物流设施规划与设计
实验报告

学 院（系）：经济管理学院

专　　业：物流管理专业

姓　　名：严梓锴

学　　号：201903020

指 导 教 师：金淳

指 导 博 士：吴世萍

完 成 日 期：2022 年 4 月 13 日

大连理工大学

Dalian University of Technology

目录

图目录

表目录

Facility Location Problem

1. P-中值问题

1.1问题描述

某牛奶公司在某地区经过一段时间的宣传后，得到了 8 个超市的订单，为了方便运输，公司决定在该地区新建 2 个仓库，经过初步的考察，确定了 4 个候选仓库，公司拟用最低的成本来满足该新地区的需求。各个候选仓库到超市的距离和每个超市的需求量如下表所示，要求建立整数规划模型，并使用 Cplex 进行求

表格 1 超市需求量表

i	j				h_i
	1	2	3	4	
	d_{ij}				
1	4	12	20	6	100
2	2	10	25	10	50
3	3	4	16	14	120
4	6	5	9	2	80
5	18	12	7	3	200
6	14	2	4	9	70
7	20	30	2	11	60
8	24	12	6	22	100

1.2数学模型建立

$$\min \sum_{i=1}^{8}\sum_{j=1}^{4} h_i d_{ij} x_{ij}$$

$$s.t. \quad \sum_{j=1}^{4} x_{ij} = 1, \forall i = 1,...,8$$

$$\sum_{j=1}^{4} y_j = 2$$

$$x_{ij} \leq y_j \quad \forall i = 1,2,...,8;\ j = 1,2,...,4$$

$$x_{ij} \in \{0,1\} \quad \forall i = 1,2,...,8;\ j = 1,2,...,4$$

$$y_j \in \{0,1\} \quad \forall j = 1,2,...,4$$

图 1 问题 1 线性规划模型

其中，h_i 表示第 i 个超市的需求量，d_{ij} 为第 i 个超市到第 j 个候选仓库的最短距离。如果超市 i 被设仓库 j 所服务，则 x_{ij}=1，否则，x_{ij}=0。y_j 表示如果仓库 j 被选中作为设施点，y_j=1，否则，y_j=0。目标是最小化超市到仓库的加权距离和。

1.3求解结论

目标函数极值：3740

表格 2 问题 1 仓库选址结论表

	yj
1	1
2	0
3	1
4	0

表格 3 问题 1 超市-仓库供应关系表

	1	**2**	**3**	**4**
1	1	0	0	0
2	1	0	0	0
3	1	0	0	0
4	1	0	0	0
5	0	0	1	0
6	0	0	1	0
7	0	0	1	0
8	0	0	1	0

求解结果为，选择 1、3 两个候选地址作为仓库，其中 1、2、3、4 四个超市由仓库 1 供应，5、6、7、8 四个超市由仓库 3 供应，最小化超市到仓库的加权距离和（目标函数值）为 3740。

2. P-中心问题

2.1问题描述

例题 1 中的牛奶公司，由于公司的战略发展需求，建设仓库的目标改为服务于每个超市的最大成本最小化，求应该在哪两个位置建设仓库？要求建立整数规划模型，并用 cplex 进行求解。

2.2数学模型建立

$$\min\ D$$

$$s.t.\quad \sum_{j=1}^{4} x_{ij} = 1, \forall i = 1,...,8$$

$$\sum_{j=1}^{4} y_j = 2$$

$$x_{ij} \leqslant y_j \quad \forall i = 1,2,...,8;\ j = 1,2,...,4$$

$$\sum_{j=1}^{4} d_{ij} x_{ij} \leqslant D \quad \forall i = 1,2,...,8$$

$$x_{ij} \in \{0,1\} \quad \forall i = 1,2,...,8;\ j = 1,2,...,4$$

$$y_j \in \{0,1\} \quad \forall j = 1,2,...,4$$

图 2 问题 2 线性规划模型

其中，h_i 表示第 i 个超市的需求量，d_{ij} 为第 i 个超市到第 j 个候选仓库的最短距离。如果超市 i 被设仓库 j 所服务，则 x_{ij}=1，否则，x_{ij}=0。y_j 表示如果仓库 j 被选中作为设施点，y_j=1，否则，y_j=0。D 为超市与最近仓库的运输最大成本。目标是极小化超市与最近仓库的运输最大成本。

2.3求解结论

目标函数极值：1200

表格 4 问题 2 仓库选址结论表

	yj
1	0
2	1
3	0
4	1

表格 5 问题 2 超市-仓库供应关系表

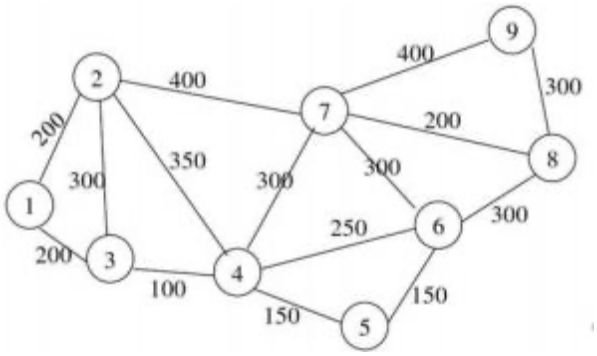
	1	**2**	**3**	**4**
1	0	0	0	1
2	0	1	0	0
3	0	1	0	0
4	0	1	0	0
5	0	0	0	1
6	0	1	0	0
7	0	0	0	1
8	0	1	0	0

求解结果为，选择 2、4 两个候选地址作为仓库，其中 2、3、4、6、8 五个超市由仓库 2 供应，1、5、7 三个超市由仓库 4 供应，最小化超市与最近仓库的运输最大成本（目标函数值）为 1200（即仓库 2 到超市 8 的运输成本）。

3. 集合覆盖问题

3.1问题描述

某物流企业要建设一些配送中心，为 9 个城市的客户服务，配送中心最远的服务距离为 300km，即任何一个城市的周边 300km 处至少有一个配送中心。不考虑配送中心的服务能力，物流企业要确定需要多少个配送中心以及配送中心的位置。其中第 6 个城市由于缺乏建立配送中心的必要条件，不可以建立配送中心。下图展示了各个城市之间的位置和距离，9 个城市的建设成本分别是：5，6，7，3，4，10，7，3，6（千万元），要求建立整数规划模型，并用 cplex 进行求解。



3.2数学模型建立

$$\min \sum_{j=1}^{9} f_j x_j$$

$$s.t. \quad \sum_{j=1}^{9} a_{ij} x_j \geq 1, \forall i = 1,2,...,9$$

$$x_6 = 0$$

$$x_j \in \{0,1\} \quad \forall j = 1,2,...,9$$

图 3 问题 3 线性规划模型

f_j 表示在候选设施点 j 建立设施的固定成本，表示如果设施点 j 可以覆盖需求点 i 则 a_{ij}=1，否则 a_{ij}=0。x_j 表示如果候选设施点被选中，则 x_j=1，否则 x_j=0。目标是最小化总的建设成本。

各城市之间的最短路径使用 Floyd 算法进行计算，相关 Python 代码见附件。

3.3求解结论

目标函数极值：10

表格 6 各城市间最短路表

	1	2	3	4	5	6	7	8	9
1	0	200	200	300	450	550	600	800	1000
2	200	0	300	350	500	600	400	600	800
3	200	300	0	100	250	350	400	600	800
4	300	350	100	0	150	250	300	500	700
5	450	500	250	150	0	150	450	450	750
6	550	600	350	250	150	0	300	300	600
7	600	400	400	300	450	300	0	200	400
8	800	600	600	500	450	300	200	0	300
9	1000	800	800	700	750	600	400	300	0

表格 7 问题 4 配送中心选址结果表

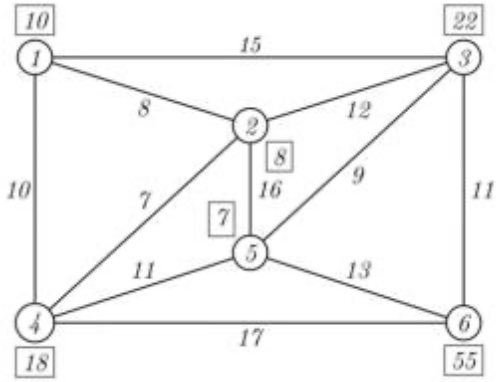
	x_j
1	0
2	0
3	1
4	0
5	0
6	0
7	0
8	1
9	0

计算结果表明，在 3、8 两个城市建配送中心最经济，且能覆盖到所有的城市，总的建设成本为 10（千万元）。

4. 范围覆盖问题

4.1问题描述

某物流企业要建设两个配送中心，为 6 个区域的客户服务，配送中心最远的服务距离为 11km。物流企业要确定这两个配送中心的位置来尽可能多的满足客户的需求。下图展示了各个区域之间的距离和各个区域的需求量（方框中数字），要求建立整数规划模型，并用 cplex 进行求解。



4.2数学模型建立

$$\max \sum_{i=1}^{6} h_i z_i$$

$$s.t. \quad \sum_{j=1}^{6} a_{ij} x_j \geq z_i, \forall i = 1, ..., 6$$

$$\sum_{j=1}^{6} x_j = 2$$

$$z_i \in \{0, 1\} \quad \forall i = 1, 2, ..., 6$$

$$x_j \in \{0, 1\} \quad \forall j = 1, 2, ..., 6$$

图 4 问题 4 线性规划模型

其中 h_i 为需求点 i 的需求量，a_{ij} 表示如果设施点 j 可以覆盖需求点 i 则 a_{ij}=1，否则 a_{ij}=0。z_i 代表当需求点 i 被覆盖时，z_i=1，否则 z_i=0。x_j 表示如果候选设施点被选中，则 x_j=1，否则 x_j=0。目标是最大化被覆盖的需求量。

4.3求解结论

目标函数极值：120

表格 8 各区域间最短路表

	1	**2**	**3**	**4**	**5**	**6**
1	0	8	15	10	21	26
2	8	0	12	7	16	23
3	15	12	0	19	9	11
4	10	7	19	0	11	17
5	21	16	9	11	0	13
6	26	23	11	17	13	0

表格 9 问题 4 选址结果表

	x_j
1	0
2	0
3	0
4	1
5	0
6	1

表格 10 问题 4 需求点被覆盖结果表

	z_i
1	1
2	1
3	1
4	1
5	1
6	1

计算结果表明，在 4、6 两个城市建配送中心能覆盖到最多的需求量（此时所有区域均被覆盖），覆盖到的需求量为 120。

Facility Layout Problem

5. 设施布局问题

5.1问题提出

医院急救中心的重新布局（课件上的题，要求根据例 1 自行完成例 2）

某医院急救中心的管理者为改进诊疗效率，计划对医院的布局进行调整。目前该医院 8 间房屋分布的示意图如下：

出入口/紧急处理室	检查室 1	检查室 2	X 光透视室
化验室/心电图室	手术室	恢复/观察室	石膏室

管理的目标是重新对病房进行布局，使前来就诊的病人在医院内往返的距离最短，即：

病人走动的最短距离=$\sum_{i=1}^{8}\sum_{j=1}^{8}N_{ij}D_{ij}$

N_{ij}表示每月中，往返于i部门和j部门之间的病人人数，或病人走动的次数；

D_{ij}表示i部门和j部门之间的距离（m），相当于在两个部门之间搬动物品的单位成本；i和j代表各个部门。

假设各个部门之间的距离都是 10m，对角相邻部门间的距离也是 10m，间隔一个部门的两个部门间距离是 20m，而医院入口与 X 光透视室间的距离为 30m。

分析一个月中病人在各个部门之间的往返次数，具体数据见下图：

表格 11 一个月中病人在各个部门之间的往返次数表

	1	2	3	4	5	6	7	8
1		100	100	0	0	0	0	0
2			0	50	20	0	0	0
3				30	20	0	0	0
4					20	0	0	20
5						20	0	10
6							30	0
7								0
8								

要求：要求列出 QAP 问题的模型，并用 Cplex 求解怎么对医院急救中心重新布置是最好的？（两个表格放到一个新建的数据文件中，不要直接写到编辑区）

5.2数学模型建立

$$\min \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} f_{ik} d_{jl} x_{ij} x_{kl}$$

$$s.t. \quad \sum_{i=1}^{N} x_{ij} = 1, \forall j = 1,...,8$$

$$\sum_{j=1}^{N} x_{ij} = 1, \forall i = 1,...,8$$

$$x_{ij} \in \{0,1\} \quad \forall i = 1,2,...,8;\ j = 1,2,...,8$$

图 5 问题 5 线性规划模型

f_{ik} 表示部门 i 与部门 k 在一个月中病人往返的次数，d_{jl} 表示位置 j 与位置 l 的距离，x_{ij} 表示如果将设施 i 放到位置 j，则 x_{ij}=1，否则，x_{ij}=0。目标函数代表所有部门对之间的加权后的人流量。

根据布局图列出部门间距离表，如表格 12 所示。其中 1—出入口/紧急处理室、2—化验室/心电图室、3—检查室 1、4—手术室、5—检查室 2、6—恢复/观察室、7—X 光透视室、8—石膏室。

表格 12 部门间距离表（单位：米）

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	0	10	10	10	20	20	30	30
2	10	0	10	10	20	20	30	30
3	10	10	0	10	10	10	20	20
4	10	10	10	0	10	10	20	20
5	20	20	10	10	0	10	10	10
6	20	30	10	10	10	0	10	10
7	30	30	20	20	10	10	0	10
8	30	30	30	30	10	10	10	0

5.3求解结论

目标函数极值：8800

表格 13 问题 5 部门布局方案表

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	1	0
3	0	0	0	0	1	0	0	0
4	0	0	0	0	0	1	0	0
5	0	0	0	1	0	0	0	0
6	1	0	0	0	0	0	0	0
7	0	1	0	0	0	0	0	0
8	0	0	1	0	0	0	0	0

如表格 13 所示，第 i 行第 j 列表示部门 i 布局在位置 j。在此布局模式下，一个月内病人的加权人流量最小，为 8800。

附录

问题 1 求解代码 language = Python

```
from docplex.mp.model import Model
import pandas as pd
m = Model(name='experimentmodel1.3.1')
n_shop = 8
n_warehouse = 4
set_shop = range(1, n_shop + 1)
set_warehouse = range(1, n_warehouse + 1)
distance = pd.read_csv("distance.csv")
distance.index = [i for i in set_shop]
distance.columns = [j for j in set_warehouse]
demand = pd.Series(pd.read_csv("demand.csv")['demand'].values, index
    = [i for i in set_shop])
x_vars = pd.DataFrame({j : pd.Series(m.binary_var(name="x_{0}_{1}".f
   ormat(i,j))
                                        for i in set_shop)
                          for j in set_warehouse})
x_vars.index = [i for i in set_shop]
y_vars = pd.Series([m.binary_var(name="y_{0}".format(j)) for j in se
   t_warehouse])
y_vars.index = [j for j in set_warehouse]
unique_supply_constrains = {i : m.add_constraint(
    sum(x_vars.loc[i,j] for j in set_warehouse) == 1,
    ctname="unique_supply_constrain_{0}".format(i))
                      for i in set_shop}
selection_constraint = m.add_constraint(
    sum(y_vars.loc[j] for j in set_warehouse) == 2,
    ctname="selection_constraint")
supply_constrains = {(i,j) : m.add_constraint(
    x_vars.loc[i,j] <= y_vars.loc[j],
    ctname="supply_constrain_{0}_{1}".format(i,j))
                      for i in set_shop
                      for j in set_warehouse}
objective = sum(demand.loc[i] * distance.loc[i,j] * x_vars.loc[i,j]
                  for j in set_warehouse
                  for i in set_shop)
m.minimize(objective)
m.print_information()
s = m.solve(log_output=True)
m.print_solution()
for i in set_shop:
```

```
    x_vars.loc[i] = x_vars.loc[i].apply(lambda item: item.solution_value)
y_vars = y_vars.apply(lambda item: item.solution_value)
x_vars.to_csv("solution_x.csv")
y_vars.to_csv("solution_y.csv")
```

问题 2 求解代码 language = Python

```
from docplex.mp.model import Model
import pandas as pd
m = Model(name='experimentmodel1.3.2')
n_shop = 8
n_warehouse = 4
set_shop = range(1, n_shop + 1)
set_warehouse = range(1, n_warehouse + 1)
distance = pd.read_csv("distance.csv")
distance.index = [i for i in set_shop]
distance.columns = [j for j in set_warehouse]
demand = pd.Series(pd.read_csv("demand.csv")['demand'].values, index = [i for i in set_shop])
x_vars = pd.DataFrame({j : pd.Series(m.binary_var(name="x_{0}_{1}".format(i,j))
                                     for i in set_shop)
                       for j in set_warehouse})
x_vars.index = [i for i in set_shop]
y_vars = pd.Series([m.binary_var(name="y_{0}".format(j)) for j in set_warehouse])
y_vars.index = [j for j in set_warehouse]
maxCost = m.continuous_var(name="maxCost")
unique_supply_constrains = {i : m.add_constraint(
    sum(x_vars.loc[i,j] for j in set_warehouse) == 1,
    ctname="unique_supply_constrain_{0}".format(i))
                    for i in set_shop}
selection_constraint = m.add_constraint(
    sum(y_vars.loc[j] for j in set_warehouse) == 2,
    ctname="selection_constraint")
supply_constrains = {(i,j) : m.add_constraint(
    x_vars.loc[i,j] <= y_vars.loc[j],
    ctname="supply_constrain_{0}_{1}".format(i,j))
                    for i in set_shop
                    for j in set_warehouse}
max_cost_constrains = {i : m.add_constraint(
    sum(demand.loc[i] * distance.loc[i,j] * x_vars.loc[i,j] for j in set_warehouse) <= maxCost,
```

```
    ctname="max_cost_constrain_{0}".format(i))
                        for i in set_shop}
objective = maxCost
m.minimize(objective)
m.print_information()
s = m.solve(log_output=True)
m.print_solution()
for i in set_shop:
    x_vars.loc[i] = x_vars.loc[i].apply(lambda item: item.solution_value)
y_vars = y_vars.apply(lambda item: item.solution_value)
x_vars.to_csv("solution_x.csv")
y_vars.to_csv("solution_y.csv")
```

问题 3 求解代码 language = Python

```
from docplex.mp.model import Model
import pandas as pd
m = Model(name='experimentmodel1.3.3')
n_city = 9
set_city = range(1, n_city + 1)
distance = pd.read_csv("min_distance.csv", usecols=[i for i in set_city])
distance.columns = [i for i in set_city]
distance.index = [i for i in set_city]
cost = pd.Series(pd.read_csv("cost.csv")['cost'].values, index = [i for i in set_city])
coverable_distance = 300
cover_matrix = pd.DataFrame(columns=[i for i in set_city], index = [i for i in set_city])
for i in set_city:
    for j in set_city:
        if distance.loc[i,j] <= coverable_distance:
            cover_matrix.loc[i,j] = True
        else:
            cover_matrix.loc[i,j] = False
cover_matrix
x_vars = pd.Series([m.binary_var(name="x_{0}".format(j)) for j in set_city])
x_vars.index = [j for j in set_city]
distance_constrains = {i : m.add_constraint(
    sum(cover_matrix.loc[i,j] * x_vars.loc[j] for j in set_city) >= 1,
    ctname="distance_constrain_{0}".format(i))
```

```
24.              for i in set_city}
25.capability_constrain = m.add_constraint(x_vars.loc[6] == 0)
26.objective = sum(cost.loc[j] * x_vars.loc[j] for j in set_city)
27.m.minimize(objective)
28.m.print_information()
29.s = m.solve(log_output=True)
30.m.print_solution()
31.x_vars = x_vars.apply(lambda item: item.solution_value)
32.x_vars.to_csv("solution_x.csv")
```

问题 4 求解代码 language = Python

```
1. from docplex.mp.model import Model
2. import pandas as pd
3. m = Model(name='experimentmodel1.3.4')
4. n_city = 6
5. set_city = range(1, n_city + 1)
6. distance = pd.read_csv("min_distance.csv", usecols=[i for i in set_c
   ity])
7. distance.columns = [i for i in set_city]
8. distance.index = [i for i in set_city]
9. demand = pd.Series(pd.read_csv("demand.csv")['demand'].values, index
    = [i for i in set_city])
10.coverable_distance = 11
11.cover_matrix = pd.DataFrame(columns=[i for i in set_city], index = [
   i for i in set_city])
12.for i in set_city:
13.    for j in set_city:
14.        if distance.loc[i,j] <= coverable_distance:
15.            cover_matrix.loc[i,j] = 1
16.        else:
17.            cover_matrix.loc[i,j] = 0
18.cover_matrix
19.x_vars = pd.Series([m.binary_var(name="x_{0}".format(j)) for j in se
   t_city])
20.x_vars.index = [j for j in set_city]
21.z_vars = pd.Series([m.binary_var(name="z_{0}".format(i)) for i in se
   t_city])
22.z_vars.index = [i for i in set_city]
23.cover_constrains = {i : m.add_constraint(
24.    sum(cover_matrix.loc[i,j] * x_vars.loc[j] for j in set_city) >=
   z_vars.loc[i],
25.    ctname="cover_constrain_{0}".format(i))
26.              for i in set_city}
```

```
capability_constrain = m.add_constraint(sum(x_vars.loc[j] for j in set_city) == 2)
objective = sum(demand.loc[i] * z_vars.loc[i] for i in set_city)
m.maximize(objective)
m.print_information()
s = m.solve(log_output=True)
m.print_solution()
x_vars = x_vars.apply(lambda item: item.solution_value)
z_vars = z_vars.apply(lambda item: item.solution_value)
x_vars.to_csv("solution_x.csv")
z_vars.to_csv("solution_z.csv")
```

问题 5 求解代码 language = Python

```
from gurobipy import *
import pandas as pd
m = Model(name='LogisticsExperiment-2.4')
n = 8
set_N = range(1, n + 1)
distance = pd.read_csv("distance.csv",usecols=[i for i in set_N], dtype=int)
distance.index = [i for i in set_N]
distance.columns = [j for j in set_N]
flow = pd.read_csv("flow.csv",usecols=[i for i in set_N], dtype=int)
flow.index = [i for i in set_N]
flow.columns = [j for j in set_N]
x_vars = pd.DataFrame({j : pd.Series(m.addVar(vtype=GRB.BINARY,name="x_{0}_{1}".format(i,j))
                                     for i in set_N)
                       for j in set_N})
x_vars.index = [i for i in set_N]
columns_constrains = {j :
                      m.addConstr(
                          sum(x_vars.loc[i,j] for i in set_N) == 1,
                          name="row_constrain_{0}".format(j))
                      for j in set_N}
rows_constraints = {i :
                    m.addConstr(
                        sum(x_vars.loc[i,j] for j in set_N) == 1,
                        name="rows_constraint_{0}".format(i))
                    for i in set_N}
objective = m.setObjective(sum(flow.loc[i, k] * distance.loc[j, l] *
    x_vars.loc[i, j] * x_vars.loc[k, l]
                               for l in set_N
```

```
                                    for k in set_N
                                    for j in set_N
                                    for i in set_N),
                                GRB.MINIMIZE)
s = m.optimize()
for i in set_N:
    x_vars.loc[i] = x_vars.loc[i].apply(lambda item: int(item.x))
x_vars.to_csv("solution_x.csv")
```

问题 3 及问题 4 的最短路径求解代码（以问题 3 为例）

```
import pandas as pd
graph = {'1': [(200, '1', '2'), (200, '1', '3')],
         '2': [(200, '2', '1'), (300, '2', '3'), (350, '2', '4'), (400, '2', '7')],
         '3': [(200, '3', '1'), (300, '3', '2'), (100, '3', '4')],
         '4': [(350, '4', '2'), (100, '4', '3'), (150, '4', '5'), (250, '4', '6'), (300, '4', '7')],
         '5': [(150, '5', '4'), (150, '5', '6')],
         '6': [(250, '6', '4'), (150, '6', '5'), (300, '6', '7'), (300, '6', '8')],
         '7': [(400, '7', '2'), (300, '7', '4'), (300, '7', '6'), (200, '7', '8'), (400, '7', '9')],
         '8': [(300, '8', '6'), (200, '8', '7'), (300, '8', '9')],
         '9': [(400, '9', '7'), (300, '9', '8')]
        }
def graph2adjacent_matrix(graph):
    vnum = len(graph)
    dict = {'1':0,'2':1,'3':2,'4':3,'5':4,'6':5,'7':6,'8':7,'9':8}
    adjacent_matrix = [[0 if row==col else float('inf') for col in range(vnum)] for row in range(vnum)]
    vertices = graph.keys()
    for vertex in vertices:
        for edge in graph[vertex]:
            w,u,v = edge
            adjacent_matrix[dict.get(u)][dict.get(v)]=w
    return adjacent_matrix
def floyd(adjacent_matrix):
    vnum = len(adjacent_matrix)
    a = [[adjacent_matrix[row][col] for col in range(vnum)] for row in range(vnum)]
    nvertex = [[-1 if adjacent_matrix[row][col]==float('inf') else col for col in range(vnum)] for row in range(vnum)]
```

```
        for k in range(vnum):
        for i in range(vnum):
            for j in range(vnum):
                if a[i][j]>a[i][k]+a[k][j]:
                    a[i][j]=a[i][k]+a[k][j]
                    nvertex[i][j] = nvertex[i][k]
    return nvertex, a
adjacent_matrix = graph2adjacent_matrix(graph)
nvertex, a = floyd(adjacent_matrix)
for i in range(len(adjacent_matrix)):
    for j in range(len(adjacent_matrix[0])):
        print(adjacent_matrix[i][j],end="\t")
    print()print()
for i in range(len(nvertex)):
    for j in range(len(nvertex[0])):
        print(nvertex[i][j],end="\t")
    print()print()
for i in range(len(a)):
    for j in range(len(a[0])):
        print(a[i][j],end="\t")
    print()min_distance = pd.DataFrame(a, columns=[i for i in range(
1, 10)], index=[i for i in range(1, 10)])
min_distance.to_csv("min_distance.csv")
```

对于问题 4，只需要修改上述代码中的第 2-11 行及 14 行为如下即可：

```
graph = {'1': [(8, '1', '2'), (15, '1', '3'), (10, '1', '4')],
         '2': [(8, '2', '1'), (12, '2', '3'), (7, '2', '4'), (16, '2
', '5')],
         '3': [(15, '3', '1'), (12, '3', '2'), (9, '3', '5'), (11, '
3', '6')],
         '4': [(10, '4', '1'), (7, '4', '2'), (11, '4', '5'), (17, '
4', '6')],
         '5': [(16, '5', '2'), (9, '5', '3'), (11, '5', '4'), (13, '
5', '6')],
         '6': [(11, '6', '3'), (17, '6', '4'), (13, '6', '5')]
        }


dict = {'1':0,'2':1,'3':2,'4':3,'5':4,'6':5}
```