HUNAN UNIVERSITY

MATHEMATICAL MODELING, JULY 2020

作业二

研究生第 12 组

甘毅辉 (组长)

李峥嵘 (组员)

李俊杰 (组员)

2020 年 7 月

1 问题描述

已知某物资有 8 个配送中心可以供货，有 15 个部队用户需要该物资。表中给出了配送中心和部队用户之间单位物资的运费，15 个部队用户的物资需求量和 8 个配送中心的物资储备量。我们需要解决两个问题：

1. 利用表中数据，求出最小运费调用计划。
2. 增加约束条件：每个配送中心，可以对用户配送物资，也可以不对用户配送物资；若配送物资的话，配送量要大于等于 1000 且小于等于 2000。求此时的最小运费调用计划。

2 问题 1 的求解

这是一个配送问题，每个物资中心的物资储备量应大于其向部队用户运输的物资总量。

这个问题可以用线性规划模型来解决，设第 j 个配送中心为 A_j，第 i 个部队用户为 B_i，$x_{ij}(i=1,2,\cdots,15;j=1,2,\cdots,8)$ 表示 A_j 向 B_i 运输的物资量。c_{ij} 表示从 A_j 向 B_i 运输物资的单位运费。d_j 表示 A_j 的物资储备量，e_i 表示 B_i 的物资需求量。

目标函数是使总的运费最小，即：

$$min \sum_{i=1}^{15} \sum_{j=1}^{8} c_{ij} x_{ij}.$$

约束条件有两类:

1. 每个配送中心配送物资总量约束，每个配送中心配送的物资总量应小于其物资储备量，即 $\sum\limits_{i=1}^{15} x_{ij} \le d_j, j=1,2,\cdots,8$
2. 每个部队用户的需求量约束，所有配送中心运到第 i 个部队用户的物资问题应等于其需求量，即 $\sum\limits_{j=1}^{8} x_{ij} = e_i, i=1,2,\cdots,15$

综上所述，建立线性规划模型如下：

$$min \sum_{i=1}^{15} \sum_{j=1}^{8} c_{ij} x_{ij}.$$

$$
s.t.\begin{cases}
\sum_{i=1}^{15} x_{ij} \le d_j, j = 1, 2, \cdots, 8 \\
\sum_{j=1}^{8} x_{ij} = e_i, i = 1, 2, \cdots, 15 \\
x_{ij} \ge 0
\end{cases}
$$

利用 python 软件求得 8 个配送中心到 15 个部队用户的最优运量如表1所示，代码见附录, 对应的最小运费为 9244730。

表 1: 8 个配送中心到 15 个部队用户的最优运量

	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
B_1	0	0	0	0	0	3000	0	0
B_2	0	0	0	0	3100	0	0	0
B_3	0	0	0	0	0	0	0	2900
B_4	0	0	3100	0	0	0	0	0
B_5	3100	0	0	0	0	0	0	0
B_6	0	0	0	3400	0	0	0	0
B_7	0	3500	0	0	0	0	0	0
B_8	0	0	0	3200	0	0	0	0
B_9	0	3000	0	0	0	0	0	0
B_{10}	0	0	0	0	0	0	0	3100
B_{11}	0	0	0	0	0	0	3300	0
B_{12}	0	0	0	0	0	0	3200	0
B_{13}	0	0	0	0	0	0	3300	0
B_{14}	0	0	0	0	0	0	0	2900
B_{15}	0	0	0	0	0	0	0	3100

3 问题 2 的求解

在问题 1 的基础上，添加了如果配送中心运输物资，物资必须大于等于 1000，小于等于 2000 的限制条件。用数学符号表示为以下的非线性约束。

$$x_{ij} \in \{0\} \cup [1000, 2000]$$

参考钢管问题，可以引入 0-1 变量，

$$t_{ij}=\begin{cases}1, 运输中心\ A_j\ 向部队用户\ B_i\ 运输物资\\0, 运输中心\ A_j\ 不向部队用户\ B_i\ 运输物资\end{cases}$$

将约束条件转为线性约束

$$\begin{cases}1000*t_{ij}\le x_{ij}\le 2000*t_{ij}\\0\le t_{ij}\le 1, t_{ij}\ 为整数\end{cases}$$

利用 python 软件求得最小运输费用为 12682750,8 个配送中心到 15 个部队用户的最优运量和运输情况如表2和3所示，代码见附录。

表 2: 8 个配送中心到 15 个部队用户的最优运量

	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
B_1	0	0	0	0	1000	2000	0	0
B_2	0	0	0	0	2000	1100	0	0
B_3	0	0	1000	0	0	0	0	1900
B_4	0	0	2000	0	1100	0	0	0
B_5	2000	0	0	1100	0	0	0	0
B_6	1400	0	0	2000	0	0	0	0
B_7	1500	2000	0	0	0	0	0	0
B_8	0	0	0	2000	1200	0	0	0
B_9	1000	2000	0	0	0	0	0	0
B_{10}	0	0	1100	0	0	0	0	2000
B_{11}	0	0	0	0	0	0	2000	1300
B_{12}	0	0	0	1200	0	0	2000	0
B_{13}	0	0	0	1300	0	0	2000	0
B_{14}	0	0	1000	0	0	0	0	1900
B_{15}	0	0	0	1100	0	0	0	2000

附录 1

表 3: 0-1 矩阵

	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
B_1	0	0	0	0	1	1	0	0
B_2	0	0	0	0	1	1	0	0
B_3	0	0	1	0	0	0	0	1
B_4	0	0	1	0	1	0	0	0
B_5	1	0	0	1	0	0	0	0
B_6	1	0	0	1	0	0	0	0
B_7	1	1	0	0	0	0	0	0
B_8	0	0	0	1	1	0	0	0
B_9	1	1	0	0	0	0	0	0
B_{10}	0	0	1	0	0	0	0	1
B_{11}	0	0	0	0	0	0	1	1
B_{12}	0	0	0	1	0	0	1	0
B_{13}	0	0	0	1	0	0	1	0
B_{14}	0	0	1	0	0	0	0	1
B_{15}	0	0	0	1	0	0	0	1

```python
import numpy as np
import cvxpy as cp
import pandas as pd

# 构造目标函数
# 1. 定义决策向量x(i, j)表示第j个配送中心向第i个部队用户运输
# 的物资量
x = cp.Variable((15, 8), pos = True)
# 2. 定义价值向量(单位物资的运费)
c = np.genfromtxt('../data/homework2_data.txt', dtype = \
        float, max_rows = 15, usecols = range(8))
# 3. 定义目标函数
obj = cp.Minimize(cp.sum(cp.multiply(c, x)))

```

```
# 定义约束条件
# 获取需求量
a = np.genfromtxt('../data/homework2_data.txt', dtype= \
        float, max_rows = 15, usecols= 8)
# 获取储备量
b = np.genfromtxt('../data/homework2_data.txt', dtype = \
        float, skip_header = 15) # 读最后一行数量
# 定义约束条件  储备量约束 + 需求量约束
con1 = [cp.sum(x, axis = 1, keepdims = True) == \
         a.reshape(15, 1),
        cp.sum(x, axis = 0, keepdims = True) <= \
         b.reshape(1, 8)]

## 求解问题
prob = cp.Problem(obj, con1)
prob.solve(solver = 'GLPK_MI')
print('最优值', obj.value)
print('最优解', x.value)

xd = pd.DataFrame(x.value, dtype = int)
xd.to_excel('../data/homework2_res1.xlsx')
```

附录 2

```
import numpy as np
import cvxpy as cp
import pandas as pd

# 构造目标函数
# 1. 定义决策向量x(i, j)表示第j个配送中心向第i个部队用户运输
# 的物资量
x = cp.Variable((15, 8), pos = True)
y = cp.Variable((15, 8), integer=True)
```

```
# 2. 定义价值向量(单位物资的运费)
c = np.genfromtxt('../data/homework2_data.txt', dtype = \
    float, max_rows = 15, usecols = range(8))
# 3. 定义目标函数
obj = cp.Minimize(cp.sum(cp.multiply(c, x)))

# 定义约束条件
# 获取需求量
a = np.genfromtxt('../data/homework2_data.txt', dtype= \
    float, max_rows = 15, usecols= 8)
# 获取储备量
b = np.genfromtxt('../data/homework2_data.txt', dtype = \
    float, skip_header = 15) # 读最后一行数量
# 定义约束条件 储备量约束 + 需求量约束
con1 = [cp.sum(x, axis = 1, keepdims = True) \
        == a.reshape(15, 1),
       cp.sum(x, axis = 0, keepdims = True) \
        <= b.reshape(1, 8),
        x >= 1000 * y, x <= 2000 * y,
        y >= 0, y <= 1]

## 求解问题
prob = cp.Problem(obj, con1)
prob.solve(solver = 'GLPK_MI')
print('最优值', obj.value)
print('最优解', x.value)
print('最优解', y.value)

# 保存决策变量与0-1矩阵

xd = pd.DataFrame(x.value, dtype = int)
xd.to_excel('../data/homework2_res2.xlsx')
```

```
xd2 = pd.DataFrame(y.value, dtype= int)
xd2.to_excel('../data/homework2_res3.xlsx')
```