**案例12：Braneast 航空公司的航线安排**

（第六章：图论；难度：中等）

**案例背景：**

Braneast 航空公司必须确定应当有多少飞机服务于波士顿一纽约一华盛顿空中走廊，以及要安排哪些航班。表1给出该公司每天可以安排的航班。飞机降落后，至少要在机场停留1小时后才能再次出发。必须确保这个模型包括运营飞机的固定成本。该公司希望能够设计一种运营方案，使该公司日利润最大。

表1 航空公司每天安排的航班

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **离开城市** | **离开时间** | **到达城市** | **到达时间** | **收入 （美元）** | **航班的可变成本 （美元）** |
| 纽约 | 上午9点 | 华盛顿 | 上午10点 | 900 | 400 |
| 纽约 | 下午2点 | 华盛顿 | 下午3点 | 600 | 350 |
| 纽约 | 上午10点 | 波士顿 | 上午11点 | 800 | 400 |
| 纽约 | 下午4点 | 波士顿 | 下午5点 | 1200 | 450 |
| 华盛顿 | 上午9点 | 纽约 | 上午10点 | 1100 | 400 |
| 华盛顿 | 下午2点 | 纽约 | 下午3点 | 900 | 350 |
| 华盛顿 | 上午10点 | 波士顿 | 中午12点 | 1500 | 700 |
| 华盛顿 | 下午5点 | 波士顿 | 晚上7点 | 1800 | 900 |
| 波士顿 | 上午 10点 | 纽约 | 上午11点 | 900 | 500 |
| 波士顿 | 下午2点 | 纽约 | 下午3点 | 800 | 450 |
| 波士顿 | 上午11点 | 华盛顿 | 下午1点 | 1100 | 600 |
| 波士顿 | 下午3点 | 华盛顿 | 下午5点 | 1200 | 650 |

**思考题**

1. 该航空公司至少要准备多少架飞机，才能让开行上述所有航线？请建立飞机数量最少的线性规划模型，并求解该模型。（假设所有飞机都正常运行）
2. 假设该航空公司拥有的飞机数量总数为4，飞机抵达某一机场后，需要至少停留在该机场1小时，才能进行下一段航程。请设计一种运营方案，使得该公司日利润最大。
3. 如果飞机停在机场超过1小时，机场将向航空公司收取每架次每小时50美元的机位占用费用。收取时间从上午9:00至晚上7:00（其他时间不收取）。请问航空公司的最大利润与运营方案是否会发生变化？

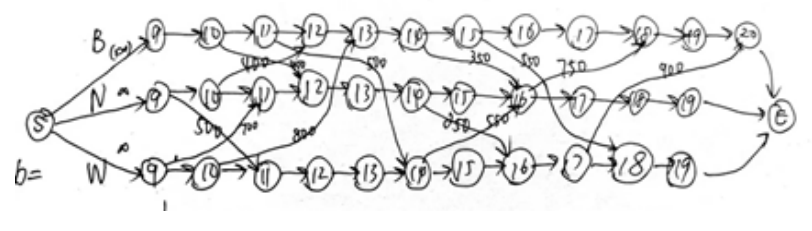
**案例解答**

1. 建立该问题的数学模型：
2. 参数设置

机场编号（1-波士顿,2-纽约，3-华盛顿）

:出发机场编号 :到达机场编号 出发时间 到达时间

节点:机场，时刻代表的节点。



:虚拟的起始节点S :虚拟的终止节点E

到的最大流量 >即从机场飞往机场，起飞时间为,到达时间为的航线的最大流量。

到的利润 >即从机场飞往机场，起飞时间为,到达时间为的航线的利润

代表了所有可行航线的集合。

代表了包括中的所有航线，以及起始节点S与结束节点E的虚拟航线的集合。

1. 决策变量

:到的流量

1. 目标函数

机队数量最少(最小化总流量)

1. 约束：

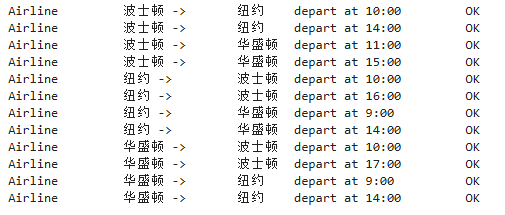
航线的流量不超过最大流量限制:

除起始节点S和结束节点E外，其他的每个节点的流入流量=流出流量：

起始节点S的流出流量=结束节点的流入流量：

由于变量较多，因此使用python进行求解，求解的代码为：

求解结果为：**最小飞机数量为6**。输出结果显示，所有的航线都在正常运行



（2）第二问的约束条件与第一问不同，第二位的约束条件为：

利润最高

此外还添加了约束：机队数量为4

使用python进行求解，求解代码为：

*# 定义线性规划问题*

**from** pulp **import** **\***

prob **=** LpProblem("Airlines2", LpMaximize)

*# 定义变量*

f **=** LpVariable**.**dicts("f",[(i,j,ts,te) **for** i **in** range(1,3**+**1) **for** j **in** range(1,3**+**1) **for** ts **in** range(8,22) **for** te **in** range(8,22)],lowBound**=**0,cat**=**'Integer')

TotalAirplane **=** LpVariable("TotalAirplane", lowBound**=**0, cat**=**'Integer')

TotalRevenue **=** LpVariable("TotalRevenue",cat**=**'Continuous')

*# 定义目标函数*

prob **+=** TotalRevenue

*# 定义约束条件*

prob **+=** TotalAirplane **==** lpSum(f[1,j,8,9] **for** j **in** range(1,3**+**1))

prob **+=** TotalRevenue **==** lpSum(f[i,j,ts,te] **\*** revenue[i**-**1][j**-**1][ts**-**8][te**-**8] **for** i **in** range(1,3**+**1) **for** j **in** range(1,3**+**1) **for** ts **in** range(8,22) **for** te **in** range(8,22) **if** flow[i**-**1][j**-**1][ts**-**8][te**-**8] **>** 0 )

prob **+=** TotalAirplane **==** 4

*# 边<i,j,ts,te>的流量不超过最大流量限制:*

**for** i **in** range(1,3**+**1):

**for** j **in** range(1,3**+**1):

**for** ts **in** range(8,21):

**for** te **in** range(8,21):

prob **+=** f[i,j,ts,te] **<=** flow[i**-**1][j**-**1][ts**-**8][te**-**8]

prob **+=** f[i,j,ts,te] **>=** 0

*# 除起始节点S和结束节点E外，其他的每个节点node[i,ts]的流入流量=流出流量*

**for** i **in** range(1,3**+**1):

**for** ti **in** range(9,21):

prob **+=** lpSum(f[h,i,th,ti] **for** h **in** range(1,4) **for** th **in** range(8,21)) **==** lpSum(f[i,j,ti,tj] **for** j **in** range(1,4) **for** tj **in** range(9,22) )

*# 起始节点S的流出流量=结束节点的流入流量*

prob **+=** lpSum(f[1,j,8,9] **for** j **in** range(1,3)) **==** f[1,1,20,21] **+** f[2,1,19,21] **+** f[3,1,19,21]

prob**.**solve()

**if** LpStatus[prob**.**status] **==** "Optimal":

print("找到最优解")

**else**:

print("No solution")

print(value(TotalRevenue))

*# 查看正常运行的航线*

city **=** ["波士顿","纽约","华盛顿"]

**for** i **in** range(1,4):

**for** j **in** range(1,4):

**for** ts **in** range(9,20):

**for** te **in** range(9,21):

**if** flow[i**-**1][j**-**1][ts**-**8][te**-**8] **==** 1:

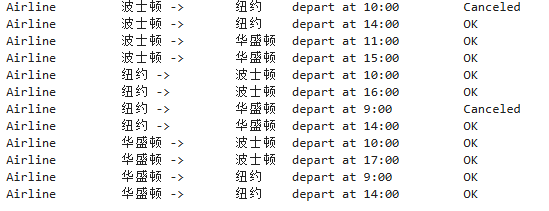
status **=** "OK"

**if** value(f[i,j,ts,te]) **==** 0:

status **=** "Canceled"

print(f"Airline \t{city[i**-**1]} -> \t{city[j**-**1]} \tdepart at {int(ts)}:00 \t{status}" )

计算得到**最大利润为5750.0**，对应的调度方案为：



其中，OK代表该航线正常运营，Canceled代表改航线不运营

（3）若考虑飞机在机场的停机费用，只需调整代表机场内部流动的边的权值。求解代码为：

*# 在机场内部每小时的流动停机成本*

REST\_COST **=** 50

**for** i **in** range(0,3):

**for** t **in** range(9,19):

revenue[i][i][t**-**8][t**+**1 **-** 8] **=** **-**1**\***REST\_COST

revenue[0][0][19**-**8][20**-**8] **=** **-**1**\***REST\_COST

*# 定义线性规划问题*

**from** pulp **import** **\***

prob **=** LpProblem("Airlines2", LpMaximize)

*# 定义变量*

f **=** LpVariable**.**dicts("f",[(i,j,ts,te) **for** i **in** range(1,3**+**1) **for** j **in** range(1,3**+**1) **for** ts **in** range(8,22) **for** te **in** range(8,22)],lowBound**=**0,cat**=**'Integer')

TotalAirplane **=** LpVariable("TotalAirplane", lowBound**=**0, cat**=**'Integer')

TotalRevenue **=** LpVariable("TotalRevenue",cat**=**'Continuous')

*# 定义目标函数*

prob **+=** TotalRevenue

*# 定义约束条件*

prob **+=** TotalAirplane **==** lpSum(f[1,j,8,9] **for** j **in** range(1,3**+**1))

prob **+=** TotalRevenue **==** lpSum(f[i,j,ts,te] **\*** revenue[i**-**1][j**-**1][ts**-**8][te**-**8] **for** i **in** range(1,3**+**1) **for** j **in** range(1,3**+**1) **for** ts **in** range(8,22) **for** te **in** range(8,22) **if** flow[i**-**1][j**-**1][ts**-**8][te**-**8] **>** 0 )

prob **+=** TotalAirplane **==** 4

*# 边<i,j,ts,te>的流量不超过最大流量限制:*

**for** i **in** range(1,3**+**1):

**for** j **in** range(1,3**+**1):

**for** ts **in** range(8,21):

**for** te **in** range(8,21):

prob **+=** f[i,j,ts,te] **<=** flow[i**-**1][j**-**1][ts**-**8][te**-**8]

prob **+=** f[i,j,ts,te] **>=** 0

*# 除起始节点S和结束节点E外，其他的每个节点node[i,ts]的流入流量=流出流量*

**for** i **in** range(1,3**+**1):

**for** ti **in** range(9,21):

prob **+=** lpSum(f[h,i,th,ti] **for** h **in** range(1,4) **for** th **in** range(8,21)) **==** lpSum(f[i,j,ti,tj] **for** j **in** range(1,4) **for** tj **in** range(9,22) )

*# 起始节点S的流出流量=结束节点的流入流量*

prob **+=** lpSum(f[1,j,8,9] **for** j **in** range(1,3)) **==** f[1,1,20,21] **+** f[2,1,19,21] **+** f[3,1,19,21]

prob**.**solve()

**if** LpStatus[prob**.**status] **==** "Optimal":

print("找到最优解")

**else**:

print("No solution")

print(value(TotalRevenue))

*# 查看正常运行的航线*

city **=** ["波士顿","纽约","华盛顿"]

**for** i **in** range(1,4):

**for** j **in** range(1,4):

**for** ts **in** range(9,20):

**for** te **in** range(9,21):

**if** flow[i**-**1][j**-**1][ts**-**8][te**-**8] **==** 1:

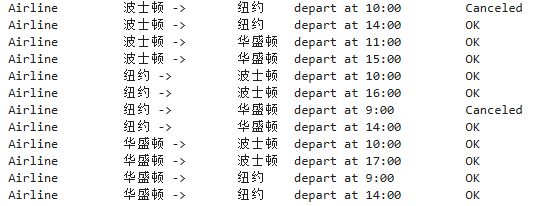
status **=** "OK"

**if** value(f[i,j,ts,te]) **==** 0:

status **=** "Canceled"

print(f"Airline \t{city[i**-**1]} -> \t{city[j**-**1]} \tdepart at {int(ts)}:00 \t{status}" )

运行上述代码，得到**最大利润为5150.0**，对应的运营方案为：



其中，OK代表该航线正常运营，Canceled代表改航线不运营。可以看到,当停机成本上升时，总利润下降了。