参赛密码 _____

（由组委会填写）

“华为杯”第十四届中国研究生数学建模竞赛

题 目 机场临时关闭时的航班恢复问题研究

摘 要：

针对航班枢纽航站临时关闭背景下的航班恢复问题，本模型同时考虑了航班取消、航班延误及飞机置换等策略来进行不正常航班调度。基于 0-1 整数线性规划思想，分别针对指定机型的航班恢复问题，不同机型间存在置换成本的航班恢复问题，由于飞机置换导致的部分旅客无法登机的航班恢复问题，以及旅客乘坐联程航班的航班恢复问题分别建立数学模型。

针对问题一，绘制了机型 9 所有航班的飞行计划图，在不考旅客信息的条件下，以该机型飞机的所有航班延误时间最短为目标函数，结合航班覆盖率，机场最大流量，同一飞机前后两个航班的间隔至少为 45 分钟等约束条件，利用改进资源指派模型进行求解。结果表明，该单机型的航班恢复问题计算复杂度相对较小，总体最小延误时间为 1104 分钟，调整后的航班执行计划符合题目要求。

针对问题二，利用时空网络图对不同机型的飞机置换进行描述，对问题一的模型进行改进，将不同机型飞机置换所产生额外延误时间添加至目标函数，并考虑不同机型间飞机置换的约束条件。此多机型航班恢复问题是一个大规模的整数线性规划模型，采用随机模拟退火算法对其进行求解，得到最小航班总体延误时间为 189.3 小时，结果表明该算法可以处理大规模的航班恢复问题，且可避免陷入局部最优解。

针对问题三，借助信号与系统的思想，创新性地引入语谱图的概念来建立航

班与时间、飞机、载客量的关系模型。在问题二的基础上对模型进行改进，将不同机型飞机置换导致所有不能登机旅客的总体延误时间添加至目标函数，在约束中对会导致航班座位数不足的飞机置换情况进行描述。该模型采用模拟退火算法进行求解，在尽量不取消航班的条件下，得到最小旅客总体延误时间为 1816851 分钟。

针对问题四，从旅客乘坐单程和联程航班两种情况分析，单程航班的调度策略与之前建模相同；联程航班执行的条件是前后两个航班均同时执行，且前一航班要在后一航班起飞前 45 分钟之前到达，否则联程航班取消，其延误时间为联程航班执行时的后一航班的延误时间。同时考虑这两种情况，以所有旅客到达最终目的地延误时间最短为目标函数，并用贪心模拟退火算法进行求解，得到新的航班规划方案，最后解得旅客总体延误时间为 490367 小时。

综上所述，0-1 整数线性规划模型能恰当地描述不同条件下不正常航班恢复问题，而采用改进的资源指派模型、随机模拟退火算法可快速求解该模型，二者的组合是一种行之有效的航班恢复方案。

关键词：航班恢复 资源指派模型 匈牙利算法 模拟退火算法 信号与系统 语谱图 贪心算法

目 录

一、问题重述.....7
二、问题分析.....8
2.1 问题一的分析.....8
2.2 问题二的分析.....9
2.3 问题三的分析.....9
2.4 问题四的分析.....10
三、符号说明.....10
四、模型假设.....12
五、模型建立与求解.....13
5.1 问题一 针对机型 9 的航班时间规划问题.....13
5.1.1 机型 9 原航班计划分析.....13
5.1.2 航班恢复的 0-1 整数规划模型.....14
5.1.3 资源指派算法最优化求解.....15
5.2 问题二 考虑不同机型间调整成本的航班规划问题.....18
5.2.1 航班恢复的时空网络模型.....18
5.2.2 基于改进时空网络模型的 0-1 整数规划.....20
5.2.3 模拟退火算法最优化求解.....21
5.3 问题三 考虑不能登机旅客成本的航班规划问题.....24
5.3.1 结合信号时频分析方法的航空规划建模.....24
5.3.2 采用通信学信号与系统理论的航空恢复模型.....26
5.3.3 模拟退火算法求解.....27
5.4 问题四 考虑旅客行程信息的航班规划问题.....28
5.4.1 针对旅客联程和同行情况的模型建立.....28
5.4.2 贪心模拟退火最优化求解.....29
六、模型的特点、推广与改进.....30
七、参考文献.....33
附录：.....34
附录一：航班时间计划图绘图程序.....34
附录二：资源指派算法主程序.....35
附录三：模拟退火算法主程序.....37
附录四：产生随机解的子函数.....39
附录五：解的稀疏矩阵形式转 0-1 矩阵形式子函数.....40
附录六：解的 0-1 矩阵形式转稀疏矩阵形式子函数.....41
附录七：$yatp$约束函数.....41
附录八：获取允许航班 f 最早起飞时间集合子函数.....42
附录九：计算所有航班最早允许起飞时间子函数.....42

图 录

图 1 机型 9 在所给时间段内所有航班的飞行计划图 13
图 2 机型 9 所有航班进行航班恢复后的飞行计划图 18
图 3 改进的时空网络图 19
图 4 模拟退火算法流程图.....22
图 5 原所有航班计划图 23
图 6 问题二规划后的所有航班计划图 24
图 7 未规划时前 200 趟航班 f 的航班计划图.....25
图 8 未规划时前 200 趟航班 f 的语谱图.....25
图 9 问题三规划后的航班计划图 27
图 10 问题四规划后的航班计划图 30
图 11 基于信号与系统思想的航班恢复问题模型示意图.....31

表 录

表 1 顺延航班的飞行计划表.....16
表 2 OVS 机场出发及到达的航班串表 16
表 3 OVS 恢复时可用飞机信息表 17
表 4 新航班飞行时刻计划表.....17
表 5 新航班飞行时刻计划表.....23
表 6 新航班飞行时刻计划表.....27
表 7 新航班飞行时刻计划表.....29

一、问题重述

在现代交通工具飞速发展的今天，航空出行已经成为人们远距离出行的主要交通工具。由于飞机的飞行易受到恶劣天气、军事演习、航空管制等不确定因素的影响，航班延误时有发生，严重影响了航空公司效益及人们的出行计划。因此，航班恢复问题成为国内外民航管理机构及各大航空公司亟待解决的问题。小面积的航班延误问题是容易解决的，通过人为的调度即可很快恢复机场航班的秩序，但是由于机场航班的延误现象会影响到后续航班的飞行计划，或是航班紊乱后，恢复方案实施的不及时，则会造成大面积的延误现象，若是在这种情况下仍然依靠人工调度来解决，则很难保证航班秩序及时恢复。因此，针对机场的实际情况建立合理的数学优化模型来及时有效调度来实现航班恢复是一个重要的课题。

本文针对航班恢复问题提出一些行之有效的解决方法。首先，将对该数学问题进行描述。航班恢复问题本质上是运营恢复问题的一部分，其包括航班恢复、机组恢复和旅客行程重新规划三部分，它们相互约束，构成一个整体上超大规模的运筹优化问题。该优化问题本身具有极高的复杂度，目前工业上可实现的计算能力很难对其直接求解。因此，采用运筹优化问题将对应地将其划分为三个子问题分别解决，先考虑航班恢复，在此基础上实现机组恢复，最后重新规划旅客的行程。其中，针对航班恢复问题，通常有三种航班调整方法：航班延误、飞机置换和航班取消，前两种可以同时发生。由于对广义的航班恢复问题采取分步解决的策略，在每步骤实现的同时可能会带来信息的缺失，使得该问题无法达到全局的最优解。因此，本文针对赛题的四个子问题，建立从单一机型的航班恢复模型，多机型恢复模型，最后考虑到旅客联程航班的恢复模型。下面将对赛题及其四个子问题分别阐述。

由于受到暴风雪的影响，管理部门决定在 2016 年 4 月 22 日的 18:00 到 21:00 之间关闭机场 OVS。在该时间段内该机场不能起飞或降落任何航班，机场关闭前的所有航班都正常执行，关闭时间过后机场可立即恢复正常起降。因此，原定在该日 18:00 至 21:00 之间（不包括 18：00 和 21：00 这两个时刻）起降的所有航班都需要重新安排，而且它们的重新安排可能造成关闭后其它航班的重新安排。由于 OVS 机场的跑道限制，该机场每 5 分钟最多能起飞 5 架飞机，同时降落 5 架飞机。

问题一，由于 OVS 机场在 2016 年 4 月 22 日的 18:00 到 21:00 之间起飞和降落航班均需重新安排在 21:00 及之后，且后续的其它航班可能会受到影响需要重新安排。不考虑旅客信息，要求针对机型 9（不考虑其它机型）的航班计划制定起飞时间表，在航班尽可能不被取消的条件下，使得航班的总体延误时间最小。

问题二，不考虑旅客信息，假定同一机型的所有飞机的载客量相同，其间航班调整没有成本，但在不同机型间调整有成本。比方说飞机 DIBPV 属于 320 机型，飞机 COBPV 属于 321 机型，航班 174774110 原计划是安排给飞机 DIBPV 执行，如果将 174774110 分配给飞机 COBPV 执行则需要产生额外的成本。假设此额外成本等价于航班延误半小时（置换和延误有可能会同时发生，则成本叠加）。需求解如何重新规划飞机航班（包括所有机型的所有航班），制定起飞时间表（给出延误分钟）使原计划航班尽可能不被取消，同时保证所有航班总体延误时间最短。

问题三，进一步考虑飞机的载客量，假设在不同机型间调整航班的成本除了航班本身延误半小时外，还要加上不能登机旅客的成本。比如飞机 DIBPV 的载客量是 140 人，COBPV 的载客量是 170 人。如果将飞机 COBPV 的航班分给 DIBPV 去执行，将会有 30 名旅客因没有座位而无法登机。但如果将 DIBPV 的原计划航班分配给 COBPV 去执行

则没有这种情况。假设一名旅客无法登机与该旅客延误 2 小时的成本相当，需求解如何重新规划航班以保证旅客总体延误时间最短。

问题四，在第二题的基础上，假设在不同机型间调整航班不考虑成本。旅客数据中提供了旅客的行程信息，包括旅客号，同行旅客数量和相应的航班。每个旅客行程中的相连航班间最少需要 45 分钟间隔时间用于中转。旅客的延误按照旅客计划到达最终目的地时间为基准计算，基于上述条件，如何重新规划航班以保证旅客总体延误时间最短。

二、问题分析

2.1 问题一的分析

问题一要求调整机型 9 所执行的航班时刻表，使得其受机场 OVS 关闭影响的所有航班总体延迟时间最小，且尽量不取消航班。该问题可以建模为一个 0-1 线性规划问题，将航班的延误、取消以及飞机置换策略考虑在约束当中。由于该问题是单机型的航班恢复问题，飞机置换不存在成本，为了尽量不取消航班，令航班取消的代价远大于航班延误，目标函数即使得所有航班延误时间和最小。约束中具体要考虑航班的覆盖率，最小飞机执行航班的间隔时间，恢复期开始航班的衔接，最大飞机延误时间，每架飞机执行航班是否首尾相连，及最大机场航班流量等。该问题涉及的航班及飞机数目很多，模型规模很大，所需变量复杂，约束条件较多，计算量大，求解困难。

利用资源指派算法寻求该模型的可行解，制定最佳航班起飞时刻表。借助集合的思想，将同一机型的所有航班看成一个集合，同一机型的所有飞机看成一个集合，航班集合到飞机集合存在一定映射关系。还可以利用信号处理的思想将每个航班看作一个持续一定时间（航班的飞行时间）的脉冲信号，每一个飞机看作一个载频。进一步将航班集合又抽象出两个子集合，一是所有到达 OVS 机场的航班到达时间点集合，另一个是所有离开 OVS 机场的航班起飞时间点集合，二者均可以看作是一系列冲击响应。该系统中只受 OVS 机场关闭的影响，所以原计划在关闭时间段到达该机场和从该机场起飞的航班均要安排在机场恢复以后。可以将机场关闭的时间段看作该机场的无线电静默时期，在该时期不允许信号的发生。因此，需要将到达时刻子集合和起飞时刻子集合的在静默时期的冲击向后延迟到恢复期开始时刻，考虑到到达和起飞均有 5 架飞机的流量控制，所以若是恢复时刻的冲击数量达到流量的上限，则按照原航班起飞或到达时刻的先后顺序，将超出上限的航班安排在下一个单位时间，以此类推，直至航班安排完毕。为了保证同一飞机执行航班的出发地和目的地是首尾相连的，航班起飞和到达的脉冲存在一一对应的关系，在排序后利用该对应关系将到达和起飞两个脉冲集合组合为调整过的航班脉冲。对于调整后的航班要指派飞机来执行，由于同一飞机执行飞离 OVS 机场的航班和飞到 OVS 机场的航班是成对出现的，所以只需考虑为从 OVS 机场起飞的航班指派飞机。对于无线电静默前在 OVS 机场起飞的航班按原计划指派其飞机，对于静默时期后从 OVS 机场起飞的航班需要从现有可用的飞机中进行指派。由于一些飞机的前一航班延误至静默结束的前几个决策时间单位，则这些飞机在前一航班达到后的 45 分钟内处于恢复期，不能执行后续航班。因此，OVS 机场在恢复后是否又待用飞机是值得判断的。若是由于机场关闭而产生延误的航班，在机场恢复后都被指派到可用的飞机，则只需计算由于机场关闭所产生的延误时间；若是部分航班无法指派到可用飞机。则航班将根据飞机的最早恢复时间再次延误。

本文将基于上述分析对该问题进行建模，并利用资源指派的思想对其进行求解，以航班的总体延误时间最短为目标，制定可行的航班恢复时刻表。

2.2 问题二的分析

问题二在前一问题的基础上，从单机型的航班恢复问题扩展到多机型。由于同一机型之间的航班调整没有成本，而在不同机型之间调整会产生额外的成本，相当于航班延误半小时。因此，在该问题的求解上要尽量避免不同机型之间的航班调整。若按照第一问的思路分别对每个机型的航班进行恢复，在一定程度上可以减小延迟。但缺少各机型之间的交互，该恢复方案不一定使得所有航班延迟时间最小。因此，需要综合考虑所有航班的延迟时间，将不同机型飞机置换所产出的额外延迟考虑到目标函数中。借助问题一中集合的思想，将所有的航班看为一个集合，所有的飞机看作一个集合。每个机型执行的所有航班为航班集合的一个子集合，每个机型的所有飞机为飞机集合的一个子集合，则航班子集合个数和飞机子集合个数均与机型数目相等。同一机型的航班子集合和飞机子集合具有一定的对应关系。首先按照第一问的思路，对于所有航班进行调整，使得其在 OVS 机场起飞和降落的航班避开机场的关闭期，在机场恢复时刻根据机场的流量安排后续航班。接下来就是为每个航班安排飞机，与问题一不同的是，尽量先在对于机型的飞机子集合中指派飞机，若该子集合中不存在可用飞机，可以采取航班取消或者不同机型的航班策略，为了尽量不取消航班，第一个策略将花费较高的代价；不同机型的航班之间置换会产生额外的代价，策略二将会有半小时的延迟代价。将这些代价都整合到目标函数中。此时可以建立时空网络模型，在模型中考虑飞行弧，置换弧，过站弧，对于需要的延误航班，采用为飞行弧添加平行弧集合的方式来表示。网络中每条弧都被赋予成本，模型的求解转化为网络中为每一个飞机寻找一条总成本最小的路线。

2.3 问题三的分析

问题三在问题二的基础上，考虑了由于飞机置换而不能登机旅客的成本。观察飞机信息表可以发现同一机型的座位数相同，则不同机型飞机的调换不仅会带来航班本身延误半小时的成本，而且可能会使得部分旅客不能登机，其成本可以看作不能登机的旅客数目乘以 2 小时的延误时间。本问题的关键需要考虑每个航班所需的座位数在飞机置换后是否满足，若不满足则还需考虑旅客溢出的代价。在之前的问题中我们试图用信号与系统的思想建立模型，利用时间和频率来等效航班起飞时间和飞机，在本问题中可以增加一维信号的强度来等效航班所需的座位数，进一步画出信号的语谱图。同一机型的飞机可以看作是具有相同强度的冲击响应，其强度为该机型的座位数。当某航班需要调换执行飞机时，将该航班对应的单位脉冲乘以实际执行飞机对应的冲击，所得的信号强度若是大于原信号强度，则原信号被增强，即该航班座位数满足需求且有剩余；若是小于原信号强度，则原信号发生衰减，即该航班的座位数不满足需求，则需考虑率无法登机的成本。

基于上述分析，拟在问题二的模型上，将旅客不能登机的成本计入目标函数，在约束中对航班座位号是否能满足旅客需求讨论。该问题的求解和问题二相似，均可利用贪心随机模拟退火算法进行求解。

2.4 问题四的分析

问题四在问题二的基础上，不考虑机型间调整航班的成本，而是考虑旅客存在联程的情况，以旅客计划到达最终目的地时间为基准计算延误时间，制定使得所有旅客延误时间和最小的航班执行计划。之前的问题都是从遍历航班的角度来计算总体延误时间，本问题从旅客号的角度来考虑。观察旅客信息可以发现，旅客号可以分为单程和联程两类（联程是指旅客从出发地到最终目的地之间存在一次中转）。单程旅客的延误问题较为简单，每个旅客号仅对应一个航班，旅客号到达最终目的地的延误时间即为其对于航班的延误时间，模型与问题一类似，只需对不同旅客号乘以同行旅客数来计算总体延误时间。假设不存在旅客签转的情况，若航班取消，则该旅客号的同行旅客无法到达目的地。对于联程旅客的延误问题建模较为复杂，最关键的是要判断联程是否执行。联程执行的条件是前后两个航班均被执行，且前一航班的到达时间和后一航班的出发时间之间有 45 分钟时间可以供旅客办理手续。该种情况下，旅客的延误时间按第二个航班的延误时间计算，同样在目标函数中要考虑所有旅客的延误时间。若前后两个航班中有一个被取消，或者前者航班延误致使旅客无法乘坐第二航班，则视为联程不被执行，旅客无法到达目的地。基于上述分析，拟在问题二的模型上，对 0-1 整数规划模型的目标函数进行改进，去除一些冗余的约束来简化模型，同时要根据本问题的要求增加新的约束。该问题的模型较前三问的模型更为复杂，数据量更为庞大，利用 MATLAB 求解困难会花费较大的时间，这导致本问题最后求得的结果只能保证为可行解，但不一定是最优解。

三、符号说明

符号设定	符号说明
x_f	航班 f 是否运营
x_{ft}	航班 f 是否在时间点 t 起飞
x_{pt}	飞机 p 是否在时间点 t 起飞
x_{ftp}	航班 f 是否在时间点 t 由飞机 p 起飞
y_{tp}	飞机 p 在时间点 t 是否停在机场 a
F	所有航班的集合
T	时间轴上所考虑的所有时间点的集合
P	所有飞机的集合
T_f	航班 f 的所有允许起飞时间点集合
Dpt_f	航班 f 的所有允许起飞时间点集合

Fly_f	航班 f 的飞行时间
$Cost_{Cancel\ f}$	取消航班 f 的代价权重
$Cost_{Delay\ f}$	延迟航班单位时间的代价权重
Π_i	指派矩阵
$x_{f,type}$	航班 f 是否由机型 $type$ 执行
$d_{f,type}$	航班 f 原计划是否由机型 $type$ 执行
x_{fp}	航班 f 是否由飞机 p 来执行
d_{fp}	航班 f 原计划是否由飞机 p 来执行
$I_{f,type}$	恢复后航班 f 对应飞机机型 $type$ 与原计划是否相同
I_f	航班 f 在航班恢复前后飞机机型对应是否一致
$Type$	所有飞机 p 属于飞机机型 $type$ 的集合
$Cost_{Change\ f}$	不同机型间的航班调整成本
U_f	问题三规划后航班 f 的不能登机旅客数
$Bool_{type_1,type_2}$	表示机型 $type_1$ 调整为机型 $type_2$ 时座位数是否增加
z_{type}	机型 $type$ 对应的座位数
$Cost_{seat\ f}$	航班 f 一名旅客无法登机带来的成本
l	旅客号
f_l	非常规意义的旅客号 l 对应航班 f
f_0	联程旅客所乘坐的前一航班
$\tilde{x}_{f_l t}$	旅客号 l 的航班 f 是否在时间 t 执行
$\tilde{x}_{f_l}$	旅客号为 l 航班 f_l 是否执行
$\tilde{x}_{f_0}$	旅客号为 l 航班 f_0 是否执行

L	所有旅客号的集合
L_s	单程的所有旅客号的集合
L_d	联程的所有旅客号的集合
N_l	旅客号对应的旅客同行者数量
Dpt_{f_l}	航班 f_l 的最早允许起飞时间点
$Cost_{cancal\ f_l}$	一名旅客乘坐的航班 f_l 不能到达目的地带来的成本

四、模型假设

1、所有航班只能延误，不能提前，最早起飞时间不早于原计划的起飞时间。

2、每架飞机的连续航班能首尾相连，即前一航班的到达机场与后一航班的起飞机场必须相同，而且前一航班到达时间与后一航班起飞时间之间的最小间隔时间为 45 分钟。

3、所有飞机的最后一个航班的到达时间不能晚于飞机的最晚可用时间

4、航班延误的决策时间点间隔为 10 分钟。

5、不考虑机场可停留飞机的容量。理论上所有机场可以全天 24 小时工作。

6、同行旅客是指一起订票并且行程完全一致的旅客，他们共享同一个旅客号，并作为一个整体考虑，即不能分乘不同的航班。

7、以 5 分钟为单位时间来调整航班飞行计划，且单位时间内最多有 5 个起飞航班和 5 个降落航班。

8、航班最大延误时间为 5 小时，即延误超过 5 小时一定取消该航班。

9、航班被取消的成本相当于延误 1000 分钟的成本。

10、问题三中的旅客行程都是直达的，并所有航班都是 100%的上座率。

11、问题三和问题四中所有旅客不得自行改变行程或者签转别的航班。

12、问题四中每个旅客行程中的相连航班间最少需要 45 分钟间隔时间用于中转。

五、模型建立与求解

5.1 问题一 针对机型 9 的航班时间规划问题

在航班规划中，通常有三种航班调整方法：航班延误、飞机置换和航班取消。考虑机型 9 受到 OVS 机场于 2016 年 4 月 22 日的 18:00 到 21:00 之间机场关闭的影响，原定计划飞行的 16 架飞机共 97 趟航班中，将有数十趟航班需要被迫延误或进行飞机置换。为简化本问题，针对这 97 趟航班，我们考虑飞机航班行程在调整过程中允许飞机延误和航班置换的情况，同时使原计划航班尽可能不被取消，在此基础上建立 0-1 整数规划模型来规划航班的起飞时间。

5.1.1 机型 9 原航班计划分析

对于机型 9 在所给时间段内的所有航班，这里绘制各飞机各航班的飞行计划图如图 1 所示。

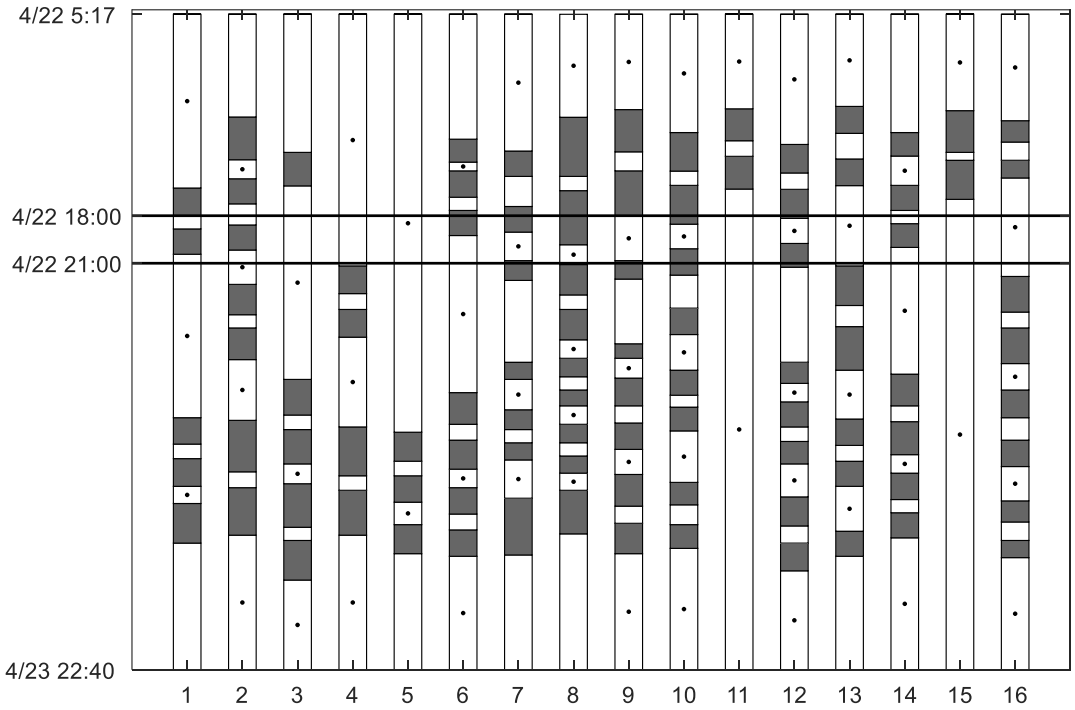

图 1 机型 9 在所给时间段内所有航班的飞行计划图

图 1 中横轴为机型 9 所有不同尾号的可用飞机，纵轴为时间刻度，图中每一条形柱表示该尾号飞机从 4 月 22 日 05:17 至 4 月 23 日 22:40 的航班规划，灰色方块表示未规划前已安排的航班时段，白色方块表示飞机位于机场等待的时段，标记有圆点“▪”的白色方块表示飞机位于受到暴风雪影响的 OVS 机场的时段。暴风雪影响导致 OVS 机场于 2016 年 4 月 22 日的 18:00 到 21:00 之间不可用，此时段用横线标注在图上。

从图 1 中可以直观地看到受到不可用时段影响的航班，航班到达 OVS 机场的时刻即为图中前一灰色方块与后一标记“▪”的白色方块连接处时刻，航班从 OVS 机场起飞的

时刻即为图中前一标记“▪”的白色方块与后一灰色方块连接处时刻，由于机场不可用时段限制，如果机型 9 的航班到达 OVS 机场的时间点和从 OVS 机场起飞的时间点落入该不可用时段内，则该航班需要将到达或起飞时间点延迟至 21:00 及以后，且后续航班可能受到影响并需要进行航班置换。

5.1.2 航班恢复的 0-1 整数规划模型

整数规划是一类问题中的全部或部分变量限制为整数的数学规划，其中，0-1 规划为一类特殊的整数规划，任何有界变量的整数规划都可以与 0-1 规划等价。由于 0-1 变量可以数量化地描述“有”与“无”等现象所反映的离散变量间的逻辑关系、顺序关系以及互斥的约束条件，因此十分适用于描述和解决指派问题、选地问题、送货问题等，本题中的航程规划和飞机指派问题即可建模为 0-1 整数规划模型[9]。

由于飞机航班延误以 10 分钟为一个决策单位，最大延误时间为 5 小时，因此可认为每个航班在某个单位时间起飞与否为一个 $\{0,1\}$ 的整数变量，为保证机型 9 的所有航班总体延误时间达到最小，在设定的约束条件内，采用 0-1 整数规划模型[3]，构造优化目标函数如式(1)所示。

目标函数：

$$\min\left(\sum_{f\in F} Cost_{Cancel\,f}\cdot\left(1-x_f\right)+\sum_{f\in F} Cost_{Delay\,f}\cdot\sum_{t\in T_f}\left(t-Dpt_f\right)\cdot x_{ft}\right) \tag{1}$$

根据问题一提出的实际问题，这里设定约束条件为：

$$x_f=\sum_{t\in T_f} x_{ft},\forall f\in F \tag{2}$$

$$x_{ft}=\sum_{p\in P} x_{ftp},\forall t\in T,\forall f\in F \tag{3}$$

$$y_{tpa}=\sum_{f\in F^I}\sum_{Dpt_f\le s<t-Fly_f} x_{fsp}-\sum_{f\in F^O}\sum_{Dpt_f\le s<t} x_{fsp},\forall a\in A,\forall t\in T,\forall p\in P,s\in T_f \tag{4}$$

$$\sum_{p\in P}\sum_{f\in F} x_{ftp}\le 5,\forall t\in T \tag{5}$$

$$\sum_{p\in P}\sum_{f\in F} x_{f(t-Dpt_f)p}\le 5,\forall t\in T \tag{6}$$

$$t\cdot\left(x_{ft}-Dpt_f\right)\le 300,\forall f\in F \tag{7}$$

$$x_{pt}\cdot\sum_{t\le s<t+45} x_{ps}=0,\forall p\in P,\forall t\in T,s\in T_f \tag{8}$$

其中，变量定义如下：

x_f 为 0-1 变量，用来表示航班 f 是否运营；

x_{ft} 为 0-1 变量，用来表示航班 f 是否在时间点 t 起飞；

x_{pt} 为 0-1 变量，用来表示飞机 p 是否在时间点 t 起飞；

x_{ftp} 为 0-1 变量，用来表示航班 f 是否在时间点 t 由飞机 p 起飞；

y_{tp} 为 0-1 变量，用来表示飞机 p 在时间点 t 是否停在机场 a。

其中，参数符号定义如下：

F 表示所有航班的集合；

T 表示时间轴上所考虑的所有时间点的集合；

P 表示所有飞机的集合；

$T_f \subseteq T$ 表示航班 f 的所有允许起飞时间点集合；

$Dpt_f = \min\{s : s \in T_f\}$ 表示航班 f 的最早允许起飞时间点；

Fly_f 表示航班 f 的飞行时间；

$F^I \subseteq F$ 表示所有到达机场 9 的航班集合；

$F^O \subseteq F$ 表示所有从机场 9 起飞的航班集合；

$Cost_{Cancel\ f}$ 表示取消航班 f 的代价权重；

$Cost_{Delay\ f}$ 表示延迟航班单位时间的代价权重。

为使原计划航班尽可能不被取消，本模型中设定 $Cost_{Cancel\ f} = 1000$ 、 $Cost_{Delay\ f} = 1$。

上述式(1)为目标函数，约束条件式(2)-式(8)为本模型中设定的约束条件，考虑机型 9 的航班运营状态、飞机起飞状态、飞机在机场停留状态、时间间隔、机场跑道限制等因素，针对目标建立相应约束，这里对式(1)-式(8)进行详细说明如下：

式(1)为目标函数，要求航班延误成本和取消成本之和最小；

式(2)是航班运行约束，表示航班 f 是否运营；

式(3)是航班 f 的起飞时间约束，表示航班 f 是否在时间点 t 起飞；

式(4)是飞机在机场的停留约束，表示飞机 p 在时间点 t 是否停在机场 a；

式(5)是机场的起飞飞机跑道限制，表示机场在时间点 t 的所有飞机 p 的所有航班 f 不大于 5；

式(6)是机场的降落飞机跑道限制，表示机场在时间点 $t - Dpt_f$ 的所有飞机 p 的所有航班 f 不大于 5；

式(7)是航班取消约束，表示航班 f 在时间 t 不大于最大延误时间 5 小时。

式(8)是最小飞行间隔约束，表示飞机 p 在时间 t 起飞其后 45 分钟内航班数为 0。

5.1.3 资源指派算法最优化求解

资源指派问题的基本思想是 N 项任务分配给 N 个人去执行，每个人完成任务的效率不同，目标是如何指派任务使得完成任务的总效率最高。将这一思想用于航班延误的恢复问题就是延误指派算法，将延误的航班动态的指派给可用飞机去执行，这个操作可能导致飞机飞行路线的改变[1]。其基本思路是将受机场关闭影响的所有航班顺延至机场恢复时刻（在机场恢复时刻起飞或降落），制定航班起飞时刻表。统计机场恢复时刻所有可用的飞机一个集合，未被执行的航班串为一个集合，建立二者的指派矩阵。将首航班与不同飞机匹配，校验是否满足约束条件，如果符合条件则将该航班指派给该飞机来执行，并且计算该航班的延误时间写入指派矩阵；如果不符合条件将匹配下一个飞机，直至所有飞机匹配完毕，若此时没有可执行该航班的飞机，则等待最早恢复的飞机来执行，若是该等待时间大于 5 小时，则该航班将取消。本文采用匈牙利算法[4]建立指派矩阵，保存各航班串上受航班的指派结果并更新后续航班串。对所有没被指派的航班串已经指派过的航班串的后续航班串迭代，直至所有航班都完成指派。

接下来，介绍算法的具体步骤，并示意求解过程。

(1) 检索出机型 9 受机场关闭影响的所有航班，将其顺延至机场恢复后，同时检索出可能会受其影响而产生延误的后续航班。考虑机场飞机起飞和降落的流量限制，制定航班顺延的飞行计划，如表 1 所示。

表 1 顺延航班的飞行计划表

航班唯一编号	起飞时间	新起飞时间	到达时间	新到达时间	起飞机场	到达机场	飞机尾号	延误时间
174773957	4/22/16 17:25	4/22/16 17:23	4/22/16 19:02	4/22/16 21:00	WTR	OVS	14098	**118**
174773460	4/22/16 20:50	4/22/16 21:45	4/22/16 22:05	4/22/16 23:00	OVS	ZOV	14098	**55**
174777548	4/23/16 3:15	4/23/16 3:15	4/23/16 4:20	4/23/16 4:20	ZOV	OVS	14098	**0**
174773636	4/22/16 16:05	4/22/16 18:33	4/22/16 18:32	4/22/16 21:00	KMM	OVS	15098	**148**
174774076	4/22/16 20:05	4/22/16 21:45	4/22/16 21:45	4/22/16 23:25	OVS	XIR	15098	**100**
174774222	4/22/16 23:50	4/23/16 00:10	4/23/16 1:50	4/23/16 1:30	XIR	OVS	15098	**20**
174773751	4/22/16 17:40	4/22/16 19:30	4/22/16 19:15	4/22/16 21:05	LLT	OVS	23098	**110**
174774144	4/22/16 18:30	4/22/16 19:35	4/22/16 20:00	4/22/16 21:05	XIR	OVS	26098	**65**
174774124	4/22/16 18:50	4/22/16 19:29	4/22/16 20:26	4/22/16 21:05	LEH	OVS	41098	**39**
174774298	4/22/16 16:25	4/22/16 17:35	4/22/16 19:50	4/22/16 21:00	GKS	OVS	44098	**70**
174774204	4/22/16 21:05	4/22/16 21:45	4/22/16 23:00	4/22/16 23:45	OVS	VOR	44098	**40**
174773919	4/22/16 23:55	4/23/16 00:39	4/23/16 1:50	4/23/16 2:25	VOR	OVS	44098	**35**
174773887	4/22/16 18:35	4/22/16 19:30	4/22/16 20:10	4/22/16 21:05	NZK	OVS	51098	**55**
174773380	4/22/16 15:10	4/22/16 18:09	4/22/16 18:01	4/22/16 21:00	MJT	OVS	64098	**179**
174773432	4/22/16 20:50	4/22/16 21:45	4/22/16 22:00	4/22/16 22:55	OVS	JOG	64098	**55**
174778458	4/23/16 2:05	4/23/16 2:05	4/23/16 3:00	4/23/16 3:00	JOG	OVS	64098	**0**
174774466	4/22/16 13:30	4/22/16 13:30	4/22/16 15:18	4/22/16 15:18	OVS	HRA	85098	**0**
174774314	4/22/16 16:20	4/22/16 19:10	4/22/16 18:10	4/22/16 21:00	HRA	OVS	85098	**170**
174774048	4/22/16 19:45	4/22/16 21:45	4/22/16 21:15	4/22/16 23:15	OVS	QSM	85098	**120**
延误航班：13 班，取消航班：0 班，航班总体延误时间：1279 分钟。								

(2) 在表 1 中找出最早的航班，检索出发生延误开始到恢复期间执行的航班串信息，接着在从剩余的航班总找出最早的航班，检索出以其为初始航班的航班串，直至所有航班均被安排咋航班串内，航班串信息如表 2 所示。

表 2 OVS 机场出发及到达的航班串表

出发地及到达时间	4/22/16 13:30	4/22/16 15:18	4/22/16 16:20	4/22/16 18:10	4/22/16 19:45	4/22/16 21:15
出发地及到达地	OVS	HRA	HRA	OVS	OVS	QSM
航班唯一编号	174774466		174774324		174774048	
出发地及到达时间	4/22/16 15:10	4/22/16 16:01	4/22/16 20:50	4/22/16 22:00	4/23/16 2:05	44/23/16 3:00
出发地及到达地	MJT	OVS	OVS	JOG	JGO	OVS
航班唯一编号	174773380		174773482		174778458	
出发地及到达时间	4/22/16 14:25	4/22/16 17:50	4/22/16 19:05	4/22/16 23:00	4/23/16 23:55	44/23/16 1:50
出发地及到达地	GKS	OVS	OVS	VOR	VOR	OVS
航班唯一编号	174774298		174774204		174773919	
出发地及到达时间	4/22/16 16:05	4/22/16 18:32	4/22/16 20:05	4/22/16 21:45	4/22/16 23:50	44/23/16 1:30
出发地及到达地	KMM	OVS	OVS	XIR	XIR	OVS
航班唯一编号	174773636		174774076		174774222	
出发地及到达时间	4/22/16 17:25	4/22/16 19:02	4/22/16 20:50	4/22/16 22:05	4/23/16 3:15	44/23/16 4:20
出发地及到达地	WTR	OVS	OVS	ZOV	ZOV	OVS
航班唯一编号	174773957		174773460		174773548	

(3) 统计出 OVS 机场恢复后可用的飞机信息，这里的可用是指该时间段飞机停留在机场，且在一定时间内不存在待执行的航班。其具体信息如表 3 所示，飞机就绪时间是指最早可用时间或者上一航班执行完毕 45 分钟后的时间。

表 3 OVS 恢复时可用飞机信息表

飞机尾号	最早可用时间	最晚可用时间	就绪时间	下一航班执行时间
32098	4/22/16 13:11	4/23/16 21:10	4/22/16 16:52	4/23/16 4:20
36098	4/22/16 11:22	4/23/16 22:40	4/22/16 17:43	4/22/16 19:30
42098	4/22/16 5:17	4/23/16 15:10	4/22/16 5:17	4/22/16 21:10
75098	4/22/16 11:15	4/23/16 21:10	4/22/16 17:04	4/23/16 21:10
82098	4/22/16 12:00	4/23/16 16:20	4/22/16 12:00	4/23/16 7:04

(4) 利用表 2 和表 3 的数据建立指派矩阵 Π_i，矩阵 Π_i 中的行是表 2 中由于前一航班受机场关闭延误导致原执行飞机无法准时起飞的航班，其航班号在矩阵的左边标注，列为表三中五个可用飞机，其先后顺序与表 3 一致。该矩阵表示单位决策时间的航班-飞机指派方阵，其维度为 5，满足飞机单位时间的流量控制策略。

$$\Pi_i = \begin{array}{r} 174774048 \\ 174773482 \\ 174774204 \\ 174774076 \\ 174773460 \end{array} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix} \tag{9}$$

(5) 更新表 1，将新的航班-飞机指派方案写入延误航班的飞行计划，再执行后续步骤，直所有航班都被指派可用的飞机。最终可得调整后的航班飞行计划如表 4 所示。

表 4 新航班飞行时刻计划表

航班编号	起飞时间	新起飞时间	到达时间	新到达时间	起飞机场	到达机场	飞机尾号	延误时间
174773957	4/22/16 17:25	4/22/16 17:23	4/22/16 19:02	4/22/16 21:00	WTR	OVS	14098	**118**
174773636	4/22/16 16:05	4/22/16 18:33	4/22/16 18:32	4/22/16 21:00	KMM	OVS	15098	**148**
174773751	4/22/16 17:40	4/22/16 19:30	4/22/16 19:15	4/22/16 21:05	LLT	OVS	23098	**110**
174774144	4/22/16 18:30	4/22/16 19:35	4/22/16 20:00	4/22/16 21:05	XIR	OVS	26098	**65**
174774204	4/22/16 21:05	4/22/16 21:05	4/22/16 23:00	4/22/16 23:00	OVS	VOR	32098	**0**
174773919	4/22/16 23:55	4/22/16 23:55	4/23/16 1:50	4/23/16 1:50	VOR	OVS	32098	**0**
174774076	4/22/16 20:05	4/22/16 21:00	4/22/16 21:45	4/22/16 22:40	OVS	XIR	36098	**55**
174774222	4/22/16 23:50	4/22/16 23:50	4/23/16 1:30	4/23/16 1:30	XIR	OVS	36098	**0**
174774124	4/22/16 18:50	4/22/16 19:29	4/22/16 20:26	4/22/16 21:05	LEH	OVS	41098	**39**
174774466	4/22/16 13:30	4/22/16 13:30	4/22/16 15:18	4/22/16 15:18	OVS	HRA	42098	**0**
174774314	4/22/16 16:20	4/22/16 19:10	4/22/16 18:10	4/22/16 21:00	HRA	OVS	42098	**170**
174774298	4/22/16 16:25	4/22/16 17:35	4/22/16 19:50	4/22/16 21:00	GKS	OVS	44098	**70**
174773887	4/22/16 18:35	4/22/16 19:30	4/22/16 20:10	4/22/16 21:05	NZK	OVS	51098	**55**
174773380	4/22/16 15:10	4/22/16 18:09	4/22/16 18:01	4/22/16 21:00	MJT	OVS	64098	**179**
174773432	4/22/16 20:50	4/22/16 21:00	4/22/16 22:00	4/22/16 22:10	OVS	JOG	75098	**10**
174778458	4/23/16 2:05	4/23/16 2:05	4/23/16 3:00	4/23/16 3:00	JOG	OVS	75098	**0**
174773460	4/22/16 20:50	4/22/16 21:00	4/22/16 22:05	4/22/16 22:15	OVS	ZOV	82098	**10**
174777548	4/23/16 3:15	4/23/16 3:15	4/23/16 4:20	4/23/16 4:20	ZOV	ZOV	82098	**0**

174774048	4/22/16 19:45	4/22/16 21:00	4/22/16 21:15	4/22/16 22:30	OVS	QSM	85098	**75**
延误航班：13 班，取消航班：0 班，航班总体延误时间：1104 分钟。								

(5) 比较表 4 与表 1 可知，采样延误指派算法对航班重新规划，可以降低航班总的延误时间，上表仅为机型 9 对应的部分航班，完整的新航班飞行时刻计划表将在附件的 EXCEL 表中给出。根据新的航班飞行计划，绘制得到航班飞行计划图（如图 2 所示）。从图中我们可以直观地看出，重新调整后的航班飞行计划，在 OVS 机场关闭时间段内没有起飞和降落的航班，且在机场恢复的同一时间起飞和降落的航班个数均满足上限，所有航班均被执行，且飞机执行航班的间隙时间大于 45 分钟，满足题目的要求。

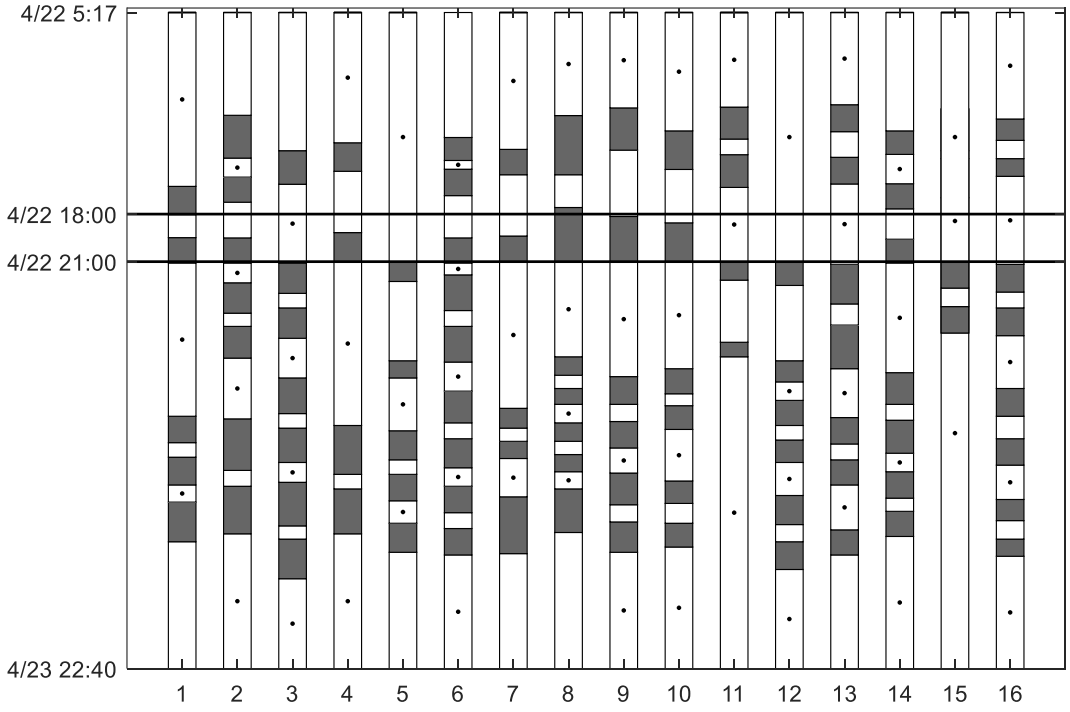

图 2 机型 9 所有航班进行航班恢复后的飞行计划图

采用上述算法对问题一的优化问题进行求解，最后的优化结果如表 4 和图 2 所示，规划后的延误航班共 13 班，取消航班 0 班，航班总体延误时间为 1104 分钟。

5.2 问题二 考虑不同机型间调整成本的航班规划问题

在问题一的基础上，此问题中设同一机型所有飞机的航班间调整没有成本，但在不同机型间调整有成本，且此额外成本等价于航班延误半小时，当置换和延误同时发生则延误时间成本叠加。在此假设下重新规划所有机型的所有飞机航班，制定起飞时间表使原计划航班尽可能不被取消，同时保证所有航班总体延误时间最短。

5.2.1 航班恢复的时空网络模型

由于此问题需要考虑所有机型的所有航班，且增加了不同机型间的航班调整成本，如果沿用第一问中的优化模型则无法实现对不同机型间航班调整的合理刻画。因此，我

们引入一个在航班规划中较为常见的时空网络优化模型[2]，用以更加明确和深入地对此问题的结构进行阐述。

时空网络由节点和有向边组成，节点含有时间和空间二维坐标。针对该问题，本文提出了一种改进时空网络如图 3 所示，以机型为横坐标，横轴上标注了机型的名称。每个机型对应一个时间的纵轴，方向自上而下，起点为每个机型中最早航班的起飞时间，终点为每个机型最晚航班的到达时间。时空网络中，纵向的连接代表时间的变化，横向的连接代表机型的变化，斜向的连接代表时间和机型都变化。

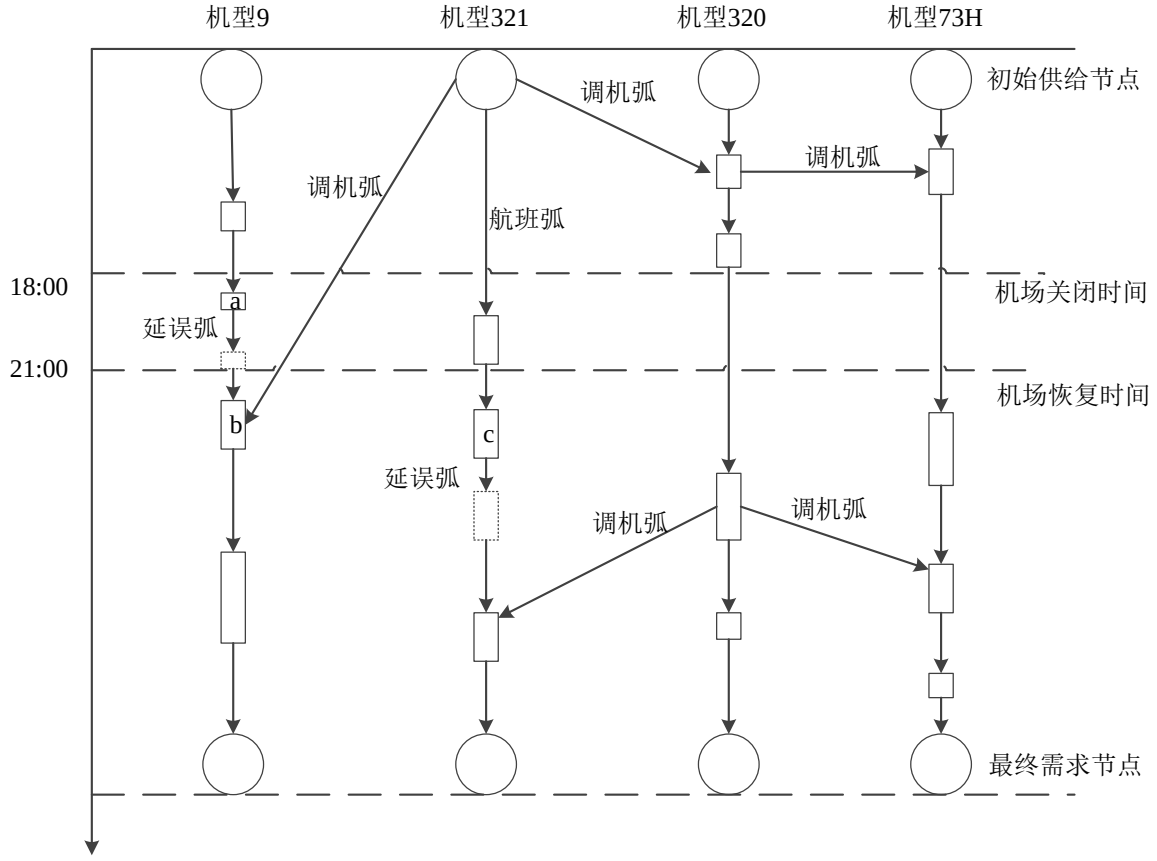

图 3 改进的时空网络图

图中每个实线矩形表示计划执行航班，虚线矩形表示延误的航班。矩阵的高表示该航班的飞行时间，其坐标可以表示航班的起飞及降落时间和所执行该航班的机型。在这个改进的时空网络中有如下节点及弧[5]：

供给节点：表示拥有一定可使用飞机数量的特定时点。供给节点可位于一天的起始时刻或者机场恢复的时间点以及机组存在空闲飞机的时间点。

需求节点：表示机组中所有飞机飞行结束后的聚集时间点，一般出现在一天的结束时刻。

航班起飞节点：表示航班起飞的时间点，以及执行该航班的机型。

航班到达节点：表示航班到达的时间点，以及执行该航班的机型。

航班弧：连接前一已被执行的航班到下一个按原计划正常执行的航班的弧，表示该过程不存在航班的延误。

延误弧：连接不能按计划正常起飞的航班到其实际执行航班的弧，表示该段时间发生了航班延误。

调机弧：连接不同机型计划执行航班的弧，表示不同机型的飞机进行了置换。

例如，图中由机型 9 执行的航班 a 由于到达机场关闭，只能推迟起飞时间使得在机场恢复的时间点到达，延迟弧表示了该过程产生了延迟。由于前一航班的延误等原因，在机型 9 中没有空闲的飞机可以执行航班 b，此时可以调用机型 321 的飞机来执行航班 b，使得航班不延误，但这一措施会产生额外的成本。由于机场刚恢复时机型 321 的飞机短缺，且其他机型无空闲飞机，航班 c 由于缺少可执行的飞机而发生延误。本小节所提的改进时空网络可以很好的刻画不同机型之间航班的调整问题，图 3 仅画出了 4 种机型的部分航用于示意，后续本节将基于该网络对问题二进行建模，从而合理安排航班的时刻表及执行机型。

5.2.2 基于改进时空网络模型的 0-1 整数规划

相对于问题一，此问题中需要考虑所有机型的所有航班，且增加了不同机型间的航班调整成本。因此，我们在目标函数中引入增加的飞机置换成本，且定义若干新的 0-1 变量用以描述飞机置换与否[7][8]。

问题二目标函数：

$$\min\left(\sum_{f\in F} Cost_{Cancel\,f}\cdot\left(1-x_f\right)+\sum_{f\in F} Cost_{Delay\,f}\cdot\sum_{t\in T_f}\left(t-Dpt_f\right)\cdot x_{ft}+\sum_{f\in F} Cost_{Change\,f}\cdot\left(1-I_f\right)\cdot x_{ft}\right) \tag{10}$$

由于问题一的全部约束在问题二中仍然需要，因此此处不再赘述，仅根据问题二提出的实际问题，增加新的约束条件如下：

$$I_f=\sum_{type} I_{f,type},\forall type\in Type \tag{11}$$

$$I_{f,type}=\sum_{p\in P_{type}} d_{fp}\cdot\sum_{p\in P_{type}} x_{fp},\forall f\in F \tag{12}$$

$$x_{f,type}=\sum_{p\in P_{type}} x_{fp},\forall f\in F,\forall type\in Type \tag{13}$$

$$d_{f,type}=\sum_{p\in P_{type}} d_{fp},\forall f\in F,\forall type\in Type \tag{14}$$

其中，新增的变量定义如下：

$x_{f,type}$ 为航班 f 是否由机型 $type$ 执行；

$d_{f,type}$ 为航班 f 原计划是否由机型 $type$ 执行

x_{fp} 为 0-1 变量，用来表示航班 f 是否由飞机 p 来执行；

d_{fp} 为 0-1 变量，用来表示航班 f 原计划是否由飞机 p 来执行；

$I_{f,type}$ 为 0-1 变量，用来表示恢复后航班 f 对应飞机机型 $type$ 与原计划是否相同；

I_f 为 0-1 变量，用来表示航班 f 在航班恢复前后飞机机型对应是否一致。

其中，新增的参数符号定义如下：

$Type$ 表示所有飞机机型的集合；

P_{type} 表示所有飞机 p 属于飞机机型 $type$ 的集合；

由于不同机型间的航班调整成本等价于航班延误半小时，因此这里令 $Cost_{Change\,f}=30\cdot Cost_{Delay\,f}$。

上述式(10)为目标函数，约束条件式(2)-式(8)及式(11)-式(14)为本模型中设定的约束条件，新增考虑飞机航班变化因素的相关约束，这里对式(10)-式(14)进行详细说明如下：

式(10)为考虑不同机型间调整成本的航班规划目标，值得一提的是，其中$\left(1-I_f\right)\cdot x_{ft}$的含义为航班 f 在时间 t 是否起飞并且发生了机型变化。

式(11)为航班变化约束，表示恢复后航班 f 对应飞机机型 $type$ 与原计划是否相同；

式(12)为航班对应机型变化约束，表示恢复后航班 f 对应飞机机型 $type$ 与原计划是否相同。

式(13)是航班执行约束，表示航班 f 是否由机型 $type$ 执行；

式(14)是原航班执行约束，表示原航班 f 是否由原机型 $type$ 执行。

5.2.3 模拟退火算法最优化求解

不同于问题一，此问题中考虑到所有不同机型的不同航班，而且飞机在不同机型间的置换会带来新的成本，因此有大量的约束需要针对决策变量 f 、 t 、 p 、 $type$ 进行判决，可想而知，如采用问题一中的资源指派模型问题，重复分次进行航班顺延和航班指派，计算量将远远超过一般计算机的求解能力。基于上述情况，我们尝试采用模拟退火算法对此优化问题进行求解，这类启发式优化算法对优化问题的实例能给出可接受的计算成本，有利于快速有效地获得最优解[6]。

模拟退火算法（Simulate Anneal Arithmetic, SAA）是一种基于概率的演算法，用来在一个区域较大的搜寻空间内找寻命题的最优解。模拟退火的思想来自冶金学中的“退火”过程，当固体的温度很高的时候，固体内能较大，其内部粒子处于快速的无序运动当中，在温度缓慢降低的过程中，固体内能慢慢减小，粒子趋于有序，最终当固体处于常温时，其内能达到最小，此时，粒子最为稳定。模拟退火算法便是基于这样的原理设计而成。

模拟退火算法将热力学的理论套用到统计学上，搜寻空间内的每一个搜索点被认为是空气分子，分子的能量就是它本身的动能。而搜寻空间内的每一点，也像空气分子一样带有“能量”，以表示该搜索点对于该命题的合适程度。模拟退火算法首先以搜寻空间内一个任意点作为起始点：每一步先选择一个“邻居”，然后再计算从现有位置到达“邻居”的概率，同时算法以一定的概率跳出这个解，在一定程度上增加了寻找到全局最优解的可能性。

针对此优化问题，本文以飞机延误时间变量作为基础,优化求解的过程实质上就是构造出新 x_{ftp} 中的 f 、 t 、 p 去替代上一个解中的相应的值。首先，计算目标函数的一个可行解作为执行解方案中的初始领域解，而后在每次迭代中，通过随机调整产生领域解，然后用该领域解来计算飞机延迟成本，若延迟成本降低，则放入延迟成本减少备选池。为保证飞机航班规划的过程不违背式(2)-式(8)的约束条件，每次调整领域解均需要判断是否满足约束，若满足约束则选取延迟成本最小的解作为最优领域解，并将结果输出。

下图图 4 所示为采用模拟退火算法求解本优化问题的算法流程图。

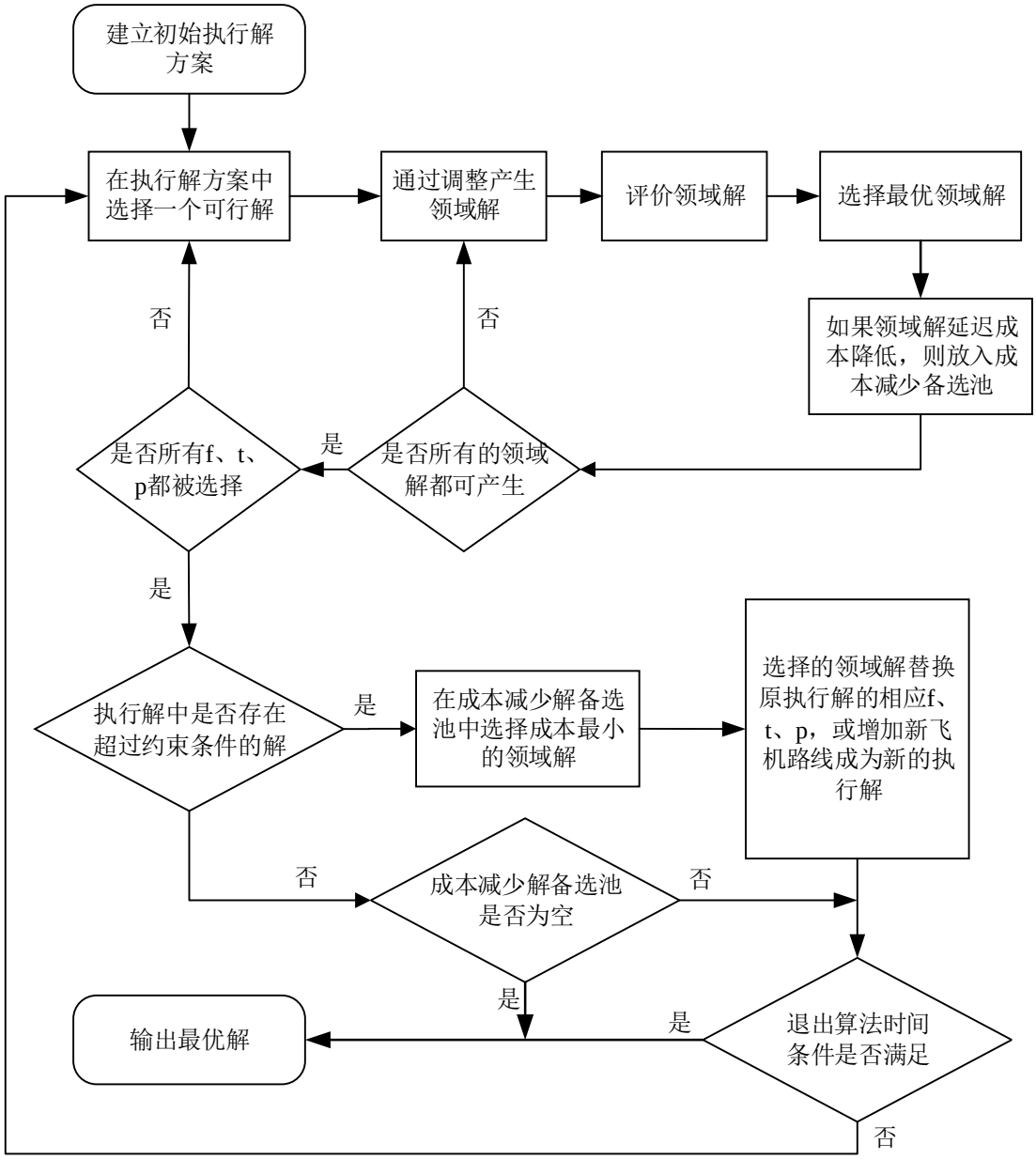

图 4 模拟退火算法流程图

需要说明的是，本算法中另设置有迭代次数最大值，当算法在经过较长时间迭代过程后仍无法趋于稳定，则重新寻找一个可行解作为初始领域解，再次进行模拟退火优化迭代。

在初始条件的设置上，令退火初始温度 $t_0=10000$；内循环的最大迭代次数 $iLk=1000$；外循环的最大迭代次数 $oLk=1000$；退火速率 $\lambda=0.99$；循环函数值方差最小值为 $ostd=0.001$；内循环保存的目标函数值个数 $olen=100$；外循环保存的目标函数值个数 $olen=100$。在退火过程的内循环中，状态转移需基于 Metropolis 稳定抽样准则：

$$p=e^{-\left(E_j-E_i\right)/T_0} \tag{15}$$

若概率 p 大于 $[0,1)$ 区间的随机数，则仍接受状态 j 为当前状态；若不成立，则保留状态 i 为当前状态。

模拟退火法得到最小航班总体延误时间为 189.3 小时，其部分结果如表 5 所示，完整的航班调整飞机计划见附件。从表 5 中可以看出，对于不同的机型飞机所执行的航班均已进行调整，从而避开机场关闭的时段。在航班冲突时，若可以找到同一机型的可用飞机进行置换，则能很好的解决冲突，且不带来额外的成本；若在同一机型能找不到可用飞机，则考虑将该航班延误至出现可用航班或者在其他机型的飞机中搜索可用飞机，前一策略将会带来延误，而后一策略会产生额外半小时的成本。

表 5 新航班飞行时刻计划表

航班编号	起飞时间	新起飞时间	到达时间	新到达时间	起飞机场	到达机场	机型	飞机尾号	延误时间
174773893	4/22/16 15:40	4/22/16 17:15	4/22/16 15:40	4/22/16 17:15	OVS	NZK	9	51098	0
174773887	4/22/16 19:25	4/22/16 21:00	4/22/16 18:35	4/22/16 20:10	NZK	OVS	9	51098	50
174773853	4/22/16 21:00	4/23/16 0:05	4/22/16 18:50	4/22/16 21:55	OVS	CGS	32A	YJBPV	130
174773700	4/22/16 23:15	4/23/16 2:35	4/22/16 23:15	4/23/16 2:35	CGS	OVS	32A	YJBPV	95
174773785	4/22/16 21:00	4/22/16 23:25	4/22/16 18:05	4/22/16 20:30	OVS	VRM	320	EMBPV	175
174773630	4/22/16 23:50	4/23/16 2:15	4/22/16 23:50	4/23/16 2:15	VRM	OVS	320	EMBPV	20
174777524	4/23/16 5:50	4/23/16 8:10	4/23/16 5:50	4/23/16 8:10	OVS	XVS	320	EMBPV	0
174774366	4/22/16 6:58	4/22/16 12:39	4/22/16 6:58	4/22/16 12:39	OVS	PGA	73H	HRBPV	0
174774372	4/22/16 16:00	4/22/16 21:00	4/22/16 13:45	4/22/16 18:45	PGA	OVS	73H	HRBPV	135
174773470	4/22/16 21:00	4/23/16 0:55	4/22/16 18:05	4/22/16 22:00	OVS	SAT	333	KEBQV	175
174773442	4/23/16 0:00	4/23/16 4:20	4/23/16 0:00	4/23/16 4:20	SAT	OVS	333	42098	95
174773817	4/22/16 21:20	4/23/16 1:15	4/22/16 21:20	4/23/16 1:15	OVS	BVO	321	44098	25
174777900	4/23/16 2:25	4/23/16 6:40	4/23/16 2:25	4/23/16 6:40	BVO	OVS	321	51098	0
174773773	4/22/16 21:00	4/23/16 4:15	4/22/16 18:35	4/23/16 1:50	OVS	KEP	77W	64098	145
174778252	4/22/16 15:40	4/22/16 17:15	4/22/16 15:40	4/22/16 17:15	KEP	OVS	77W	75098	0

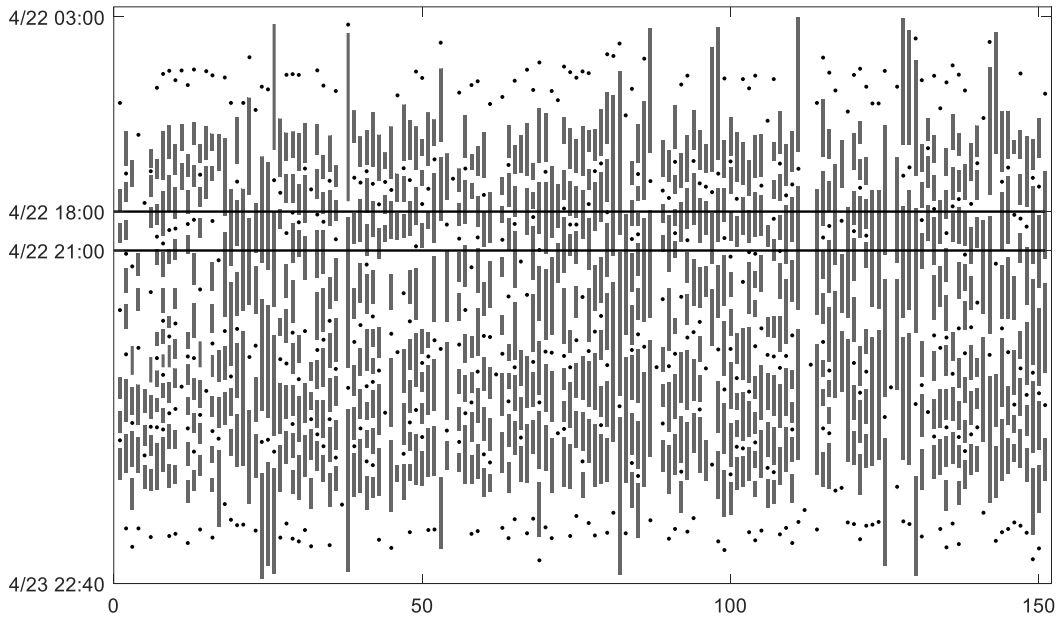

图 5 原所有航班计划图

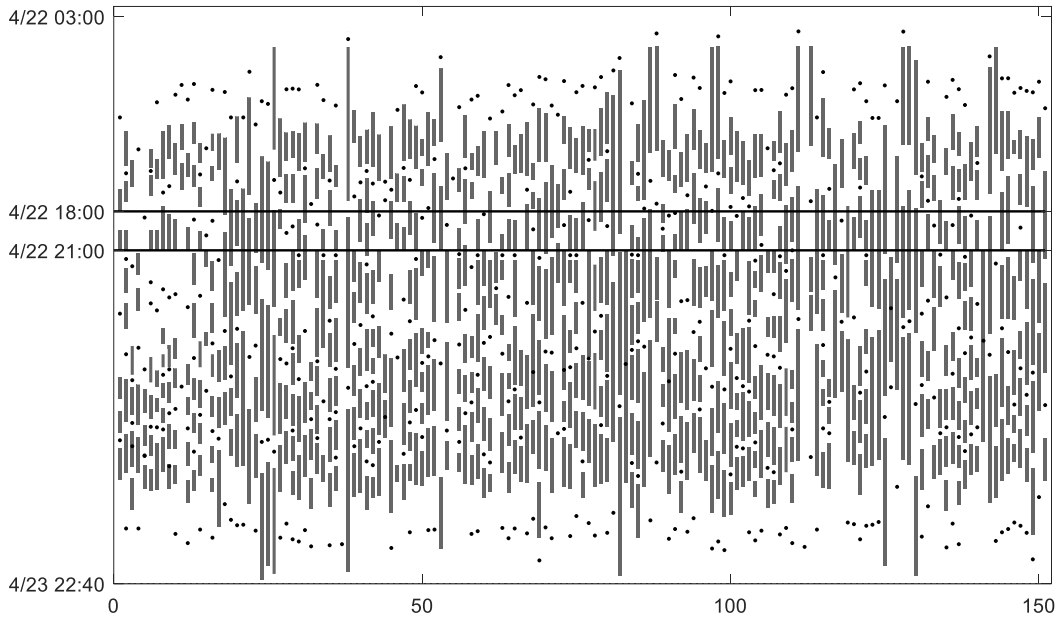

图 6 问题二规划后的所有航班计划图

为了直观的显示航班的调整，将原所有飞机执行航班的计划图（图 5 所示）与问题二规划后的航班计划图（图 6 所示）进行比较。从图中可以看出，已将机场关闭时期的航班进行调整，使得在该段时间内没有航班在 OVS 机场起飞或降落，所有需要起飞和降落的航班均安排在机场恢复后，且满足机场的流量控制要求。对于后续受影响的航班通过飞机置换及航班延迟等策略进行调整，使得航班计划满足要求。

5.3 问题三 考虑不能登机旅客成本的航班规划问题

在问题一、二的基础上，此问题中设定在飞机调整过程中需考虑不能登机旅客的成本，且此额外成本等价于该旅客延误 2 小时的成本，如载客量较大的飞机被置换给载客量较小的飞机，则会产生不能登机旅客数所对应的延误成本。在此假设下重新规划所有机型的所有飞机航班，制定起飞时间表使原计划航班尽可能不被取消，同时保证所有航班总体延误时间最短。

5.3.1 结合信号时频分析方法的航空规划建模

在上述问题一中，为描述机型 9 的 16 架飞机在不同时间的飞行计划，我们绘制了图 1 以刻画不同飞机不同时间的航班信息。在此问题中，由于属于不同机型的航班调整会受到飞机座位数变化影响，进而可能导致部分旅客不能登机，而航班的座位数是一个与航班一一对应的数值，且与时间无关，因此，我们需要在图 1 中灰色方块上增加第三个维度的信息，用以表示每个航班的乘客数。一般情况下，三维立体图并不利于显示平面上的变量信息，然而在此问题中，二维平面上 f 和 t 的变化情况对于表征航班的飞行计划尤为重要。

针对此问题，本文创造性地结合通信中信号与系统的相关理论来对航空规划进行建模。受到图 1 航班飞行计划图的启发，我们尝试采用信号的语谱图对带有飞机座位数的航班进行类比和描述[10]。

语谱图，也可称为时频图，是二战时期发明的用于描述语音数据的一种频谱分析视图。其横坐标是时间，纵坐标是频率，坐标值为语音数据的能量。由于语谱图采用二维平面表达三维信息，所以能量值的大小通过颜色来表示，颜色越深，则代表该点的语音能量越强。同时，在一个通信系统中，固定的频率一个时刻只能被一对通信用户占用，这与一架飞机在一个时刻只能执行一个航班十分相似。

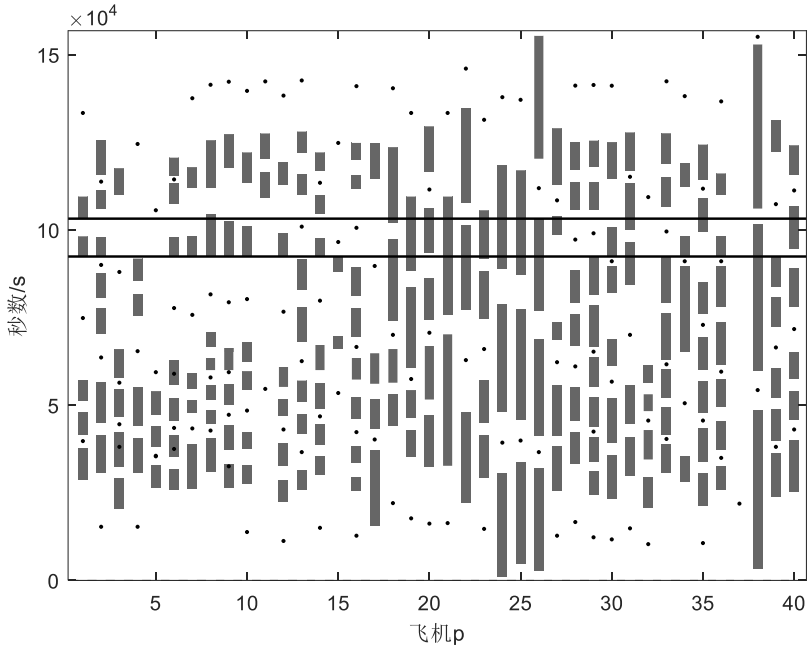

图 7 未规划时前 200 趟航班 f 的航班计划图

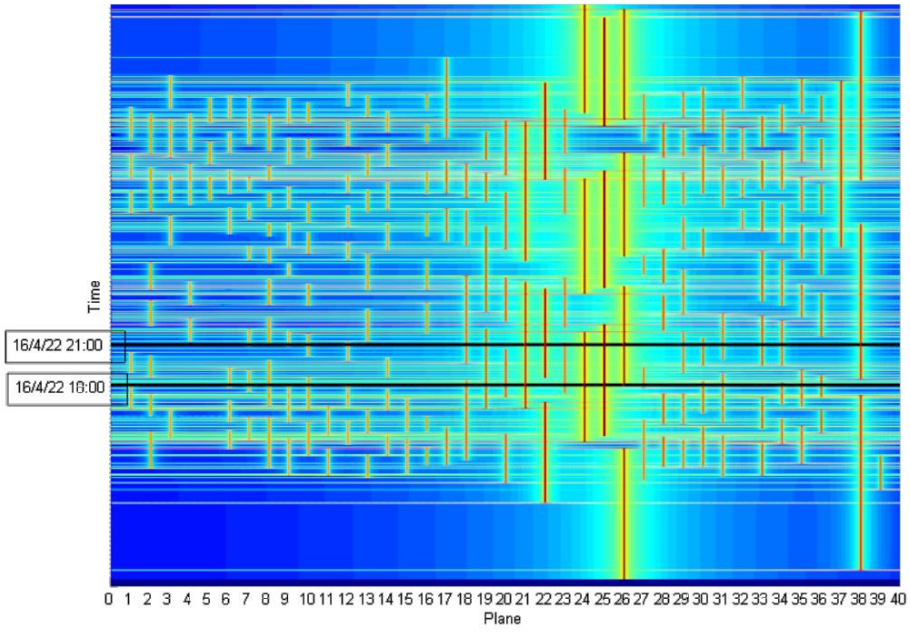

图 8 未规划时前 200 趟航班 f 的语谱图

如图 5 所示，原计划中前 200 个航班可以用语谱图的形式清晰地表示，不同时段对应多个频率信号的语谱图。对于本文中具体的航班规划问题，航班飞机 p 可认为是语谱图横轴中的频率，图中的航班 f 可认为是语谱图纵轴中的时段，旅客数（或衡量航班重要性的其他指标）则可认为是语谱图中的坐标值对应的语音能量。

语谱图可有效刻画不同飞机不同座位数不同航班的飞机时间计划，并且不会损失航班 f 和时间 t 的任何信息。同时，信号与系统中的部分研究理论可用于本问题的建模与理解，通过短时傅里叶变换（STFT）等时频分析技术，可以简单快速地得到某一时刻的在飞航班，即将起飞的航班，即将降落的航班，执行某航班的飞机尾号以及该航班的重要程度，这将十分有利于我们从多个维度分析航班的恢复问题。

5.3.2 采用通信学信号与系统理论的航空恢复模型

相对于问题一、二，此问题中不仅需要考虑所有机型、所有航班以及不同机型间的航班调整成本，还需要考虑由于旅客不能登机带来的对应航班延迟成本。因此，我们在前文设定的目标函数中引入不能登机旅客成本，且定义若干新的 0-1 变量和一个 0-1 变量矩阵用以描述航班调整过程中座位数增减与否。

问题三目标函数：

$$\min\left(\begin{array}{l}\sum\limits_{f\in F}Cost_{Cancel\,f}\cdot\left(1-x_f\right)+\sum\limits_{f\in F}Cost_{Delay\,f}\cdot\sum\limits_{t\in T_f}\left(t-Dpt_f\right)\cdot x_{ft}\\+\sum\limits_{f\in F}Cost_{Change\,f}\cdot\left(1-I_f\right)\cdot x_{ft}+\sum\limits_{f\in F}Cost_{seat,f}\cdot U_f\cdot\left(1-I_f\right)\cdot x_{ft}\end{array}\right) \tag{16}$$

问题三在问题一、二的全部约束基础上，即可实现全部约束的限制，因此此处直接沿用约束条件式(2)-式(8)、式(11)-式(14)，具体约束式不再赘述。这里仅根据问题三提出的实际问题，对新增加的变量和参数进行详细说明。

在参数说明之前，我们引入通信学信号与系统理论中的一个常用序列，即单位阶跃序列[10]。此序列将运用于本问题的变量表示中，可使变量定义更加简洁。该单位阶跃序列定义为：

$$\varepsilon\left(k\right)\overset{def}{=}\begin{cases}1,\ k\geq 0\\0,\ k<0\end{cases} \tag{17}$$

显见，单位阶跃序列 $\varepsilon\left(k\right)$ 在 $k<0$ 的各点为零，在 $k\geq 0$ 的各点为 1。

问题三新增的变量定义如下：

$U_f=\sum\limits_{type_1}\sum\limits_{type_2}Bool_{type_1,type_2}\left(x_{f,type_1}\cdot z_{type_1}-d_{f,type_2}\cdot z_{type_2}\right),\forall f\in F$ 表示规划后航班 f 的不能登机旅客数；

$Bool_{type_1,type_2}=\varepsilon\left(z_{type_2}-z_{type_1}\right),\forall type_1,type_2\in Type$ 为 0-1 变量，表示机型 $type_1$ 调整为机型 $type_2$ 时座位数是否增加。则 $Bool$ 矩阵为一个 0-1 矩阵，行表示 $type_1$，列表示 $type_2$，$\forall type_1,type_2\in Type$，当机型从 $type_1$ 调整为 $type_2$ 时座位数如果增加，则 $Bool_{type_1,type_2}=1$，否则 $Bool_{type_1,type_2}=0$，可以看出 $Bool$ 矩阵为一个对称矩阵。

新增的参数符号定义如下：

z_{type} 为机型 $type$ 对应的座位数；

$Cost_{seat\,f}$ 为航班 f 一名旅客无法登机带来的成本，由于一名旅客无法登机与该旅客延误 2 小时的成本相当，因此这里令 $Cost_{seat\,f}=120\cdot Cost_{Delay\,f}$。

5.3.3 模拟退火算法求解

相比问题二，问题三目标函数发生了小幅变化，并增加了部分约束，模型仍可采用模拟退火算法求解，根据算法找到的最优解，绘制规划后的航班计划图如图 9 所示。

表 6 新航班飞行时刻计划表

航班编号	起飞时间	新起飞时间	到达时间	新到达时间	起飞机场	到达机场	机型	飞机尾号	延误时间
174773460	4/22/16 21:00	4/22/16 22:15	4/22/16 20:50	4/22/16 22:05	OVS	ZOV	9	36098	**10**
174777548	4/23/16 3:15	4/23/16 4:20	4/23/16 3:15	4/23/16 4:20	ZOV	OVS	9	36098	**0**
174777404	4/23/16 11:00	4/23/16 13:00	4/23/16 11:00	4/23/16 13:00	OVS	DGK	32A	YJBPV	**15**
174777428	4/23/16 13:55	4/23/16 15:45	4/23/16 13:55	4/23/16 15:45	DGK	OVS	32A	YJBPV	**5**
174774406	4/22/16 11:56	4/22/16 14:51	4/22/16 11:56	4/22/16 14:51	OVS	GRP	320	CKBPV	**0**
174774412	4/22/16 18:36	4/22/16 21:00	4/22/16 15:50	4/22/16 18:14	GRP	OVS	320	CKBPV	**166**
174773877	4/22/16 21:00	4/22/16 23:05	4/22/16 19:25	4/22/16 21:30	OVS	AFU	320	CKBPV	**140**
174777678	4/23/16 3:30	4/23/16 5:40	4/23/16 3:30	4/23/16 5:40	AFU	OVS	320	CKBPV	**75**
174773741	4/22/16 21:00	4/23/16 0:20	4/22/16 19:25	4/22/16 22:45	OVS	SMO	32A	LLBPV	**95**
174773610	4/22/16 23:50	4/23/16 3:15	4/22/16 23:50	4/23/16 3:15	SMO	OVS	32A	LLBPV	**0**
174774242	4/22/16 21:00	4/22/16 23:25	4/22/16 19:15	4/22/16 21:40	OVS	KEC	321	COBPV	**105**
174778044	4/23/16 2:50	4/23/16 5:15	4/23/16 2:50	4/23/16 5:15	KEC	OVS	321	COBPV	**0**
174773506	4/22/16 21:00	4/23/16 2:35	4/22/16 20:55	4/23/16 2:30	OVS	TKI	73H	RRBPV	**5**
174778400	4/23/16 4:00	4/23/16 10:05	4/23/16 4:00	4/23/16 10:05	TKI	OVS	73H	RRBPV	**0**

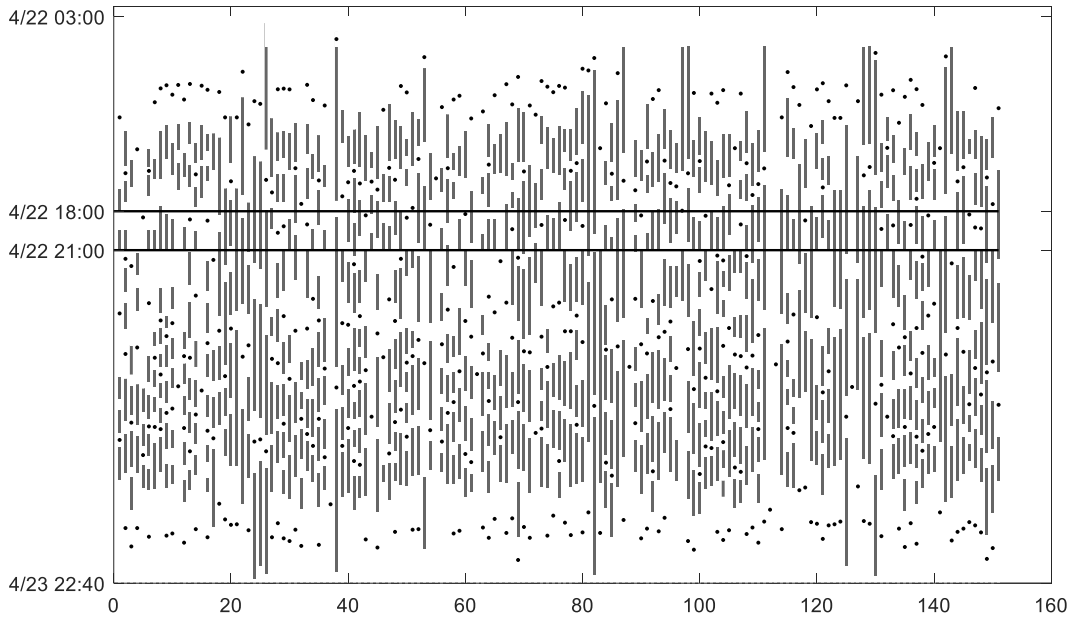

图 9 问题三规划后的航班计划图

本问题涉及 9 个机型共 151 架飞机，来执行 749 个航班，每个机型的座位数相同，既考虑不同机型调换会产生航班延误半小时的代价，又考虑座位少的飞机置换座位多的飞机导致部分旅客无法登机的总延误时间，在尽量不取消航班的条件下，得到旅客总体延误时间为 1816851 分钟，部分航班的调整计划表如表 1 所示，完整的航班调整计划见

附件，图 9 给出了航班调整飞行计划图，图中无航班冲突，机场关闭时期无在 OVS 机场起飞或降落的航班。

5.4 问题四 考虑旅客行程信息的航班规划问题

不同于问题二和问题三，这里不再考虑不同机型间调整航班的成本，基于问题二中对所有飞机建立的时空网络数学模型，问题四加入旅客行程信息，在航班尽量不取消的情况下，考虑旅客到达目的地延迟时间成本，对所有飞机的所有航班进行规划，使得规划后的总延迟成本达到最小。

5.4.1 针对旅客联程和同行情况的模型建立

观察问题所提供旅客信息，不难发现旅客行程可以分为单程和联程（即旅客从出发地到最终目的地之间存在一次中转）两类。

单程的延误问题较简单，每个旅客号仅对应一个航班，旅客号到达最终目的地的延误时间即为其对于航班的延误时间，模型与问题一类似，计算延误成本时只需对不同旅客号计算的延误成本进行叠加。若航班取消，则该旅客号的同行旅客无法到达目的地。

联程的延误问题建模则较为复杂，最关键的是要判断联程是否执行。联程执行的条件是前后两个航班均被执行，且前一航班的到达时间与后一航班的出发时间之间间隔时间需大于 45 分钟。该种情况下，旅客的延误时间按第二个航班的延误时间计算，同样在目标函数中需要考虑所有旅客的延误时间。若前后两个航班中有一个被取消，或前者航班延误致使旅客无法乘坐第二航班，则旅客无法到达目的地。

根据上述分析，我们在问题一设定的目标函数中引入旅客延误成本，且针对旅客信息设定必要的约束，定义若干新的 0-1 变量用以描述旅客的行程执行情况。

问题四目标函数：

$$\min\left(\begin{array}{l}\sum_{f\in F} Cost_{Cancel\ f}\cdot\left(1-x_f\right)+\sum_{f\in F} Cost_{Delay\ f}\cdot\sum_{t\in T_f}\left(t-Dpt_f\right)\cdot x_{ft}\\ +\sum_{l\in L} Cost_{Delay\ f}\cdot N_l\cdot\left(t-Dpt_{f_l}\right)\cdot x_{f_l t}\cdot\tilde{x}_{f_l}+Cost_{cancal\ f_l}\cdot N_l\cdot\left(1-\tilde{x}_{f_l}\right)\end{array}\right) \tag{18}$$

在问题一的约束条件式(2)-式(8)的基础上，这里需要引入有关旅客信息的一个必要约束：

$$\tilde{x}_{f_l}=\sum_{t\in T}\tilde{x}_{f_l t},\ f_l\in F \tag{19}$$

其中，新增的变量定义如下：

l 为旅客号，$l\in L$；

f_l 分为两个部分，对于单程旅客号，f_l 为旅客号 l 所乘坐的航班；对于联程旅客号，由于联程旅客的最终到达目的地时间仅由其乘坐后一航班的到达时间决定，因此此时 f_l 表示联程旅客号所乘坐的后一航班；

f_0 表示联程旅客所乘坐的前一航班；

$\tilde{x}_{f_l t}$ 为 0-1 变量，表示旅客号 l 的航班 f 是否在时间 t 执行，对于单程旅客号，$\tilde{x}_{f_l t}=x_{f_l t}$，对于联程旅客号，$\tilde{x}_{f_l t}=\varepsilon\left(x_{f_0 t}\cdot\left(t+Fly_{f_l}\right)+45\le x_{f_l t}\cdot t\right)$；

$\tilde{x}_{f_l}$ 为 0-1 变量，表示旅客号为 l 航班 f_l 是否执行；

$\tilde{x}_{f_0}$ 为 0-1 变量，表示旅客号为 l 航班 f_0 是否执行；

其中，新增的参数符号定义如下：

L 表示所有旅客号的集合；

L_s 表示单程的所有旅客号的集合；

L_d 表示联程的所有旅客号的集合；

N_l 表示旅客号对应的旅客同行者数量；

Dpt_{f_l} 表示航班 f_l 的最早允许起飞时间点；

$Cost_{cancal\ f_l}$ 为一名旅客乘坐的航班 f_l 不能到达目的地带来的成本，由于一名旅客无法到达目的地相当于延误了 24 小时，因此这里令 $Cost_{cancal\ f_l}=24\times 60\cdot Cost_{Delay\ f}$。

上述式(18)为目标函数，沿用问题一中的约束条件式(2)-式(8)，并且针对本问题设定约束条件式(19)，考虑旅客到达目的地的延迟时间，建立相应约束，现对式(19)进行详细说明如下：

式(18)为目标函数，要求多种成本等效后的航班总延误成本最小；

式(19)是航班 f_l 的运行约束，表示航班 f_l 是否运营。

针对问题四的建模，其关键就在于对于航班 f_l 这一变量的定义，f_l 的物理意义分为两个部分，针对单程旅客，f_l 的意义与前文相同，代表旅客号 l 所乘坐的航班 f，但针对联程旅客，f_l 在通用意义上发生了变化，不再代表某一个旅客号对应的某单个飞机，而是作为联程旅客号的后一航班的航班号来考虑，这样有利于在目标函数中对旅客到达目的地时间进行刻画，可明晰地将航班到达时间转化为航班的起飞时间。

5.4.2 贪心模拟退火最优化求解

由于考虑了每次航班实际承载旅客的人数，同一航班内存在同行旅客，不同航班间存在联程旅客，航班变动产生的延误时间会因为旅客群体的不同而动态变化，导致此模型的复杂度远高于 5.1~5.3 中建立的模型，求解过程中需要执行的约束条件过多且过于复杂，导致判断的计算量极大，利用 MATLAB 在短时间内求解此 NP-Hard 问题的最优解十分困难，因此本问题放弃求解全局最优解，而是利用满足 5.2 模型约束的可行解，根据贪心策略，在可接受的时间开销内寻求一个可以接受的局部最优解。根据求解的结果，在该航班计划方案下，所有旅客总体延误时间为 490367 小时，方案的新航班飞行时刻表（部分）和航班计划图分别如表 7 和图 10 所示。

表 7 新航班飞行时刻计划表

航班编号	起飞时间	新起飞时间	到达时间	新到达时间	起飞机场	到达机场	机型	飞机尾号	延误时间
174774150	4/22/16 16:15	4/22/16 18:02	4/22/16 16:15	4/22/16 18:02	OVS	LEH	9	41098	c
174774124	4/22/16 19:24	4/22/16 21:00	4/22/16 18:50	4/22/16 20:26	LEH	OVS	9	41098	34
174774226	4/22/16 14:25	4/22/16 16:05	4/22/16 14:25	4/22/16 16:05	OVS	FUK	32A	RLBPV	0
174773592	4/22/16 19:24	4/22/16 21:00	4/22/16 17:05	4/22/16 18:41	FUK	OVS	32A	RLBPV	139
174773859	4/22/16 3:10	4/22/16 12:56	4/22/16 3:10	4/22/16 12:56	TRN	OVS	3KR	YMBQV	0
174773502	4/22/16 16:15	4/23/16 1:55	4/22/16 16:15	4/23/16 1:55	OVS	TKH	333	SNBQV	0
174777608	4/23/16 4:10	4/23/16 14:20	4/23/16 4:10	4/23/16 14:20	TKH	OVS	333	SNBQV	0
174774154	4/22/16 9:16	4/22/16 13:46	4/22/16 9:16	4/22/16 13:46	OVS	NCB	321	HOBQV	0
174773510	4/22/16 16:45	4/22/16 21:00	4/22/16 14:50	4/22/16 19:05	NCB	OVS	321	HOBQV	115
174773835	4/22/16 21:00	4/23/16 1:05	4/22/16 20:10	4/23/16 0:15	OVS	FOT	77W	IOBQV	80

174778340	4/23/16 1:20	4/23/16 5:35	4/23/16 1:20	4/23/16 5:35	FOT	OVS	77W	IOBQV	0
174773456	4/22/16 21:00	4/22/16 23:00	4/22/16 20:20	4/22/16 22:20	OVS	EEP	32A	WPBQV	85
174777542	4/23/16 3:00	4/23/16 5:00	4/23/16 3:00	4/23/16 5:00	EEP	OVS	32A	WPBQV	0
174773464	4/22/16 21:00	4/22/16 23:40	4/22/16 20:45	4/22/16 23:25	OVS	MJT	3KR	HSBQV	45
174777538	4/23/16 0:25	4/23/16 3:10	4/23/16 0:25	4/23/16 3:10	MJT	OVS	3KR	HSBQV	0

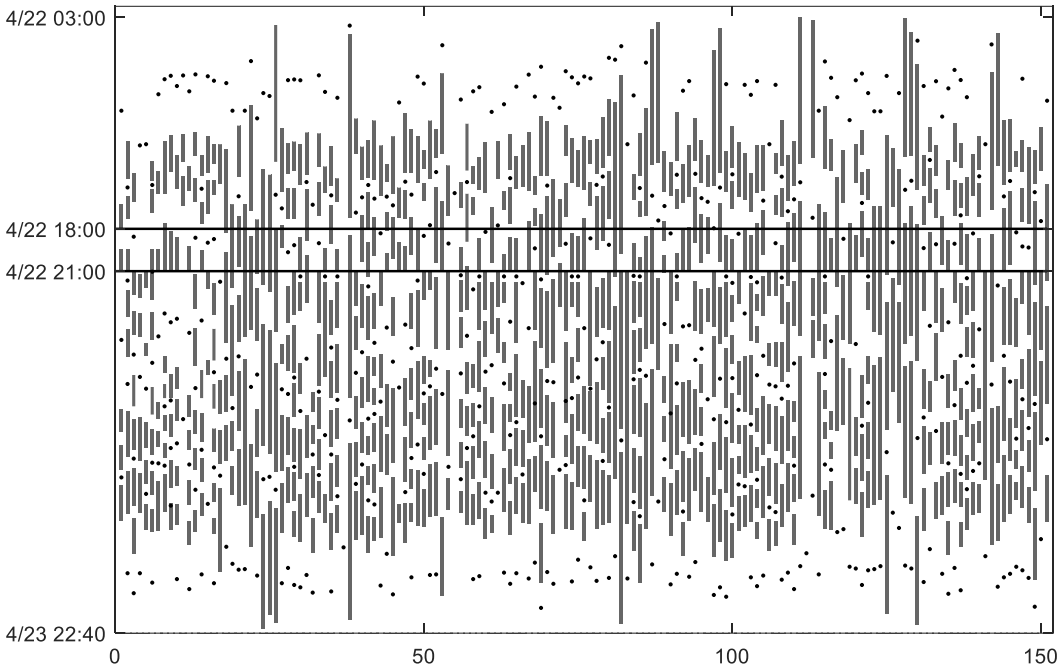

图 10 问题四规划后的航班计划图

值得注意的是，该结果虽然只是该优化问题的一个可行解，并不一定是最优解，但考虑到计算能力限制，直接寻找最优解可能会付出较大的时间开销，在应对一些不可知的突发事件时，如恐怖袭击、飞机故障或意外事故等，需要尽快调整航班规划，恢复运行，此时快速的找到可行的次优解，根据实施状况进一步动态优化，可能会起到更好的效果。

六、模型的特点、推广与改进

模型建立的特点：本文的不正常航班恢复问题采用 0-1 整数线性规划模型，该模型常用于描述任务分配、资源管理等的优化问题。0-1 整数规划模型形式多变，具有很强的灵活性，通过改进目标函数、添加或减少约束，可以与实际问题很好的匹配，具有较大的应用价值。

问题一采用了改进的资源指派模型来解决单机型的航班恢复问题，该模型易于理解，对飞机调度问题描述准确、简洁，切合同一机型所有航班恢复的策略，对于单机型的问题求解较为有效。

问题二针对多机型航班采用了时空网络模型，该模型可以直观的描述不同机型之间的调配问题，便于刻画不同机型飞机置换所产生的额外成本。但该模型由于数据庞大，求解较为困难。因此我们采用了贪心随机模拟退火算法，可以避免陷入局部最优解。

问题三创造性地借助了信号与系统的思想建立语谱图模型来考虑不同机型的载客情况，该模型可以直观的刻画不同机型飞机置换而带来的部分旅客无法登机的问题。但目前该模型还缺少严谨的解算方法，因此我们最终还是回到整数规划模型，采用问题二中的算法求解。

问题四中模型从同行旅客的角度来建立目标函数，由于只需考虑联程航班的最终目的地来计算延误时间，对模型进行优化，降低了计算难度。

本文针对机场临时关闭情况建立 0-1 整数规划模型，经过适当的改进可以还用于其他场景航班恢复问题。如飞机临时出现故障需要维修，多个枢纽航站发生长时间关闭等情况。

对于模型后续的改进方向，我们提出以下几点：

一、可以进一步考虑用信号与系统的思想来对该航班恢复问题进行建模，系统地利用信号处理的手段简化求解过程，从而避免整数规划求解计算量大的问题。

二、可以利用集合的思想，借助集合间的映射关系来将航班与飞机进行匹配，从而制定合理的航班飞行计划。

三、在模型及算法中考虑旅客失望的溢出成本、飞机调配的成本、航班延误赔偿成本等更多因素，并令最长延误时间最小来制定航班执行计划。

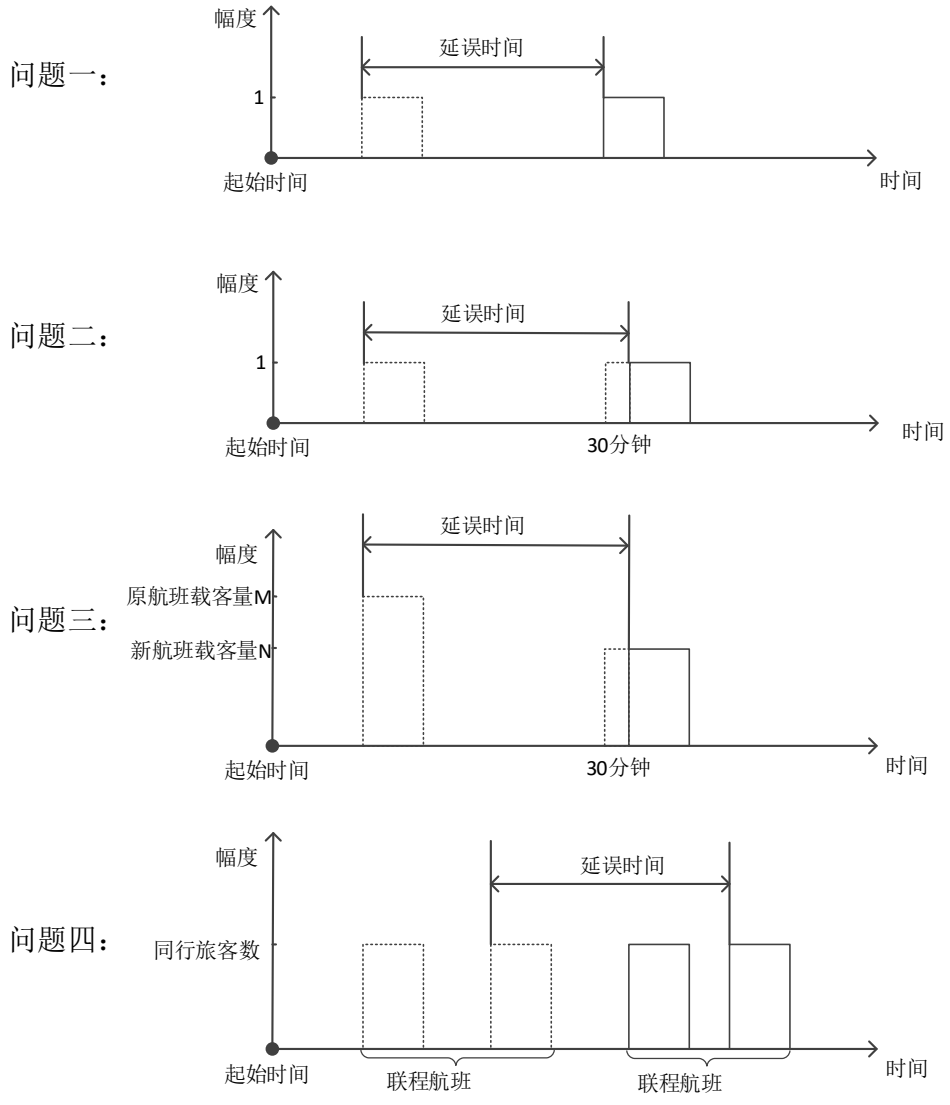

图 11 基于信号与系统思想的航班恢复问题模型示意图

四、利用信号与系统方法对不正常航班的恢复问题建立模型，利用信号的傅里叶变换及相关性质对模型进行简化求解。具体的模型建立如下图所示，下面分别针对本文中的四个问题对此进行说明。

(1) 问题一中左边的单位脉冲表示原航班，右侧表示调整后的新航班，两个脉冲之间的时间间隔记为延误时间，决策变量为每个脉冲右移的距离，目标为所有脉冲右移距离和最小，约束为脉冲只能后移或者丢失，同一时刻的不超过 5 个脉冲发生和消失。

(2) 问题二左侧单位脉冲代表原航班，右侧实线单位脉冲表示另一机型的航班，中间的虚线脉冲表示由于不同机型飞机置换产生的额外 30 分钟延迟。决策标量，约束及目标函数同问题一。

(3) 问题三在问题二的模型上以航班的载客数为幅度，将原航班记为幅度为 M 的脉冲，将新航班记为幅度为 N 的脉冲，决策变量和约束与之前相同，目标为所有脉冲右移时间与脉冲幅度衰减等效延迟时间的加和。

(4) 问题四以同行旅客数为幅度，将两个幅度相同的脉冲看作一个信号来表示联程航班，航班的延误则为信号结束时刻右移的时间。目标则为所有信号右移距离与其幅度乘积的加和。

七、参考文献

[1] 赵秀丽. 航空公司不正常航班恢复模型及算法研究[D]. 南京航空航天大学, 2010.

[2] 张力菠，鲍和映. 基于离散时空网络的不正常航班调度模型[J]. 系统工程，2013(12):60-68.

[3] Jon D. Petersen, Gustaf Sölveling, John-Paul Clarke, Ellis L. Johnson, and Sergey Shebalov (2012). An Optimization Approach to Airline Integrated Recovery. Transportation Science, 46(4), 482-500.

[4] 卢开澄，卢华明，图论及其应用（第二版），北京：清华大学出版社，1995.

[5] J. M. Rosenberger, E. L. Johnson, G. L. Nemhauser(2003). Rerouting Aircraft for Airline Recovery. Transportation Science, 37(4), 408–42.

[6] 唐小卫，高强，朱金福. 不正常航班恢复模型的贪婪模拟退火算法研究[J]. 预测，2010, 29(1):66-70.

[7] Ahmad I. Z. Jarrah, Gang Yu, Nirup Krishnamurthy, and Ananda Rakshit (1993). A Decision Support Framework for Airline Flight Cancellations and Delays. Transportation Science, 27(3), 266-280.

[8] Stephen J. Maher (2015). Solving the Integrated Airline Recovery Problem Using Column-and-Row Generation. Transportation Science, 50(1), 216-237.

[9] 赵鹏. 不正常航班恢复问题研究[D]. 北京邮电大学, 2015.

[10]吴大正，信号与线性系统分析（第 4 版），北京：高等教育出版社，P95，2005.

附录：

附录一：航班时间计划图绘图程序

程序名：timeflightplot.m
代码：

```
clear stack;
[a,airport,c]=xlsread('1.xlsx');
open=1461302220; %飞机 9 最早起飞航班时间  4/22/16 5:17
close=1461451200; %飞机 9 最晚起飞航班时间  4/23/16 22:40
startunusetime=1461348000; %机场故障起点时间 18:00
endunusetime=1461358800; %机场故障终点时间 21:00
for i=1:length(flight)
    x=find(Flysta(:,2)==flight(i));
    flightnum(i)=length(x);
end
isOVS=ones(length(flight),2*max(flightnum)+1);
for i=1:length(flight)
    x=find(Flysta(:,2)==flight(i));
    thisflysta=Flysta(x,1);
    thisflyend=Flyend(x,1);
    startairport=char(airport(:,1));
    endairport=char(airport(:,2));
    thisstartairport=startairport(x,1:3);
    thisendairport=endairport(x,1:3);

    for j=1:length(x)
        %为 0 表示在 OVS 机场，起点在 OVS 则停留在 OVS
        isOVS(i,2*flightnum(i)+1+1-(2*j-1))=sum(thisstartairport(j,1:3)-'OVS');
    end % 1 表示在飞行过程，0 表示在 OVS，其他表示不在 OVS
    isOVS(i,1)=sum(thisendairport(flightnum(i),1:3)-'OVS');
    enddduration=close-thisflyend(length(x));
    startduration=thisflysta(1)-open;
    thisflyduration=thisflyend-thisflysta;
    restduration=thisflysta(2:end)-thisflyend(1:end-1);
    stack(i,1)=enddduration;
    for j=1:length(x)-1
        stack(i,2*j)=thisflyduration(length(x)-j+1);
        stack(i,2*j+1)=restduration(length(x)-j);
    end
    stack(i,2*(j+1))=thisflyduration(length(x)-j);
    stack(i,2*(j+1)+1)=startduration;
    clear thisflyduration;
    clear restduration;
end
h=bar(stack,0.5,'stacked');
for i=1:(size(stack,2)-1)/2
    set(h(2*i-1),'facecolor','w');
```

```
        set(h(2*i),'facecolor',[0.4 0.4 0.4]);
    end
    set(h(2*i+1),'facecolor','w');

    for i=1:length(flight)
        locat(i,1)=stack(i,1)/2;
        for j=2:2*max(flightnum)+1
            locat(i,j)=sum(stack(i,1:j-1))+stack(i,j)/2;
        end
    end

    hold on;
    xgrid=meshgrid(1:2*max(flightnum)+1);
    for i=1:length(flight)
        for j=1:2*max(flightnum)+1
            if isOVS(i,j)==0
                plot(xgrid(j,i),locat(i,j),'k.');
            end
        end
    end
    line([0,17],[close-endunusetime close-endunusetime],'LineWidth',1.2,'color','k');
    line([0,17],[close-startunusetime close-startunusetime],'LineWidth',1.2,'color','k');
    set(gca,'xtick',1:16);
    set(gca,'xticklabel',1:16);
    xlim([0,17]);
```

附录二：资源指派算法主程序

程序名：main.m

代码：

```
%2017.9.16
clear all;
[Aircrafts_NUM,Aircrafts_TXT] = xlsread('Aircrafts.xlsx');
AirportClosures = xlsread('AirportClosures.xlsx');
Paxinfo = xlsread('Paxinfo.xlsx');
[Schedules_NUM,Schedules_TXT] = xlsread('Schedules.xlsx','A2:G750');

global T F A Flysta Flyend Dptf Schedules_TXT;
T = 1461294000:300:1461438300;                          %时间点的集合
F = Schedules_NUM(1:97,1);
%机型为 9 的航班集合--航班唯一编号
A = unique([unique(Schedules_TXT(1:97,1));unique(Schedules_TXT(1:97,2))]);
%9 型号所有机场的集合
P = Aircrafts_TXT(2:17,1);
%9 型号飞机的集合--飞机尾号
P_loc = Aircrafts_TXT(2:17,5);
%飞机初始机场
Flysta = Schedules_NUM(1:97,2);
```

```
%9 型号飞机计划起飞时间
Flyend = Schedules_NUM(1:97,3);
%9 型号飞机计划到达时间
Fly = Flyend - Flysta;
FtoOVS = Schedules_NUM( find(strcmp(Schedules_TXT(1:97,2),'OVS')),1);
FdeOVS = Schedules_NUM( find(strcmp(Schedules_TXT(1:97,1),'OVS')),1);
% date = datestr(Aircrafts_NUM(:,1)/86400 + datenum(1970,1,1));
% Dptf = Schedules_NUM(1:97,2);       %9 型号飞机最早允许起飞时间
Dptf = func_Dpt(FtoOVS,FdeOVS);       %9 型号飞机最早允许起飞时间

loc = zeros(length(P),length(A));
for i = 1:length(P)
    loc(i,strcmp(A , P_loc(i)) ) = 1;
end

f = [];
Aeq2 = sparse(length(F),length(f));
beq2 = ones(length(F),1);
Aeq = [];
beq = length(F);
temp = 1;
Tf = [];
for i = 1:length(F)
    lp = length(P);
    f = [ f , repmat(func_Tf(F(i)) - Dptf(i) , [1,lp]) ];
    Aeq2(i,temp:temp+lp*length(func_Tf(F(i))    )-1    )    =
ones(1,length(func_Tf(F(i) ))*lp );
    temp = temp+ lp*length(func_Tf(F(i)));
    Aeq = [ Aeq , ones(1,lp*length(func_Tf(F(i)) )) ];
    Tf = [Tf , repmat(func_Tf(F(i)),[1,lp] )] ;
end
AA = sparse(length(f)/lp,length(f));
bb = ones(length(f)/lp,1);
for i = 1:length(f)/lp
AA(i,(i-1)*lp+1:i*lp) = ones(1,lp);
end
AAA = sparse(length(F),length(f));
temp1 = ones(1,length(F));
for i = 2:length(F)
    temp1(i) = temp1(i-1) + lp*length(func_Tf(F(i-1)));
end
for i = 1:length(F)
    AAA(i,i + temp1 ) = 1;
end
bbb = ones(97,1);

% [x , fval] = bintprog(f,AA,b,Aeq,beq);
intcon = 1:length(f);
lb = zeros(length(f),1);
ub = ones(length(f),1);
```

```
[x , fval] = intlinprog(f,intcon,[AA;AAA],[bb;bbb],[Aeq;Aeq2],[beq;beq2],lb,ub);
ind = find(x==1);

order = Tf(ind)';
order1 = datestr(order/86400 + datenum(1970,1,1));
```

附录三：模拟退火算法主程序

程序名：main.m

代码：

```
load('data.mat');
load('ccc');
global A T F P Ps Flysta Flyend Fly FtoOVS FdeOVS Dptf lt lf lp;
T=1461294000:300:1461438300;
%时间点的集合
lt = length(T);
F = Schedules_NUM(1:97,1);
%机型为 9 的航班集合--航班唯一编号
A = unique([unique(Schedules_TXT(1:97,1));unique(Schedules_TXT(1:97,2))]);
%9 型号所有机场的集合
P = str2num(cell2mat(Aircrafts_TXT(2:17,1)));
%9 型号飞机的集合--飞机尾号
Ps = str2num(cell2mat(Schedules_TXT(1:97,4)));
P_num = 12;
Flysta = Schedules_NUM(1:97,2);
%9 型号飞机计划起飞时间
Flyend = Schedules_NUM(1:97,3);
%9 型号飞机计划到达时间
Fly = Flyend - Flysta;
lt = length(T);
lf = length(F);
lp = length(P);
FtoOVS = [Schedules_NUM( find(strcmp(Schedules_TXT(1:97,2),'OVS')),1) ,
str2num(cell2mat(Schedules_TXT( find(strcmp(Schedules_TXT(1:97,2),'OVS')),4))) ];
%所有到达机场为 OVS 的航班集合
FdeOVS = [Schedules_NUM( find(strcmp(Schedules_TXT(1:97,1),'OVS')),1),
str2num(cell2mat(Schedules_TXT( find(strcmp(Schedules_TXT(1:97,1),'OVS')),4))) ];
%所有起飞机场为 OVS 的航班集合
% date = datestr(Aircrafts_NUM(:,1)/86400 + datenum(1970,1,1));
% Dptf = Schedules_NUM(1:97,2);      %9 型号飞机最早允许起飞时间
Dptf = func_Dpt(FtoOVS,FdeOVS);      %9 型号飞机最早允许起飞时间
t0 = 10000;                                  %  初温 t0
iLk = 10000 ;                                %内循环最大迭代次数 iLk
oLk = 1000 ;                                 %外循环最大迭代次数 oLk
lam = 0.99 ;                                 % λlambda
istd = 0.001 ;                               %  若内循环函数值方差小于 istd 则停止
ostd = 0.001 ;                               %若外循环函数值方差小于 ostd 则停止
```

```
ilen = 50 ;                                  %  内循环保存的目标函数值个数
olen = 50 ;                                  %外循环保存的目标函数值个数
%主程序
ft = [];
ft_len = zeros(1,lf+1);
for i = 1:lf
      ft = [ft , T ];
      ft_len(i+1) = ft_len(i) + length(func_Tf(F(i)) );
end
% path = sparse(lf*lt*lp,1);                                  %初始路径
path = ftp2path(ccc,F,T,P);
val = func_J(ft,path);                                       %初始路径的评价
ores = zeros( 1,olen) ;                                      %外循环保存的目标函数值
e0 = val ;                                                   %  能量初值 e0
t = t0 ; %  温度 t
for out = 1 : oLk
   ires = zeros( 1 , ilen ) ;
   tic
    for in = 1 : iLk                                         %内循环模拟热平衡过程
          flag = 0;
        [ newpath , ~ ] = swap( path ) ;                     %  产生新状态
        fullnewpath = full(newpath);
        [a,b] = find(newpath==1);
        %===============约束条件=======================
              %约束 0    最早起飞时间约束
        for k = 1:lf
              ftp = path2ftp(newpath);
              if (ftp(k,2) < floor(Dptf(k)/300)*300)
                    flag = 1;
                    break;
              end
        end
        if flag
              continue;
        end
        %约束 1
        for k = 1:lf
              if (sum(fullnewpath( (k-1)*lt*lp+1: k*lt*lp ) ) ~= 1)
                    flag = 1;
                    break;
              end
        end
        if flag
              continue;
        end
        %约束 2
        for k = 1:lp
              if (sum( fullnewpath( (k-1)*lp+1:lp: k*lp ) ) > 1)
```

```
                flag = 1;
                break;
            end
        end
        if flag
            continue;
        end
        %约束 3
        if (func_check4(newpath))
            continue;
        end
        %=========================================================
        e1 = func_J( ft , newpath ) ;          %新状态能量（新状态下目标函数的取值）
        % Metropolis 抽样稳定准则（以一定的概率选择新的状态）
        if e1 < e0
            path = newpath ; %  更新最佳状态
            e0 = e1 ;
        elseif    exp( - ( e1 - e0 ) / t )>rand
            path = newpath ; %  更新最佳状态
            e0 = e1;
        else
            continue;
        end
        ires = [ ires( 2 : end ) e0 ] ; %保存新状态能量
        %  内循环终止准则：连续 ilen 个状态能量波动小于 istd
          if std( ires , 1 ) < istd
             break ;
          end
          ftp = path2ftp(path);
          ftp = [ftp ftp(:,2) + Fly, ftp(:,2) - Flysta];
          delay = sum(ftp(:,5))/60
    end
             toc
    ores = [ ores( 2 : end ) e0 ] ; %  保存状态能量
     %  外循环终止准则：连续 olen 个状态能量波动小于 ostd
     if std( ores , 1 ) < ostd
          break ;
     end
     t = lam * t ;%降温
end
% [a,b] = find(path==1);
% ff = F;
% tt = ft(a)';
% pp = P(b);
```

附录四：产生随机解的子函数

程序名：swap.m

代码：

```
function [ newpath , position ] = swap( oldpath    )
%对 oldpath 的一个起飞时间进行随机选择
% number  为产生的新路径的个数
% position  为对应 newpath 互换的位置

a = length(oldpath);
len = length(find(oldpath==1));
newpath = sparse(a,1);
fight = randi(    len, 1 , 1 );                                    %随机选择一个航班
position = randi( a , 1 , 100 ) ;                                  %  随机产生的位置
ind = find(oldpath==1);
ind(fight) = position(1);
newpath(ind) = 1;
for i = 2:100
      if(length(find(newpath)) == len )
            break;
      else
            ind(fight) = position(i);
            newpath(ind) = 1;
      end
end
end
```

附录五：解的稀疏矩阵形式转 0-1 矩阵形式子函数

程序名：path2ftp.m

代码：

```
function [ ftp ] = path2ftp( path)
%UNTITLED  此处显示有关此函数的摘要
%     此处显示详细说明
global F T P;
lt = length(T);
lf = length(F);
lp = length(P);
a = find(path==1);
pp = zeros(lf,1);
tt = zeros(lf,1);
ff = zeros(lf,1);
% for i = 1:lf
%         pp(i) = mod(a(i),lp);
%         a(i) = (a(i) - mod(a(i),lp))/lp;
%         tt(i) = mod(a(i),lt);
%         a(i) = (a(i) - mod(a(i),lt))/lt;
%         ff(i) = mod(a(i),lf);
% end
indpp = mod(a,lp);
indpp(indpp==0) = lp;
pp = P(indpp);
```

```
a = (a - indpp)/lp + 1;
indtt = mod(a,lt);
indtt(indtt==0) = lt;
tt = T(indtt);
a = (a - indtt)/lt + 1;
indff = mod(a,lf);
indff(indff==0) = lf;
ff = F(indff);
ftp = [ff tt' pp];
end
```

附录六：解的 0-1 矩阵形式转稀疏矩阵形式子函数

程序名：ftp2path.m

代码：

```
function [ path ] = ftp2path( ftp,F,T,P    )
%UNTITLED  此处显示有关此函数的摘要
%     此处显示详细说明
[a,b] = size(ftp);
lf = length(F);
lt = length(T);
lp = length(P);
path = sparse(lf*lt*lp,1);
for i = 1:a
ff = find(F == ftp(i,1));
tt = find(T > ftp(i,2),1) - 1;
pp = find(P == ftp(i,3));
path((ff-1)*lt*lp+(tt-1)*lp+pp) = 1;
end
```

附录七：y_{atp}约束函数

程序名：check3.m

代码：

```
function [ flag ] = check3( path )
%UNTITLED  此处显示有关此函数的摘要
%     此处显示详细说明
%    flag = 1      表示解不满足此约束
global F T P Ps    Dptf Fly FtoOVS FdeOVS lp lt lf;
dddd = floor(Dptf/300)*300;
ftp = path2ftp(path);
for k = 1:lp
for i = 1:lf
      xfsp1 = 0;
      xfsp2 = 0;
      f = ftp(i,1);
      t = ftp(i,2);
      p = ftp(i,3);
```

```
        comf = F(Ps==p);
        for j = 1:length(comf)
                if(dddd(ftp(:,1)==comf(j)      )<=  ftp(  (ftp(:,1)==comf(j)  )  ,2)  &&
ftp((ftp(:,1)==comf(j)  )  ,2)  <  t  -  Fly(i)  &&~isempty(find(FtoOVS(:,1)==comf(j)))  &&
~isempty(find( FtoOVS(:,2)==p ) )   )
                    xfsp1 = xfsp1 +1;
                end
                if(dddd(ftp(:,1)==comf(j)      )<=      ftp((ftp(:,1)==comf(j)  )  ,2)  &&
ftp((ftp(:,1)==comf(j)              )              ,2)<              t                    &&
~isempty(find(FdeOVS(:,1)==comf(j)))&&~isempty(find( FdeOVS(:,2)==p ) )     )
                    xfsp2 = xfsp2 +1;
                end
        end
        if(xfsp1-xfsp2<=1)
            flag =0;
             break;
        else
            flag = 1;
            break;
        end
    end
    end
    end
```

附录八：获取允许航班 f 最早起飞时间集合子函数

程序名：func_Tf.m

代码：

```
function [ Tf ] = func_Tf( f )
%UNTITLED  此处显示有关此函数的摘要
%
%     输入为航班编号 f
%     输出为航班 f 的所有允许起飞时间集合,
global T F Dptf;
t = Dptf( (F==f) );
Tf = T( (T>=t));
end
```

附录九：计算所有航班最早允许起飞时间子函数

程序名：func_Dpt.m

代码：

```
function [ dptime ] = func_Dpt( arrive_fight,departure_fight )
%UNTITLED  此处显示有关此函数的摘要
%     此函数用来计算航班的最早允许起飞时间
%     输入到达的航班号、离开的航班号
%     输出最早允许的起飞时间
global F Flysta Flyend;
```

```
arrive_num = length(arrive_fight);                    % 到达的飞机数目
departture_num = length(departure_fight);             % 离开的飞机数目
dptime = zeros(arrive_num+departture_num,1);
for i = 1:arrive_num
if( 1461348000<Flyend( F==arrive_fight(i) ) && Flyend(F==arrive_fight(i)) < 1461358800)
    dptime( F==arrive_fight(i)) = Flysta(F==arrive_fight(i) ) + 1461358800 - Flyend(F==arrive_fight(i));
else
    dptime(F==arrive_fight(i)) = Flysta(F==arrive_fight(i)) ;
end
end

for i = 1:departture_num
if( 1461348000<Flysta(F==departure_fight(i) ) && Flysta(F==departure_fight(i) )< 1461358800)
    dptime(F==departure_fight(i) ) = Flysta(F==departure_fight(i) ) + 1461358800 - Flysta(F==departure_fight(i));
else
    dptime(F==departure_fight(i) ) = Flysta(F==departure_fight(i) );
end
end
end
```