固定路线数，组合叠加第一部分中的可行解，通过调整各路线的起跑间隔，得到该路线数下的最多参赛人数，而后迭代路线数，选取最多参赛人数。

### 4.2.2 选取终点及必经检查点

终点及必经检查点的选取原则：终点及必经检查点的位置由区域形状和起点位置决定，整个区域尽量关于起点与终点的连线段对称，两个必经点尽量关于起点与终点的连线段对称。若区域为圆形，起点在圆上，则终点与起点的连线段为圆的直径，且与两必经点的连线段垂直。在该原则下，区域内检查点被充分利用，各路线上的选手分散，减轻检查点的承载容量，满足约束条件的基础上可增大路线数，从而一定程度上增加参赛选手数。

### 4.2.3 非必经点不可重复条件下的路线规划

遗传算法是求解复杂的组合优化问题的有效方法，其思想来自于达尔文进化论和门德尔松遗传学说。它对种群重复地进行选择、交叉、变异等基本遗传操作，

不断产生出比父代更适应环境的新一代种群，直到满足要求条件。遗传算法通过模拟生物进化过程来从庞大的搜索空间中筛选出较优秀的解，高效而且具有强鲁棒性[4]。

传统的多旅行商问题[5]所有节点只能被访问一次，但本题的必经点被访问次数为路线数，所以改进得到基于多旅行商的插入必经点路径规划遗传算法[6]-[10]。由于路线数未定，设定初始路线数为 3，得到初始路线数下的可行解后迭代路线数。由于本题共有 98 个检查点，每条路线至少访问 20 个检查点，所以非必经点不可重复条件下至多 4 条路线，并且由于在该条件下，非必经点只可被路线访问一次，所以可以类比多旅行商问题，即节点只可被访问一次，于是设计了基于多旅行商的插入必经点路径规划遗传算法，如图 3 所示，对编码、交叉、变异过程增加了是否满足必经点条件的判决，从而生成具有必经点的多旅行商路径规划解。

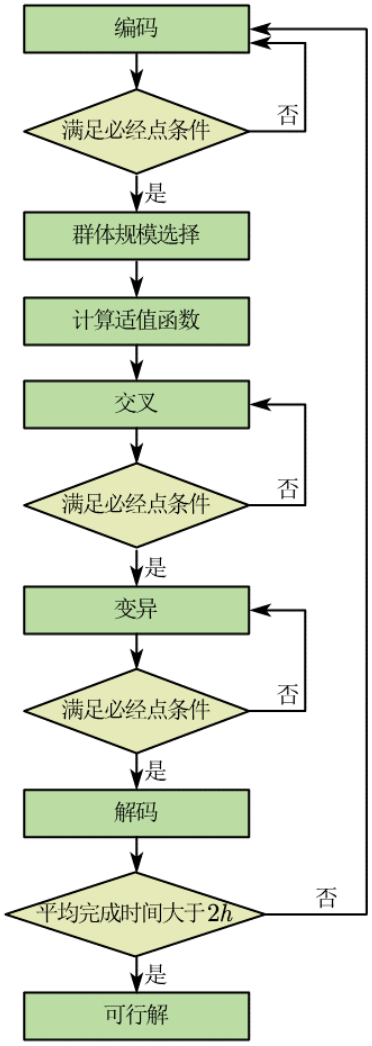


图 3 基于多旅行商的插入必经点路径规划遗传算法流程图

**Step1:编码**

问题一有 98 个检查点，集合为$\Phi_1$。假设每条路线除了经过两个必经检查点只经过 18 个检查点。集合$\Phi_1$除去必经点$j^*$、$j^{**}$得到集合$\phi_1^{'}$，染色体编码从$\phi_1^{'}$中随机选取$18\times m$个检查点，并对$18\times m$个检查点编号，然后随机排列得到一个

$1\times 18m$ 的样本，随后插入 $m$ 个必经点 $j^*$ 和 $m$ 个 $j^{**}$，得到一个 $1\times 20m$ 的样本，如图 4。每个样本可均匀地划分为连续的 $m$ 段，每段 20 个检查点，代表一条线路。由于每条线路有必经点 $j^*$、$j^{**}$，所以每 20 个检查点中必须含有一个 $j^*$、一个 $j^{**}$。若产生的样本不符合条件，则删除该样本。

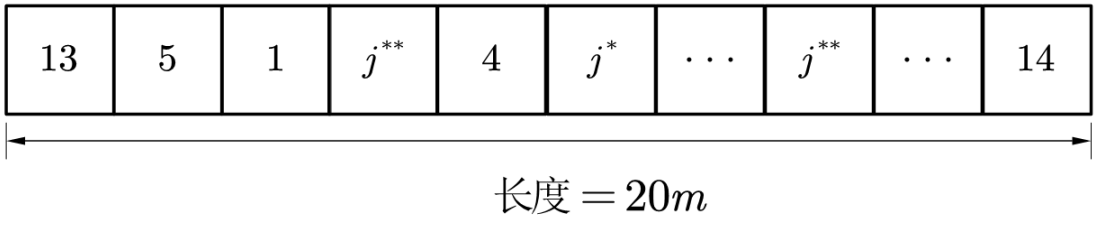


图 4 某随机样本

**Step2: 群体规模选择**

群体太小难以求得满意的结果，群体太大则计算复杂。群体规模一般 10—160。根据经验，本题群体规模取 120，则总共获取 120 个符合要求的随机排列样本。

**Step3:适值函数**

适值函数的设计非常关键。在遗传进化的初期，通常会产生一些超常的个体，如若按照比例选择法，超常个体可能会因竞争力太强而控制选择过程，影响算法的全局优化性能；在遗传进化的后期，即算法接近收敛时，由于种群中个体适应值差异较小时，继续优化的潜能降低，可能获得某个局部最优解。

由于非必经点不可重复，所以非必经点没有承载容量约束，最大由于次优化目标为总长度的最小方差，令目标函数作指数变换得到适值函数：

$$f = \alpha e^{-\beta \times H} \tag{4-25}$$

上式中，$\alpha$、$\beta$ 为正实数，$H = -\sigma_D$ 为目标函数。

**Step4:选择**

由于该题包括起点和终点共有 100 个节点，所以转移距离矩阵阶数为 $100\times 100$，转移距离矩阵 $T_{100\times 100}$ 为

$$T_{100\times 100} = \begin{pmatrix} \infty & \cdots & d_{1i} & \cdots & d_{1,100} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ d_{i1} & \cdots & \infty & \cdots & t_{i,100} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ d_{100,1} & \cdots & d_{100,i} & \cdots & \infty \end{pmatrix} \tag{4-26}$$

选择是用来确定重组或交叉的个体，以及备选个体将产生多少个子代个体。选择的第一步是计算适值，采用按比例的适值分配，是利用比例于各个个体适值的概率决定其子孙的遗留可能性。若某个体 $i$，其适值为 $f_i$，则其被选择的概率表示为

$$q_i = \sum_{i=1}^{M} p_i \tag{4- 27}$$

第二轮使用轮盘赌选择法，为了选择交配个体，需要进行多轮选择，每一轮产生一个[0,1]均匀随机数，将该随机数作为选择指针来确定备选个体。

**Step5:交叉与变异**

关于交叉本题采用对调交叉操作，每两个父体进行一次交叉。随机选取两个交叉编号，如第 1 号和第 3 号检查点，确定一个匹配段，然后将匹配区域对调形成新染色体。若除必经点外有与匹配段基因值相同的部分，则删除交叉变异得到的新染色体，重新进行交叉操作，如图 5。判断每 20 个检查点中必须含有一个 $j^*$、一个 $j^{**}$，不符合则重新进行交叉操作。

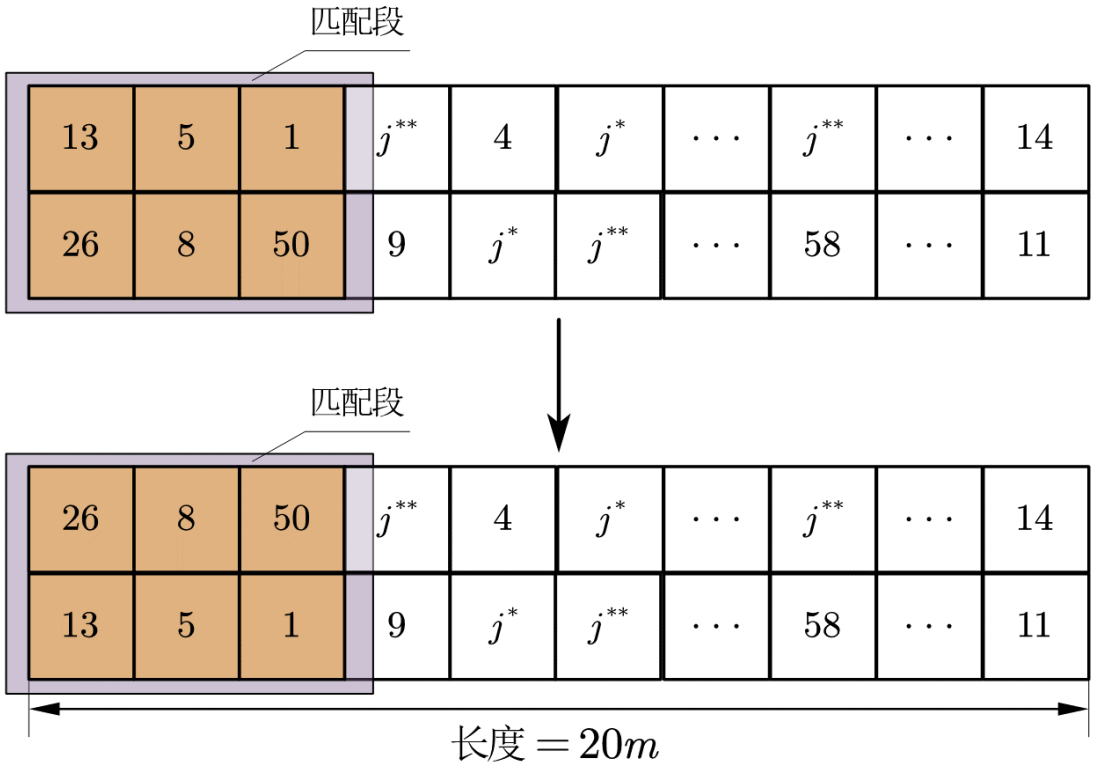


图 5 交叉示意图

变异在遗传操作中属于辅助性的搜索操作，主要目的是维持群体的多样性。较低的变异概率可以防止群体中重要的单一基因丢失，但降低了遗传算法开辟新搜索空间的能力；较高的变异概率将使遗传操作趋于纯粹的随机搜索，降低了算法的收敛速度和稳定性。本题采用对换变异方法，在染色体随机确定两个对换位置，如第 2 号和第 4 号检查点，进行基因对换，如图 6 所示。判断每 20 个检查点中是否含有一个 $j^*$、一个 $j^{**}$，不符合则重新进行变异操作。

本题为适当减小计算量，将 120 个样本分成 15 组，每组 8 个样本进行交叉变异。每组适应值低的父代样本将会被适应值高的子代代替，适应值相对高的父代样本将被保留。

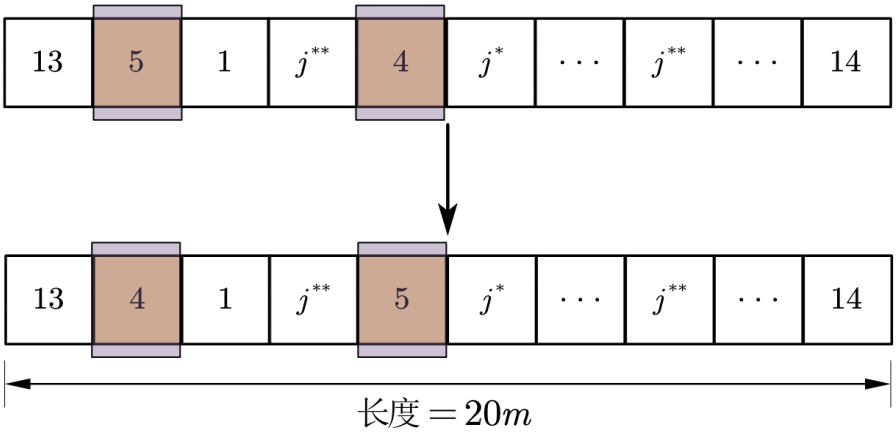


图 6 对换变异示意图

**Step6:解码**

经过上述步骤获得适应值最高的样本序列，且序列被均匀地划分为 $m$ 段，即得到各条行进路线，如图 7 所示，此情况下第一条、第二条、第三条线路为 $13-4-1-j^{**}-\ldots$、…、$\ldots-14$。而后计算该路线的平均完成时间，若小于 2 小时，则回到 Step1，重新计算，直到满足条件。

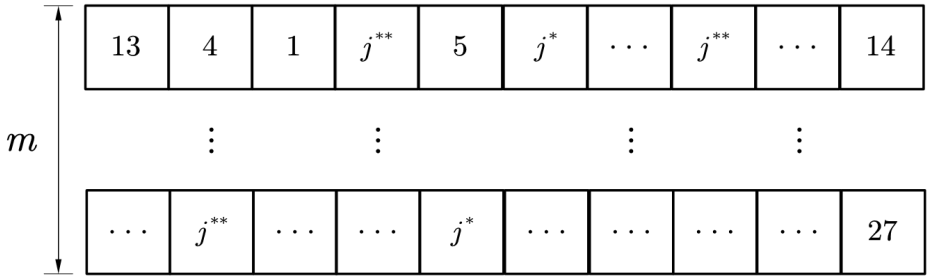


图 7 某最优样本示意图

### 4.2.4 非必经点可重复条件下的路线规划

通过非必经点不可重复条件下较为分散的路线规划组合叠加得到非必经点不可重复条件下的路线规划,示意图如图 8，左图为第一模块计算出的非必经点不重复的一组路线，与另一组路线叠加得到图右，图右有两个非必经点被重复访问，则得到了非必经点可重复的路线规划。虽然该方法难以全局最优，但满足了部分检查点被所有路线访问、少部分检查点被多次访问、大部分检查点仅被访问一次的要求，并且在满足约束条件和符合优化目标的可行解[11]。

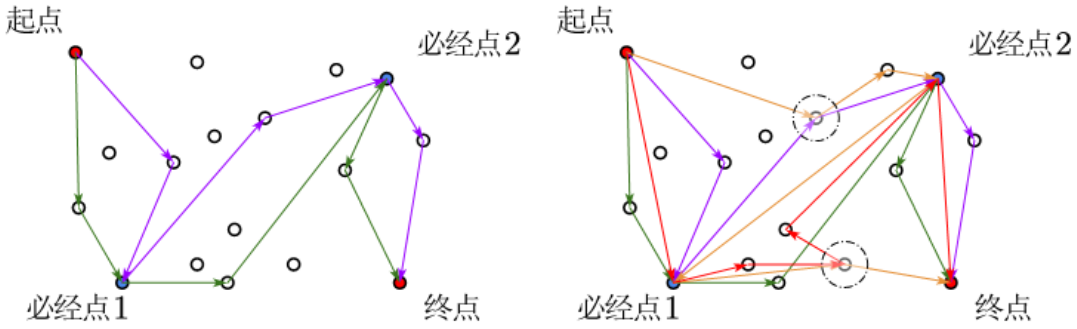


图 8 路径可行解叠加示意图

### 4.3 问题一结果

由于本比赛的区域为1500$m$×1500$m$的平面方形区域，所以起点大约在区域左上角，为充分利用区域空间，减少选手在检查点的拥塞，应可能使选手在方形区域内均匀分布，所以终点大约在区域的右下角，两个必经检查点在区域的左下和右上区域，如图 9，得到终点、必经点 1 和必经点 2 位置分别为 (1445,327)、(462,416)、(1137,1082)。

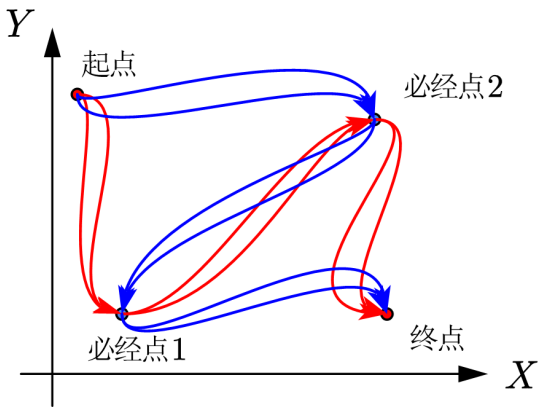


图 9 问题一的必经点和终点位置示意图

使用基于多旅行商的插入必经点路径规划遗传算法得到了路线数分别为 2、3、4 时的多个可行解，随后得到了非必经点可重复条件下不同路线数时的路线规划组合，即可行解集合元素的叠加，筛选得到了不同路线数下满足条件的路线规划及参赛人数最大值。

经计算检验，获得参赛人数最大值的路线数为 14，各路线检查点序号见附录，当中的某一条路线如图 10 所示，各路线总长度、总难度、选手数量如表 1 所示。

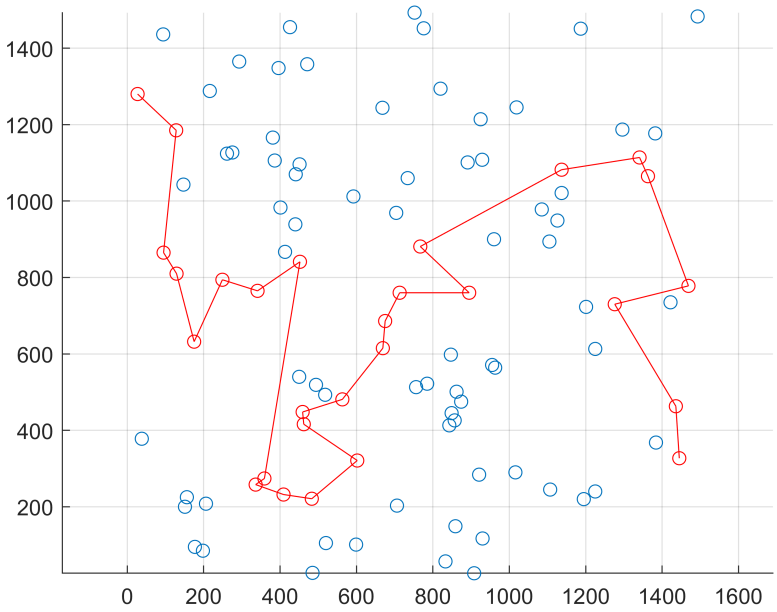


图 10 问题一中的某一条路线

表 1 问题一各路线详情

| 路径编号 | 路径总长度/km | 路径总难度/min | 路径总爬高量/m | 该路线选手数量 |
|---|---|---|---|---|
| 1 | 5.430 | 76 | 0 | 11 |
| 2 | 4.527 | 70 | 0 | 14 |
| 3 | 4.495 | 68 | 0 | 30 |
| 4 | 4.231 | 63 | 0 | 22 |
| 5 | 5.011 | 73 | 0 | 27 |
| 6 | 5.241 | 68 | 0 | 11 |
| 7 | 4.728 | 72 | 0 | 22 |
| 8 | 5.361 | 75 | 0 | 27 |
| 9 | 4.911 | 69 | 0 | 32 |
| 10 | 4.812 | 70 | 0 | 13 |
| 11 | 5.169 | 70 | 0 | 10 |
| 12 | 4.656 | 71 | 0 | 30 |
| 13 | 4.630 | 67 | 0 | 34 |
| 14 | 4.587 | 74 | 0 | 13 |

# 5 问题二的模型建立与求解

## 5.1 模型调整

设$\Phi_2$为问题二的候选检查点编号的集合，$\Phi_2=\{1,2,\ldots \mathrm{N}\}$，由附件二数据易知，$\mathrm{N}=148$，即共有 148 个元素。$i$、$j$为任一候选检查点，即$i$、$j$为集合内任一元素，且$i\neq j$。设$i$、$j$的三维坐标分别为$(x_i,y_i,z_i)$、$(x_j,y_j,z_j)$。设$d_{ij}$为检查点$i$到检查点$j$的三维距离（m），则

$$d_{ij}=\sqrt{(x_i-x_j)^2+(y_i-y_j)^2+(z_i-z_j)^2} \tag{5-1}$$

本问题的决策变量、决策矩阵与问题一相同。

约束条件：

设检查点$i$到检查点$j$的爬高量为$h_{ij}$（m）,则

$$h_{ij}=\frac{\left|z_i-z_j\right|+z_i-z_j}{2} \tag{5-2}$$

所以路线$m$的总爬高量为

$$H_m=\sum_{i=1}^{\mathrm{N}}\sum_{j=1}^{\mathrm{N}}x_{m,ij}h_{ij} \tag{5-3}$$

其余约束与问题一相同。

目标函数：

$$C = \max(\sum_{m=1}^{\mathrm{M}} \mathrm{R}_m) \tag{5- 4}$$

$$\sigma_H = \min\left(\frac{\sum_{m=1}^{M}(H_m - \bar{H})^2}{M}\right) \tag{5- 5}$$

$$\sigma_D = \min\left(\frac{\sum_{m=1}^{M}(D_m - \bar{D})^2}{M}\right) \tag{5- 6}$$

$$\sigma_W = \min\left(\frac{\sum_{m=1}^{M}(W_m - \bar{W})^2}{M}\right) \tag{5- 7}$$

$$U = \min\left(\frac{\sum_{m=2}^{\mathrm{M}}\sum_{m'=1}^{m-1}\left(u_{mm'}\right)}{C_{\mathrm{M}}^2}\right) \tag{5- 8}$$

相比问题一，本题增加了目标函数$\sigma_H$。$\sigma_H$表示各路线的总爬高量的最小方差。本题使用总爬高量的方差来衡量路线总爬高量的相近程度，各路线总爬高量的方差越小，代表各路线的总爬高量越相近。根据本题要求，目标函数$C$、$\sigma_H$、$\sigma_D$、$\sigma_W$、$R$的重要程度依次减小。

### 5.2 算法调整

在求解时改变距离的计算方式，即计算三维坐标下的直线距离；为将爬高量作为第二优化目标，将爬高量方差作为基于多旅行商的插入必经点路径规划遗传算法中的适值函数，尽量使各路线的爬高量相近。

### 5.3 问题二结果

考虑本比赛的区域为$1500m \times 1500m \times 9m$的三维空间区域，终点和两个必经检查点的坐标分别为(1485,682,1)、(802,321,0)、(754,1108,7)，示意图如图 11。

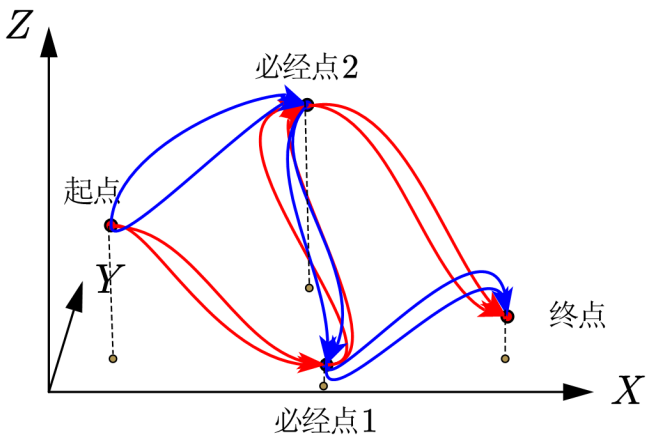


图 11 问题二的必经点和终点位置示意图

经计算检验，获得参赛人数最大值的路线数为 16，各路线检查点序号见附录，当中的某一条路线如图 12 所示，各路线总长度、总难度、选手数量如表 2 所示。

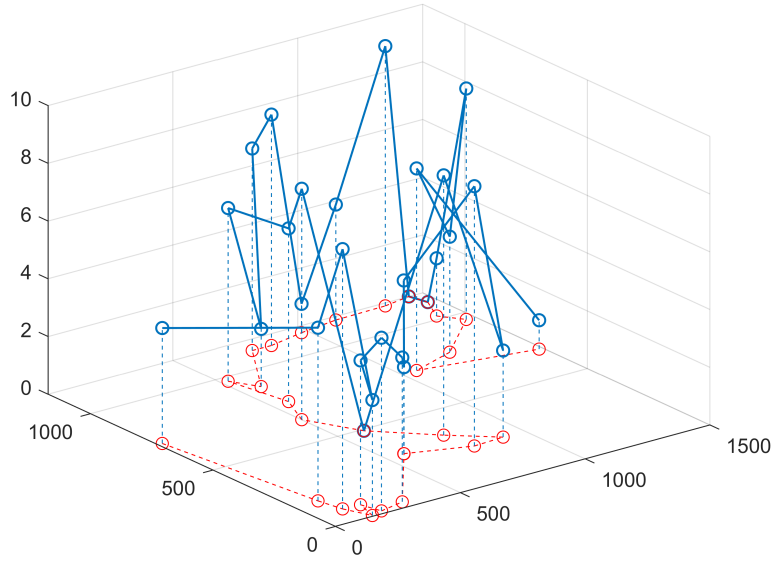


图 12 问题二的某一条路线

表 2 问题二各路线详情

| 路径编号 | 路径总长度 | 路径总难度 | 路径总爬高量 | 该路线选手数量 |
|---|---|---|---|---|
| 1 | 4.872 | 69 | 52 | 19 |
| 2 | 5.318 | 69 | 42 | 20 |
| 3 | 6.228 | 76 | 52 | 22 |
| 4 | 4.866 | 70 | 49 | 20 |
| 5 | 4.748 | 70 | 32 | 21 |
| 6 | 5.125 | 74 | 56 | 27 |
| 7 | 4.598 | 72 | 42 | 19 |
| 8 | 5.267 | 72 | 39 | 19 |

| | | | | |
|---|---|---|---|---|
| 9 | 6.596 | 69 | 39 | 20 |
| 10 | 4.920 | 65 | 35 | 26 |
| 11 | 5.057 | 73 | 47 | 19 |
| 12 | 4.787 | 75 | 48 | 17 |
| 13 | 5.676 | 70 | 53 | 22 |
| 14 | 4.981 | 73 | 45 | 18 |
| 15 | 5.571 | 64 | 48 | 18 |
| 16 | 4.609 | 72 | 50 | 21 |

# 6 问题三的模型建立与求解

## 6.1 模型调整

设$\Phi_3$为问题三的候选检查点编号的集合，$\Phi_3=\{1,2,\ldots \mathrm{N}\}$，由附件三数据易知，$\mathrm{N}=198$，即共有 198 个元素。$i$、$j$为任一候选检查点，即$i$、$j$为集合内任一元素，且$i\neq j$。设$i$、$j$的三维坐标分别为$(x_i,y_i,z_i)$、$(x_j,y_j,z_j)$。

本问题的决策变量、决策矩阵与问题二相同。

约束条件：

由于参赛选手共有 120 人，所以

$$\sum_{m=1}^{\mathrm{M}}\mathrm{R}_m=120 \tag{6-1}$$

设线路$m$的第$r$个选手的速度为$v_{m,r}$（Km/h），线路$m$的第$r$个选手到达第$s$个检查点的时刻为$T^{(s)}{}_{mr}$（h），线路$m$安排$\mathrm{R}_m$个选手，则$T^{(s)}{}_{mr}$由下式可计算得到

$$\begin{cases} T_{m,r}{}^{(s)}=T_{m,r}{}^{(s-1)}+\dfrac{d_{p_m(s-1)p_m(s)}}{1000\times v_{m,r}}+\dfrac{w_{p_m(s-1)}}{60} \\ T_{m,r}{}^{(1)}=T_{m,r-1}{}^{(1)}+\Delta_m \quad ,\ r\in[1,\mathrm{R}_m] \\ T_{1,1}{}^{(1)}=\cdots=T_{m,1}{}^{(1)}=\cdots T_{\mathrm{M},1}{}^{(1)}=0 \end{cases} \tag{6-2}$$

由于选手平均完成任务时间大于 2 小时且竞赛总时长小于 3.5 小时，则

$$\begin{cases} \dfrac{\displaystyle\sum_{m=1}^{\mathrm{M}}\left(\sum_{r=1}^{\mathrm{R}_m}\left(\dfrac{\displaystyle\sum_{i=1}^{\mathrm{N}}\sum_{j=1}^{\mathrm{N}}x_{m,ij}d_{ij}}{1000\times v_{m,r}}+\dfrac{\displaystyle\sum_{i=1}^{\mathrm{N}}y_{mi}w_i}{60}\right)/\mathrm{R}_m\right)}{\mathrm{M}}\geq 2 \\ \max(T^{(\mathrm{S}_m)}{}_{m,\mathrm{R}_m})\leq 3.5, m\in[1,\mathrm{M}] \end{cases} \tag{6-3}$$

其余约束与问题二相同。

目标函数：

$$M = \min(\mathrm{M}) \tag{6- 4}$$

$$T = \min(\max(T^{(\mathrm{S}_m)}{}_{m,\mathrm{R}_m})), m \in [1, \mathrm{M}] \tag{6- 5}$$

$$\sigma_H = \min\left(\frac{\sum_{m=1}^{M}(H_m - \bar{H})^2}{M}\right) \tag{6- 6}$$

$$\sigma_D = \min\left(\frac{\sum_{m=1}^{M}(D_m - \bar{D})^2}{M}\right) \tag{6- 7}$$

$$\sigma_W = \min\left(\frac{\sum_{m=1}^{M}(W_m - \bar{W})^2}{M}\right) \tag{6- 8}$$

$$U = \min\left(\frac{\sum_{m=2}^{\mathrm{M}}\sum_{m'=1}^{m-1}\left(u_{mm'}\right)}{C_{\mathrm{M}}^2}\right) \tag{6- 9}$$

### 6.2 算法调整

根据参赛人员的速度对选手进行分类，使得同一组中有各种速度成分的选手，在比赛的过程中速度快的先出发，速度慢的后出发，避免同一条路线中选手相互赶超，“追赶”的现象。

由于最小竞赛总时长为第二优化目标，将总长度作为基于多旅行商的插入必经点路径规划遗传算法中的适值函数，在平均完成时间不低于 2 小时这一约束条件下使得竞赛总时长尽可能短。

### 6.3 问题三结果

考虑本比赛的区域为的三维空间区域，终点、必经检查点 1 和必经检查点 2 的坐标分别为(1746,427,1)，(598,473,4)，(1290,1273,13)。

经计算检验，获得参赛人数最大值的路线数为 8，各路线检查点序号见附录 4，当中的某一条路线如图 13 所示，各路线总长度、总难度、选手数量如表 3 所示。

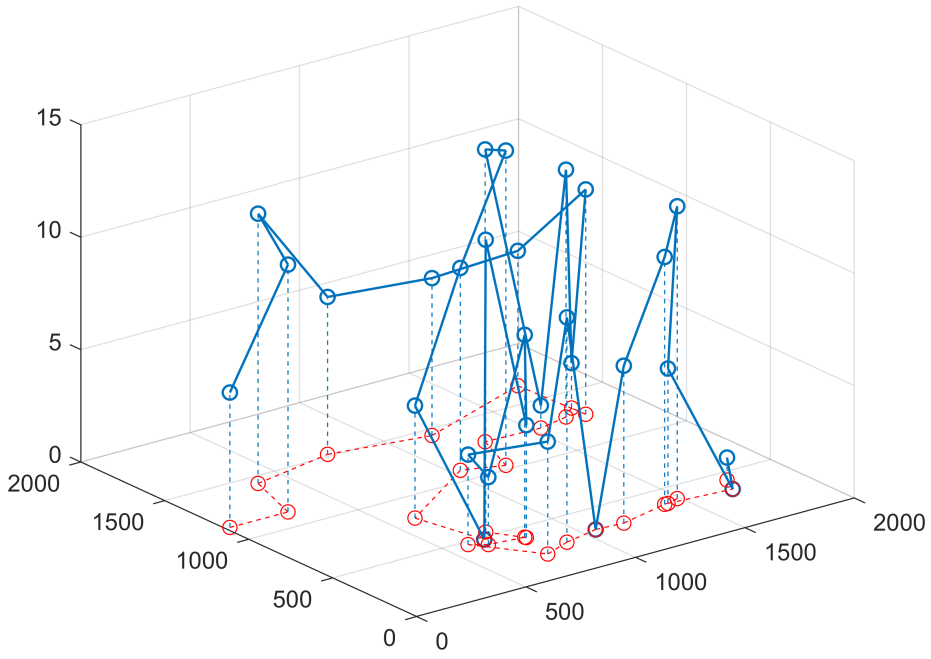


图 13 问题三的某一条路线

表 3 问题三各路线详情

| 路径编号 | 路径总长度 | 路径总难度 | 路径总爬高量 | 该路线选手数量 |
|---|---|---|---|---|
| 1 | 23.30642 | 115 | 88 | 15 |
| 2 | 24.08997 | 115 | 65 | 15 |
| 3 | 22.30582 | 105 | 67 | 15 |
| 4 | 22.55612 | 110 | 60 | 15 |
| 5 | 20.49934 | 96 | 72 | 15 |
| 6 | 24.40634 | 118 | 69 | 15 |
| 7 | 25.5565 | 116 | 83 | 15 |
| 8 | 22.32895 | 106 | 58 | 15 |

# 7 模型评价与敏感性分析

## 7.1 模型优点

（1） 用数学语言完善地描述了定向越野路径规划问题及要求，并通过有效的假设简化了问题，抓住了定向越野问题的主要矛盾。

（2） 针对参赛人数最多、爬高量相近等多目标问题，结合问题的实际需求建立了基于分层序列法的多目标优化模型，该模型的解有效满足了主优化目标，并一定程度上满足了次优化目标的需求。

（3） 针对复杂的必经点及检查点可重复性问题，创新地设计插入必经点的旅行商算法，采取分步策略解决了必经点路径规划、检查点可重复路径规划问题的策略，并满足了时间约束和检查点的承载容量要求。

## 7.2 模型缺点

（1） 模型自变量较多，敏感性较强。如终点及必经检查点的位置、优化目标

的先后排序等，对结果影响较大。在问题一的基于多旅行商的插入必经点路径规划遗传算法中，适值函数为总长度方差，而不是总难度方差等其他优化目标，原因在于建立问题一模型时人为地将总长度方差设为了第二优化目标，如果设为其他优化目标，路线的解空间变化较大。

（2） 算法较复杂，效率不高。本问题采用分步方式逐步实现两检查点必经和其他检查点可重复的要求，但该方式只是实现了该要求，在效率并不高，效果也有待加强，同时需要人为多次干预，不能直接由输入得到最终的输出。

### 7.1 模型敏感性分析

终点及必经检查点的选取是结合题目要求和比赛场地人为分析得到的，具有一定误差，不同终点和必经检查点可能会导致结果产生差异。以终点和必经检查点的位置为变量，分析位置误差对结果的影响。

## 参考文献


[1] https://en.wikipedia.iwiki.eu.org/wiki/Route_choice_(orienteering),2020.11.28.

[2] Gunawan A, Lau H C, Vansteenwegen P. Orienteering Problem: A survey of recent variants, solution approaches and applications[J]. European Journal of Operational Research, 2016:315-332.

[3] 卓金武.MATLAB 在数学建模中的应用[M].北京：北京航空航天大学出版社.2011.

[4] 吴孟达、毛紫阳、王丹，数学建模教程[M]，长沙：高等教育出版社.2011.

[5] 周辉仁,唐万生,魏颖辉.基于 GA 的最小旅行时间的多旅行商问题研究[J].计算机应用研究, 2009, 26(007):2526-2529.

[6] Xu Y, Kang S, Lin H. The MTSP optimal model suited for local-cross repeat path[C]// Control & Decision Conference. IEEE, 2013.

[7] 冯培伦.改进遗传算法在强边界路径规划中的应用[J].火力与指挥控制，45（9）：171-173，2020.

[8] 周鹏,张骏,史忠科.分段路径寻优算法研究及实现[J].计算机应用研究,（12）:241-243,2005.

[9] 罗勇， 陈治亚． 基于改进遗传算法的物流配送路径优化[J]． 系统工程,2012(08):118-122.

[10] 李鸿培， 王新梅． 具有局部重复路径的多路旅行商问题的研究[J]． 西安公路交通大学学报, 2000.

[11] Pieter Vansteenwegen, Wouter Souffriau. Iterated local search for the team orienteering problem with time windows[J]. Computers & Operations Research,2009,36(12): 3281-3290.

# 附 录

1.main 主函数：

```
clc

clear all

suoyin=xlsread("区域划分1.xlsx");

xy_origin=xlsread("zuobiao.xlsx");

xy=suoyin(:,1);    %可选择1 2 3

xy=xy(~isnan(xy));

xy=xy_origin(xy,:,:);

x_pos = xy(:,1);

y_pos = xy(:,2);

difficulty = xy(:,4);


D = pdist(xy(:,1:2));

temp_mat_time_dis = squareform(D)./100;%时间单位为分钟

mat_time_dis = zeros(length(x_pos),length(x_pos));

%加上搜寻的时间

for i=1:length(x_pos)

    for j=i+1:length(x_pos)


                mat_time_dis(i,j) = temp_mat_time_dis(i,j) +

difficulty(j);


    end
```

```
end
mat_time_dis = mat_time_dis+mat_time_dis.';
dmat=mat_time_dis;

nSalesmen = 4;
minTour = 7;
popSize = 120;
numIter = 5e4;
[optRoute,optBreak,minDist] =
mtspof_ga(xy(:,1:2),dmat,nSalesmen,minTour,popSize,numIter,1,1);
lujing=xy(optRoute,3)

shijian=xlsread("shijian.xlsx");
dian=xlsread("dian.xlsx");
renshu=0;
for i=1:14
    shijian_zong=0;
    for j=1:27
        shijian_zong = shijian_zong+shijian(dian(i,j),dian(i,j+1));
    end
    renshu(i)=(180-shijian_zong);
end
```

2.mtspof_ga 遗传算法函数

```
function varargout =
mtspof_ga(xy,dmat,salesmen,min_tour,pop_size,num_iter,show_prog,sho
w_res)
nargs = 8;
for k = nargin:nargs-1
    switch k
        case 0
            xy = 10*rand(40,2);
        case 1
            N = size(xy,1);
            a = meshgrid(1:N);
            dmat = reshape(sqrt(sum((xy(a,:)-xy(a',:)).^2,2)),N,N);
        case 2
            salesmen = 5;
        case 3
            min_tour = 1;
        case 4
            pop_size = 80;
        case 5
            num_iter = 5e3;
        case 6
            show_prog = 1;
```

```
            case 7
                show_res = 1;
            otherwise
        end
    end

    [N,dims] = size(xy);
    [nr,nc] = size(dmat);
    if N ~= nr || N ~= nc
        error('Invalid XY or DMAT inputs!')
    end
    n = N - 2;

    salesmen = max(1,min(n,round(real(salesmen(1)))));
    min_tour = max(1,min(floor(n/salesmen),round(real(min_tour(1)))));
    pop_size = max(8,8*ceil(pop_size(1)/8));
    num_iter = max(1,round(real(num_iter(1))));
    show_prog = logical(show_prog(1));
    show_res = logical(show_res(1));

    num_brks = salesmen-1;
    dof = n - min_tour*salesmen;
    addto = ones(1,dof+1);
    for k = 2:num_brks
```

```
    addto = cumsum(addto);
end
cum_prob = cumsum(addto)/sum(addto);

pop_rte = zeros(pop_size,n);
pop_brk = zeros(pop_size,num_brks);
for k = 1:pop_size
    pop_rte(k,:) = randperm(n)+1;
    pop_brk(k,:) = randbreaks();
end

clr = [1 0 0; 0 0 1; 0.67 0 1; 0 1 0; 1 0.5 0];
if salesmen > 5
    clr = hsv(salesmen);
end

global_min = Inf;
total_dist = zeros(1,pop_size);
dist_history = zeros(1,num_iter);
tmp_pop_rte = zeros(8,n);
tmp_pop_brk = zeros(8,num_brks);
new_pop_rte = zeros(pop_size,n);
new_pop_brk = zeros(pop_size,num_brks);
if show_prog
```

```
        pfig = figure('Name','MTSPOF_GA | Current Best
Solution','Numbertitle','off');
end
for iter = 1:num_iter
        for p = 1:pop_size
                d = 0;
                p_rte = pop_rte(p,:);
                p_brk = pop_brk(p,:);
                rng = [[1 p_brk+1];[p_brk n]]';
                for s = 1:salesmen
                        d = d + dmat(1,p_rte(rng(s,1)));
                        for k = rng(s,1):rng(s,2)-1
                                d = d + dmat(p_rte(k),p_rte(k+1));
                        end
                        d = d + dmat(p_rte(rng(s,2)),N);
                end
                total_dist(p) = d;
        end

        [min_dist,index] = min(total_dist);
        dist_history(iter) = min_dist;
        if min_dist < global_min
                global_min = min_dist;
```

```
            opt_rte = pop_rte(index,:);
            opt_brk = pop_brk(index,:);
            rng = [[1 opt_brk+1];[opt_brk n]]';
            if show_prog
                figure(pfig);
                for s = 1:salesmen
                    rte = [1 opt_rte(rng(s,1):rng(s,2)) N];
                    if dims == 3, plot3(xy(rte,1),xy(rte,2),xy(rte,3),'.-
','Color',clr(s,:));
                    else plot(xy(rte,1),xy(rte,2),'.-','Color',clr(s,:)); end
                    title(sprintf('Total Distance = %1.4f, Iteration
= %d',min_dist,iter));
                    hold on
                end
                if dims == 3,
plot3(xy(1,1),xy(1,2),xy(1,3),'ko',xy(N,1),xy(N,2),xy(N,3),'ko');
                else plot(xy(1,1),xy(1,2),'ko',xy(N,1),xy(N,2),'ko'); end
                hold off
            end
        end

        rand_grouping = randperm(pop_size);
        for p = 8:8:pop_size
```

```
rtes = pop_rte(rand_grouping(p-7:p),:);
brks = pop_brk(rand_grouping(p-7:p),:);
dists = total_dist(rand_grouping(p-7:p));
[ignore,idx] = min(dists);
best_of_8_rte = rtes(idx,:);
best_of_8_brk = brks(idx,:);
rte_ins_pts = sort(ceil(n*rand(1,2)));
I = rte_ins_pts(1);
J = rte_ins_pts(2);
for k = 1:8 % Generate New Solutions
    tmp_pop_rte(k,:) = best_of_8_rte;
    tmp_pop_brk(k,:) = best_of_8_brk;
    switch k
        case 2
            tmp_pop_rte(k,I:J) = fliplr(tmp_pop_rte(k,I:J));
        case 3
            tmp_pop_rte(k,[I J]) = tmp_pop_rte(k,[J I]);
        case 4
            tmp_pop_rte(k,I:J) = tmp_pop_rte(k,[I+1:J I]);
        case 5
            tmp_pop_brk(k,:) = randbreaks();
        case 6
```

```
                    tmp_pop_rte(k,I:J) = fliplr(tmp_pop_rte(k,I:J));
                    tmp_pop_brk(k,:) = randbreaks();
                case 7
                    tmp_pop_rte(k,[I J]) = tmp_pop_rte(k,[J I]);
                    tmp_pop_brk(k,:) = randbreaks();
                case 8
                    tmp_pop_rte(k,I:J) = tmp_pop_rte(k,[I+1:J I]);
                    tmp_pop_brk(k,:) = randbreaks();
                otherwise
            end
        end
        new_pop_rte(p-7:p,:) = tmp_pop_rte;
        new_pop_brk(p-7:p,:) = tmp_pop_brk;
    end
    pop_rte = new_pop_rte;
    pop_brk = new_pop_brk;
end

if show_res
    figure('Name','MTSPOF_GA | Results','Numbertitle','off');
    subplot(2,2,1);
    if dims == 3, plot3(xy(:,1),xy(:,2),xy(:,3),'k.');
    else plot(xy(:,1),xy(:,2),'k.'); end
```

```
        title('City Locations');
        subplot(2,2,2);
        imagesc(dmat([1 opt_rte N],[1 opt_rte N]));
        title('Distance Matrix');
        subplot(2,2,3);
        rng = [[1 opt_brk+1];[opt_brk n]]';
        for s = 1:salesmen
            rte = [1 opt_rte(rng(s,1):rng(s,2)) N];
            if dims == 3, plot3(xy(rte,1),xy(rte,2),xy(rte,3),'.-
','Color',clr(s,:));
            else plot(xy(rte,1),xy(rte,2),'.-','Color',clr(s,:)); end
            title(sprintf('Total Distance = %1.4f',min_dist));
            hold on;
        end
        if dims == 3,
plot3(xy(1,1),xy(1,2),xy(1,3),'ko',xy(N,1),xy(N,2),xy(N,3),'ko');
        else plot(xy(1,1),xy(1,2),'ko',xy(N,1),xy(N,2),'ko'); end
        subplot(2,2,4);
        plot(dist_history,'b','LineWidth',2);
        title('Best Solution History');
        set(gca,'XLim',[0 num_iter+1],'YLim',[0 1.1*max([1
dist_history])]);
```

```
end

if nargout
    varargout{1} = opt_rte;
    varargout{2} = opt_brk;
    varargout{3} = min_dist;
end

    function breaks = randbreaks()
        if min_tour == 1
            tmp_brks = randperm(n-1);
            breaks = sort(tmp_brks(1:num_brks));
        else
            num_adjust = find(rand < cum_prob,1)-1;
            spaces = ceil(num_brks*rand(1,num_adjust));
            adjust = zeros(1,num_brks);
            for kk = 1:num_brks
                adjust(kk) = sum(spaces == kk);
            end
            breaks = min_tour*(1:num_brks) + cumsum(adjust);
        end
    end
end
```