中国研究生创新实践系列大赛

“华为杯”第十八届中国研究生数学建模竞赛

学　　校	中国计量大学
参赛队号	21103560049
队员姓名	1.吴楷文
	2.陈晨煜
	3.高　琪

中国研究生创新实践系列大赛
“华为杯”第十八届中国研究生
数学建模竞赛

题　目　基于整数线性规划的机组航班匹配问题研究

摘　　要：

随着国民经济的进步，我国航空产业得到了快速发展，航班线路以及机场数量规模不断增大，这种变化虽然便利了旅客的出行，但也导致航空企业的运营管理日渐趋于复杂，经济并高效地解决机组人员排班问题显得尤为重要。针对此问题本文从航班的实际约束出发，建立了**单目标整数规划模型**，用于解决机组人员排班难问题，满足航班的起飞条件，提高机组的整体利用率，模型切实匹配了机组人员的实际需求。

针对问题一，分析航班，机场以及人员的信息属性，引入时间戳的概念创建航班、机场、人员数据集，根据航班到达、出发的机场和起飞、降落的时间，进行航段间连接时间约束并据此构建**可换航班矩阵**，该矩阵包含着相应航班之间可进行更换航班处理的信息，这简化了后续模型的建立与求解。利用上述数据信息进行所有航班以及人员的分配，以实现满足机组配置的航班数最多，总体机组人员的乘机次数最少以及替补机组人员数最少的优化目标，并将多目标的权重参数赋值为$\omega_1=1000,\omega_4=1,\omega_7=0.001$。据此构建单目标规划模型。采用**启发式算法**，根据**时序优先**的启发信息对模型进行求解，在满足约束的条件下，针对 A 组数据此时求得的最优机组人员分配方案中不满足机组配置航班数为 0 架，满足机组配置航班数为 **206 架**，机组人员总体乘机次数为 **8 人次**，替补资格使用次数为 **0 人次**。针对 B 组数据求得的分配方案中不满足机组配置航班数为 **130 架**，满足机组配置航班数为 **13824 架**，机组人员总体乘机次数为 **240 人次**，替补资格使用次数为 **0 人次**，程序运行时间为 1 分钟。

针对问题二，为简化模型的建立与求解我们引入航班集合 FS_r，在问题一的优化目标基础上添加机组人员的总体执勤成本最少以及机组人员间的执勤时长尽可能平衡优化次要目标，针对执勤时长尽可能平衡，利用各机组人员执勤时长与平均时长差值的最大值最小化方法量化表达时长尽可能平衡，并利用一类最大最小化问题的线性规划方法转化为线性目标，将多目标的权重参数赋值为$\omega_2=100$，$\omega_5=0.1$，据此构建**单目标规划模型**，采用根据时间优先的启发信息对模型进行求解，在满足约束条件的情况下，针对 A 组数据此时求得的最优机组人员分配方案中不满足机组配置航班数为 0 架，满足机组配置航班数为 **206 架**，机组人员总体乘机次数为 **8 人次**，替补资格使用次数为 **0 人次**，机组总体利用率 76.4%，最小/平均/最大一次执勤飞行时长分别为 **1.5**，**4.07**，**7.08**；最小/平均/最大一次执勤执勤时长分别为 **1.5**，**5.32**，**11.58**；总体执勤成本为 **44.149 万元**；针对 B 组数据此时求得的最优机组人员分配方案中不满足机组配置航班数为 **2034 架**，满足机组配置航班数为 **11920 架**，机组人员总体乘机次数为 **2528 人次**，替补资格使用次数为 **0 人次**，机组总体利用率 **65.15%**，最小/平均/最大一次执勤飞行时长分别为 **0.75**，**6.34**，**12**；最小/平均/最大一次执勤执勤时长分别为 **0.75**，**7.01**，**12**；总体执勤成本为 **4214.5 万元**。

针对问题三，类比可换航矩阵，我们构建**任务环间连接矩阵**，该矩阵包含着相应航班

间可进行任务环连接信息，并携带着连接时间的约束，这简化了后续模型的建立和求解。结合问题一、二进行机组人员排班计划任务分配，在前两问的优化目标基础上添加机组人员总体任务环成本最低以及任务环时长尽可能平衡优化目标，与问题二同样的处理方式，利用一类最大最小化问题的线性规划方法转化为线性目标，将多目标的权重参数赋值为 $\omega_3=10$，$\omega_6=0.01$，据此构建单目标规划模型，采用启发式算法根据时序优先的启发信息对模型进行求解，在满足约束条件的情况下，针对 *A* 组数据此时求得的最优机组人员分配方案中不满足机组配置航班数为 **0 架**，满足机组配置航班数为 **206 架**，机组人员总体乘机次数为 **48 人次**，替补资格使用次数为 **30 人次**；针对 *B* 组数据此时求得的最优机组人员分配方案中不满足机组配置航班数为 **2256 架**，满足机组配置航班数为 **11698 架**，机组人员总体乘机次数为 **3564 人次**，替补资格使用次数为 **124 人次**。

关键词：机组排班；线性模型；0-1 整数规划；任务环

目录

一、问题重述

1.1 问题背景

随着国民经济的进步，我国航空产业得到了快速地发展，但由于受到诸多因素与条件的影响，使得航空机组的排班问题成为困扰航空企业运营管理的重要部分，排班策略的优劣直接决定了航空企业的管理运营水平的高低。

面对机组排班问题，需要构造特定时间段的机组人员的日程安排，具体内容包括对每个机组人员确定其在何时何地以及哪个航班执行相应的任务，其中广义的机组人员包括飞行员，乘务员和空警。高质量的机组排班计划，不但可以有效地节省航空公司的运营成本，并且在满足机组人员合理福利需求上有着良好的表现。由于不同类型的机组人员不相互通用，所以机组排班问题一般上可表示为对其进行分别求解，并且不同机型的飞行员不能通用，所以对不同机型也考虑进行分别的求解。

现阶段的研究中，形成的比较成熟的解决方案考虑将排班问题分解为两个子问题进行求解：第一步即完成满足规范以及节约成本目的的任务环生成；第二步则是对任务环进行机组人员的分配。上述两个子问题的解决主要通过建立网络流模型并利用列生成法。但在实际应用中，两阶段的解法并不能完美的解决一些不完整的任务环问题，而且在优化空间上两阶段解法会减小相应的解空间，因此并不能得到最优解。对于列生成法，优化模型本身缺乏对各种规章约束参数的显性表达，因此难以实现指导作用。

1.2 相关信息

本次机组排班问题中假设航班的规划问题已经解决，机型的分配任务也已结束，航班对机组人员数量以及资格的要求已经确定，并且相应的待分配机组人员数固定，为此相应的优化排班问题的具体描述有以下几个方面。

时间：所有的时间按照单一的指定时区定义，不考虑秒及其以下的精度，相邻两天以给定时区的半夜零点作为分割点。

机场：作为航班起飞、到达以及机组人员的出发和到达的节点，机场按照国际民航组织 *IATA* 标准进行相应的标识。

机组人员：本题只考虑飞行员这一类别，现代民航飞行需要两种资格的飞行员：正机长和副机长。

1、机组人员的固定基地以其所在的机场表示。

2、机组人员都具有并且仅具有一个主要资格，但也可具有其它替补资格。机组安排优先安排主要资格，若主要资格不足时，安排替补资格。本题假定机长的主要资格只有两类：正机长和副机长。部分正机长可以代替副机长执行飞行任务。

3、机组人员除执行飞行任务外还可执行乘机任务。乘机任务表示机组人员乘坐航班从一机场前往另一机场执行飞行任务，为符合客运航班的需求，乘机人数存在着一定的限制。

对机组人员进行的完整的排班周期，应当由＜航班—执勤—任务环＞三部分组成。

<table>
<tr><td colspan="6">排班周期</td></tr>
<tr><td colspan="4">任务环1</td><td rowspan="3">休假</td><td>……</td></tr>
<tr><td colspan="3">执勤1</td><td rowspan="2">休息</td><td>……</td><td>……</td></tr>
<tr><td>航班1</td><td>连接</td><td>……</td><td>……</td><td>……</td></tr>
</table>

图 1 排班周期示意图

航班：飞机的一次飞行过程，包括起飞和降落。当航班应用于机组人员排班时也叫做航段。每个航班均包含以下的信息：给定的起飞时间、机场和到达时间、机场以及航班的最低机组配置，其中最低机组配置以 $C\langle\cdot\rangle F\langle\cdot\rangle$ 表示，$C\langle\cdot\rangle$ 表示为正机长数，$F\langle\cdot\rangle$ 表示为副机长数。只有当航班满足最低机组配置才能执行飞行任务，一种机型只有一种配置要求。

执勤：执勤是指由一连串航段以及间隔连接时间组成，考虑同一次执勤中的航段空间以及时间的问题，为此应当满足上一航段的到达应与下一航段的出发相对应；相邻航段之间的间隔时间必须满足最短连接时间的约束。以及在同一次执勤中，每个航段的起飞时间必须约束在同一天内，但到达时间不受这一约束。

航段之间的连接时间被算入执勤时间，但不计入飞行时间，执勤时间从当次的第一趟航班的起飞时间到最后一趟航班的到达时间进行计算。

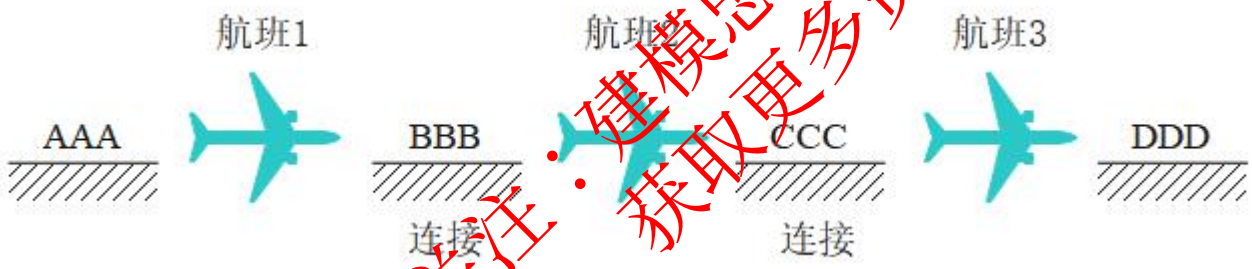


图 2 执勤示意图

图中 *AAA* 等表示机场称号信息。

任务环：表示由一系列的执勤与休息时间组成，机组人员需要完成从自己基地出发并最终回到出发基地的任务操作，同一任务环内执勤之间与航段之间的约束类似，应当满足同一执勤结束时到达机场应与下一执勤开始时的出发机场一致；相邻执勤间的休息时间必须满足最少休息时间约束，且不存在休息时间上限；每次任务环的第一次执勤必须从基地出发并且最后一次执勤需要回到基地。

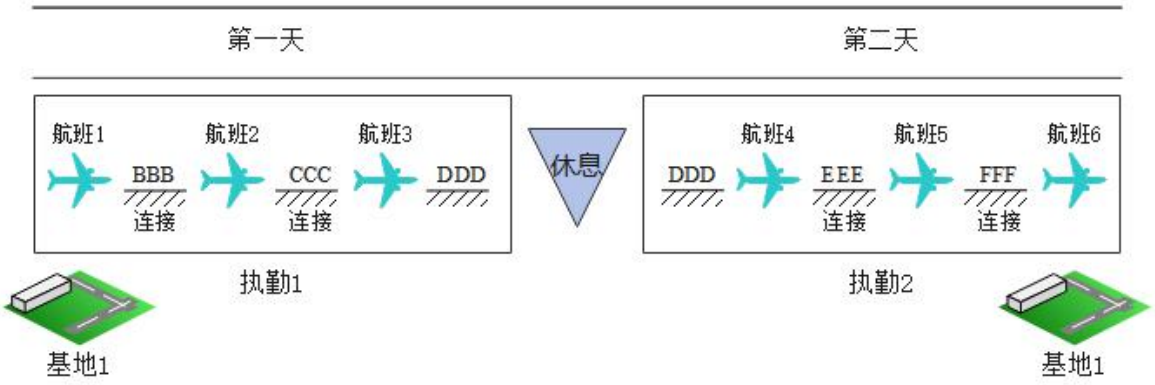


图 3 任务环示意图

图中执勤航班数只作为示意，并不代表实际执勤一次只包括三次航班数。

排班计划：排班计划由一系列任务环和休假组成，任务环之间需要满足一定的休假天数。排班计划的时间为排版周期，排版周期通常分为双周周期或者月周期。排班计划在每个周期开始前完成编排任务并向机组人员发布。

在实际应用中，排版周期的最后一个任务环可能出现跨越时间分割点而进入下一个排班周期的情况。为此，为减小模型的复杂度，假定排版周期可以进行适当的延展。

<table>
<tr><td colspan="20">排班周期</td></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td></tr>
<tr><td colspan="3"></td><td colspan="2">休假</td><td colspan="7"></td><td colspan="3">休假</td><td colspan="5"></td></tr>
<tr><td colspan="3">任务环1</td><td colspan="2"></td><td colspan="7">任务环2</td><td colspan="3"></td><td colspan="5">任务环3</td></tr>
</table>

图 4 排班计划示意图

1.3 待求解问题

对于问题的求解计算，在问题的表述中我们考虑一些必要的情况并假定以下各种信息：

1、机组人员可实现任意的组合。

2、航班允许由于无法满足最低机组资格配置而不能起飞的情况。

3、对于不能满足最低机组配置的航班不需要配置机组人员。

4、由于机组人员可以进行乘机任务，所以存在实际机组配置超过最低配置要求，此时实行乘机任务的机组人员的航段时间计入执勤时间，但不计入飞行时间。

问题一：

通过建立相应的线性规划模型实现给航班分配机组人员的任务并按照编号顺序依次满足以下的优化目标：

1、 航班完成尽可能多的机组配置；

4、机组人员的总乘机次数尽可能少；

7、 使用的替补资格次数也尽可能少。

问题一中机组人员以及相应的时间需要满足以下约束：

1、 机组人员从基地出发并需要最终回到基地。

2、 机组人员的下一航段的起飞机场与上一航段的到达机场保持一致。

3、 每个机组人员在两个相邻航段间的连接时间不得小于规定的 *MinCT* 分钟。

问题二：

在本问题中，引入执勤的概念，并假定每个机组人员的单位小时执勤成本已知。在满足问题一的优化目标的基础上，补充以下目标并按照相应的编号进行优化：

2、 总体机组人员的执勤成本最低；

5、 执勤时长在每个机组人员间尽可能保持平衡。

同样的，在问题一约束的基础上作以下的补充：

1、对于每个机组人员，一天至多执行一次执勤；

2、每次执勤的飞行时间不能超过 *MaxBlk* 分钟；

3、每次执勤的时长最多不超过 *MaxDP* 分钟；

4、对于每个机组人员，下一执勤的出发机场应与上一之前的到达机场相一致，

5、每位机组人员的相邻执勤间的休息时间不得小于 *Min*Re*st* 分钟。

问题三：

在本题中考虑任务环的概念，假定每个机组人员的没单位小时的任务环成本已知，在任务环成本中不包含执勤成本，在满足问题一、二的所有目标外，还需依编号满足以下目标：

3、总体机组人员的任务环成本最低；
6、任务环时长在每个机组人员间尽可能保持平衡。

同样的，在问题一、二约束的基础上作以下的补充：

1、 对于每个机组人员的排班周期的任务环总时长不得超过 *MaxTAFB* 分钟；
2、 对于每个机组人员相邻任务环之间至少有 *MinVacDay* 天休息；
3、 对于每个机组人员连续执勤天数不得超过 *MaxSuccOn* 天。

对于三问需要给出相应的编程代码，能够运行所给的数据，并将输出结果按照相应的格式给出。

1.4 问题输入参数

对于求解问题中提到的参数取值说明如下表所示：

表 1 输入参数具体说明及取值

输入参数	具体含义	具体取值
MinCT	航段之间最小连接时间	40 分钟
MaxBlk	一次执勤最大飞行时间	600 分钟
MaxDP	最长执勤时长	720 分钟
MinRest	相邻执勤间最短休息时间	660 分钟
MaxDH	航班最多乘机人数	5
MaxTAFB	每个机组人员在排版周期中最大任务环总时长	14400 分钟
MaxSuccOn	连续最长执勤天数	4 天
MinVacDay	相邻任务环间最少休息天数	2 天

二、问题分析

2.1 问题一分析

针对问题一，问题要求给出航班分配机组人员的策略方案，期望是尽可能多的航班满足机组配置，且使得总体乘机次数尽可能少，替补资格尽可能少使用，据此建立单目标线性规划模型。模型的优化目标有着准确的表达，这是一个多目标规划模型，考虑引入权重参数实现单目标线性模型的建立。

问题一的优化模型思路如图所示，在问题的分析中，为简化模型的表述与建立，我们考虑引入航班的可换乘变量以及航班的可飞行状态变量。为了简化时间的计算，我们引入时间戳的概念，将航班时间转化为从最早航班所在日期的零时时分至航班起飞或到达的总分钟数。

综上所述，建立一个包含有航班满足机组配置最多，总体乘机次数最少，使用替补资格最少组合的多目标规划模型，依靠目标间的权重参数设置将多目标规划模型转化为单目标模型求解，考虑使用启发式算法进行最优解的求解。

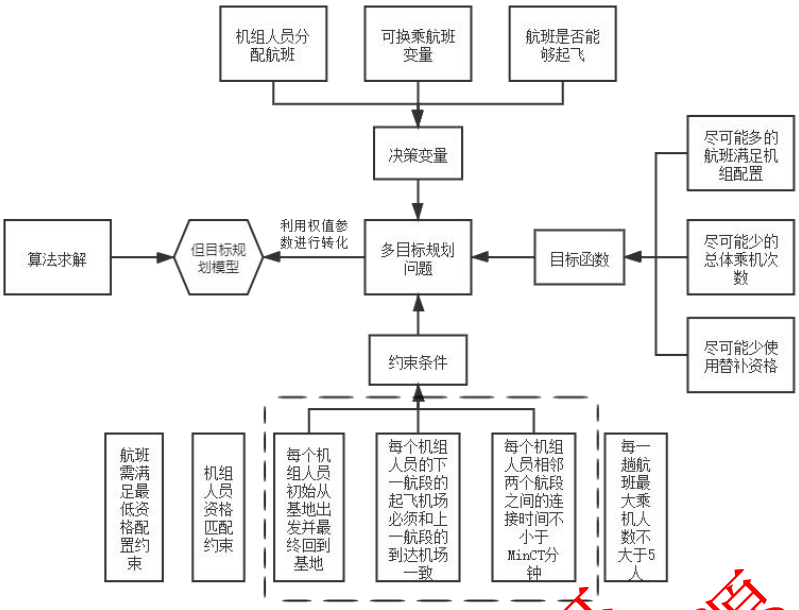


图 5 问题一分析思路图

2.2 问题二分析

针对问题二，在问题一的基础上引入执勤概念，同样的问题一中需要优化的目标函数以及所需满足的约束条件在问题二中同样需要考虑。执勤由航段组成，一组满足条件的航班组合即可表示机组人员的一次执勤，执勤之间需要满足一定的时间约束，在空间的连贯性上同样需要满足上一执勤的到达机场与下一执勤出发机场为同一机场。每个机组人员每天至多 执行一次执勤，在上述约束的条件下尽可能减少机组人员的总体执勤成本已经各机组人员之间执勤时长应尽量保持平衡。

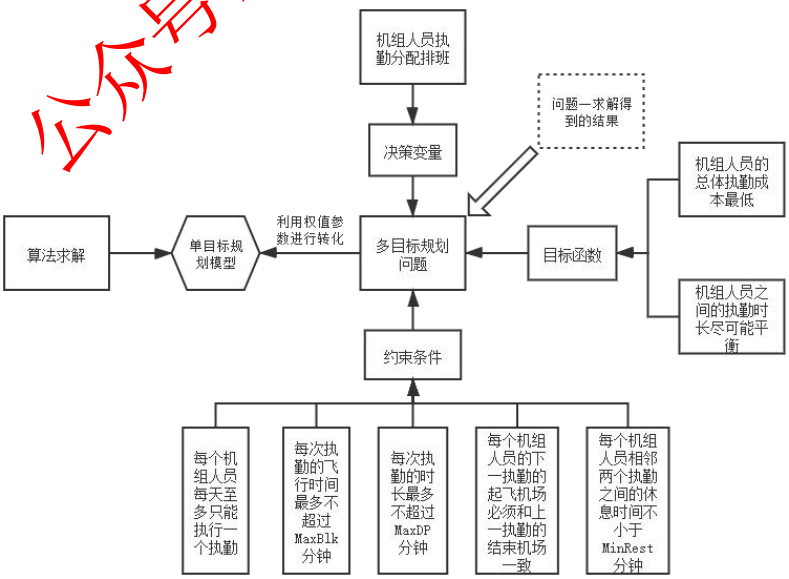


图 6 问题二分析思路图

为此我们考虑建立一个结合问题一目标的多目标规划模型，并同样的设置多目标间的权重参数将其转化为单目标模型，采用问题一的算法思路进行求解。

2.3 问题三分析

针对问题三，是问题一、二模型上的深化，任务环由执勤组成，执勤约束在这一问中同样需要被考虑，并且任务环间同样需要考虑时间关系的约束，每一个单独的任务环均从同一起点出发回到出发起点，每个排版周期内的任务环有最长时间约束，任务环之间有最短休息时间约束并且为避免构建的任务环过大，对连续执勤天数也有着一定的限制条件。在上述约束下，尽可能使总体任务环成本最低以及机组人员间的任务环时长尽可能平衡。

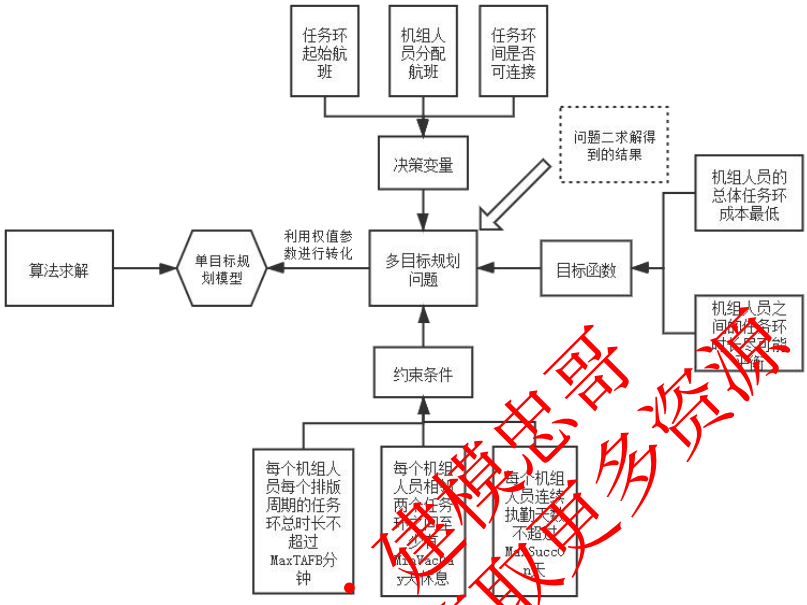


图 7 问题三分析思路图

综上所述，建立一个由多优化目标进行线性组合形成的单目标规划模型，采用问题一、二中使用的算法求解思路，考虑利用优先进行最早航班排班的启发式思想来对问题进行可行解的求解。

三、模型假设

1、机组人员之间可任意组合成一个机组，且不存在机组成员对航班的偏好；
2、假设在给定时间段前的航班安排与待安排人员无关，即执勤和任务环的全新开始；
3、当航班未达到最低配置要求不可起飞，且不满足起飞条件的航班不可配置机组人员；
4、实际机组成员配置可以超过最低配置要求，超过最低配置要求的成员均作为乘机人员。
5、题中所给航班安排表中航班完全准时，不存在延误等意外情况；
6、题中所给机组人员属性和信息完全准确，且不存在因个人情况违背安排导致航班无法起飞的意外情况。

四、符号说明

符号	符号含义	单位
i	第 i 位机组人员，i 的顺序按照所给数据集进行排序	
j	第 j 架航班，j 的顺序按照航班出发的时间顺序进行排序	
D_{jk}	可换乘矩阵，由第 j 架航班是否可以换乘至第 k 架航班	
$\mathbf{C}$	机组人员信息矩阵	
$\mathbf{F}$	航班信息矩阵	
$x_{ij}^{c}, x_{ij}^{f}, x_{ij}^{p}$	第 i 位机组人员是否以 $c=$ 正机长，$f=$ 副机长，$p=$ 乘机人员身份分配至 j 航班	
T_{ijk}	第 i 位机组人员由第 j 航班换乘至第 k 航班	
h_j	第 j 架航班是否满足最低配置	
$j_{u,ini}$	以 u 作为出发地的航班	
$j_{u,fin}$	以 u 作为终点地的航班	
FS_r	第 r 天的所有起飞航班集合	
$t_{i,r}^{1}$	第 i 位机组人员在第 r 天的飞行时间	分钟
$t_{i,r}^{2}$	第 i 位机组人员在第 r 天的连接时间	分钟
$\bar{t}$	所有机组人员的平均执勤时长	分钟
R_{ijk}	任务环间连接矩阵	
$RF_{i,j}$	机组人员第一架起飞航班的搜索矩阵	
TH_i	第 i 位机组人员的任务环时长	分钟
TH_i^2	第 i 位机组人员最后一次到达时间与第一次航班到达时间的差值	分钟
TH_i^3	第 i 位机组人员的第一次航班的飞行时间	分钟
TH_i^1	第 i 位机组人员任务环间隙时间	分钟
$\bar{TH}$	所有机组人员任务环总时长的平均值	分钟

五、数据预处理

5.1 机场、航班数据处理

所有问题中均包含有航班的出发、到达时间以及机场信息，给出的数据集中对此等信息的表述对于数据建模可行性方面有着不小的阻碍，为此需要将原始数据转化为可用的连续整数形式，考虑到具体航班号对问题的建模求解影响并不是很大，因此，我们仅对航班的出发、到达时间以及航班的出发、到达机场进行必要的数据处理。航班编号次序按照数航班起飞时间的先后顺序进行排序。

对航班的出发以及到达机场进行相应的编号处理，数据 *A* 中一共存在着 7 个机场信息，为此将具体的机场标识转化为 1-7 的编号类别，具体的对应关系如下表所示：

表 2 数据 *A* 机场编号表

机场标识	CTH	NKX	PDK	PGX	PLM	PXB	XGS
编号	1	2	3	4	5	6	7

数据 *B* 做同样的处理，共存在 39 个机场信息，部分机场标识编号如下表所示：

表 3 数据 *B* 机场编号表

机场标识	AXO	BCJ	FBX	GKS	GKU	GOQ	HOM
编号	1	2	3	4	5	6	7
机场标识	HUK	HWJ	HWX	MMC	MMY	MYJ	NOU
编号	8	9	10	11	12	13	14
机场标识	OAX	OOJ	OSY	OXU	SGJ	SHO	SXA
编号	15	16	17	18	19	20	21
机场标识	SXJ	TAN	TGD	THJ	TUK	TXD	UDJ
编号	22	23	24	25	26	27	28
机场标识	UTC	UYS	WXZ	XBT	XMH	XMJ	XMQ
编号	29	30	31	32	33	34	35
机场标识	XNZ	XSJ	XXJ	YKJ			
编号	36	37	38	39			

为了便于计算时间相关问题，在此将时间进行相应的转化，对于数据 *A* 的处理，以 2021 年 8 月 11 日 0 时 0 分作为基准时间，并以分钟作为基础单位，将航班出发、到达时间的标准形式进行转化，对于数据 *B* 的处理，以 2019 年 8 月 1 日 0 时 0 分作为基准时间(此处出现的题目勘误，正确数值以实际数据为准)，依据同样的处理方法进行时间数据的整理，同时为便于后续数据计算方便，对数据集序列进行一定的处理，此处给出部分结果展示，如下表所示：

表 4 数据 *A* 处理部分结果表

航班号	出发时间	到达时间	出发机场	到达机场	最低资格配置
FA2	2050	2140	4	2	C1F1
FA3	2065	2140	4	2	C1F1
FA680	480	570	2	4	C1F1
FA680	1920	2010	2	4	C1F1
FA680	3360	3450	2	4	C1F1

数据 *B* 做同样处理，结果如下表所示：

表 5 数据 *B* 处理部分结果表

航班号	出发时间	到达时间	出发机场	到达机场	最低资格配置
FB8559	5	125	7	21	C1F1
FB8559	1445	1565	7	21	C1F1
FB8559	2885	3005	7	21	C1F1
FB8559	7205	7325	7	21	C1F1
FB8559	8645	8765	7	21	C1F1
FB8559	10085	10205	7	21	C1F1
FB8559	11525	11645	7	21	C1F1

如表 4 中所示的 *FA2* 航班，其出发时间为 2050，即表示该飞机在 2021 年 8 月 12 日 10 时 10 分出发，起飞机场编号为 4，其表示为 *PGX*，其到达时间为 2140，即表示该飞机在 2021 年 8 月 12 日 11 时 40 分到达，到达机场编号为 2，其表示为 *NKX* 。并且可利用表 4，5 构建航班信息矩阵。分析讨论所有航班的最低资格配置均为 *C1F1*,故在航班信息矩阵表示中将这一信息组剔除，具体形式如 $\mathbf{F}_{1,:}=[2050 \quad 2140 \quad 4 \quad 2]$ 表示数据 *A* 中第一架航班的出发时间、到达时间，出发机场、到达机场数值分别为 2050，2140，4，2。矩阵的行向量表示航班的架次顺序，按照航班的起飞先后顺序进行排序。

5.2 机组人员信息预处理

对于机组人员信息的处理是进行航班分配机组人员必不可少的环节，此处只考虑飞行员这一类机组人员，飞行员信息包括有资格信息，基地、每单位小时执勤成本以及每单位小时任务环成本，为方便数据处理，现将数据做如下处理，满足主副机长资格进行赋1处理，不满足则进行赋0处理，基地标识按照机场编号规则进行处理，据此可构建包含有各飞行员信息的数据矩阵。员工编号次序按照数据集所给的顺序进行排序。处理后数据表格如下表所示：

表 6 数据 *A* 机组人员部分信息整理表

员工号	正机长	副机长	允许乘机	基地	每单位小时执勤成本	每单位小时任务环成本
A0001	1	0	1	2	680	20
A0002	1	0	1	2	680	20
A0003	1	0	1	2	680	20
⋮	⋮	⋮	⋮	⋮	⋮	⋮
A0020	0	1	1	2	600	20
A0021	0	1	1	2	600	20

表 7 数据 *B* 机组人员部分信息整理表

员工号	正机长	副机长	允许乘机	基地	每单位小时执勤成本	每单位小时任务环成本
B0001	1	1	1	7	640	20
B0002	1	1	1	7	640	20
B0003	1	1	1	7	640	20
B0004	1	1	1	7	640	20

⋮	⋮	⋮	⋮	⋮	⋮	⋮
B0462	0	1	1	24	600	20
B0463	0	1	1	24	600	20
B0464	0	1	1	24	600	20
B0465	0	1	1	24	600	20

利用表 5，6 构建机组人员的信息矩阵 $\mathbf{C}$ 具体形式如 $\mathbf{C}_1=[1\quad 0\quad 1\quad 2\quad 680\quad 20]$，表示数据 A 中第一位机组人员的正机长资格、副机长资格、允许登机资格、基地、每单位小时执勤成本、每单位小时任务环成本值分别为 1，0，1，2，680，20。矩阵的行向量表示机组人员的次序。

5.3 航班可换乘数据处理

机组人员为实现相应的排班周期任务，考虑航班间的空间连续性是必不可少的环节，只有可行换乘航班才可表示路径的导通意义，这对机组人员来说才有可能进行连续的航段，执勤以及任务环任务。考虑到排班周期的最小执行单位为航段任务，航班的换乘需要满足上一航班的到达机场与下一航班的出发机场为同一机场，为此对所有的航班的空间连接进行数据的可视化处理。

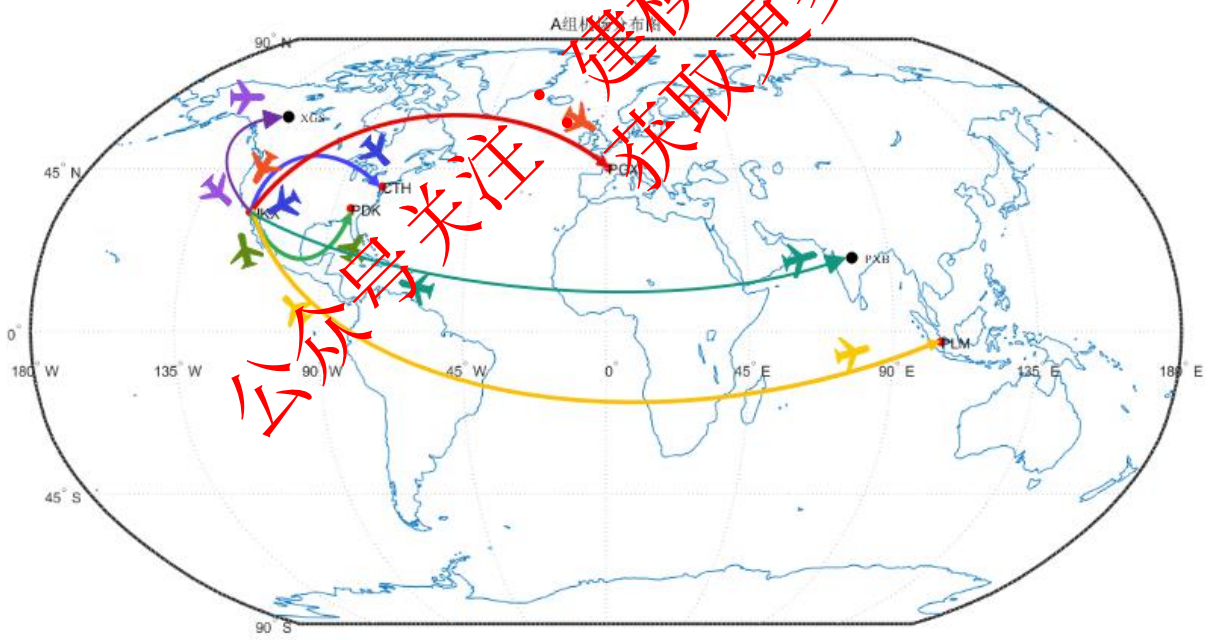


图 8　A 组航班空间连接示意图

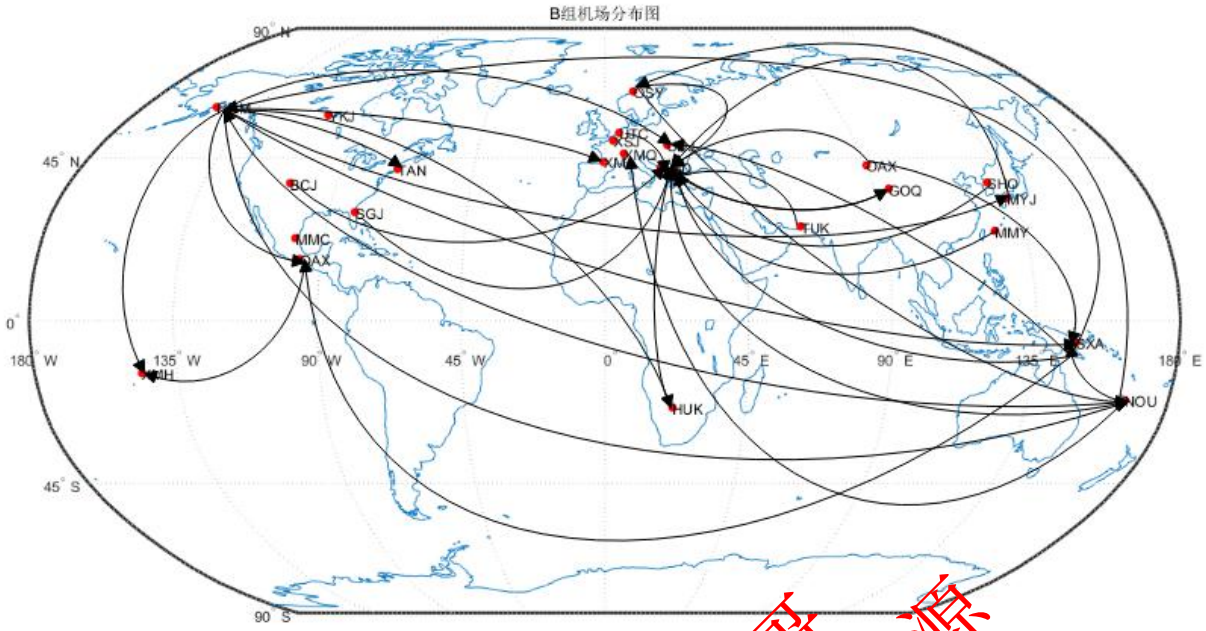


图 9　B 组航班空间连接示意图

但在实际问题的解决中可执行换乘并不简单的等价于航班在空间层面上的次序连接，还应当考虑换乘航班间满足的最小连接时间问题，即前一架航班的到达时间与可换乘航班的起飞时间应当有不小于 $MinCT=40$分钟 的时间间隔。

$$t_{k,1}-t_{j,2}\geq 40 \tag{1}$$

其中，$t_{k,1}$ 表示第 k 架航班的出发时间，$t_{j,2}$ 表示第 j 架航班的到达时间。为此对所有航班的连接进行重新的判断处理，并将结果储存在可执行换乘矩阵 D_{jk} 内，换乘矩阵的可视化数据如下图所示：

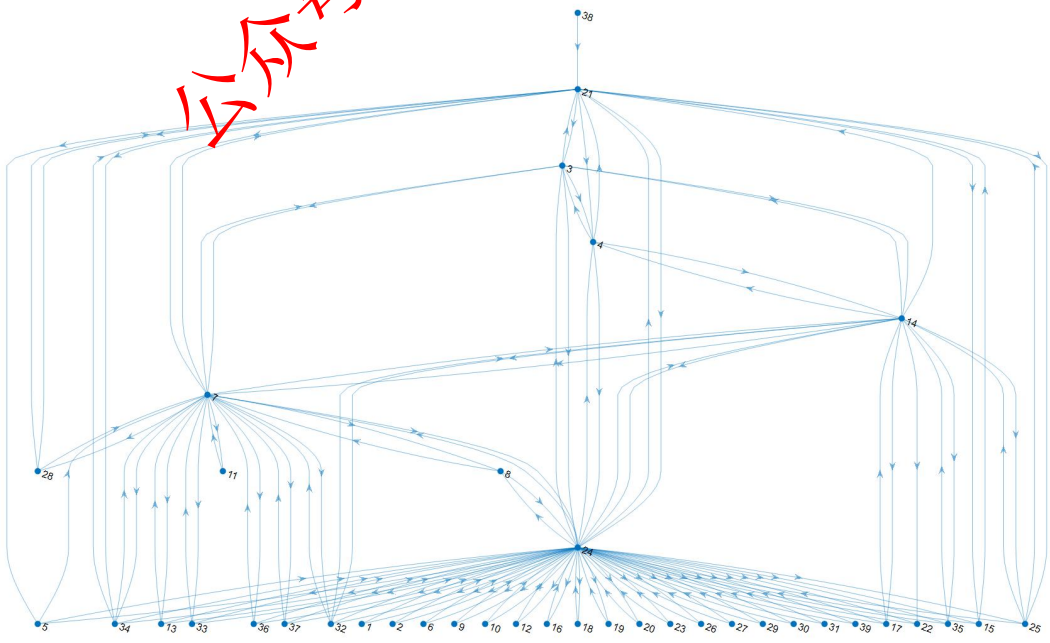


图 10　数据 B 考虑连接时间约束的航班连接图

六、模型的建立与求解

6.1 航班情况下机组人员分配问题

问题一的求解实际上是在只考虑航段情况下的机组人员排班分配，飞机的每一航班即表示机组人员的一次航段，航段之间需要满足一定的时间需求，空间上的连贯性也要求相邻两个航段需通过同一机场，并且为满足机组人员的实际生活需要，在所有的航段结束后，机组人员必须得回到出发时的基地。在满足以上实际约束条件的基础上，尽可能多的实现航班的最少机组配置，并且从时间利用的有效性上来说，总体乘机次数越多，时间利用效率越低。过多地使用替补资格也体现出分配方案模型的可行性不够高。为此，针对上述提到的优化目标以及需要考虑的约束，我们选择建立 0-1 整数规划模型进行求解。

优化的目标是要满足尽可能多的航班机组配置，同时满足尽可能少的总体乘机次数以及少使用替补资格，这是一个多目标规划问题，最大化航班满足机组配置数为最高规划目标，为了模型的方便求解，我考虑添加相应的权重将多目标合并成一个目标，其中第二、三目标添加反向权重将其转化为最大化目标，以达到多目标问题的求解。

6.1.1 航段排班分配整数规划模型

（一）决策变量

（1）$\mathbf{X}_{ij}$：第 i 位机组人员是否分配至第 j 架航班，$\mathbf{X}_{ij}$ 作为 0-1 变量，其具体的取值代表着实际分配方案，若表示第 i 位机组人员分配至第 j 架航班则对应的 $\mathbf{X}_{ij}$ 取值为 1，不分配则取值为 0。

实际中，每位机组人员根据其执行的任务可表示有三种临时状态身份，即在该次航班中作为正机长或者副机长的身份执行飞行任务，以及只是作为搭乘人员执行乘机任务，为此，我们考虑对决策变量进行分量化处理 $\mathbf{X}_{ij}=\left[x_{ij}^{c}\quad x_{ij}^{f}\quad x_{ij}^{p}\right]$，其内部值 $x_{ij}{}^{c}$、$x_{ij}{}^{f}$ 以及 $x_{ij}{}^{p}$ 分别表示三种可能的状态$\left[\text{正机长}\quad \text{副机长}\quad \text{搭乘人员}\right]$，$x_{ij}{}^{c}$、$x_{ij}{}^{f}$ 以及 $x_{ij}{}^{p}$ 作为 0-1 变量，其具体取值对应着第 i 位机组人员以何种身份分配至第 j 架航班。

（2）T_{ijk}：在进行换乘问题分析时，我们考虑引入换乘执行变量 T_{ijk}，换乘变量的具体含义表示为第位 i 机组人员是否由航班 j 换乘至航班 k，T_{ijk} 同样作为 0-1 变量，若第 i 位机组人员是否由航班 j 换乘至航班 k 则为 1，否则为 0。

（3）h_j：航班 j 是否满足起飞条件，若航班 j 的机组配置超过最低配置要求即航班 j 满足起飞条件故 $h_j=1$，反之航班 j 不满足起飞条件 $h_j=0$。

（二）目标函数

为了使机组人员的航班分配策略更加合理，需要满足以下三个目标：

1、主要目标是满足机组配置的航班数量最多：

$$Z_1=\max\sum_{j=1}^{n}h_j \tag{2}$$

其中 n 表示航班的总数，在 A 数据组中航班总数 $n=206$，在 B 数据组中航班总数 $n=13954$。

2、次要目标是机组人员的总体乘机次数最少，即机组人员仅作为乘坐人员搭乘航班的次数最少：

$$Z_4=\min\sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}{}^{p} \tag{3}$$

其中 m 表示待分配的机组人员个数，在 A 套数据中机组人员个数 $m=21$，在 B 套数据中机组人员个数 $m=465$。

3、次要目标是机组人员作为替补资格执行飞行任务的次数最少，即同时具有主要资格为正机长的机组人员替补副机长执行任务的次数最少：

$$Z_7=\min\sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}^{\ f}\cdot C_{i,1}\cdot C_{i,2} \tag{4}$$

在这里我们利用数据预处理中机组人员信息数据矩阵 $\mathbf{C}_{i,v}$，其中 $C_{i,1}$ 表示机组人员 i 是否具有正机长资格，若具有则为 1，没有则为 0；$C_{i,2}$ 表示机组人员 i 是否具有副机长资格，若具有则为 1，没有则为 0。

实际上上述目标函数的构建是多目标线性规划模型，求解这一类模型的简单思路是将以上三个目标函数赋以不同的权重，再通过线性加权的方式将多目标规划转化为单目标函数，设其权重分别为 ω_1、ω_4 和 ω_7，则转化后得到的单目标函数表示为：

$$\begin{aligned}Z&=\max(\omega_1Z_1-\omega_4Z_4-\omega_7Z_7)\\&=\max[\omega_1(\sum_{j=1}^{n}h_j)-\omega_4(\sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}^{\ p})-\omega_7(\sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}^{\ f}\cdot C_{i,1}\cdot C_{i,2})]\end{aligned} \tag{5}$$

（三）约束条件

1、状态约束：机组人员 i 搭乘航班 j 时的状态应满足其资格要求，即仅具有正机长资格的飞行员不能替补副机长执行飞行任务，仅具有副机长资格的飞行员只能作为副机长或者乘机人员搭乘飞机，同时具有正副机长资格的飞行员才能在必要时替补副机长执行飞行任务：

$$x_{ij}^{\ c}\le C_{i,1} \tag{6}$$

$$x_{ij}^{\ f}\le C_{i,2} \tag{7}$$

$$x_{ij}^{\ p}\le C_{i,3} \tag{8}$$

其中 i 与 j 的取值范围为 $i=1,2,\cdots,m$；$j=1,2,\cdots,n$。$C_{i,1}$ 表示机组人员 i 是否具有正机长资格，$C_{i,2}$ 表示机组人员 i 是否具有副机长资格，$C_{i,3}$ 表示机组人员 i 是否具有乘机资格，取值满足机组人员具有某类资格则为 1，没有则为 0。因为 x_{ij} 取值为 0 或 1，且所有机组人员均具有乘机资格，所以 $x_{ij}^{\ p}\le C_{i,3}$ 恒成立，故在后续模型中此约束条件略去。

2、最低机组资格配备约束：机组人员 i 搭乘航班 j 均有三种可能状态，但最多存在一种状态，不同状态不能同时存在，当其不搭乘时，三种状态都不存在；因为所有航班的最低机组配置相同均为 C1F1，故对任意一趟航班 j 而言，所有搭乘它的机组人员有且仅有一人担任正机长，一人担任副机长：

$$x_{ij}^{\ c}+x_{ij}^{\ p}+x_{ij}^{\ f}\le h_j(j=1,2,\cdots,n) \tag{9}$$

$$\sum_{i=1}^{m}x_{ij}^{\ c}=h_j(j=1,2,\cdots,n) \tag{10}$$

$$\sum_{i=1}^{m}x_{ij}^{\ f}=h_j(j=1,2,\cdots,n) \tag{11}$$

3、乘机人数约束：每一趟航班的乘机人数不大于 5 人：

$$\sum_{i=1}^{m}(x_{ij}^{\ c}+x_{ij}^{\ f}+x_{ij}^{\ p})\le 5h_j(j=1,2,\cdots,n) \tag{12}$$

4、**换乘约束**（相邻航段经过机场一致）：只有当航班 j 和航班 k 之间满足换乘条件时，机组人员 i 才可能从航班 j 换乘到航班 k；对于任一机组人员而言，在任一航班 j 之后不能同时换乘多趟航班，最多只能选择换乘一趟航班 k，且在换乘前后均需满足状态约束，在此换乘约束中利用了 D_{ij} 数据矩阵，在问题的考虑上已经将航班之间的连接时间约束考虑在内：

$$T_{ijk}\le D_{jk} \tag{13}$$

$$\sum_{k=1}^{n}T_{ijk}\le 1 \tag{14}$$

$$x_{ij}^{\ c}+x_{ij}^{\ f}+x_{ij}^{\ p}\ge T_{ijk} \tag{15}$$

$$x_{ik}^{\ c}+x_{ik}^{\ f}+x_{ik}^{\ p}\ge T_{ijk} \tag{16}$$

其中 i、j、k 的取值范围为 $i=1,2,\cdots,m$；$j=1,2,\cdots,n$；$k=1,2,\cdots,n$。其中 D_{jk} 为可执行换乘矩阵，具体数据由预处理中给出。

5、**机场进出约束**（每个机组人员初始从基地出发并最终回到基地）：对于任意机组人员 i 在任一机场进入和出去的次数应该相等；对于任一机场 u，以 u 作为出发地的航班记为 $j_{u,ini}$，总数为 $n_{u,ini}$，以 u 作为终点地的航班记为 $j_{u,fin}$，总数为 $n_{u,fin}$，有：

$$\sum_{j_{u,ini}=1}^{n_{u,ini}}x_{ij_{u,ini}}^{c,f,p}=\sum_{j_{u,fin}=1}^{n_{u,fin}}x_{ij_{u,fin}}^{c,f,p} \tag{17}$$

6、搭乘次数约束：任一机组人员 i 搭乘航班次数与其换乘航班次数相差恒为 1：

$$\sum_{j=1}^{n}x_{ij}^{c,f,p}-1=\sum_{j=1}^{n}\sum_{k=1}^{n}T_{ijk} \tag{18}$$

7、**连接时间约束：**

数据处理部分，在构建可执行换乘矩阵时，已将此约束考虑在内。

综上所述，问题一建立的基于机组人员航班分配问题的多目标线性规划模型总结如下：

$$Z=\max[\omega_1(\sum_{j=1}^{n}h_j)-\omega_4(\sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}^{\ p})-\omega_7(\sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}^{\ f}\cdot C_{i,1}\cdot C_{i,2})] \tag{19}$$

$$
s.t.\begin{cases}
x_{ij}^{c} \le C_{i,1} \\
x_{ij}^{f} \le C_{i,2} \\
x_{ij}^{p} \le C_{i,3} \\
x_{ij}{}^{c} + x_{ij}{}^{f} + x_{ij}{}^{p} \le h_j \ (j = 1,2,\cdots,n) \\
\sum_{i=1}^{m} x_{ij}{}^{c} = h_j \ (j = 1,2,\cdots,n) \\
\sum_{i=1}^{m} x_{ij}{}^{f} = h_j \ (j = 1,2,\cdots,n) \\
\sum_{i=1}^{m} (x_{ij}{}^{c} + x_{ij}{}^{f} + x_{ij}{}^{p}) \le 5h_j \ (j = 1,2,\cdots,n) \\
T_{ijk} \le D_{jk} \\
\sum_{k=1}^{n} T_{ijk} \le 1 \\
x_{ij}{}^{c} + x_{ij}{}^{f} + x_{ij}{}^{p} \ge T_{ijk} \\
x_{ik}{}^{c} + x_{ik}{}^{f} + x_{ik}{}^{p} \ge T_{ijk} \\
\sum_{j_{u,ini}=1}^{n_{u,ini}} x_{ij_{u,ini}} = \sum_{j_{u,fin}=1}^{n_{u,fin}} x_{ij_{u,fin}} \\
\sum_{j=1}^{n} x_{ij}^{c,f,p} - 1 = \sum_{j=1}^{n} \sum_{k=1}^{n} T_{ijk} \\
x_{ij}^{c,f,p}, T_{ijk}, h_j = 0 \ or \ 1
\end{cases}
$$

6.1.2 航段排班分配整数规划模型求解

（1）启发式算法步骤

Step1：导入预处理后按时间顺序排列的航班数据矩阵 $F_{n,4}$，机组人员数据矩阵 $C_{m,6}$ 以及可执行转机矩阵 $D_{n,n}$；

Step2：由数据矩阵计算得到属于基地所在机场的起飞航班编号数组 *base_1* 和降落航班编号数组 *base_2*；

Step3：随机选择一个编号 $i \in base\_1$，依据可执行转机矩阵和转接时间最短的启发条件，找到一条以 i 为起始航班，同时终止航班 $k \in base\_2$ 的航线。考虑到航班不重复利用，删去属于该航线的所有航班；

Step4：在集合 *base_1* 中剩余的编号 i 中重复 Step2 的流程，直至所有从该基地出发且最终回到基地的路线全部求得；

Step5：若存在多个基地，基于新基地再次执行 Step2-Step4；

Step6：重复 Step2-Step5 若干次，求得在编号先后次序随机的情况下得到未安排航班数最少的航线安排；

Step7：将可执行转机矩阵和转接时间最短作为启发条件，依次将航线进行组合作为一个机组成员组合即 1C1F 的航段任务编号数组，使得航线组合最少。

Step8：将航段任务编号数组对应可执行任务的机组成员个人并填入机组成员对应任务 0-1

变量矩阵 $x_{i,j}^{C,F,P}$ 。

Step9：根据 Step8 中得到的 0-1 变量矩阵进行部分求和得到总体乘机次数和总体替补资格。

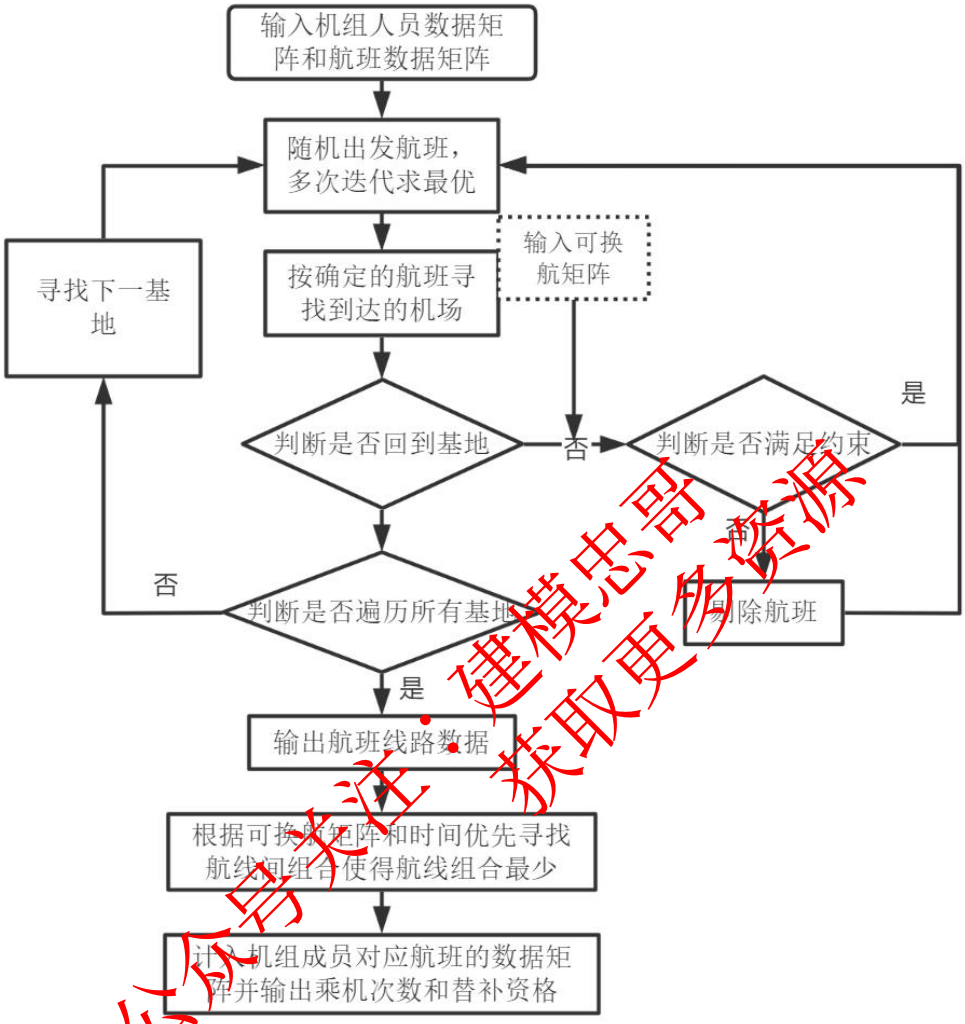


图 11 问题一求解流程图

（2）问题一模型求解结论

对于 A 组数据，将所有航班分为从基地出发的以及飞往基地的。将机组人员按属性分为只具有正机长资格，只具有副机长资格，正副资格兼有三种类别，率先考虑派遣主要资格人员执行飞行任务。

假设所有飞往基地的航班均满足最低配置要求，以 C1F1 的配置飞向基地，并且额外需要的正副飞行员数量由上一次从基地飞向该机场的航班搭载，可以据此倒推出每班从基地出发飞往各个机场的航班上需配备的正副机长数量。将最大乘机人数纳入考虑，将超量的乘机人员进一步分散到前几次从基地出发飞往各机场的航班上去，此时可得到人员分配的初始矩阵。

由于 A 组数据有 206 次飞机的起落，按时间顺序分为 412 个时间节点。因为 A 组中所有机组人员基地均为 2，所以初始时刻三种类型的机组人员以 5，10，6 的数量存在于机场 2 中，按照之前确定的人员分配规则并根据 412 个时间节点的划分，计算出各个时刻点 1-7 号机场地面上存在的三种人员的数量。根据得到的结果以最小连接时间 40 分钟为判断依据，判断时间维度下人员的流动是否满足最小连接时间约束，若不满足，则更新人员分配

矩阵，添加新的乘机人员使所有机组人员的换乘满足最小时间约束，如此进行迭代可以得到较优的人员分配方案。

根据上述方法计算得到机组人员航班分配如表 8 所示：

表 8 A 套数据航班分配结果评价

各项指标	结果
不满足机组配置航班数	0
满足机组配置航班数	206
机组人员总体乘机次数	8
替补资格使用次数	0
程序运行分钟数	0.1

需要参与飞行任务的机组人员其飞行任务的执行如表 9 所示：

表 9 A 套数据航班分配结果评价

机组人员	机场	航班	机场	航班	机场	航班	机场	航班	
A0001	2	1	4	2	2	7	1	9	…
A0002	2	1	4	14	2	19	3	22	…
A0003	2	3	7	5	2	11	5	13	…
A0004	2	3	7	17	2	26	5	28	…
A0011	2	4	3	6	2	8	6	10	…
A0012	2	1	4	2	2	7	1	9	…
A0013	2	1	4	14	2	19	3	22	…
A0014	2	3	7	5	2	11	5	13	…
A0015	2	3	7	17	2	26	5	28	…
A0016	2	4	3	6	2	8	6	10	…

此外，需要参与工作的正机长 5 人，副机长 5 人，需要进行乘机任务的机组人员将于航班 3、151、4、154 上进行搭载，均需搭载正副机长各一人。

对于 B 组数据，首先筛选出出发点不在基地，且起飞时刻之前未与基地存在航班交接的航班，这些航班不满足起飞要求，不予配置机组人员，根据计算，共有 11 架航班不满足要求。

由于较之 A 组，B 的航班数量更大，且存在非基地机场之间的人员交流，人员分配矩阵的迭代更为困难，故以首发航班作为初始航班，根据可执行换乘矩阵来判断其可建立的航班路径，以覆盖的航班数更大为优建立航班搜索路径，不能被覆盖到的航班定为不能起飞成功的航班，不予配置机组人员，以此来实现机组人员的合理分配。

得到的机组人员航班分配如表 10 所示：

表 10 B 套数据航班分配结果评价

各项指标	结果
不满足机组配置航班数	130
满足机组配置航班数	13824
机组人员总体乘机次数	240
替补资格使用次数	0
程序运行分钟数	1.0

6.2 机组人员执勤航班分配问题

问题二的求解实际上是在问题一航段的基础上引入执勤概念，执勤由航段组成，一组符合条件的航班组合即表示机组人员的一次执勤，在航段约束的基础上，执勤之间同样需要满足一定的时间需求，并且对于同一机组人员执勤之间也许符合空间上的连贯性，即上一执勤到达机场与下一执勤必须是同一机场，为满足机组人员的实际需求，每次执勤的飞行时间有着一定的时间限制，并且每个机组人员每天至多执行一次执勤，相邻执勤间有着规定的最短休息时间，在满足以上实际约束条件的基础上，尽可能减少机组人员的总体执勤成本，以及为体现出公平性，各机组人员之间的执勤时长应尽量保持平衡。为此，针对上述提到的优化目标以及需要考虑的约束，引入新的决策变量，我们同样选择建立 0-1 整数规划模型进行求解。

6.2.1 机组人员执勤整数规划模型

（一）决策变量

第二问的决策变量实际上是延续第一问中的决策变量，为避免重复性，此处不再具体给出。

（二）目标函数

1、对于每一位机组人员来说，其进行执勤任务时，每单位小时的执勤成本固定，相应的执勤成本包含在矩阵 $C_{i,5}$ 中，据此构建总体机组人员执勤成本最低目标函数为：

$$Z_2=\min\sum_i\sum_r t_{i,r}\cdot C_{i,5}/60$$
$$t_{i,r}=t_{i,r}^1+t_{i,r}^2=\sum_{j\in FS_r}(F_{j,2}-F_{j,1})+\sum_{j\in FS_r}\sum_{k\in FS_r}T_{ijk}(F_{k,1}-F_{j,2}) \tag{20}$$

此处为了便于模型的表述引入常量 FS_r，该常量表示第 r 天起飞的所有航班集合，则 $t_{i,r}$ 表示第 i 个机组人员在第 r 天的飞行时间，$t_{i,r}^1$ 表示第 i 位机组人员在第 r 天的飞行时间，$t_{i,r}^2$ 表示第 i 位机组人员在第 r 天的连接时间，式中的 T_{ijk} 引用问题一得到的求解结果，其具体的含义表示为第 i 位机组人员由第 j 架航班换乘至第 k 架航班，$F_{j,1},F_{j,2}$ 数据由航班信息矩阵给出，相应的 $F_{j,1},F_{j,2}$ 表示为第 j 架航班的出发时间以及到达时间。

2、各机组人员之间的执勤时长应保持平衡：

$$\begin{aligned}Z_5&=\min(\max|t_i-\bar{t}|)\\ t_i&=\sum_r t_{i,r}\\ \bar{t}&=(\sum_i\sum_r t_{i,r})/m\end{aligned} \tag{21}$$

包含有 min(max) 的非线性约束转化为线性约束，这里简单的思路想法是将单个 min(max)约束转化为 m 个 min 约束，对于非线性的绝对值形式是找出 $t_i-\bar{t}$ 的上下界 λ_i^1，λ_i^2，对上下界做差即可，即满足约束 $t_i-\bar{t}\le\lambda_i^1, t_i-\bar{t}\ge\lambda_i^2$，此时目标转化线性目标[1]为：

$$\begin{aligned}Z_5&=\min(\sum_i^m\lambda_i^1-\lambda_i^2)\\ t_i-\bar{t}&\le\lambda_i^1, t_i-\bar{t}\ge\lambda_i^2\end{aligned} \tag{22}$$

其中，t_i 表示第 i 位机组人员的总执勤时长，$\bar{t}$ 表示所有机组人员的平均执勤时长。为达到机组人员之间的执勤时长平衡的目标，我们考虑可以将所有的执勤时长中最大的时长

进行最小化约束，以此达到优化目的。

同样的，通过添加权重系数将多目标的线性规划模型转化为单目标模型，为此我们添加权重参数 ω_2，ω_5，转换后的单目标规划模型形式表示为：

$$Z=-\omega_2\cdot Z_2-\omega_5\cdot Z_5=\max[-\omega_2\cdot(\sum_i\sum_r t_{i,r}\cdot C_{i,5}/60)-\omega_5\cdot(\sum_i^m\lambda_i^1-\lambda_i^2)] \tag{23}$$

（三）约束条件

1、每个机组人员每天**至多只能执行一个执勤**：

该约束在构建常量 FS_t 的过程已将该条件进行了约束。

2、每次飞行时间的约束，每次**执勤飞行时间最多**不超过 $MaxBlk$ 分钟：

$$t_{i,r}^1=\sum_{j\in FS_t}(F_{j,2}-F_{j,1})\le MaxBlk \tag{24}$$

$t_{i,r}^1$ 表示第 i 位机组人员在第 r 天的执勤飞行时间。

3、**执勤时长最多**不超过 $MaxDP$ 分钟：

$$t_{i,r}=t_{i,r}^1+t_{i,r}^2=\sum_{j\in FS_t}(F_{j,2}-F_{j,1})+\sum_{j\in FS_t}(\sum_{k\in FS_t}T_{jik}(F_{k,1}-F_{j,2}))\le MaxDP \tag{25}$$

4、每个机组人员下一执勤的起始机场与上一执勤的结束**机场需保持一致**。在对这一约束进行分析时，可以得到，由于利用了可换乘航班矩阵 D_{jk} 的处理信息，这在实际上是已经直接考虑这一约束。

5、每个机组人员在相邻的两个**执勤之间需要满足不小于** $Min\,Rest$ 分钟的休息时间：

$$T_{ijk}(F_{k,1}-F_{j,2})\ge Min\,Rest，\text{其中}j\in FS_t,k\in FS_{t^\wedge},t<t^\wedge \tag{26}$$

为避免 T_{ijk} 在为 0 的情况下陷入无法满足上述初始约束的问题，对上述提出的约束条件进行改进优化，得到下式：

$$T_{ijk}(F_{k,1}-F_{j,2})+(1-T_{ijk})\cdot Min\,Rest\ge Min\,Rest，\text{其中}j\in FS_t,k\in FS_{t^\wedge},t<t^\wedge \tag{27}$$

综上所述，问题二建立的引入执勤后机组人员航班分配问题的多目标线性规划模型总结如下，由于执勤情况下机组人员航班分配问题是在问题一的基础上建立的，为此需要同时将问题一建立的模型引入：

$$\begin{aligned}Z=\max[&\omega_1(\sum_{j=1}^n h_j)-\omega_2\cdot(\sum_i\sum_r t_{i,r}\cdot C_{i,5}/60)\\&-\omega_4(\sum_{i=1}^m\sum_{j=1}^n x_{ij}^p)-\omega_5(\sum_i^m\lambda_i^1-\lambda_i^2)-\omega_7(\sum_{i=1}^m\sum_{j=1}^n x_{ij}^f\cdot C_{i,1}\cdot C_{i,2})]\end{aligned} \tag{28}$$

$$
s.t.\begin{cases}
x_{ij}^{c} \le C_{i,1} \\
x_{ij}^{f} \le C_{i,2} \\
x_{ij}^{p} \le C_{i,3} \\
x_{ij}^{\ c} + x_{ij}^{\ f} + x_{ij}^{\ p} \le h_j \ (j = 1,2,\cdots,n) \\
\sum_{i=1}^{m} x_{ij}^{\ c} = h_j \ (j = 1,2,\cdots,n) \\
\sum_{i=1}^{m} x_{ij}^{\ f} = h_j \ (j = 1,2,\cdots,n) \\
\sum_{i=1}^{m} (x_{ij}^{\ c} + x_{ij}^{\ f} + x_{ij}^{\ p}) \le 5h_j \ (j = 1,2,\cdots,n) \\
T_{ijk} \le D_{jk} \\
\sum_{k=1}^{n} T_{ijk} \le 1 \\
x_{ij}^{\ c} + x_{ij}^{\ f} + x_{ij}^{\ p} \ge T_{ijk} \\
x_{ik}^{\ c} + x_{ik}^{\ f} + x_{ik}^{\ p} \ge T_{ijk} \\
\sum_{j_{u,ini}=1}^{n_{u,ini}} x_{ij_{u,ini}} = \sum_{j_{u,fin}=1}^{n_{u,fin}} x_{ij_{u,fin}} \\
\sum_{j=1}^{n} x_{ij}^{c,f,p} - 1 = \sum_{j=1}^{n}\sum_{k=1}^{n} T_{ijk} \\
t_{i,r}^{1} = \sum_{j \in FS_t} (F_{j,2} - F_{j,1}) \le MaxBlk \\
t_i - \bar{t} \le \lambda_i^1, \bar{t} - t \ge \lambda_i^2 \\
t_{i,r} = t_{i,r}^{1} + t_{i,r}^{2} = \sum_{j \in FS_t} (F_{j,2} - F_{j,1}) + \sum_{j \in FS_t} (\sum_{k \in FS_t} T_{jik}(F_{k,1} - F_{j,2})) \le Max\ DP \\
T_{ijk}(F_{k,1} - F_{j,2}) \ge Min\,\mathrm{Re}\,st, \ \text{其中} j \in FS_t, k \in FS_{t^{\wedge}}, t < t^{\wedge} \\
x_{ij}^{c,f,p}, \ T_{ijk}, h_j = 0 \ or \ 1
\end{cases}
$$

6.2.2 机组人员执勤整数规划求解

（1）启发式算法步骤

Step1：导入预处理后按时间顺序排列的航班数据矩阵 $F_{n,4}$，机组人员数据矩阵 $C_{m,6}$ 以及可执行转机矩阵 $D_{n,n}$；

Step2：将航班数据矩阵拆分成共 r 个航班数据矩阵 F_r，r 为航班数据所涵盖的天数，分别求出第 r 天的可执行转机矩阵 D_r；

Step3：初始化目标满足配置航班数 $H=0$，总体执勤成本 $Cost=\infty$，总体乘机次数 $x=\infty$，执勤时长标准差 $T_{std}=\infty$，替补资格人次 $num=\infty$；

Step4：依据时序和可执行转机矩阵 D_r 的约束，随机生成执勤单元 $duty_{r,s}$，执勤单元满足包含航班的总飞行时长 $T_{飞} \le MaxBlk$，执勤单元总时长 $T_{duty} \le MaxDP$，相邻执勤单位之间的时间 $t_{duty1,2} \le MinRest$；

Step5：由随机生成的执勤单元 $duty_{r,s}$ 计算 Step3 中的所有参数的现有值；
Step6：若新生成的满足配置的航班数大于原有航班数，则替换 H；
Step7：H 相等时，若新生成的执勤成本小于原执勤成本，则替换 Cost;
Step8：Cost 相等时，若总体乘机次数小于原乘机次数，则替换 x;
Step9：x 相等时，若执勤时长标准差小于原标准差，则替换 T_{std};
Step10：T_{std} 相等时，若替补资格人次小于原人次，则替换 num;
Step11：设置迭代次数 n 为 10，50，100，1000 等，重复 Step3-Step10，在不断迭代的情况下逐渐求取执勤单元 $duty_{r,s}$ 的较优可行解；
Step12：输出对应执勤单元 $duty_{r,s}$ 较优可行解的机组人员航班任务矩阵。

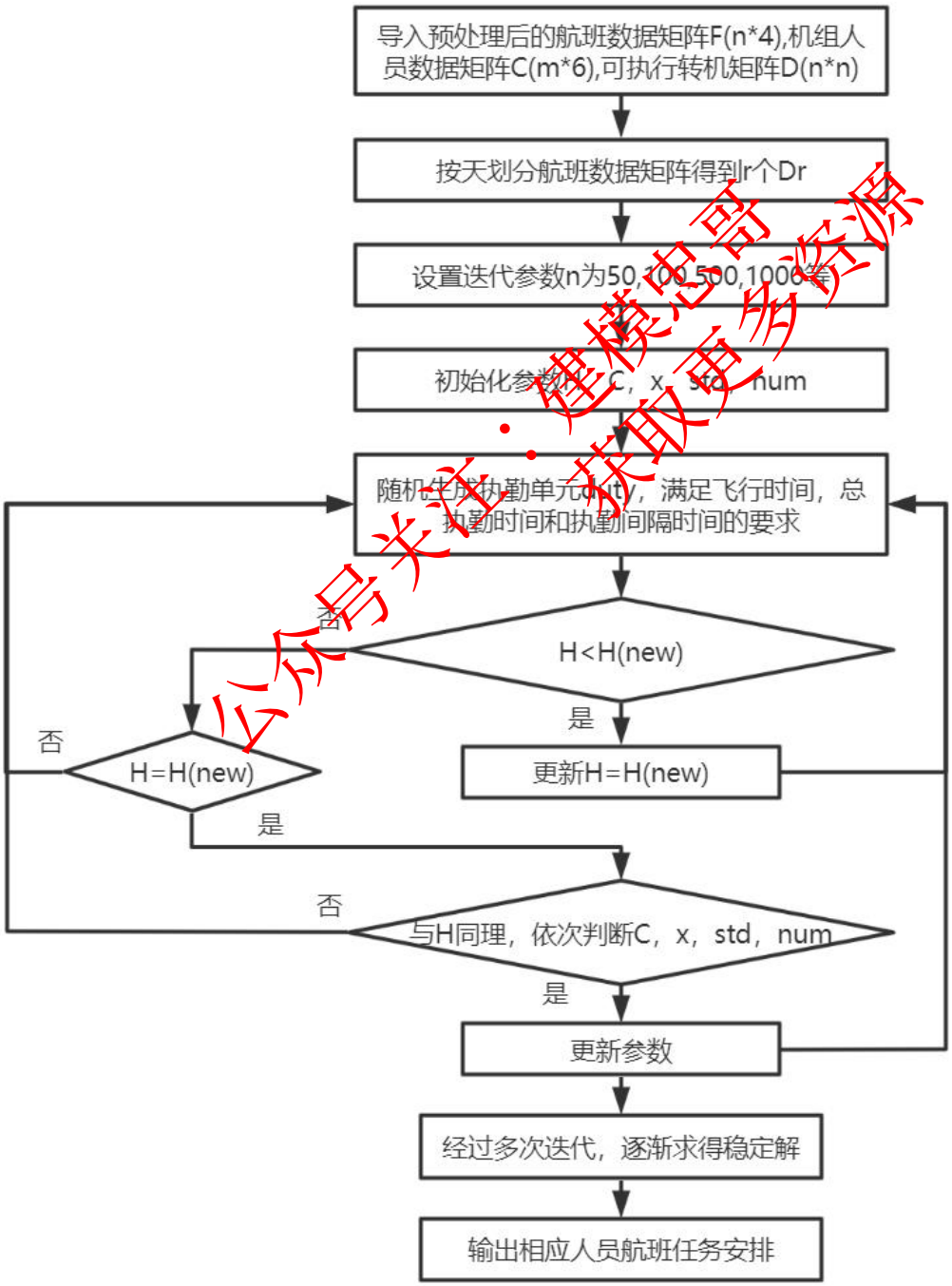


图 12 问题二求解流程图

（2）问题二模型求解

对于 A 套数据，在问题一的基础上进一步增加执勤次数、飞行时间、执勤时间等约束，在问题一得到的机组人员航班配置的方案的基础上验算其是否满足新增加的约束条件。

因为执勤里的航班须在同一天起飞，故将所有航班以起飞时间为准按照天数进行划分，从 8 月 11 日到 8 月 25 日共计 15 天，按照时间顺序排列航班，15 天依次起飞的航班数为 10、15、12、14、13、14、14、16、14、14、14、14、14、14、14 架。

对问题一中 A 组的解进行验算，发现其满足飞行时间，但不满足执勤时间，故在其本身机组人员分配方案的基础上进行调整，在某一机组人员当天执勤时间超过最长执勤时间 720 分钟时，分配其他机组人员来完成其当天的剩余飞行任务，，以总执勤成本最低以及执勤时长平衡分配为优化目标，得到 A 套数据优化后的机组人员航班分配，共计需要正机长资格人员 5 人，副机长资格人员 10 人，兼具正副机长资格人员 5 人，其航班分配结果评价如表 11 所示：

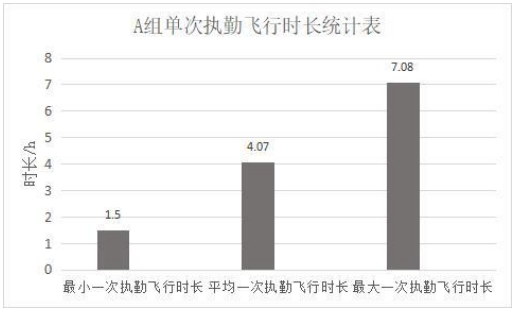
表 11 A 套数据执勤航班分配结果评价

各项指标	结果
不满足机组配置航班数	0
满足机组配置航班数	206
机组人员总体乘机次数	8
替补资格使用次数	0
程序运行分钟数	0.1
机组总体利用率	72.4%
最小/平均/最大一次执勤飞行时长	1.5;4.07;7.08
最小/平均/最大一次执勤执勤时长	1.5;5.32;11.58
最小/平均/最大机组人员执勤天数	1；2.2826；4
总执勤成本/万元	44.149
程序运行分钟数/分钟	0.1

对于 B 套数据，航班分配结果如表 12 所示。

表 12 B 套数据执勤航班分配结果评价

各项指标	结果
不满足机组配置航班数	2034
满足机组配置航班数	11920
机组人员总体乘机次数	2528
替补资格使用次数	0
程序运行分钟数	1.2
机组总体利用率	65.15%
最小/平均/最大一次执勤飞行时长	0.75;6.34;12
最小/平均/最大一次执勤执勤时长	0.75;7.01;12
最小/平均/最大机组人员执勤天数	1;6.5648;10
总执勤成本/万元	4214.5
程序运行分钟数/分钟	1.5



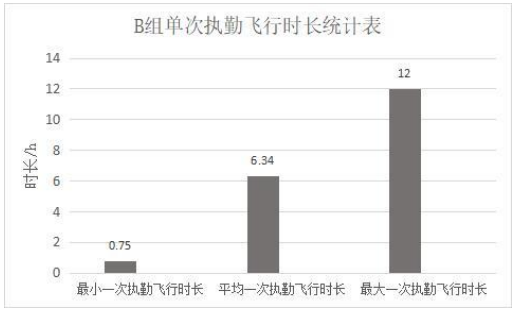


图 13 单次执勤飞行时长统计图

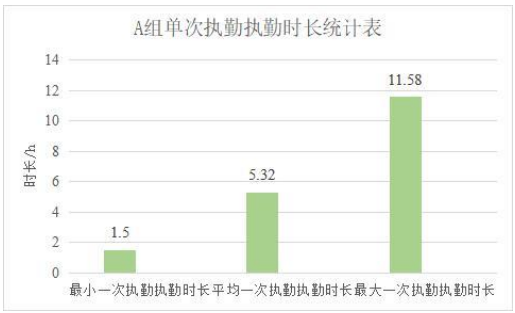
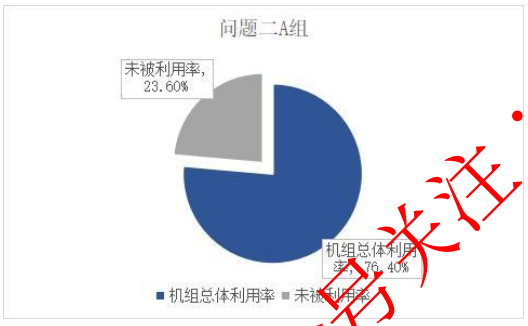


图 14 单次执勤执勤时长统计图



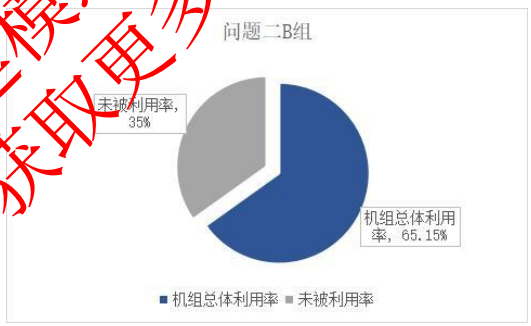


图 15 机组总体利用率统计图

6.3 机组人员排班计划问题

若能做到每一环节求解结果是最优的，则问题三的处理实际上是利用问题二构建的单个任务环计划进行组合形成排班计划。排班计划由一系列任务环和休假组成，休假的时间同样需要满足一点的约束，并且排班周期通常是两周计划或者是月计划。在执勤约束的基础上，任务环之间需要满足一定的时间约束，但对于任务环空间上的连接问题，由于每一任务环结束时均会回到起始起点，所以并不需要单独考虑。为满足机组人员的实际需求，每个排版周期的任务环总时长有最长时间约束，任务环之间有着最短的休息时间，为避免构建的任务环过大，对连续执勤天数也有着一定的限制。在上述约束的基础上，尽可能使总体任务环成本最低，机组人员间的任务环时长尽可能平衡，为此我们考虑建立任务环连接规划模型。

6.3.1 任务环连接规划模型

（一）决策变量

（1）类比于可换乘矩阵，这里我们构建任务环间的可连接矩阵 R_{ijk}，其中 $j \in j_{u,ini}, k \in j_{u,fin}$，任务环间的连接时间有着最短休息时间的约束，即 $F_{k,1} - F_{j,2} \geq MinVacDay \cdot 1440$

$$R_{ijk}=\begin{cases}0\\1\end{cases},j\in j_{u,fin},k\in j_{u,ini},F_{k,1}-F_{j,2}\geq MinVacDay\cdot 1440 \tag{29}$$

R_{ijk} 表示第 i 位机组人员由第 j 架航班换乘至第 k 架航班，此处的换乘概念不同于航段换乘，是属于两个不同任务环之间的换乘，可以由之后的约束体现，$j_{u,fin}$ 表示以 u 作为到达地的航班，表示以 u 作为起飞地的航班。$F_{k,1}$ 表示 k 航班的起飞时间，$F_{j,2}$ 表示 j 航班的到达时间，$MinVacDay$ 表示相邻任务环间最少休息天数，转化为相应的分钟量纲是需要乘以1440。

（2）为方便计算求解，我们这里引入机组人员第一架起飞航班的搜索矩阵 $RF_{i,j}$，若第 j 架航班是第 i 位机组人员的第一架起飞航班则对应矩阵元素取 1，其余元素对应为 0。为求得这一搜索矩阵，我们构建以下约束：

$$x_{ij}\leq\sum_{v=1}^{j}RF_{i,v}\leq 1,j=(1,2,...,n) \tag{30}$$

此处的 j 为实现寻找第一架航班的目标，已将其按照时间的先后顺序进行处理，n 表示航班的总架数。此处对于 i 来说需要同时满足 n 个约束。

（二）目标函数

1、机组人员排版周期任务环总时长约束：

$$Z_3=\min\sum_i TH_i\cdot(C_{i,6}/60) \tag{31}$$

其中，TH_i 表示第 i 位机组人员的任务环时长，$C_{i,6}$ 表示第 i 位机组人员的每单位小时任务环成本，任务环总时长由以下算式得出：

$$\begin{aligned}TH_i&=TH_i^2+TH_i^3-TH_i^1\\&=\sum_j\sum_k T_{ijk}\cdot(F_{k,2}-F_{j,2})+\sum_v RF_{i,v}\cdot(F_{v,2}-F_{v,1})-\sum_j\sum_k T_{ijk}\cdot B_{ijk}\cdot(F_{k,1}-F_{j,2})\end{aligned} \tag{32}$$

$TH_i^2=\sum_j\sum_k T_{ijk}\cdot(F_{k,2}-F_{j,2})$ 表示第 i 位机组人员最后一次到达时间与第一次航班到达时间的差值；$TH_i^3=\sum_v RF_{i,v}\cdot(F_{v,2}-F_{v,1})$ 表示第 i 位机组人员的第一次航班的飞行时间；$TH_i^1=\sum_j\sum_k T_{ijk}\cdot B_{ijk}\cdot(F_{k,1}-F_{j,2})$ 表示第 i 位机组人员任务环间隙时间。

2、各机组人员之间的任务环时长尽可能平衡约束：

$$\begin{aligned}Z_6&=\min(\max|TH_i-\bar{TH}|)\\\bar{TH}&=(\sum_i TH_i)/m\end{aligned} \tag{33}$$

其中，$\bar{TH}$ 表示所有机组人员任务环总时长的平均值。

同样的处理方法，取 $TH_i-\bar{TH}\leq\lambda_i^3,TH_i-\bar{TH}\geq\lambda_i^4$，此时目标函数转化为

$$\begin{aligned}&\min[\sum_i^m(\lambda_i^3-\lambda_i^4)]\\&TH_i-\bar{TH}\leq\lambda_i^3,TH_i-\bar{TH}\geq\lambda_i^4\end{aligned} \tag{34}$$

此处我们添加权重参数 ω_3,ω_6 将多目标规划模型转化为单目标模型，转化后形式如下所示：

$$Z=\max(-\omega_3\cdot Z_3-\omega_6\cdot Z_6)=\max[-\omega_3\cdot\sum_i TH_i\cdot(C_{i,6}/60)-\omega_6\cdot\sum_i^m(\lambda_i^3-\lambda_i^4)] \tag{3}$$

（三）约束条件

1、机组人员排班周期的**任务环总时长约束：**

$$TH_i\le MaxTAFB \tag{34}$$

TH_i表示第i位机组人员的任务环时长，对于每一位机组人员均要满足上述约束；

2、**任务环间隔约束：**

在构造任务环连接矩阵时，其中已将该约束考虑在内，故此处不再重复给出。

3、每位机组人员**连续执勤天数约束：**

$$\sum_{r=r_0}^{r_0+g}\sum_{\substack{j\in FS_r\\k\in FS_{r+1}}}T_{ijk}\le g, r_0=1,2,...,R-MaxSuccOn$$

$$g=MaxSuccOn-1 \tag{35}$$

此处是利用航段换乘矩阵，换乘的次数实际上与执勤次数之间存在着个数为 1 的差值，因此取$g=MaxSuccOn-1$，$j\in FS_r$表示j航班属于第r天的航班集合内，$k\in FS_{r+1}$表示k航班属于第$r+1$天的航班集合内。R表示排班的周期，根据双周计划以及月计划取不同值。

综上所示，结合问题一、问题二建立规划模型如下所示：

$$\begin{aligned}Z=&=\max[\omega_1(\sum_{j=1}^n h_j)-\omega_2\cdot(\sum_i\sum_r t_{i,r}\cdot C_{i,5}/60)-\omega_3\cdot\sum_i TH_i\cdot(C_{i,6}/60)\\&-\omega_4(\sum_{i=1}^m\sum_{j=1}^n x_{ij}^p)-\omega_5(\sum_i^m(\lambda_i^1-\lambda_i^2))-\omega_6\cdot\sum_i^m(\lambda_i^3-\lambda_i^4)-\omega_7(\sum_{i=1}^m\sum_{j=1}^n x_{ij}^f\cdot C_{i,1}\cdot C_{i,2})]\end{aligned} \tag{36}$$

$$
s.t.\begin{cases}
x_{ij}^{c} \le C_{i,1}, x_{ij}^{f} \le C_{i,2}, x_{ij}^{p} \le C_{i,3} \\
x_{ij}{}^{c} + x_{ij}{}^{f} + x_{ij}{}^{p} \le h_j (j = 1,2,\cdots,n) \\
\sum_{i=1}^{m} x_{ij}{}^{c} = h_j (j = 1,2,\cdots,n) \\
\sum_{i=1}^{m} x_{ij}{}^{f} = h_j (j = 1,2,\cdots,n) \\
\sum_{i=1}^{m} (x_{ij}{}^{c} + x_{ij}{}^{f} + x_{ij}{}^{p}) \le 5h_j (j = 1,2,\cdots,n) \\
T_{ijk} \le D_{jk}, \sum_{k=1}^{n} T_{ijk} \le 1 \\
x_{ij}{}^{c} + x_{ij}{}^{f} + x_{ij}{}^{p} \ge T_{ijk} \\
x_{ik}{}^{c} + x_{ik}{}^{f} + x_{ik}{}^{p} \ge T_{ijk} \\
\sum_{j_{u,ini}=1}^{n_{u,ini}} x_{ij_{u,ini}} = \sum_{j_{u,fin}=1}^{n_{u,fin}} x_{ij_{u,fin}} \\
\sum_{j=1}^{n} x_{ij}^{c,f,p} - 1 = \sum_{j=1}^{n} \sum_{k=1}^{n} T_{ijk} \\
t_{i,r}^{1} = \sum_{j \in FS_t} (F_{j,2} - F_{j,1}) \le MaxBlk \\
t_i - \bar{t} \le \lambda_i^1, t_i - \bar{t} \ge \lambda_i^2 \\
R_{ijk} = \begin{cases} 0 \\ 1 \end{cases}, j \in j_{u,fin}, k \in j_{u,ini}, F_{k,1} - F_{j,2} \ge MinVacDay \cdot 1440 \\
x_{ij} \le \sum_{v=1}^{j} RF_{i,v} \le 1, j = (1,2,...,n) \\
TH_i = \sum_{j} \sum_{k} T_{ijk} \cdot (F_{k,2} - F_{j,2}) + \sum_{v} RF_{i,v} \cdot (F_{v,2} - F_{v,1}) - \sum_{j} \sum_{k} T_{ijk} \cdot B_{ijk} \cdot (F_{k,1} - F_{j,2}) \\
TH_i \le MaxTAFB \\
\bar{TH} = (\sum_{i} TH_i) / m \\
\sum_{r=r_0}^{r_0+g} \sum_{\substack{j \in FS_r \\ k \in FS_{r+1}}} T_{ijk} \le g, r_0 = 1,2,...,R - MaxSuccOn, g = MaxSuccOn - 1 \\
TH_i - \bar{TH} \le \lambda_i^3, TH_i - \bar{TH} \ge \lambda_i^4 \\
t_{i,r} = t_{i,r}^{1} + t_{i,r}^{2} = \sum_{j \in FS_t} (F_{j,2} - F_{j,1}) + \sum_{j \in FS_t} (\sum_{k \in FS_t} T_{jik} (F_{k,1} - F_{j,2})) \le MaxDP \\
T_{ijk} (F_{k,1} - F_{j,2}) \ge Min\,\mathrm{Re}\,st, \text{ 其中} j \in FS_t, k \in FS_{\hat{t}}, t < t\hat{} \\
x_{ij}^{c,f,p}, \ T_{ijk}, h_j = 0 \ or \ 1
\end{cases}
$$

6.3.2 任务环连接规划模型求解

（1）启发式算法步骤

Step1：导入预处理后按时间顺序排列的航班数据矩阵 $F_{n,4}$，机组人员数据矩阵 $C_{m,6}$ 以及可执行转机矩阵 $D_{n,n}$；

Step2：将航班数据矩阵拆分成共 r 个航班数据矩阵 F_r，r 为航班数据所涵盖的天数，分别求出第 r 天的可执行转机矩阵 D_r；

Step3：初始化目标满足配置航班数 $H=0$，总体执勤成本 $Cost=\infty$，总体乘机次数 $x=\infty$，执勤时长标准差 $T_{std}=\infty$，替补资格人次 $num=\infty$，任务环总成本 $Pay=\infty$，任务环时长标准差 $R_{std}=\infty$；

Step4：依据时序和可执行转机矩阵 D_r 的约束，随机生成执勤单元 $duty_{r,s}$ 和任务环单元 $pairing_t$ 满足包含航班的总飞行时长 $T_{飞} \le MaxBlk$，执勤单元总时长 $T_{duty} \le MaxDP$，相邻执勤单位之间的时间 $t_{duty1,2} \le MinRest$，机组人员排班周期任务环总时长 $T_{pairing} \le MaxTAFB$，相邻任务环之间的时长 $t_{pairing1,2} \le MinVacDay$，机组人员连续执勤天数 $\sum duty \le MaxSuccOn$；

Step5：由随机生成的执勤单元 $duty_{r,s}$ 计算 Step3 中的所有参数的现有值；

Step6：若新生成的满足配置的航班数大于原有航班数，则替换 H；

Step7：H 相等时，若新生成的执勤成本小于原执勤成本，则替换 Cost；

Step8：Cost 相等时，若新生成的任务环成本小于原任务环成本，则替换 Pay；

Step9：Pay 相等时，若总体乘机次数小于原乘机次数，则替换 x；

Step10：x 相等时，若执勤时长标准差小于原标准差，则替换 T_{std}；

Step11：T_{std} 相等时，若任务环时长标准差小于原标准差，则替换 R_{std}；

Step12：R_{std} 相等时，若替补资格人次小于原人次，则替换 num；

Step13：设置迭代次数 n 为 10，50，100，1000 等，重复 Step3-Step10，在不断迭代的情况下逐渐求取执勤单元 $duty_{r,s}$ 的较优可行解；

Step14：输出对应执勤单元 $duty_{r,s}$ 较优可行解的机组人员航班任务矩阵。

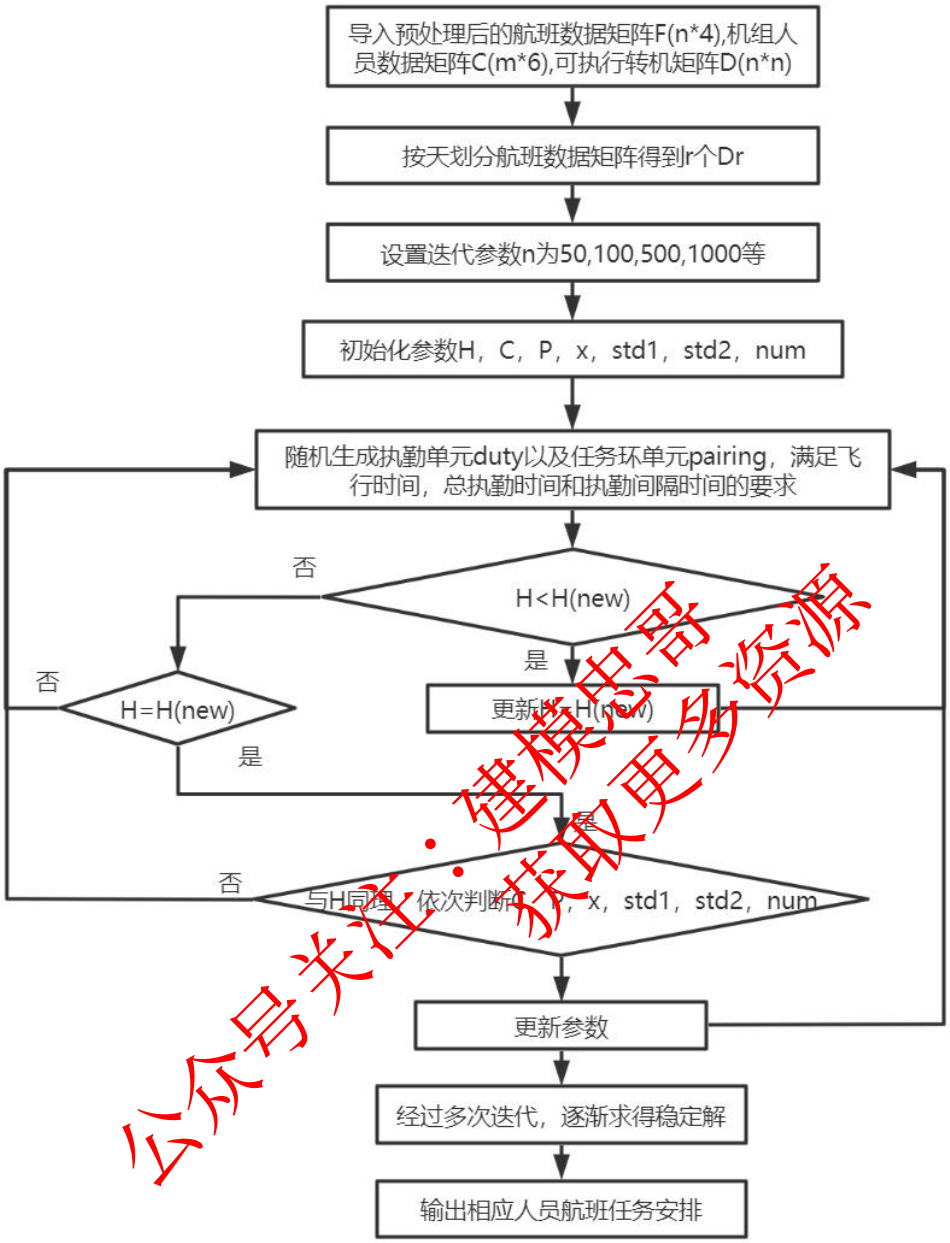


图 16 问题三求解流程图

（2）问题三模型求解

对于 A 套数据，航班分配结果如表 13 所示。

表 13 A 套数据执勤航班分配结果评价

各项指标	结果
不满足机组配置航班数	0
满足机组配置航班数	206
机组人员总体乘机次数	48
替补资格使用次数	30
程序运行分钟数	0.2

对于 B 套数据，航班分配结果如表 14 所示。

表 14 B 套数据执勤航班分配结果评价

各项指标	结果
不满足机组配置航班数	2256
满足机组配置航班数	11698
机组人员总体乘机次数	3564
替补资格使用次数	124
程序运行分钟数	12.54

七、模型的评价与推广

7.1 模型的优点

（1）本文针对航空公司机组人员的排班问题进行研究，建立了单目标线性优化模型，并利用两组实验数据进行可行性分析。求解的结果表明，本文建立的模型能够有效地解决机组人员排班问题，并能够得到合理的排班方案。

（3）建立的机组人员排班线性模型可解释性强，便于人们理解。

（2）基于本文构建的模型，设计启发式算法，有效降低直接求解模型带来的困难，算法难度的降低有效地增强了模型的实用性。

7.2 模型的缺点

（1）建立的线性模型对数据的准确性要求高，且求解该模型的计算量大，在高维数据下难以进行快速地求解。

（2）设计地启发式算法并不能找到全局最优解，只能找到局部最优解。

（3）模型的约束进行了理论上的简化处理，这对于模型应用到实际中有着一定的影响。

7.3 模型的推广

本文从航空公司机组排班问题出发，综合考虑航班的约束条件，使模型有一定的应用价值。航空机组人员的排班模型的建立对于生活中其他类似规划问题也起着一定的借鉴作用。

八、参考文献

[1]何尚录,白利华.解一类最大值最小化问题的线性规划方法[J].兰州铁道学院学报,1997(03):83-86.

[2]赵天洋. 航班机组自动排班系统研究[D].沈阳理工大学,2016.

[3]潘海洋. 无初始解的大规模机组排班问题建模与求解优化[D].清华大学,2014.

[4]张米. 航空公司机组排班模型研究[D].清华大学,2014.

[5]吴宇光. 航空公司机组排班的优化模型研究与算法实现[D].复旦大学,2012.

九、附件

附录 1：pretreatment.m 预处理，将 A 组和 B 组的时间转变为时间戳，机场按编号转换。

```
clear
clc
%A 组时间转换
Adptrdate=importdata('Adptrdate.txt');
Adptrdate=str2num(cell2mat(Adptrdate));
Adptrtime=importdata('Adptrtime.txt');
Aarrvdate=importdata('Aarrvdate.txt');
Aarrvdate=str2num(cell2mat(Aarrvdate));
Aarrvtime=importdata('Aarrvtime.txt');
for i=1:length(Adptrdate)
    t1(i)=A_transformation(Adptrdate(i),Adptrtime(i,:));
    t2(i)=A_transformation(Aarrvdate(i),Aarrvtime(i,:));
end
t=[t1;t2;(t2-t1)];
t=t';
%xlswrite('E:\A_flight.xlsx',t,'A1');
%A 组机场标号
Adptrstn=importdata('Adptrstn.txt');
Aarrvstn=importdata('Aarrvstn.txt');
Adptrstn=char(Adptrstn);
Aarrvstn=char(Aarrvstn);
for i=1:length(Adptrstn)
    for j=1:3
        stn1(i,j)=abs(Adptrstn(i,j));
        stn2(i,j)=abs(Aarrvstn(i,j));
    end
end
for i=1:206
    if stn1(i,1)==67
        Astation(i)=1;
    elseif stn1(i,1)==78
        Astation(i)=2;
    elseif stn1(i,1)==80&&stn1(i,2)==68
        Astation(i)=3;
    elseif stn1(i,1)==80&&stn1(i,2)==71
        Astation(i)=4;
    elseif stn1(i,1)==80&&stn1(i,2)==76
        Astation(i)=5;
    elseif stn1(i,1)==80&&stn1(i,2)==88
        Astation(i)=6;
```

```
    else
        Astation(i)=7;
    end
end
Astation=Astation';
for i=1:206
    if stn2(i,1)==67
        Astation(i,2)=1;
    elseif stn2(i,1)==78
        Astation(i,2)=2;
    elseif stn2(i,1)==80&&stn2(i,2)==68
        Astation(i,2)=3;
    elseif stn2(i,1)==80&&stn2(i,2)==71
        Astation(i,2)=4;
    elseif stn2(i,1)==80&&stn2(i,2)==76
        Astation(i,2)=5;
    elseif stn2(i,1)==80&&stn2(i,2)==88
        Astation(i,2)=6;
    else
        Astation(i,2)=7;
    end
end
pretreatmentA=[t,Astation];
xlswrite('E:\A_flight.xlsx',pretreatmentA,'A1');
%B 组时间转换
Bdptrdate1=importdata('Bdptrdate.txt');
for i=1:length(Bdptrdate1)
    Bdptrdate(i)=str2num(cell2mat(Bdptrdate1(i)));
end
Bdptrtime=importdata('Bdptrtime.txt');
Barrvdate1=importdata('Barrvdate.txt');
for i=1:length(Barrvdate1)
    Barrvdate(i)=str2num(cell2mat(Barrvdate1(i)));
end
Barrvtime=importdata('Barrvtime.txt');
for i=1:length(Bdptrdate)
    Bt1(i)=B_transformation(Bdptrdate(i),Bdptrtime(i,:));
    Bt2(i)=B_transformation(Barrvdate(i),Barrvtime(i,:));
end
Bt=[Bt1;Bt2;(Bt2-Bt1)];
Bt=Bt';
%B 组机场标号
Bdptrstn=importdata('Bdptrstn.txt');
Barrvstn=importdata('Barrvstn.txt');
```

```
Bdptrstn=char(Bdptrstn);
Barrvstn=char(Barrvstn);
for i=1:13954
        if Bdptrstn(i,1)=='A'&&Bdptrstn(i,2)=='X'&&Bdptrstn(i,3)=='O'
            Bstation(i)=1;
        elseif Bdptrstn(i,1)=='B'&&Bdptrstn(i,2)=='C'&&Bdptrstn(i,3)=='J'
            Bstation(i)=2;
        elseif Bdptrstn(i,1)=='F'&&Bdptrstn(i,2)=='B'&&Bdptrstn(i,3)=='X'
            Bstation(i)=3;
        elseif Bdptrstn(i,1)=='G'&&Bdptrstn(i,2)=='K'&&Bdptrstn(i,3)=='S'
            Bstation(i)=4;
        elseif Bdptrstn(i,1)=='G'&&Bdptrstn(i,2)=='K'&&Bdptrstn(i,3)=='U'
            Bstation(i)=5;
        elseif Bdptrstn(i,1)=='G'&&Bdptrstn(i,2)=='O'&&Bdptrstn(i,3)=='Q'
            Bstation(i)=6;
        elseif Bdptrstn(i,1)=='H'&&Bdptrstn(i,2)=='O'&&Bdptrstn(i,3)=='M'
            Bstation(i)=7;
        elseif Bdptrstn(i,1)=='H'&&Bdptrstn(i,2)=='U'&&Bdptrstn(i,3)=='K'
            Bstation(i)=8;
        elseif Bdptrstn(i,1)=='H'&&Bdptrstn(i,2)=='W'&&Bdptrstn(i,3)=='J'
            Bstation(i)=9;
        elseif Bdptrstn(i,1)=='H'&&Bdptrstn(i,2)=='W'&&Bdptrstn(i,3)=='X'
            Bstation(i)=10;
        elseif Bdptrstn(i,1)=='M'&&Bdptrstn(i,2)=='M'&&Bdptrstn(i,3)=='C'
            Bstation(i)=11;
        elseif Bdptrstn(i,1)=='M'&&Bdptrstn(i,2)=='M'&&Bdptrstn(i,3)=='Y'
            Bstation(i)=12;
        elseif Bdptrstn(i,1)=='M'&&Bdptrstn(i,2)=='Y'&&Bdptrstn(i,3)=='J'
            Bstation(i)=13;
        elseif Bdptrstn(i,1)=='N'&&Bdptrstn(i,2)=='O'&&Bdptrstn(i,3)=='U'
            Bstation(i)=14;
        elseif Bdptrstn(i,1)=='O'&&Bdptrstn(i,2)=='A'&&Bdptrstn(i,3)=='X'
            Bstation(i)=15;
        elseif Bdptrstn(i,1)=='O'&&Bdptrstn(i,2)=='O'&&Bdptrstn(i,3)=='J'
            Bstation(i)=16;
        elseif Bdptrstn(i,1)=='O'&&Bdptrstn(i,2)=='S'&&Bdptrstn(i,3)=='Y'
            Bstation(i)=17;
        elseif Bdptrstn(i,1)=='O'&&Bdptrstn(i,2)=='X'&&Bdptrstn(i,3)=='U'
            Bstation(i)=18;
        elseif Bdptrstn(i,1)=='S'&&Bdptrstn(i,2)=='G'&&Bdptrstn(i,3)=='J'
            Bstation(i)=19;
        elseif Bdptrstn(i,1)=='S'&&Bdptrstn(i,2)=='H'&&Bdptrstn(i,3)=='O'
            Bstation(i)=20;
        elseif Bdptrstn(i,1)=='S'&&Bdptrstn(i,2)=='X'&&Bdptrstn(i,3)=='A'
```

```
            Bstation(i)=21;
        elseif Bdptrstn(i,1)=='S'&&Bdptrstn(i,2)=='X'&&Bdptrstn(i,3)=='J'
            Bstation(i)=22;
        elseif Bdptrstn(i,1)=='T'&&Bdptrstn(i,2)=='A'&&Bdptrstn(i,3)=='N'
            Bstation(i)=23;
        elseif Bdptrstn(i,1)=='T'&&Bdptrstn(i,2)=='G'&&Bdptrstn(i,3)=='D'
            Bstation(i)=24;
        elseif Bdptrstn(i,1)=='T'&&Bdptrstn(i,2)=='H'&&Bdptrstn(i,3)=='J'
            Bstation(i)=25;
        elseif Bdptrstn(i,1)=='T'&&Bdptrstn(i,2)=='U'&&Bdptrstn(i,3)=='K'
            Bstation(i)=26;
        elseif Bdptrstn(i,1)=='T'&&Bdptrstn(i,2)=='X'&&Bdptrstn(i,3)=='D'
            Bstation(i)=27;
        elseif Bdptrstn(i,1)=='U'&&Bdptrstn(i,2)=='D'&&Bdptrstn(i,3)=='J'
            Bstation(i)=28;
        elseif Bdptrstn(i,1)=='U'&&Bdptrstn(i,2)=='T'&&Bdptrstn(i,3)=='C'
            Bstation(i)=29;
        elseif Bdptrstn(i,1)=='U'&&Bdptrstn(i,2)=='Y'&&Bdptrstn(i,3)=='S'
            Bstation(i)=30;
        elseif Bdptrstn(i,1)=='W'&&Bdptrstn(i,2)=='X'&&Bdptrstn(i,3)=='J'
            Bstation(i)=31;
        elseif Bdptrstn(i,1)=='X'&&Bdptrstn(i,2)=='B'&&Bdptrstn(i,3)=='T'
            Bstation(i)=32;
        elseif Bdptrstn(i,1)=='X'&&Bdptrstn(i,2)=='M'&&Bdptrstn(i,3)=='H'
            Bstation(i)=33;
        elseif Bdptrstn(i,1)=='X'&&Bdptrstn(i,2)=='M'&&Bdptrstn(i,3)=='J'
            Bstation(i)=34;
        elseif Bdptrstn(i,1)=='X'&&Bdptrstn(i,2)=='M'&&Bdptrstn(i,3)=='Q'
            Bstation(i)=35;
        elseif Bdptrstn(i,1)=='X'&&Bdptrstn(i,2)=='N'&&Bdptrstn(i,3)=='Z'
            Bstation(i)=36;
        elseif Bdptrstn(i,1)=='X'&&Bdptrstn(i,2)=='S'&&Bdptrstn(i,3)=='J'
            Bstation(i)=37;
        elseif Bdptrstn(i,1)=='X'&&Bdptrstn(i,2)=='X'&&Bdptrstn(i,3)=='J'
            Bstation(i)=38;
        elseif Bdptrstn(i,1)=='Y'&&Bdptrstn(i,2)=='K'&&Bdptrstn(i,3)=='J'
            Bstation(i)=39;
    end
end
Bstation=Bstation';
for i=1:13954
        if Barrvstn(i,1)=='A'&&Barrvstn(i,2)=='X'&&Barrvstn(i,3)=='O'
            Bstation(i,2)=1;
        elseif Barrvstn(i,1)=='B'&&Barrvstn(i,2)=='C'&&Barrvstn(i,3)=='J'
```

```
        Bstation(i,2)=2;
elseif Barrvstn(i,1)=='F'&&Barrvstn(i,2)=='B'&&Barrvstn(i,3)=='X'
        Bstation(i,2)=3;
elseif Barrvstn(i,1)=='G'&&Barrvstn(i,2)=='K'&&Barrvstn(i,3)=='S'
        Bstation(i,2)=4;
elseif Barrvstn(i,1)=='G'&&Barrvstn(i,2)=='K'&&Barrvstn(i,3)=='U'
        Bstation(i,2)=5;
elseif Barrvstn(i,1)=='G'&&Barrvstn(i,2)=='O'&&Barrvstn(i,3)=='Q'
        Bstation(i,2)=6;
elseif Barrvstn(i,1)=='H'&&Barrvstn(i,2)=='O'&&Barrvstn(i,3)=='M'
        Bstation(i,2)=7;
elseif Barrvstn(i,1)=='H'&&Barrvstn(i,2)=='U'&&Barrvstn(i,3)=='K'
        Bstation(i,2)=8;
elseif Barrvstn(i,1)=='H'&&Barrvstn(i,2)=='W'&&Barrvstn(i,3)=='J'
        Bstation(i,2)=9;
elseif Barrvstn(i,1)=='H'&&Barrvstn(i,2)=='W'&&Barrvstn(i,3)=='X'
        Bstation(i,2)=10;
elseif Barrvstn(i,1)=='M'&&Barrvstn(i,2)=='M'&&Barrvstn(i,3)=='C'
        Bstation(i,2)=11;
elseif Barrvstn(i,1)=='M'&&Barrvstn(i,2)=='M'&&Barrvstn(i,3)=='Y'
        Bstation(i,2)=12;
elseif Barrvstn(i,1)=='M'&&Barrvstn(i,2)=='Y'&&Barrvstn(i,3)=='J'
        Bstation(i,2)=13;
elseif Barrvstn(i,1)=='N'&&Barrvstn(i,2)=='O'&&Barrvstn(i,3)=='U'
        Bstation(i,2)=14;
elseif Barrvstn(i,1)=='O'&&Barrvstn(i,2)=='A'&&Barrvstn(i,3)=='X'
        Bstation(i,2)=15;
elseif Barrvstn(i,1)=='O'&&Barrvstn(i,2)=='O'&&Barrvstn(i,3)=='J'
        Bstation(i,2)=16;
elseif Barrvstn(i,1)=='O'&&Barrvstn(i,2)=='S'&&Barrvstn(i,3)=='Y'
        Bstation(i,2)=17;
elseif Barrvstn(i,1)=='O'&&Barrvstn(i,2)=='X'&&Barrvstn(i,3)=='U'
        Bstation(i,2)=18;
elseif Barrvstn(i,1)=='S'&&Barrvstn(i,2)=='G'&&Barrvstn(i,3)=='J'
        Bstation(i,2)=19;
elseif Barrvstn(i,1)=='S'&&Barrvstn(i,2)=='H'&&Barrvstn(i,3)=='O'
        Bstation(i,2)=20;
elseif Barrvstn(i,1)=='S'&&Barrvstn(i,2)=='X'&&Barrvstn(i,3)=='A'
        Bstation(i,2)=21;
elseif Barrvstn(i,1)=='S'&&Barrvstn(i,2)=='X'&&Barrvstn(i,3)=='J'
        Bstation(i,2)=22;
elseif Barrvstn(i,1)=='T'&&Barrvstn(i,2)=='A'&&Barrvstn(i,3)=='N'
        Bstation(i,2)=23;
elseif Barrvstn(i,1)=='T'&&Barrvstn(i,2)=='G'&&Barrvstn(i,3)=='D'
```

```
            Bstation(i,2)=24;
        elseif Barrvstn(i,1)=='T'&&Barrvstn(i,2)=='H'&&Barrvstn(i,3)=='J'
            Bstation(i,2)=25;
        elseif Barrvstn(i,1)=='T'&&Barrvstn(i,2)=='U'&&Barrvstn(i,3)=='K'
            Bstation(i,2)=26;
        elseif Barrvstn(i,1)=='T'&&Barrvstn(i,2)=='X'&&Barrvstn(i,3)=='D'
            Bstation(i,2)=27;
        elseif Barrvstn(i,1)=='U'&&Barrvstn(i,2)=='D'&&Barrvstn(i,3)=='J'
            Bstation(i,2)=28;
        elseif Barrvstn(i,1)=='U'&&Barrvstn(i,2)=='T'&&Barrvstn(i,3)=='C'
            Bstation(i,2)=29;
        elseif Barrvstn(i,1)=='U'&&Barrvstn(i,2)=='Y'&&Barrvstn(i,3)=='S'
            Bstation(i,2)=30;
        elseif Barrvstn(i,1)=='W'&&Barrvstn(i,2)=='X'&&Barrvstn(i,3)=='J'
            Bstation(i,2)=31;
        elseif Barrvstn(i,1)=='X'&&Barrvstn(i,2)=='B'&&Barrvstn(i,3)=='F'
            Bstation(i,2)=32;
        elseif Barrvstn(i,1)=='X'&&Barrvstn(i,2)=='M'&&Barrvstn(i,3)=='H'
            Bstation(i,2)=33;
        elseif Barrvstn(i,1)=='X'&&Barrvstn(i,2)=='M'&&Barrvstn(i,3)=='J'
            Bstation(i,2)=34;
        elseif Barrvstn(i,1)=='X'&&Barrvstn(i,2)=='M'&&Barrvstn(i,3)=='Q'
            Bstation(i,2)=35;
        elseif Barrvstn(i,1)=='X'&&Barrvstn(i,2)=='N'&&Barrvstn(i,3)=='Z'
            Bstation(i,2)=36;
        elseif Barrvstn(i,1)=='X'&&Barrvstn(i,2)=='S'&&Barrvstn(i,3)=='J'
            Bstation(i,2)=37;
        elseif Barrvstn(i,1)=='X'&&Barrvstn(i,2)=='X'&&Barrvstn(i,3)=='J'
            Bstation(i,2)=38;
        elseif Barrvstn(i,1)=='Y'&&Barrvstn(i,2)=='K'&&Barrvstn(i,3)=='J'
            Bstation(i,2)=39;
    end
end
pretreatmentB=[Bt,Bstation];
xlswrite('E:\B_flight.xlsx',pretreatmentB,'A1');
%Acrew
Acrew=importdata('A_crew.txt');
Acrew=char(Acrew);
for i=1:21
    if Acrew(i,1)=='Y'
        A(i,1)=1;
        if Acrew(i,3)=='Y'
            A(i,2)=1;
        else
```

```
            A(i,2)=0;
        end
    else
        A(i,1)=0;
        A(i,2)=1;
    end
end
A=[A,ones(21,1)];
%Bcrew
Bcrew=importdata('B_crew.txt');
Bcrew=char(Bcrew);
for i=1:length(Bcrew)
    if Bcrew(i,1)=='Y'
        B(i,1)=1;
        if Bcrew(i,3)=='Y'
            B(i,2)=1;
        else
            B(i,2)=0;
        end
    else
        B(i,1)=0;
        B(i,2)=1;
    end
end
B=[B,ones(length(Bcrew),1)];
%A 组甘特图
axis([0,22000,0,210]);
for i=1:206
    recx=t(i,1);
    recy=(i-1)*1;
    recw=t(i,3);
    rech=1;
    rectangle('Position',[recx,recy,recw,rech],'facecolor','b');
    hold on
    if i==206
        hold off
    end
end
```


附件 2：A_transformation.m 函数文件，将 A 组数据日期转为时间戳

```
function [t]=A_transformation(date,time)
%date=8/x/2021(num)
x=8/(2021*date);
t1=(x-11)*24*60;
```

```
%time=a:b(array)
a=time(1);
b=time(2);
t2=60*a+b;
t=t1+t2;
end
```

附件 3：B_transformation.m 函数文件，将 B 组数据日期转为时间戳

```
function [t]=B_transformation(date,time)
%date=8/x/2019(num)
x=8/(2019*date);
t1=(x-1)*24*60;
%time=a:b(array)
a=time(1);
b=time(2);
t2=60*a+b;
t=t1+t2;
end
```

附件 4：yansai_q1 求解 A 组数据在问题一中的最优解

```
clear
clc
MinCT=40;
%运行预处理程序
Adptrdate=importdata('Adptrdate.txt');
Adptrdate=str2num(cell2mat(Adptrdate));
Adptrtime=importdata('Adptrtime.txt');
Aarrvdate=importdata('Aarrvdate.txt');
Aarrvdate=str2num(cell2mat(Aarrvdate));
Aarrvtime=importdata('Aarrvtime.txt');
for i=1:length(Adptrdate)
    t1(i)=A_transformation(Adptrdate(i),Adptrtime(i,:));
    t2(i)=A_transformation(Aarrvdate(i),Aarrvtime(i,:));
end
t=[t1;t2;(t2-t1)];
t=t';
%xlswrite('E:\A_flight.xlsx',t,'A1');
%A 组机场标号
Adptrstn=importdata('Adptrstn.txt');
Aarrvstn=importdata('Aarrvstn.txt');
Adptrstn=char(Adptrstn);
Aarrvstn=char(Aarrvstn);
for i=1:length(Adptrstn)
    for j=1:3
```

```
            stn1(i,j)=abs(Adptrstn(i,j));
            stn2(i,j)=abs(Aarrvstn(i,j));
        end
end
for i=1:206
        if stn1(i,1)==67
            Astation(i)=1;
        elseif stn1(i,1)==78
            Astation(i)=2;
        elseif stn1(i,1)==80&&stn1(i,2)==68
            Astation(i)=3;
        elseif stn1(i,1)==80&&stn1(i,2)==71
            Astation(i)=4;
        elseif stn1(i,1)==80&&stn1(i,2)==76
            Astation(i)=5;
        elseif stn1(i,1)==80&&stn1(i,2)==88
            Astation(i)=6;
        else
            Astation(i)=7;
        end
end
Astation=Astation';
for i=1:206
        if stn2(i,1)==67
            Astation(i,2)=1;
        elseif stn2(i,1)==78
            Astation(i,2)=2;
        elseif stn2(i,1)==80&&stn2(i,2)==68
            Astation(i,2)=3;
        elseif stn2(i,1)==80&&stn2(i,2)==71
            Astation(i,2)=4;
        elseif stn2(i,1)==80&&stn2(i,2)==76
            Astation(i,2)=5;
        elseif stn2(i,1)==80&&stn2(i,2)==88
            Astation(i,2)=6;
        else
            Astation(i,2)=7;
        end
end
pretreatmentA=[t,Astation];
A1test=pretreatmentA;%按航班号排列未按时间升序
%xlswrite('E:\A_flight.xlsx',pretreatmentA,'A1');
for i=1:206
        A1test(i,6)=i;%加一列序列号
```

```
end
Acrew=importdata('A_crew.txt');
Acrew=char(Acrew);
for i=1:21
    if Acrew(i,1)=='Y'
        A(i,1)=1;
        if Acrew(i,3)=='Y'
            A(i,2)=1;
        else
            A(i,2)=0;
        end
    else
        A(i,1)=0;
        A(i,2)=1;
    end
end
A2=[A,ones(21,1)];
%
[a1,I]=sort(A1test(:,1));
for i=1:206
    for j=1:6
        A1(i,j)=A1test(I(i),j);%A1 已按时间进行升序排列，第六列为航班号
    end
end
%计算 A_path
for i=1:206
    for j=1:206
        if i==j
            A_path(i,j)=0;
        else
            if A1(i,5)==A1(j,4)&&A1(j,1)-A1(i,2)>=MinCT%i 航班的终点为 j 航班的起点
且 j 航班出发时间据 i 航班到达时间超过最小连接时间
                A_path(i,j)=1;
            else
                A_path(i,j)=0;
            end
        end
    end
end
%从基地出发的航班的集合
fd=[];%flight depart from basement2
for i=1:206
    if A1(i,4)==2
        fd=[fd,i];
```

```
    end
end
fd1=[];fd3=[];fd4=[];fd5=[];fd6=[];fd7=[];
for i=1:length(fd)
    if A1(fd(i),5)==1%航班终点为 1
        fd1=[fd1,fd(i)];
    elseif A1(fd(i),5)==3
        fd3=[fd3,fd(i)];
    elseif A1(fd(i),5)==4
        fd4=[fd4,fd(i)];
    elseif A1(fd(i),5)==5
        fd5=[fd5,fd(i)];
    elseif A1(fd(i),5)==6
        fd6=[fd6,fd(i)];
    elseif A1(fd(i),5)==7
        fd7=[fd7,fd(i)];
    end
end
%到达基地的航班的集合
fa=[];%flight arrive at basement2
for i=1:206
    if A1(i,5)==2
        fa=[fa,i];
    end
end
fa1=[];fa3=[];fa4=[];fa5=[];fa6=[];fa7=[];
for i=1:length(fa)
    if A1(fa(i),4)==1%航班从 1 出发到基地
        fa1=[fa1,fa(i)];
    elseif A1(fa(i),4)==3
        fa3=[fa3,fa(i)];
    elseif A1(fa(i),4)==4
        fa4=[fa4,fa(i)];
    elseif A1(fa(i),4)==5
        fa5=[fa5,fa(i)];
    elseif A1(fa(i),4)==6
        fa6=[fa6,fa(i)];
    elseif A1(fa(i),4)==7
        fa7=[fa7,fa(i)];
    end
end
%1.计算从基地出发到 1 的各航班需配备多少 C1F1
count1=zeros(1,length(fd1));jj=1;
for i=1:length(fd1)
```

```
    for j=jj:length(fa1)
        if i<length(fd1)
            if fa1(j)>fd1(i)&&fa1(j)<fd1(i+1)
                count1(i)=count1(i)+1;
                jj=j+1;
            elseif fa1(j)>fd1(i+1)
                break;
            end
        elseif i==length(fd1)
            count1(i)=length(fa1)-jj+1;
        end
    end
end
cp1=fd1';
for i=1:length(count1)
    cp1(i,2)=count1(i);
end
%3.计算从基地出发到 3 的各航班需配备多少 C1F1
count3=zeros(1,length(fd3));jj=1;
for i=1:length(fd3)
    for j=jj:length(fa3)
        if i<length(fd3)
            if fa3(j)>fd3(i)&&fa3(j)<fd3(i+1)
                count3(i)=count3(i)+1;
                jj=j+1;
            elseif fa3(j)>fd3(i+1)
                break;
            end
        elseif i==length(fd3)
            count3(i)=length(fa3)-jj+1;
        end
    end
end
cp3=fd3';
for i=1:length(count3)
    cp3(i,2)=count3(i);
end
%4.计算从基地出发到 4 的各航班需配备多少 C1F1
count4=zeros(1,length(fd4));jj=1;
for i=1:length(fd4)
    for j=jj:length(fa4)
        if i<length(fd4)
            if fa4(j)>fd4(i)&&fa4(j)<fd4(i+1)
                count4(i)=count4(i)+1;
```

```
                jj=j+1;
            elseif fa4(j)>fd4(i+1)
                break;
            end
        elseif i==length(fd4)
            count4(i)=length(fa4)-jj+1;
        end
    end
end
cp4=fd4';
for i=1:length(count4)
    cp4(i,2)=count4(i);
end
%5.计算从基地出发到 5 的各航班需配备多少 C1F1
count5=zeros(1,length(fd5));jj=1;
for i=1:length(fd5)
    for j=jj:length(fa5)
        if i<length(fd5)
            if fa5(j)>fd5(i)&&fa5(j)<fd5(i+1)
                count5(i)=count5(i)+1;
                jj=j+1;
            elseif fa5(j)>fd5(i+1)
                break;
            end
        elseif i==length(fd5)
            count5(i)=length(fa5)-jj+1;
        end
    end
end
cp5=fd5';
for i=1:length(count5)
    cp5(i,2)=count5(i);
end
%6.计算从基地出发到 6 的各航班需配备多少 C1F1
count6=zeros(1,length(fd6));jj=1;
for i=1:length(fd6)
    for j=jj:length(fa6)
        if i<length(fd6)
            if fa6(j)>fd6(i)&&fa6(j)<fd6(i+1)
                count6(i)=count6(i)+1;
                jj=j+1;
            elseif fa6(j)>fd6(i+1)
                break;
            end
```

```
            elseif i==length(fd6)
                count6(i)=length(fa6)-jj+1;
            end
        end
end
cp6=fd6';
for i=1:length(count6)
        cp6(i,2)=count6(i);
end
%7.计算从基地出发到 7 的各航班需配备多少 C1F1
count7=zeros(1,length(fd7));jj=1;
for i=1:length(fd7)
        for j=jj:length(fa7)
            if i<length(fd7)
                if fa7(j)>fd7(i)&&fa7(j)<fd7(i+1)
                    count7(i)=count7(i)+1;
                    jj=j+1;
                elseif fa7(j)>fd7(i+1)
                    break;
                end
            elseif i==length(fd7)
                count7(i)=length(fa7)-jj+1;
            end
        end
end
cp7=fd7';
for i=1:length(count7)
        cp7(i,2)=count7(i);
end
cp=[cp1;cp3;cp4;cp5;cp6;cp7];
for i=1:length(cp)
        A1(cp(i,1),7)=cp(i,2);
end
A1(1,7)=2;
A1(12,7)=2;%限制人员为 5，故 1 3 改为 2 2
for i=1:206
        At1(i,1)=A1(i,1);
        At1(i,2)=i;
        At1(i,3)=1;%表示出发
end
for i=1:206
        At2(i,1)=A1(i,2);
        At2(i,2)=i;
        At2(i,3)=2;%表示到达
```

```
end
At=[At1;At2];
[Atsort,Isort]=sort(At(:,1));
for i=1:206*2
    Atsort(i,2)=At(Isort(i),2);
    Atsort(i,3)=At(Isort(i),3);
end%按时间顺序排列
a=[0 5 0 0 0 0 0];b=[0 10 0 0 0 0 0];c=[0 6 0 0 0 0 0 ];
for i=1:length(Atsort)%412 个时间节点（包含重复时间点）
    a=[a;a(end,:)];b=[b;b(end,:)];
    if Atsort(i,3)==1%出发
        if A1(Atsort(i,2),4)==2%从基地出发
            a(i+1,A1(Atsort(i,2),4))=a(i,A1(Atsort(i,2),4))-A1(Atsort(i,2),7);
            b(i+1,A1(Atsort(i,2),4))=b(i,A1(Atsort(i,2),4))-A1(Atsort(i,2),7);
        else%从别的机场出发
            a(i+1,A1(Atsort(i,2),4))=a(i,A1(Atsort(i,2),4))-1;
            b(i+1,A1(Atsort(i,2),4))=b(i,A1(Atsort(i,2),4))-1;
        end
    else%Atsort(i,3)==2 到达
        if A1(Atsort(i,2),5)==2%到达基地
            a(i+1,A1(Atsort(i,2),5))=a(i,A1(Atsort(i,2),5))+1;
            b(i+1,A1(Atsort(i,2),5))=b(i,A1(Atsort(i,2),5))+1;
        else%到达别的机场
            a(i+1,A1(Atsort(i,2),5))=a(i,A1(Atsort(i,2),5))+A1(Atsort(i,2),7);
            b(i+1,A1(Atsort(i,2),5))=b(i,A1(Atsort(i,2),5))+A1(Atsort(i,2),7);
        end
    end
end
%迭代更新
%于是决定手动迭代
%A 组数据满足要求，不用更新
for i=1:412
    Atsort(i,4)=A1(Atsort(i,2),4);
    Atsort(i,5)=A1(Atsort(i,2),5);
end
a_all=[a,[zeros(1,5);Atsort]];
xlswrite('E:\a.xlsx',a_all,'A1');
f_crew=load('f_crew.txt');
find(f_crew>5);%1 3 12 44
for j=1:206
    if j==1
        x(1,A1(j,6))=1;
        x(2,A1(j,6))=1;
    elseif j==3
```

```
            x(3,A1(j,6))=1;
            x(4,A1(j,6))=1;
        elseif j==12
            x(1,A1(j,6))=1;
            x(5,A1(j,6))=1;
        elseif j==44
            x(3,A1(j,6))=1;
            x(5,A1(j,6))=1;
        else
            x(f_crew(j),A1(j,6))=1;
        end
end
xij=[x;x;zeros(11,206)];
for i=1:21
        for j=1:201
            if i<=4
                Xij(i,j)=xij(i,j);
            elseif i>=5&&i<=10
                Xij(i,j)=0;
            elseif i==11
                Xij(i,j)=xij(i-6,j);
            elseif i>=12&&i<=16
                Xij(i,j)=xij(i-6,j);
            else
                Xij(i,j)=0;
            end
        end
end
xlswrite('E:\Xij.xlsx',Xij,'A1');
```

附件 5：yansai_q2 求解 A 组数据在问题二中的最优解

```
clear
clc
yansai_q1;
for i=1:15%总计 15 天
    day(i)=i*24*60;
end
apart=[];
for i=1:15
    for j=1:206-1
        if A1(j,1)<=day(i)&&A1(j+1,1)>=day(i)
            apart=[apart,j];
        end
    end
```

```
end
apart=[0,apart];
%按时间命名航班号对应的 xxij
for j=1:206
    if j==1
        xx(1,1)=1;
        xx(2,1)=1;
    elseif j==3
        xx(3,3)=1;
        xx(4,3)=1;
    elseif j==12
        xx(1,12)=1;
        xx(5,12)=1;
    elseif j==44
        xx(3,44)=1;
        xx(5,44)=1;
    else
        xx(f_crew(j),j)=1;
    end
end
xxij=[xx;xx;zeros(11,206)];
%飞行时间
for i=1:10
    for j=1:length(apart)
        if j<length(apart)
            fly_time(i,j)=sum(xxij(i,apart(j)+1:apart(j+1)).*(A1(apart(j)+1:apart(j+1),3))');
        else
            fly_time(i,j)=sum(xxij(i,apart(j)+1:end).*(A1(apart(j)+1:end,3))');
        end
    end
end
%max(max(fly_time))=560<600
%执勤时间
for i=1:10
    for j=1:length(apart)
        if j<length(apart)
            p=min(find(xxij(i,apart(j)+1:apart(j+1))==1))+apart(j);
            q=max(find(xxij(i,apart(j)+1:apart(j+1))==1))+apart(j);
            duty_time(i,j)=A1(q,2)-A1(p,1);
        else
            p=min(find(xxij(i,apart(j)+1:end)==1))+apart(j);
            q=max(find(xxij(i,apart(j)+1:end)==1))+apart(j);
            duty_time(i,j)=A1(q,2)-A1(p,1);
        end
```

```
    end
end
panduan=duty_time>720;
%更改 xxij
for i=1:5
    for j=1:15
        if panduan(i,j)==1&&j<15
            cishu(i,j)=sum(xxij(i,apart(j)+1:apart(j+1)));
        elseif panduan(i,j)==1&&j==15
            cishu(i,j)=sum(xxij(i,apart(j)+1:end));
        end
    end
end
add=zeros(5,206);
for i=1:5
    for j=1:15
        if j<15
            if j<14
                if
cishu(i,j)==4&&A1(min(find(xxij(i,apart(j)+1:apart(j+1))==1))+apart(j),4)==2
                    number=find(xxij(i,apart(j)+1:apart(j+1))==1)+apart(j);
                    add(i,number(3):number(4))=xxij(i,number(3):number(4));
                    xxij(i,number(3):number(4))=zeros(1,number(4)-number(3)+1);
                elseif
cishu(i,j)==4&&A1(min(find(xxij(i,apart(j)+1:apart(j+1))==1))+apart(j),4)~=2
                    number=find(xxij(i,apart(j)+1:apart(j+1))==1)+apart(j);
                    number2=find(xxij(i,apart(j+1)+1:apart(j+2))==1)+apart(j+1);
                    add(i,number(4):number2(1))=xxij(i,number(4):number2(1));
                    xxij(i,number(4):number2(1))=zeros(1,number2(1)-number(4)+1);
                elseif cishu(i,j)==5
                    number=find(xxij(i,apart(j)+1:apart(j+1))==1)+apart(j);
                    number2=find(xxij(i,apart(j+1)+1:apart(j+2))==1)+apart(j+1);
                    add(i,number(3):number2(1))=xxij(i,number(3):number2(1));
                    xxij(i,number(3):number2(1))=zeros(1,number2(1)-number(3)+1);
                end
            elseif j==14
                if
cishu(i,j)==4&&A1(min(find(xxij(i,apart(j)+1:apart(j+1))==1))+apart(j),4)==2
                    number=find(xxij(i,apart(j)+1:apart(j+1))==1)+apart(j);
                    add(i,number(3):number(4))=xxij(i,number(3):number(4));
                    xxij(i,number(3):number(4))=zeros(1,number(4)-number(3)+1);
                elseif
cishu(i,j)==4&&A1(min(find(xxij(i,apart(j)+1:apart(j+1))==1))+apart(j),4)~=2
                    number=find(xxij(i,apart(j)+1:apart(j+1))==1)+apart(j);
```

```
                    number2=find(xxij(i,apart(j+1)+1:end)==1)+apart(j+1);
                    add(i,number(4):number2(1))=xxij(i,number(4):number2(1));
                    xxij(i,number(4):number2(1))=zeros(1,number2(1)-number(4)+1);
                end
            end
        elseif j==15&&i==3
            number=find(xxij(i,apart(j)+1:end)==1)+apart(j);
            add(i,number(3):number(4))=xxij(i,number(3):number(4));
            xxij(i,number(3):number(4))=zeros(1,number(4)-number(3)+1);
        end
    end
end
xtest=[xxij(1:5,:);xxij(1:5,:);add;add;zeros(1,206)];
%将时间航班序号转成原始航班序号，以及更改 C、P、C/P 的位置
for i=1:21
    for j=1:206
        xtest2(i,A1(j,6))=xtest(i,j);
    end
end
for i=1:21
    for j=1:201
        if i<=4
            X2ij(i,j)=xtest2(i,j);
        elseif i>=5&&i<=9
            X2ij(i,j)=xtest2(i+11,j);
        elseif i==10
            X2ij(i,j)=0;
        elseif i==11
            X2ij(i,j)=xtest2(i-6,j);
        elseif i>=12&&i<=21
            X2ij(i,j)=xtest2(i-6,j);
        end
    end
end
xlswrite('E:\X2ij.xlsx',X2ij,'A1');
```

附件 6：q1_B.m 启发式算法求解第一问 B 的可行解

```
clc;clear;
% 计时开始
tic
%% 初始数据导入 其中 A_flight 为预处理后的 A 套数据，B 同理
[B_flight,txt] = xlsread('B_flight.xlsx');
txt(1,:) = [];
B_crew = xlsread('B-crew.xlsx');
```

```
n = length(B_flight);


% for i = 1:n
%         B_flight(i,5) = i;
% end

P = zeros(n);
for i = 1:n
    for j = 1:n
        if B_flight(i,4) == B_flight(j,3) && B_flight(i,2) <= B_flight(j,1) - 40
            P(i,j) = 1;
        end
    end
end

%% 求所有存在路径
Q = lujing(B_flight);
% 对所有存在路径进行计数
Q(:,3) = 0; %初始化计数值
for i = 1:n
    for j = 1:length(Q)
        if Q(j,1) == B_flight(i,3) && Q(j,2) == B_flight(i,4)
            Q(j,3) = Q(j,3) + 1;
        end
    end
end

%% 确认存在往返的航班和单向航班
k = 1; %计数初始化
for i = 1:length(Q)
    for j = 1:i-1
        if Q(i,1) == Q(j,2) && Q(i,2) == Q(j,1)
            W(k,:) = Q(i,:);
            k = k+1;
        end
    end
end
U = Q;
for i = 1:length(U)
    for j = 1:length(W)
        if U(i,1) == W(j,1) && U(i,2) == W(j,2)
            U(i,:) = 0;
        elseif U(i,1) == W(j,2) && U(i,2) == W(j,1)
```

```
                U(i,:) = 0;
            end
        end
end

U(all(U==0,2),:)=[]; %去除路径矩阵中的全零行

%  绘制有向图
s = Q(:,1)';
t = Q(:,2)';
G = digraph(s,t);
p = plot(G,'Layout','layered');

%  清除多余变量
clearvars -except B_crew B_flight n P Q U W

%%  求每一天的航班及路径
Day = cell(31,1);
dayline = [0];
for i=1:n-1
    day1 = floor(B_flight(i,1)/1440)+1;
    day2 = floor(B_flight(i+1,1)/1440)+1;
    if day1 == day2 - 1
        dayline = [dayline,i];
        Day(day1,1) = {B_flight([dayline(day1)+1:dayline(day2)],:)};
    end
end
Day(31,1) = {B_flight([13518:13954],:)};
dayline1 = dayline + 1;
dayline2 = dayline;
dayline2(1) = [];
dayline2(1,31) = n;

%P_day = cell(31,1);
% Q_day = cell(31,1);
% for i = 1:31
%     f = cell2mat(Day(i,1)); %提取机组人员 i 的原航班任务矩阵用于更新
%     %Pd = P([dayline1(i):dayline2(i)],[dayline1(i):dayline2(i)];
%     %P_day(i,1) = {Pd};
%     Qd = lujing(f);
%     Q_day(i,1) = {Qd};
%     %  绘制有向图
%     figure(i)
%     s = Qd(:,1)';
```

```
%         t = Qd(:,2)';
%         G = digraph(s,t);
%         p = plot(G,'Layout','layered');
% end
%%   启发式算法求解机组人员排班问题
m = length(B_crew); %人数
l = max(B_flight(:,3)); %机场数
J = zeros(l,3); %机场目前含可调度三类人的人数，第一列 C，第二列 F，第三列 CF
for i = 1:m
    if B_crew(i,1) == 1 && B_crew(i,2) == 0
        J(B_crew(i,4),1) = J(B_crew(i,4),1) + 1;
    elseif B_crew(i,1) == 1 && B_crew(i,2) == 1
        J(B_crew(i,4),2) = J(B_crew(i,4),2) + 1;
    elseif B_crew(i,1) == 0 && B_crew(i,2) == 1
        J(B_crew(i,4),3) = J(B_crew(i,4),3) + 1;
    end
end
S = zeros(n,3);
for j = 1:n
    a = 1;
    for k =j:n
        if P(j,k) == 1 && a <=3
            S(j,a) = k;
            a = a + 1; %存 3 个起飞时间与 j 航班相距最近的航班的编号，数量可以改，
作为灵敏度分析的一个点
        end
    end
end
% 初始化最优安排航班个数
bestdone = 0;
% 迭代求可行解中的最优解
for x = 1:1
    % 初始化安排航班个数
    done = 0;
    % 随机生成航班路线
    %S_rand = random3(S);
    S_rand = S;
    % 根据生成的航班路线从两个基地出发分别进行路线选择，若回到基地视为一条航班
路线，航班唯一，使用后删去
    [base7_1,base7_2,con7] = basepoint(B_flight,7);
    [base24_1,base24_2,con24] = basepoint(B_flight,24);
    k = 1;
    for i = base7_1
        sd = [];
```

```
            sd = [sd,i];
            while ismember(i,con7)
                if S_rand(i,1) == 0
                    break
                end
                sd = [sd,S_rand(i,1)];
                i = S_rand(i,1);
            end
            if S_rand(i,1) == 0
                sd = [];
            else
                S_rand(sd,1) = 0;
            end
            L_7(k,1) = {sd};
            done = done +length(sd);
            k = k + 1;
        end
        k = 1;
        for i = base24_1
            sd = [];
            sd = [sd,i];
            while ismember(i,con24)
                if S_rand(i,1) == 0
                    break
                end
                sd = [sd,S_rand(i,1)];
                i = S_rand(i,1);
            end
            if S_rand(i,1) == 0
                sd = [];
            else
                S_rand(sd,1) = 0;
            end
            L_24(k,1) = {sd};
            k = k + 1;
            done = done +length(sd);
        end

        if bestdone < done
            bestdone = done;
        end
    end
    k = 1;
    for i = 1:length(L_7)
```

```
        l3 = cell2mat(L_7(i,1));
        if sum(l3) ~= 0
            L(k,1) = {l3};
            k = k+1;
        end
end
for i = 1:length(L_24)
        l3 = cell2mat(L_24(i,1));
        if sum(l3) ~= 0
            L(k,1) = {l3};
            k = k+1;
        end
end
%  机组人员任务分配
% R = cell(m,1); %初始化元胞数组，每个数组作为机组人员 i 的航班任务矩阵
% cell2mat(R(i,1)); %提取机组人员 i 的原航班任务矩阵用于更新
first = [];
last = [];
for i = 1:length(L)
        sd = cell2mat(L(i,1));
        first = [first,sd(1)];
        last = [last,sd(end)];
end
%  计时结束
toc
```

附件 6：q2_B.m 启发式算法求解第二问 B 的可行解

```
clc;clear;
%  计时开始
tic
%%  初始数据导入  其中 A_flight 为预处理后的 A 套数据，B 同理
[B_flight,txt] = xlsread('B_flight.xlsx');
txt(1,:) = [];
B_crew = xlsread('B-crew.xlsx');
n = length(B_flight);


% for i = 1:n
%         B_flight(i,5) = i;
% end

P = zeros(n);
for i = 1:n
        for j = 1:n
```

```
            if B_flight(i,4) == B_flight(j,3) && B_flight(i,2) <= B_flight(j,1) - 40
                P(i,j) = 1;
            end
        end
end

%% 求所有存在路径
Q = lujing(B_flight);
% 对所有存在路径进行计数
Q(:,3) = 0; %初始化计数值
for i = 1:n
        for j = 1:length(Q)
            if Q(j,1) == B_flight(i,3) && Q(j,2) == B_flight(i,4)
                Q(j,3) = Q(j,3) + 1;
            end
        end
end

%% 确认存在往返的航班和单向航班
k = 1; %计数初始化
for i = 1:length(Q)
        for j = 1:i-1
            if Q(i,1) == Q(j,2) && Q(i,2) == Q(j,1)
                W(k,:) = Q(i,:);
                k = k+1;
            end
        end
end
U = Q;
for i = 1:length(U)
        for j = 1:length(W)
            if U(i,1) == W(j,1) && U(i,2) == W(j,2)
                U(i,:) = 0;
            elseif U(i,1) == W(j,2) && U(i,2) == W(j,1)
                U(i,:) = 0;
            end
        end
end

U(all(U==0,2),:)=[]; %去除路径矩阵中的全零行

% 绘制有向图
s = Q(:,1)';
t = Q(:,2)';
```

```
G = digraph(s,t);
p = plot(G,'Layout','layered');

% 清除多余变量
clearvars -except B_crew B_flight n P Q U W

%% 求每一天的航班及路径
Day = cell(31,1);
dayline = [0];
for i=1:n-1
    day1 = floor(B_flight(i,1)/1440)+1;
    day2 = floor(B_flight(i+1,1)/1440)+1;
    if day1 == day2 - 1
        dayline = [dayline,i];
        Day(day1,1) = {B_flight([dayline(day1)+1:dayline(day2)],:)};
    end
end
Day(31,1) = {B_flight([13518:13954],:)};
dayline1 = dayline + 1;
dayline2 = dayline;
dayline2(1) = [];
dayline2(1,31) = n;

%P_day = cell(31,1);
% Q_day = cell(31,1);
% for i = 1:31
%     f = cell2mat(Day(i,1)); %提取机组人员 i 的原航班任务矩阵用于更新
%     %Pd = P([dayline1(i):dayline2(i)],[dayline1(i):dayline2(i)];
%     %P_day(i,1) = {Pd};
%     Qd = lujing(f);
%     Q_day(i,1) = {Qd};
%     % 绘制有向图
%     figure(i)
%     s = Qd(:,1)';
%     t = Qd(:,2)';
%     G = digraph(s,t);
%     p = plot(G,'Layout','layered');
% end
%%  启发式算法求解机组人员排班问题
m = length(B_crew); %人数
l = max(B_flight(:,3)); %机场数
J = zeros(l,3); %机场目前含可调度三类人的人数，第一列 C，第二列 F，第三列 CF
for i = 1:m
    if B_crew(i,1) == 1 && B_crew(i,2) == 0
```

```
            J(B_crew(i,4),1) = J(B_crew(i,4),1) + 1;
        elseif B_crew(i,1) == 1 && B_crew(i,2) == 1
            J(B_crew(i,4),2) = J(B_crew(i,4),2) + 1;
        elseif B_crew(i,1) == 0 && B_crew(i,2) == 1
            J(B_crew(i,4),3) = J(B_crew(i,4),3) + 1;
        end
end
S = zeros(n,3);
for j = 1:n
        a = 1;
        for k =j:n
            if P(j,k) == 1 && a <=3
                S(j,a) = k;
                a = a + 1; %存 3 个起飞时间与 j 航班相距最近的航班的编号，数量可以改，作为灵敏度分析的一个点
            end
        end
end
% 初始化最优安排航班个数
bestdone = 0;
% 迭代求可行解中的最优解
for x = 1:1
        % 初始化安排航班个数
        done = 0;
        % 随机生成航班路线
        %S_rand = random3(S);
        S_rand = S;
        % 根据生成的航班路线从两个基地出发分别进行路线选择，若回到基地视为一条航班路线，航班唯一，使用后删去
        [base7_1,base7_2,con7] = basepoint(B_flight,7);
        [base24_1,base24_2,con24] = basepoint(B_flight,24);
        k = 1;
        for i = base7_1
            sd = [];
            sd = [sd,i];
            while ismember(i,con7)
                if S_rand(i,1) == 0
                    break
                end
                sd = [sd,S_rand(i,1)];
                i = S_rand(i,1);
            end
            if S_rand(i,1) == 0
                sd = [];
```

```
            else
                S_rand(sd,1) = 0;
            end
            L_7(k,1) = {sd};
            done = done +length(sd);
            k = k + 1;
        end
        k = 1;
        for i = base24_1
            sd = [];
            sd = [sd,i];
            while ismember(i,con24)
                if S_rand(i,1) == 0
                    break
                end
                sd = [sd,S_rand(i,1)];
                i = S_rand(i,1);
            end
            if S_rand(i,1) == 0
                sd = [];
            else
                S_rand(sd,1) = 0;
            end
            L_24(k,1) = {sd};
            k = k + 1;
            done = done +length(sd);
        end

        if bestdone < done
            bestdone = done;
        end
    end
end
k = 1;
for i = 1:length(L_7)
    l3 = cell2mat(L_7(i,1));
    if sum(l3) ~= 0
        L(k,1) = {l3};
        k = k+1;
    end
end
for i = 1:length(L_24)
    l3 = cell2mat(L_24(i,1));
    if sum(l3) ~= 0
        L(k,1) = {l3};
```

```
            k = k+1;
        end
end
%  机组人员任务分配
% R = cell(m,1); %初始化元胞数组，每个数组作为机组人员 i 的航班任务矩阵
% cell2mat(R(i,1)); %提取机组人员 i 的原航班任务矩阵用于更新
first = [];
last = [];
for i = 1:length(L)
        sd = cell2mat(L(i,1));
        first = [first,sd(1)];
        last = [last,sd(end)];
end
%  计时结束
toc
```

附件 8：basepoint.m 函数文件，用于选择从某基地出发和到达某基地的航班号

```
function [base_1,base_2,con] = basepoint(F,base)
%BASEPOINT  该函数用于选择从某基地出发和到达某基地的航班号
%     此处显示详细说明
n = length(F);
base_1 = [];
base_2 = [];
con = [];
for i = 1:n
        if F(i,3) == base
            base_1 = [base_1,i];
        end
        if F(i,4) == base
            base_2 = [base_2,i];
        else
            con = [con,i];
        end
end
```

附件 9：Lingo 求解机组排班问题线性模型

```
model:
!机组人员排班问题 0-1规划;

sets:
crew/1..21/:c1,c2,c3,c4,c5,c6;
flight/1..206/:f1,f2,f3,f4,h;
plan(crew,flight):xc,xf,xp;
transport(crew,flight,flight):T;
```

```
turnaround(flight,flight):D;
endsets
min=1000*@sum(flight(j):h)-10*@sum(crew(i):@sum(flight(j):xp(i,j)))-0.1*@sum(
crew(i):@sum(flight(j):xf(i,j)*c1(i)*c2(i)));
@for(crew(i):@for(flight(j):xc(i,j)<=c1(i)));
@for(crew(i):@for(flight(j):xf(i,j)<=c2(i)));
@for(crew(i):@for(flight(j):xp(i,j)<=c3(i)));
@for(flight(j):@for(crew(i):xc(i,j)+xf(i,j)+xp(i,j)<=h(j)));
@for(flight(j):@sum(crew(i):xc(i,j))=h(j));
@for(flight(j):@sum(crew(i):xf(i,j))=h(j));
@for(flight(j):@sum(crew(i):xc(i,j)+xf(i,j)+xp(i,j))<=5*h(j));
@for(crew(i):@for(flight(j):@for(flight(k):T(i,j,k)<=D(j,k))));
@for(crew(i):@for(flight(j):@sum(flight(k):T(i,j,k))<=1));
@for(crew(i):@for(flight(j):@for(flight(k):xc(i,j)+xf(i,j)+xp(i,j)>=T(i,j,k)))
);

@for(crew(i):@for(flight(j):@for(flight(k):xc(i,k)+xf(i,k)+xp(i,k)>=T(i,j,k)))
);
@for(crew(i):@sum(flight(j):xc(i,j)+xf(i,j)+xp(i,j))-1=@sum(flight(j):@sum(fl
ight(k):T(i,j,k))));

Data: !此处省略输入数据;
enddata
END
```