

赛区评阅编号（由赛区组委会填写）：

2019 年高教社杯全国大学生数学建模竞赛

承 诺 书

我们仔细阅读了《全国大学生数学建模竞赛章程》和《全国大学生数学建模竞赛参赛规则》（2019 年修订稿，以下简称为“竞赛章程和参赛规则”，可从全国大学生数学建模竞赛网站下载）。

我们完全清楚，在竞赛开始后参赛队员不能以任何方式，包括电话、电子邮件、“贴吧”、QQ 群、微信群等，与队外的任何人（包括指导教师）交流、讨论与赛题有关的问题；无论主动参与讨论还是被动接收讨论信息都是严重违反竞赛纪律的行为。

我们完全清楚，抄袭别人的成果是违反竞赛章程和参赛规则的行为；如果引用别人的成果或资料（包括网上资料），必须按照规定的参考文献的表述方式列出，并在正文引用处予以标注。

我们以中国大学生名誉和诚信郑重承诺，严格遵守竞赛章程和参赛规则，以保证竞赛的公正、公平性。如有违反竞赛章程和参赛规则的行为，我们将受到严肃处理。

我们授权全国大学生数学建模竞赛组委会，可将我们的论文以任何形式进行公开展示（包括进行网上公示，在书籍、期刊和其他媒体进行正式或非正式发表等）。

我们参赛选择的题号（从 A/B/C/D/E 中选择一项填写）： A

我们的报名参赛队号（12 位数字全国统一编号）： 12

参赛学校（完整的学校全称，不含院系名）： 西安财经大学

参赛队员（打印并签名）： 1. 赵佳敏

2. 金珂

3. 张贵玉

指导教师或指导教师组负责人（打印并签名）：

（指导教师签名意味着对参赛队的行为和论文的真实性负责）

日期： 年 月 日

（请勿改动此页内容和格式。此承诺书打印签名后作为纸质论文的封面，注意电子版论文中不得出现此页。以上内容请仔细核对，如填写错误，论文可能被取消评奖资格。）

赛区评阅编号（由赛区组委会填写）：

2019 年高教社杯全国大学生数学建模竞赛

编 号 专 用 页

赛区评阅记录（可供赛区评阅时使用）：

评 阅 人						
备 注						

送全国评阅统一编号（赛区组委会填写）：

全国评阅随机编号（全国组委会填写）：

（请勿改动此页内容和格式。此编号专用页仅供赛区和全国评阅使用，参赛队打印后装订到纸质论文的第二页上。注意电子版论文中不得出现此页。）

供应链网络的建立与道路破坏问题

摘要

本次我们主要讨论的是供应链网络的建立与道路破坏问题，总共有三个问题，主要运用了 Floyd 算法，借助 Matlab 软件，然后通过分析各个城市的数据，建立一个 0-1 规划模型以及独立事件概率模型，最后得出了怎样建立供应点的最优方案以及在道路破坏程度不同的情况下，用 Floyd 算法重新求出两个城市之间的最短距离，算出增加的总费用，得出最优的破坏方案来使得总费用增加最多；然后破坏的概率确定的情况下，运用数学期望的知识，算出他们的平均总费用，得出最优的破坏方案。

对于问题一：问题一主要是解决怎样在 49 个供应点中找出适应数量的供应点从而使站点的固定费用加上运输费用，即总费用最少（其中有 21 个供应点因为固定费用特别高，所以不做考虑）。可以把他看成 0-1 规划问题，然后运用 Floyd 算法求出每个城市之间的最短距离，根据题目列出相应的目标函数以及约束条件，利用 Matlab 进行求解，就可以得到确定的供应点，最后根据表 1 中的城市坐标作出每一个供应点到需求点的连接图。

对于问题二：从表 4 中可以看出，若是破坏第 8 条道路，49 号城市将无法得到供应，所以不考虑这条破坏情况。所以只有八条道路可破坏，接下来就是考虑破坏道路的条数，如果只破坏一条道路，那就分别破坏这 8 条道路，根据 Floyd 算法求出最短距离，然后分别求出它们的总费用增加数量以及各自增加的总费用的比，得出总费用增加最多的那条路；如果破坏两条道路，分别破坏除了上次求出的总费用增加最多的那条路剩下的 7 条道路，根据 Floyd 算法求出最短距离，分别求出它们的总费用增加数量以及各自增加的总费用的比率，然后求出总费用增加最多的那条道路，一直求到同时破坏 8 条路增加的总费用。最后根据顺序选出最优的破坏方案。

对于问题三：由于破坏方破坏时，这些道路不一定会被破坏，而破坏情况服从一定的概率分布。因为道路被破坏是独立事件，相互之间不受影响。所以就可以建立一个独立事件概率模型，由于破坏方破坏时，这些道路不一定会被破坏，而破坏情况服从一定的概率分布。所以就可以利用数学期望来计算增加的费用值的平均值，即计算出被破坏的道路的平均总费用。

关键字：0-1 模型，最短距离，Floyd 算法

一、问题重述

1.1 背景知识

全球化竞争的加剧促使越来越多的企业开始采用供应链管理策略，以实现企业的一体化管理。供应链是一个复杂的网状结构系统，每一部分都面临着各种潜在的风险，任何一部分出现问题都可能给整个供应链带来严重的影响，因此如何分析、评价和提高供应链系统的可靠性变得日益迫切。

设施系统是供应链的核心，在供应链研究中有着极其重要的地位。在一个设施系统中，某些设施由于自然灾害或者其他因素的影响可能失效，例如 911 恐怖袭击事件、2004 年的印度洋海啸、2008 年的汶川地震等都对诸多行业的设施系统造成了严重的破坏。现有某物流公司要在全中国各城市之间建立供应链网络。需要选定部分城市作为供应点，将货物运输到各城市。通常每个供应点的货物是充足的，可以充分满足相应城市的需求。设该公司考虑 49 个城市的网络，城市的坐标见表 1。城市之间的道路连接关系见表 2。在每个城市建立配送中心的固定费用和需求量的见表 3，并假定作为供应点的城市其供应量可以满足有需求的城市的需要。现要建立一个供应网络，为各城市提供货物供应。货物运输利用汽车进行公路运输。设每吨每公里运输费用为 0.5 元。

1.2 问题提出

1. 现在要从 49 个城市中选取部分城市做为供给点供应本城市及其它城市。建立供给点会花费固定费用，从供应点运输到需求点会产生运输费用，要使总费用最小，问建立多少个供应点最好。给出选中作为供应点的城市，并给出每个供应点供应的城市。同时根据坐标作出每一个供应点到需求点的连接图。

2. 假定有某组织对该供应网络的道路进行破坏。并非所有的道路都可以被破坏，当某条道路被破坏后，该条道路就不能再被使用，以前运输经过该道路的只有改道，但总是沿最短路运输。如果破坏方选取的策略是使对方总费用增加 25%，而每破坏一条道路都需要成本和代价，因此需要破坏最少的道路。问破坏方选取哪几条线路进行破坏。给出具体的破坏道路和总费用。

3. 假定各道路能否被破坏具有随机性，当某条道路被破坏后，该条道路就不能再被使用，以前运输经过该道路的只有改道，但总是沿最短路运输。由于破坏方选取一些

边进行破坏时，这些边不一定被破坏，而是服从一定的概率分布。运输时产生的费用可按照各种情况下的平均费用来考虑。如果破坏方选取的策略是使对方平均总费用增加最大。给出具体的破坏道路和平均总费用。

1.3 相关数据

1. 各座城市的坐标（表 1）
2. 各座城市之间的公路段及里程表（表 2）
3. 各城市作为配送中心的固定费用（表 3）
4. 可破坏的道路及其概率（表 4）

二、问题分析

2.1 对总体问题的分析

主要运用了 Floyd 算法，借助 Matlab 软件，然后通过分析各个城市的数据，建立一个 0-1 规划模型以及独立事件概率模型，最后得出了怎样建立供应点的最优方案以及在道路破坏程度不同的情况下，用 Floyd 算法重新求出两个城市之间的最短距离，算出增加的总费用，得出最优的破坏方案来使得总费用增加最多；然后破坏的概率确定的的情况下，运用数学期望的知识，算出他们的平均总费用，得出最优的破坏方案。

2.2 具体问题分析

1. 对问题 1 的分析：问题一主要是解决怎样在 49 个供应点中找出适应数量的供应点从而使站点的固定费用加上运输费用，即总费用最少（其中有 21 个供应点因为固定费用特别高，所以不做考虑）。可以把他看成 0-1 规划问题，然后运用 Floyd 算法求出每个城市之间的最短距离，根据题目列出相应的目标函数以及约束条件，利用 Matlab 进行求解，就可以得到确定的供应点，最后根据表 1 中的城市坐标作出每一个供应点到需求点的连接图。

2. 对问题 2 的分析：从表 4 中可以看出，若是破坏第 8 条道路，49 号城市将无法得到供应，所以不考虑这条破坏情况。所以只有八条道路可破坏，接下来就是考虑破坏道路的条数，如果只破坏一条道路，那就分别破坏这 8 条道路，根据 Floyd 算法求出最短距离，然后分别求出它们的总费用增加数量以及各自增加的总费用的比，得出总费用增

加最多的那条路；如果破坏两条道路，分别破坏除了上次求出的总费用增加最多的那条路剩下的 7 条道路，根据 Floyd 算法求出最短距离，分别求出它们的总费用增加数量以及各自增加的总费用的比率，然后求出总费用增加最多的那条道路……一直求到同时破坏 8 条路增加的总费用。最后根据顺序选出最优的破坏方案。

3. 对问题 3 的分析：由于破坏方破坏时，这些道路不一定会被破坏，而破坏情况服从一定的概率分布。因为道路被破坏是独立事件，相互之间不受影响。所以就可以建立一个独立事件概率模型，由题二可知，破坏 6 号道路对总费用没有影响，所以先不考虑，只用考虑剩下的 7 条道路。由于破坏方破坏时，这些道路不一定会被破坏，而破坏情况服从一定的概率分布。所以就可以利用数学期望来计算增加的费用值的平均值，最后计算出被破坏的道路的平均总费用。

三、模型假设

1. 假设所有的道路之间的运输方式只有公路运输，没有其他的运输方式。
2. 假设每个城市只能接受一个供应点的货物。
3. 假设道路 8 不能被破坏。
4. 假设问题二中的道路被破坏每次都破坏完全了。
5. 假设供应点的货物可以满足所有城市的需求。
6. 假设这条道路被破坏时，可以用一个很大的值来代替，比如 100000。

四、符号说明及名词解释

4.1 符号说明

符号	意义
$x(i)$	判断是否在第 i 个城市建立供应点
$D(I, j)$	表示两个城市之间的最短距离
$x(j)$	表示第 j 个城市的货物需求量
$h(i)$	表示第 i 个城市作为供应点的固定费用
$s(i, j)$	判断第 j 个城市能不能接受第 i 个城市的货物
$p(i)$	破坏第 i 条道路的概率
$r(i)$	破坏第 i 条道路所增加的费用

p	总费用增加率
N(i)	破坏 i 条道路后增加的总费用
M	破坏前的总费用

4.2 名词解释

Floyd 算法：Floyd 算法又称为插点法，是一种利用动态规划的思想寻找给定的加权图中多源点之间最短路径的算法，通过一个图的权值矩阵求出它的每两点间的最短路径矩阵。

数学期望：（或均值，亦简称期望）是试验中每次可能结果的概率乘以其结果的总和，是最基本的数学特征之一。它反映随机变量平均取值的大小。

五、模型建立与求解

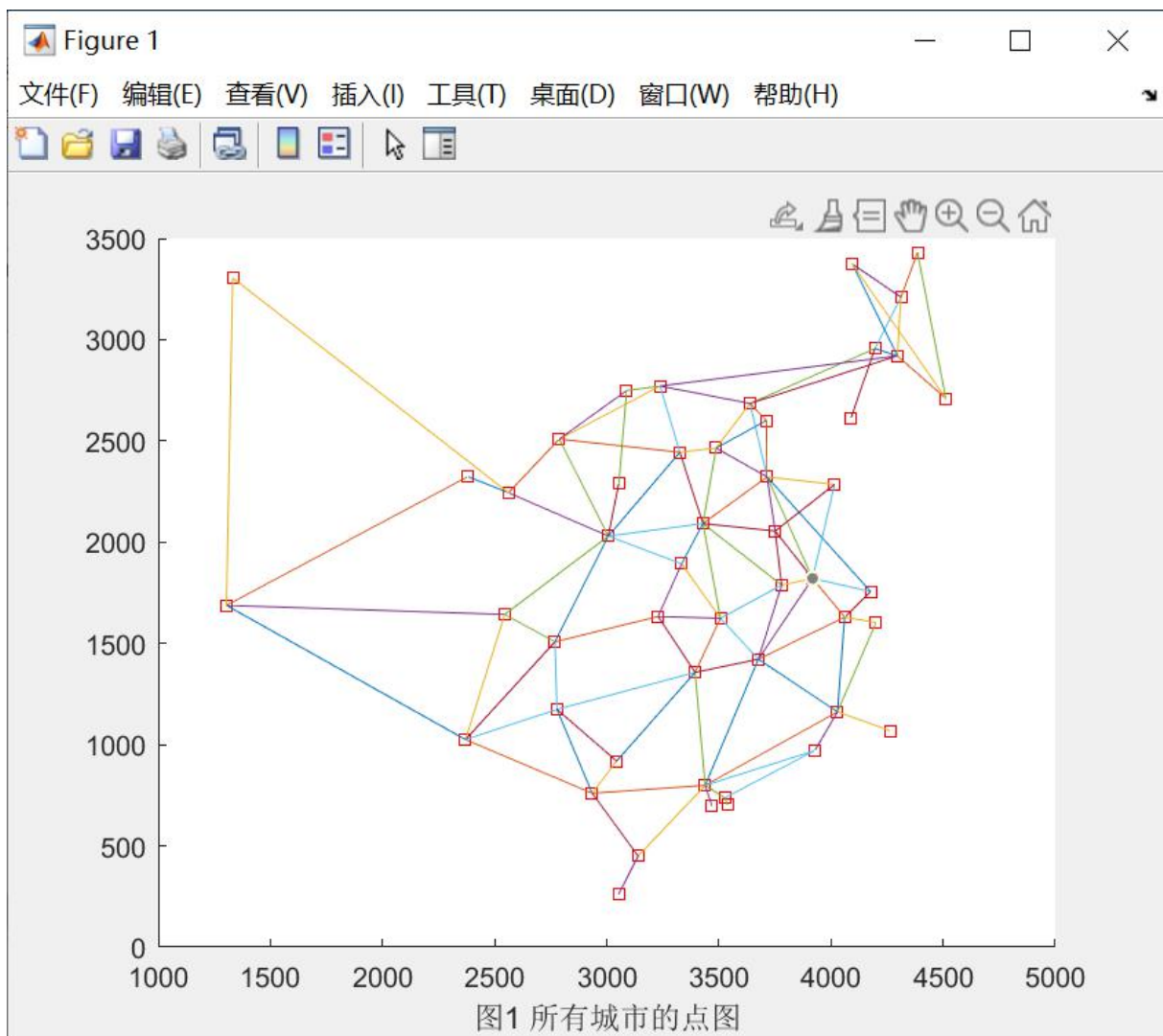
5.1 问题一模型的建立与求解

5.1.1 模型的分析

问题一主要研究的是怎样可以使总费用最少，而总费用由固定费和运输费用组成，若是增加供应点，会使得固定费用增加，运输费用减少；若是减少供应的，会使固定费用减少，运输费用增加。所以我们可以运用 0-1 规划问题，根据 Floyd 算法求出每个城市之间的最短距离（Floyd 算法：Floyd 算法又称为插点法，是一种利用动态规划的思想寻找给定的加权图中多源点之间最短路径的算法，通过一个图的权值矩阵求出它的每两点间的最短路径矩阵。），结合约束条件，找出对应的供应点的数量，再利用 Matlab 进行求解。

5.1.1 数据处理

根据表 1 中各个城市的坐标以及表 2 中各个城市的距离，利用 Matlab 绘制出每个城市之间的道路图。（代码 1）



然后根据 Floyd 算法求出每个城市的最短距离。

首先构造一个 49×49 的矩阵，根据表中的数据来对 $C[i][j]$ 进行赋值，若从 i 到 j 有路径，则将数据赋给 $C[i][j]$ ，若是没有路径就认为 $C[i][j]$ 的值为 ∞ ，然后通过弗洛伊德算法求出 i 到 j 的最短距离，弗洛伊德算法的思想如下：

第一步：令 $i=1, j=1, k=1$

第二步：令 $d=C[i][k]+C[k][j]$ ，若是 $d < C[i][j]$ ，则令 $d=C[i][j]$ ，否则进入下一步。

第三步：若 $j < 49$ ， $j=j+1$ ，然后进入第二步。

第四步：若 $i < 49$ ， $i=i+1$ ，然后进入第二步。

第五步：若 $k < 49$ ， $k=k+1$ ，然后进入第二步。

根据递归对矩阵进行 n 次更新，产生矩阵系列， $A_0, A_1, A_2, A_3, \dots, A_n$ 。

最后得到一个 49×49 的矩阵（下面是矩阵的一部分）就是两个城市之间的最短距离。

表 1：各个城市的最短距离矩阵（ 10×10 ）

城市编号	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
C1	0	120	270	480	540	799	1129	1359	1260	1094
C2	120	0	370	580	660	919	1249	1479	1200	1034
C3	270	370	0	210	740	1069	1399	1629	1151	985
C4	480	580	210	0	530	1279	1609	1839	1361	1195
C5	540	660	740	530	0	1339	1669	1899	1800	1634
C6	799	919	1069	1279	1339	0	330	560	2059	1893
C7	1129	1249	1399	1609	1669	330	0	230	2389	2223
C8	1359	1479	1629	1839	1899	560	230	0	2619	2453
C9	1260	1200	1151	1361	1800	2059	2389	2619	0	28
C10	1094	1034	985	1195	1634	1893	2223	2453	280	0

5.1.2 模型的建立

0-1 模型的建立

$$\text{约束条件: } x(i) = \begin{cases} 1, & \text{在第 } i \text{ 个城市建立供应点} \\ 0, & \text{不在第 } i \text{ 个城市建立供应点} \end{cases} \quad (5.1.1)$$

$$\text{一个城市只能接受一个供应点的货物,即 } \sum_{i=1}^{49} s(i,j) = 1, \quad j = 1, 2, 3, 4, \dots, 49 \quad (5.1.2)$$

若 $x(i) = 0$, 则 $s(i,j) = 0$; 若 $x(i) = 1$, $s(i,j) = 0$ 或 1 , 则 $x(i) \geq s(i,j)$

因为总费用 = 固定费用 + 运输费用

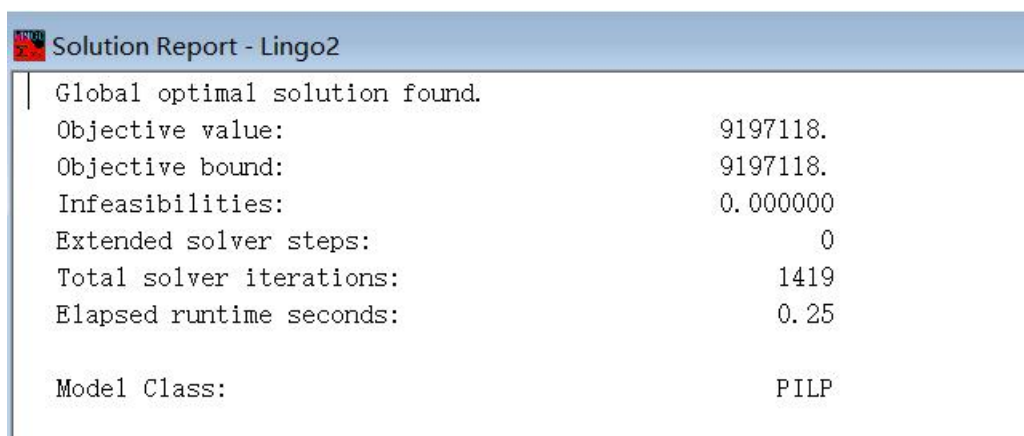
$$\text{于是我们可以建立 0-1 模型, } Z_{\min} = \sum_{i=1}^{49} h(i) * x(i) + 0.5 * \left[\sum_{i=1}^{49} \sum_{j=1}^{49} D(i,j) * x(j) * s(i,j) \right] \quad (5.1.3)$$

其中, $D(i,j)$ 表示两个城市之间的最短距离, $x(j)$ 表示第 j 个城市的货物需求量, $h(i)$ 表示第 i 个城市作为供应点的固定费用。

$$\text{约束条件有 } \begin{cases} x(i) = 0 \text{ 或 } 1, \quad i = 1, 2, 3, \dots, 49 \\ s(i,j) = 0 \text{ 或 } 1, \quad i, j = 1, 2, \dots, 49 \\ \sum_{i=1}^{49} s(i,j) = 1 \\ x(i) \geq s(i,j) \end{cases} \quad (5.1.4)$$

5.1.3 模型求解

用 Lingo 编写程序，求出结果（代码 3）

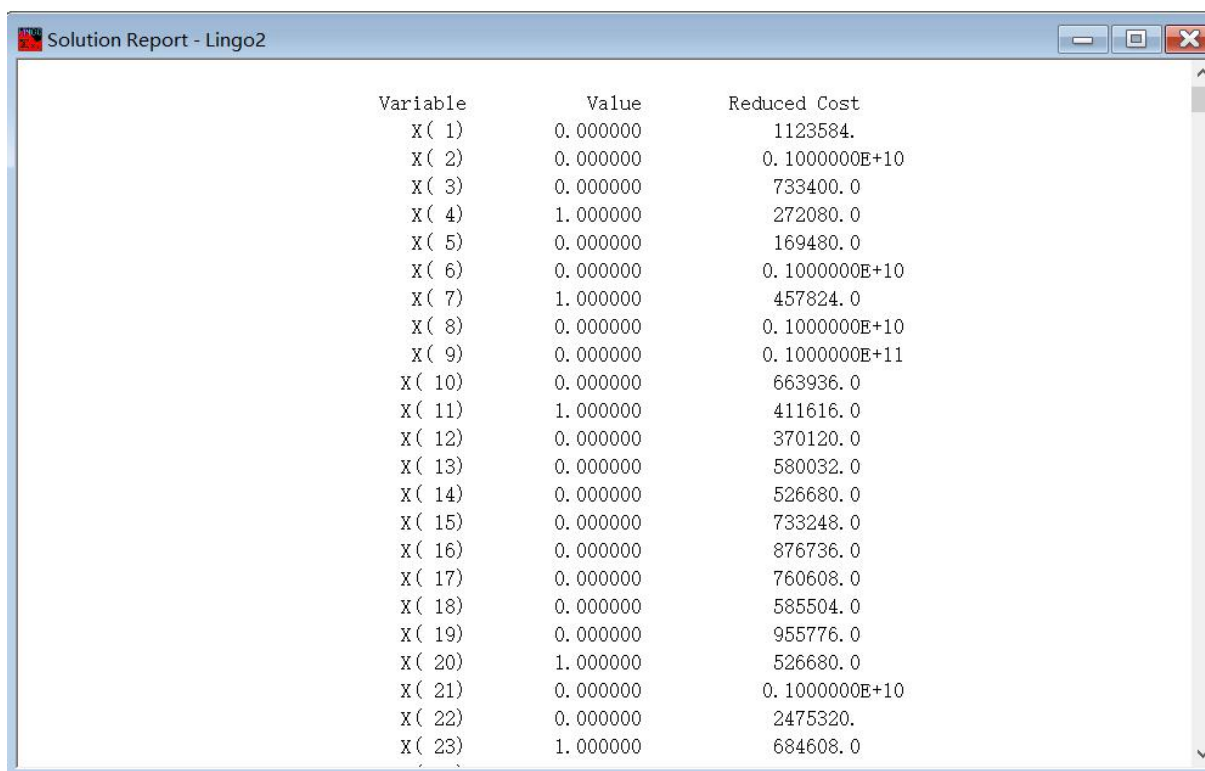


The image shows a screenshot of the 'Solution Report - Lingo2' window. It displays the results of a global optimal solution found. The report includes the objective value, objective bound, infeasibilities, extended solver steps, total solver iterations, elapsed runtime seconds, and the model class.

Global optimal solution found.	
Objective value:	9197118.
Objective bound:	9197118.
Infeasibilities:	0.000000
Extended solver steps:	0
Total solver iterations:	1419
Elapsed runtime seconds:	0.25
Model Class:	PILP

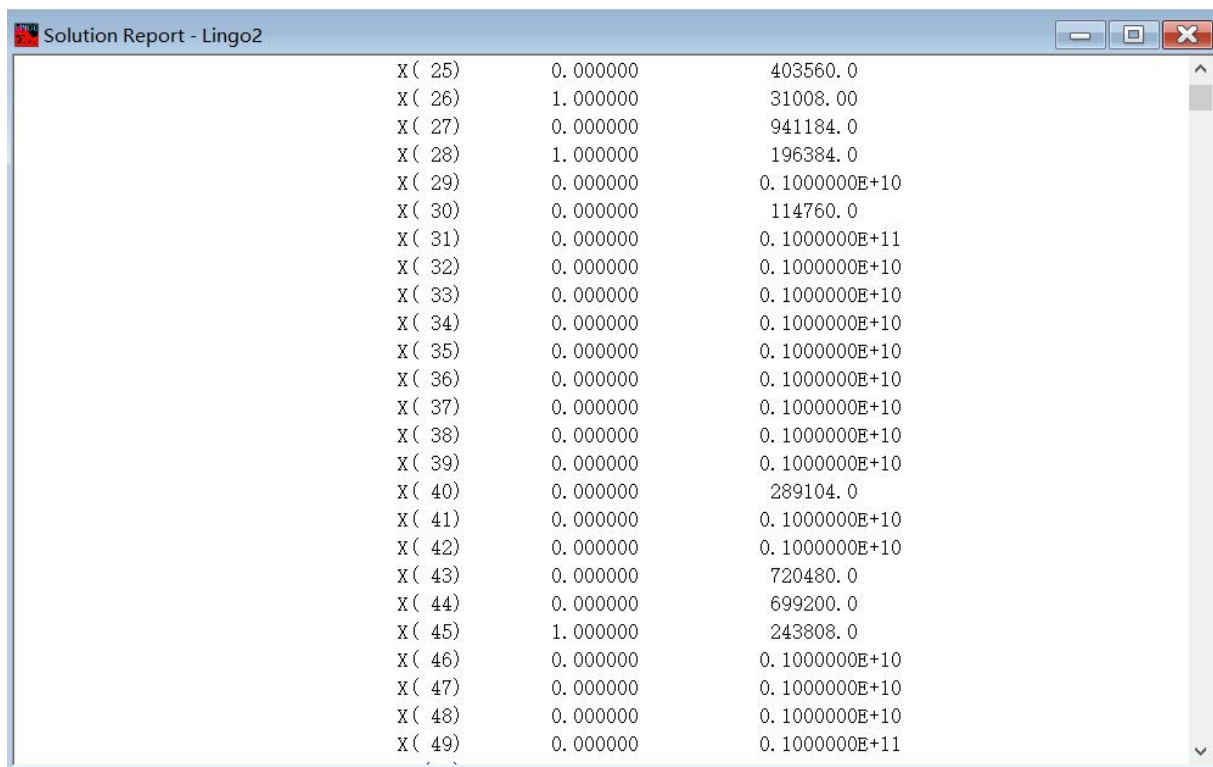
图 4：总费用最小值结果图

以下 Value 为 1 的点就是作为供应点的城市



The image shows a screenshot of the 'Solution Report - Lingo2' window displaying a table of variable values and reduced costs. The table lists 23 variables, X(1) through X(23), with their corresponding values and reduced costs.

Variable	Value	Reduced Cost
X(1)	0.000000	1123584.
X(2)	0.000000	0.1000000E+10
X(3)	0.000000	733400.0
X(4)	1.000000	272080.0
X(5)	0.000000	169480.0
X(6)	0.000000	0.1000000E+10
X(7)	1.000000	457824.0
X(8)	0.000000	0.1000000E+10
X(9)	0.000000	0.1000000E+11
X(10)	0.000000	663936.0
X(11)	1.000000	411616.0
X(12)	0.000000	370120.0
X(13)	0.000000	580032.0
X(14)	0.000000	526680.0
X(15)	0.000000	733248.0
X(16)	0.000000	876736.0
X(17)	0.000000	760608.0
X(18)	0.000000	585504.0
X(19)	0.000000	955776.0
X(20)	1.000000	526680.0
X(21)	0.000000	0.1000000E+10
X(22)	0.000000	2475320.
X(23)	1.000000	684608.0



Variable	Value	Value
X(25)	0.000000	403560.0
X(26)	1.000000	31008.00
X(27)	0.000000	941184.0
X(28)	1.000000	196384.0
X(29)	0.000000	0.1000000E+10
X(30)	0.000000	114760.0
X(31)	0.000000	0.1000000E+11
X(32)	0.000000	0.1000000E+10
X(33)	0.000000	0.1000000E+10
X(34)	0.000000	0.1000000E+10
X(35)	0.000000	0.1000000E+10
X(36)	0.000000	0.1000000E+10
X(37)	0.000000	0.1000000E+10
X(38)	0.000000	0.1000000E+10
X(39)	0.000000	0.1000000E+10
X(40)	0.000000	289104.0
X(41)	0.000000	0.1000000E+10
X(42)	0.000000	0.1000000E+10
X(43)	0.000000	720480.0
X(44)	0.000000	699200.0
X(45)	1.000000	243808.0
X(46)	0.000000	0.1000000E+10
X(47)	0.000000	0.1000000E+10
X(48)	0.000000	0.1000000E+10
X(49)	0.000000	0.1000000E+11

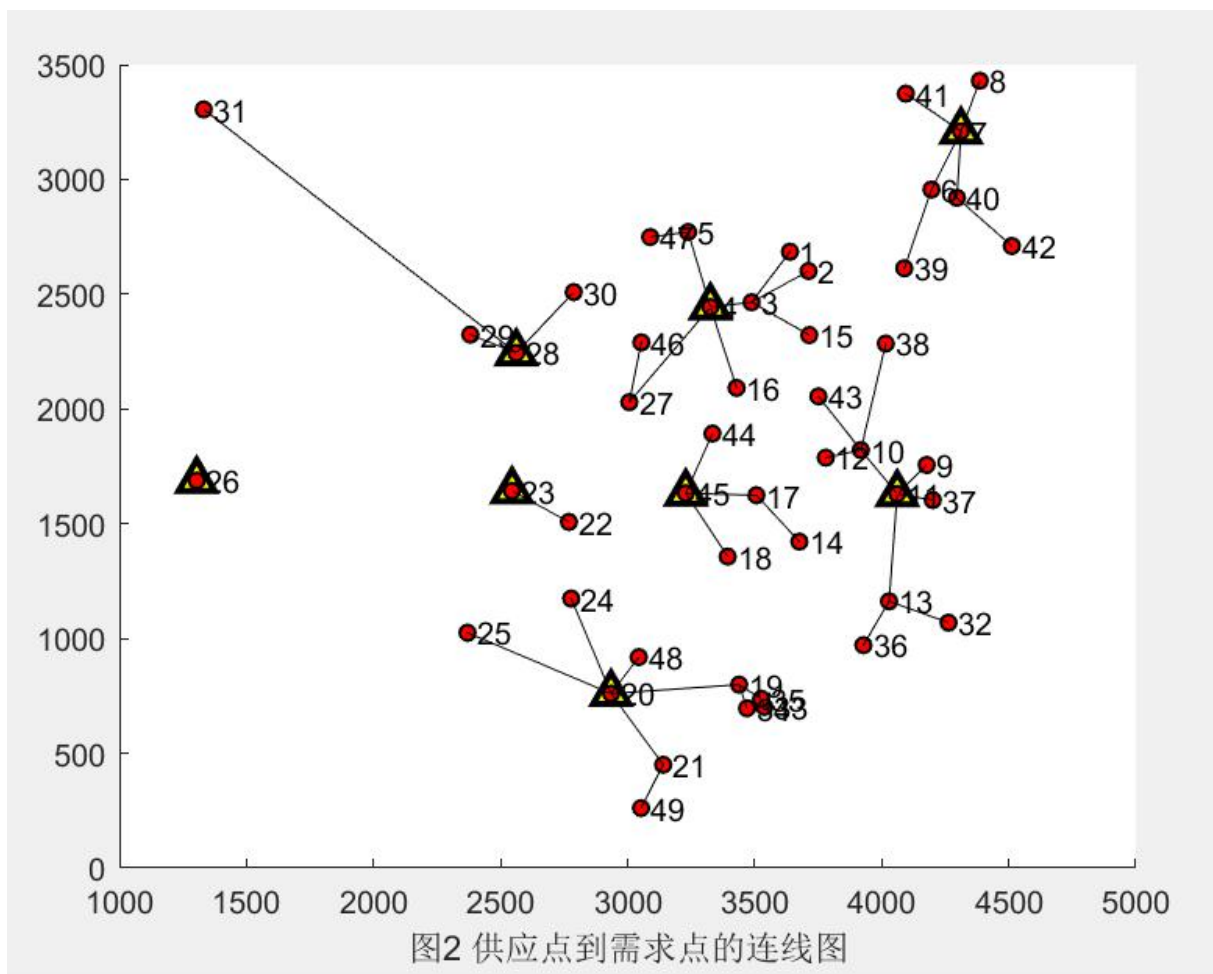
图 5：供应点图

由结果可知，可以作为供应点的城市有 8 个，分别是 4, 7, 11, 20, 23, 26, 28, 45。他们供货的城市如下表：

表 2：供货点及其对应的供货城市

编号	供货点城市	供货的城市	固定费用
1	4	1, 2, 3, 4, 5, 15, 16, 27, 46, 47	272080
2	7	6, 7, 8, 39, 40, 41, 42	457824
3	11	9, 10, 11, 12, 13, 32, 36, 37, 38, 43	411616
4	20	19, 20, 21, 24, 25, 33, 34, 35, 48, 49	526680
5	23	22, 23	684608
6	26	26	31008
7	28	28, 29, 30, 31	196384
8	45	14, 17, 18, 44, 45	243808

然后用 Matlab 画出每一个供应点到需求点的连接图（见代码 4）



最后用根据结果可知最小总费用 = 固定费用+运输费用=9197118

(5.1.5)

5.2 问题二模型的建立与求解

5.2.1 模型分析

从表 4 中可以看出，若是破坏第 8 条道路，49 号城市将无法得到供应，所以不考虑这条破坏情况。所以只有八条道路可破坏，接下来就是考虑破坏道路的条数，如果只破坏一条道路，那就分别破坏这 8 条道路，分别求出它们的总费用增加数量以及各自增加的总费用的比，得出总费用增加最多的那条路；如果破坏两条道路，分别破坏除了上次求出的总费用增加最多的那条路剩下的 7 条道路，分别求出它们的总费用增加数量以及各自增加的总费用的比率，然后求出总费用增加最多的那条道路……一直求到同时破坏 8 条路增加的总费用。最后根据顺序选出最优的破坏方案。

5.2.2 模型建立

1. 若只破坏一条路，分别破坏这八条道路，用 Floyd 算法，借助 C 语言程序（见附录程

5), 求出破坏后的最短距离, 求出他们增加的总费用, 最后找出增加的总费用最多的那条路进行破坏。

表 3: 破坏一条道路后增加的费用

优先破坏顺序	道路序号	增加的费用
1	2	1245420
2	5	381625
3	7	338895
4	4	269310
5	1	107800
6	9	77015
7	3	65379
8	6	0

总费用增加率 $p = \frac{N(i)}{M}$ ($N(i)$ 为破坏 i 条道路后增加的总费用, M 为破坏前的总费用)
(5.2.1)

由题可知, 约束条件为总费用增加率 $p = \frac{N(i)}{M} * 100\% \geq 25\%$ (5.2.2)

2. 然后在破坏第一步里面的那条道路的情况下对剩下的道路分别进行破坏, 然后找出剩下的道路里面破坏后总费用增加最多的那条路。接着考虑第三条, 第四条……第八条就可以得出优先破坏顺序。

表 4: 破坏多条道路后增加的费用及其增加率

破坏道路数	破坏道路序号	破坏后增加的总费用	总费用增加率
1	2	1245420	13.54%
2	2、5	1627045	17.80%
3	2、5、7	1965940	18.08%
4	2、5、7、4	2235250	24.30%
5	2、5、7、4、1	2243050	24.39%
6	2、5、7、4、1、9	2420065	26.31%

7	2、5、7、4、1、9、3	2485444	27.02%
8	2、5、7、4、1、9、3、 6	2485444	27.02%

5.2.3 模型求解

由表可知当破坏道路数为6条时, $p = (N(6))/M = 26.31\% > 25\%$ (5.2.3)

则总费用 = 2420065 (5.2.4)

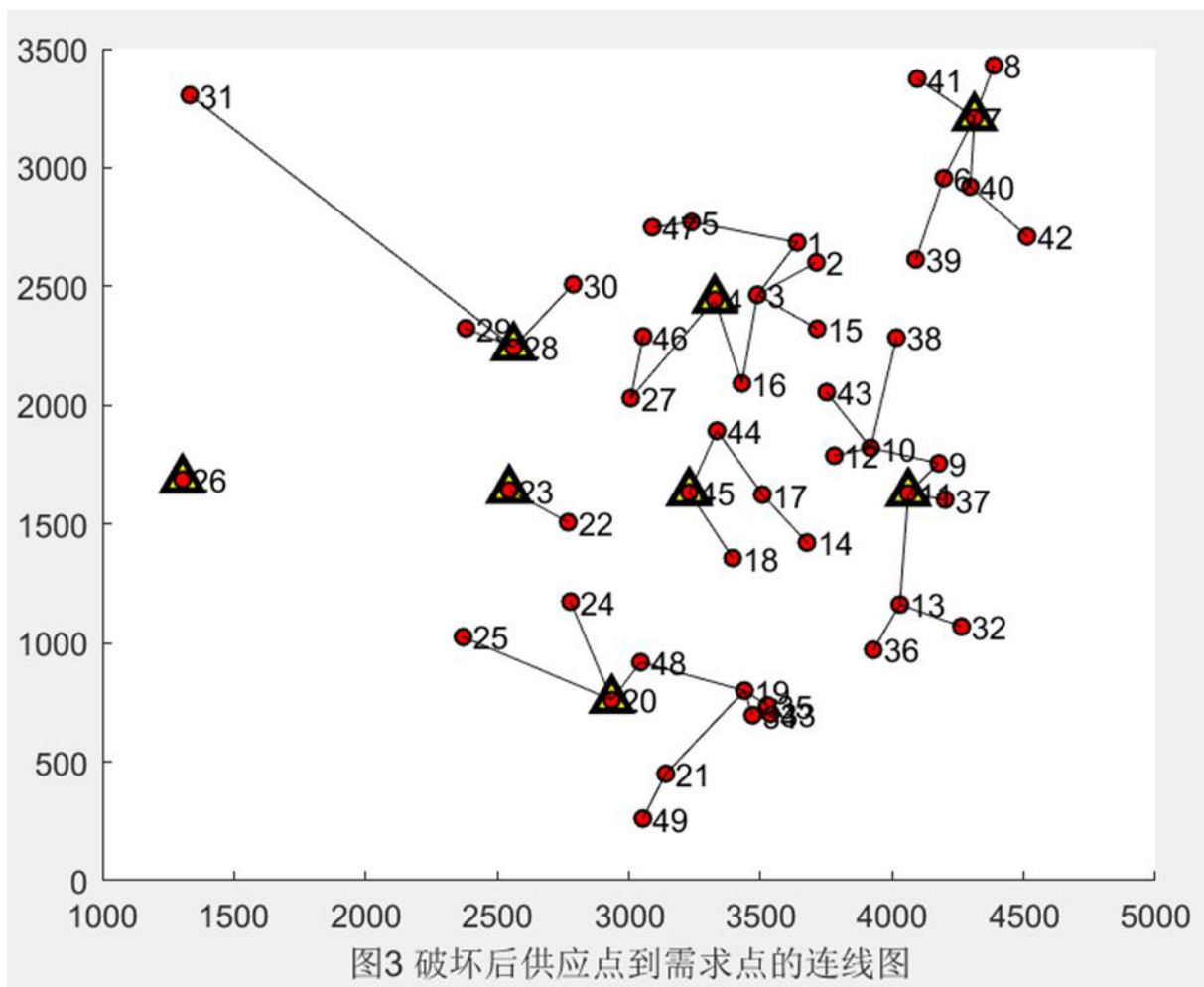
破坏的道路的道路序号为 2、5、7、4、1、9

破坏的道路为下表所示

表 5: 破坏的道路所连接的城市表

道路序号	道路所连接的两个城市
2	3-4
5	19-20
7	17-45
4	10-11
1	4-5
9	20-21

用 Matlab 画出道路被破坏之后的供应点到需求点的最短距离图



5.3 问题三模型的建立与求解

5.3.1 模型分析

由于破坏方破坏时，这些道路不一定会被破坏，而破坏情况服从一定的概率分布。因为道路被破坏是独立事件，相互之间不受影响。所以就可以建立一个独立事件概率模型，由于破坏方破坏时，这些道路不一定会被破坏，而破坏情况服从一定的概率分布。所以就可以利用数学期望来计算增加的费用平均值，即计算出被破坏的道路的平均总费用。要使破坏的平均总费用最大，就是破坏的道路越多，增加费用也就越多。不过，由题二可以看出，破坏6号道路对总费用没有影响，所以我们只需要破坏剩下的七条道路就可以使平均总费用最大。

5.3.2 模型建立

根据概率论中数学期望的定义，可以得出

$$\text{Max_平均总费用} = E(i) = \sum_i r(i) * p(i), \quad i=1, 2, 3, 4, 5, 6, 7, 9 \quad (5.3.1)$$

$p(i)$ 就是破坏道路 i 的概率, $r(i)$ 就是破坏第 i 条道路所增加的费用

然后由题二可以得到破坏第 i 条道路所增加的费用

5.3.3 模型求解

$$r(i) = [107800, 1245420, 65379, 269310, 381625, 338895, 77015] \quad (5.3.2)$$

$$p(i) = [0.6 \ 0.7 \ 0.45 \ 0.5 \ 0.55 \ 0.5 \ 0.6] \quad (5.3.3)$$

$$\text{Max_平均费用} = \sum_i r(i) * p(i) = 1526099.8 \quad (5.3.4)$$

所以, 破坏的道路序号为 1、2、3、4、5、7、9, 平均总费用最大为 1526099.8

表 6: 道路被破坏后所增加的平均总费用

道路序号	破坏后增加的平均总费用
2	871794
5	209873.75
7	169447.5
4	134655
1	64680
9	46209
3	29420.55

六、模型评价

模型优点:

1. Floyd 算法容易理解, 可以算出任意两个点的最短距离, 能使一个很复杂的问题通过模型快速的解出来。
2. 结果采用了专业的数学软件, 结果更具有真实性, 比人为计算结果更精确。
3. 很容易找出最优解, 便于快速解答问题。

模型缺点:

1. floyd 算法时间复杂度比较高, 不适合计算大量数据。

2. 当样本点很多时，无法通过一一列举的方法求出他所增加的费用。
3. 模型复杂因素较多，不能对其进行全面的考虑，与实际可能会有一些不符。

七、参考文献

- [1]姜启源，谢金星，叶俊. 数学模型（第五版）[M]. 北京：，高等教育出版社，2019.
- [2]肖华勇. 实用数学建模与软件应用[M]. 西安:西北工业大学出版社，2014.
- [3]姚泽清，郑旭东，赵颖. 全国大学生数学建模竞赛赛题与优秀论文评析[M]. 北京：国防工业出版社，2012.
- [4]白其峥，数学建模案例分析[M]. 北京:海洋出版社，2000.
- [5]沈继红. 数学建模[M]. 哈尔滨:哈尔滨工程大学出版社，1998.

附录

代码 1:

```
x=[3639 3712 3488 3326 3238 4196 4312 4386 4177 3918 4061 3780 4029 3676 3715 3429 3507 3394  
3439 2935 3140 2769 2545 2778 2370 1304 3007 2562 2381 2788 1332 4263 3538 3470 3526 3928 4201  
4016 4089 4296 4095 4512 3751 3334 3229 3054 3089 3044 3053];
```

```
y=[2685 2601 2465 2444 2771 2956 3210 3430 1756 1821 1630 1788 1162 1422 2322 2092 1624 1357  
799 760 450 1508 1643 1174 1025 1688 2030 2244 2324 2509 3305 1069 702 696 737 971 1603 2285  
2613 2920 3374 2710 2055 1893 1633 2290 2749 919 261];
```

```
z=[1 2 120;1 3 270;1 5 540;1 6 799;1 15 420;1 40 844;2 3 370;2  
15 360;3 4 210;3 15 311;3 16 440;4 5 530;4 16 430;4 27 630;4  
30 760;5 30 720;5 40 1521;5 47 186;6 7 330;6 39 387;6 40 727; 8  
230;7 40 429;7 41 347;8 42 819;9 10 280;9 11 190;9 15 840;10 11  
279;10 12 160;10 14 660;10 15 680;10 38 598;10 43 325;11 13 880;11 14  
640;11 37 153;12 14 610;12 16 650;12 17 540;12 43 435;13 14 680;13 19  
1020;13 32 490;13 36 266;13 37 592;14 17 270;14 18 640;14 19 860;15 16  
430;15 38 361;15 43 349;16 17 540;16 27 550;16 43 473;16 44 285;17 18  
380;17 44 406;17 45 362;18 19 780;18 24 1010;18 45 508;18 48 664;19 20  
710;19 21 580;19 34 130;19 35 127;19 36 688;20 21 560;20 24 650;20 25  
820;20 48 305;21 49 270;22 23 340;22 24 490;22 25 1090;22 27 910;22 45  
795;23 25 990;23 26 2170;23 27 920;24 25 650;24 48 560;25 26 2320;26 29  
1940;26 31 2672;27 28 700;27 30 640;27 44 637;27 46 304;28 29 230;28 30  
500;28 31 1980;30 47 554;33 35 36;35 36 591;38 43 368;40 41 304;40 42  
929;41 42 669;44 45 466;46 47 541];
```

```
figure;
```

```
plot(x,y,'rs');
```

```
[m n]=size(z);
```

```
for i=1:m
```

```

hold on;

plot( [x(z(i,1)),x(z(i,2))],[y(z(i,1)),y(z(i,2))] );

end

```

代码 2:

```
M=inf;
```

```

a=[0  120 270 M  540 799 M  M  M  M  M  M  M  M  420 M  M  M  M  M  M
    M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  844 M  M
    M  M  M  M  M  M  M
120 0   370 M  M  M  M  M  M  M  M  M  M  M  360 M  M  M  M  M  M
    M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M
    M  M  M  M  M  M  M
270 370 0   210 M  M  M  M  M  M  M  M  M  M  311 440 M  M  M  M  M
    M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M
    M  M  M  M  M  M  M
    M  M  210 0   530 M  M  M  M  M  M  M  M  M  M  430 M  M  M  M  M
    M  M  M  M  M  630 M  M  760 M  M  M  M  M  M  M  M  M  M  M  M
    M  M  M  M  M  M  M
540 M  M  530 0   M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M
    M  M  M  M  M  M  M  M  720 M  M  M  M  M  M  M  M  M  1521  M
    M  M  M  M  M  186 M  M
799 M  M  M  M  0   330 M  M  M  M  M  M  M  M  M  M  M  M  M  M  M
    M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  387 727 M  M
    M  M  M  M  M  M  M
    M  M  M  M  M  330 0   230 M  M  M  M  M  M  M  M  M  M  M  M  M
    M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  429 347 M
    M  M  M  M  M  M  M
    M  M  M  M  M  M  230 0   M  M  M  M  M  M  M  M  M  M  M  M  M
    M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M  M
819 M  M  M  M  M  M  M  M
    M  M  M  M  M  M  M  M  0   280 190 M  M  M  840 M  M  M  M  M  M

```

M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	280	0	279	160	M	660	680	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	598	M	M	M	M
325	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	190	279	0	M	880	640	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	153	M	M	M	M
M	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	160	M	0	M	610	M	650	540	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
435	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	880	M	0	680	M	M	M	M	1020		M
M	M	M	M	M	M	M	M	M	M	M	490	M	M	M	266	592	M	M	M	M
M	M	M	M	M	M	M	M													
M	M	M	M	M	M	M	M	M	660	640	610	680	0	M	M	270	640	860	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M														
420	360	311	M	M	M	M	M	840	680	M	M	M	M	0	430	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	361	M	M	M	M
349	M	M	M	M	M	M														
M	M	440	430	M	M	M	M	M	M	M	650	M	M	430	0	540	M	M	M	M
M	M	M	M	M	550	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
473	285	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	540	M	270	M	540	0	380	M	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	406	362	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	640	M	M	380	0	780	M	M
M	M	1010		M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	508	M	M	664	M													
M	M	M	M	M	M	M	M	M	M	M	M	1020		860	M	M	M	780	0	

710	580	M	M	M	M	M	M	M	M	M	M	M	M	M	130	127	688	M	M	M	M
M	M	M	M	M	M	M	M	M													
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	710	0	
560	M	M	650	820	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	305	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	580	560	0
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	270															
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
0	340	490	1090		M	910	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	795	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
340	0	M	990	2170		920	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	1010		M	
650	M	490	M	0	650	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	560	M													
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	820	M
1090		990	650	0	2320		M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M													
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	2170		M	2320	0	M	M	1940		M	2672		M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M												
M	M	M	630	M	M	M	M	M	M	M	M	M	M	M	550	M	M	M	M	M	M
910	920	M	M	M	0	700	M	640	M	M	M	M	M	M	M	M	M	M	M	M	M
M	637	M	304	M	M	M															
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	700	0	230	500	1980		M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M

M	M	M	M	1940	M	230	0	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M													
M	M	M	760	720	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	640	500	M	0	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	554	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	2672	M	1980		M	M	0	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M												
M	M	M	M	M	M	M	M	M	M	M	M	490	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	0	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M	0	M	36	M	M	M	M	M	M	M
M	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	130	M	M
M	M	M	M	M	M	M	M	M	M	M	M	0	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	127	M	M
M	M	M	M	M	M	M	M	M	M	M	36	M	0	591	M	M	M	M	M	M
M	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	153	M	592	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	0	M	M	M	M	M
M	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	598	M	M	M	M	361	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	0	M	M	M	M
368	M	M	M	M	M	M														
M	M	M	M	M	387	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M

M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	0	M	M	M
M	M	M	M	M	M	M														
844	M	M	M	1521		727	429	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	0	
304	929	M	M	M	M	M	M	M												
M	M	M	M	M	M	347	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	304	0	
669	M	M	M	M	M	M	M													
M	M	M	M	M	M	M	819	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	929	669	0
M	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	325	M	435	M	M	349	473	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	368	M	M	M	M
0	M	M	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	285	406	M	M	M	M
M	M	M	M	M	637	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	0	466	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	362	508	M	M	M
795	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	466	0	M	M	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	304	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	0	541	M	M														
M	M	M	M	186	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	554	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	541	0	M	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	664	M	305	M
M	M	560	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	0	M														
M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	

```

270 M M M M M M M M M M M M M M M M M M M M M M
M M M M M M M 0

```

```

];
n=size(a,1);
D=a;
path=zeros(n,n);
for i=1:n
    for j=1:n
        if D(i,j)~=M
            path(i,j)=j;
        end
    end
end
end
for k=1:n
    for i=1:n
        for j=1:n
            if D(i,k)+D(k,j)<D(i,j)
                D(i,j)=D(i,k)+D(k,j);
                path(i,j)=path(i,k);
            end
        end
    end
end
end
D;path;
代码 3:
model:
sets:
weizhi/1..49/:x,gufei,xuqiu;
assign(weizhi,weizhi):C,y;
endsets

```


data:

gufei

=1123584,1000000000,733400,272080,169480,1000000000,457824,1000000000,1000000000,663936,4
11616,370120,580032,526680,733248,876736,760608,585504,955776,526680,1000000000,2475320,684
608,276640,403560,31008,941184,196384,1000000000,114760,1000000000,1000000000,1000000000,
1000000000,1000000000,1000000000,1000000000,1000000000,1000000000,289104,1000000000,10000
00000,720480,699200,243808,1000000000,1000000000,1000000000,1000000000;

xuqiu=1232,974,965,358

223,715,753,989,1391,624,677,487,636,495,603,721,834,642,786,693,156,3257,1126,364,531,51,774,32
3,194,151,234,246,701,55,233,174,568,761,583,317,204,272,948,1150,401,224,217,366,55;

C= 0 120 270 480 540 799 1129 1359 1260 1094 1373 1204 2118 1520 420 710 1250 1630 2380 2599

2960 2020 2030 2510 3020 3910 1110 1740 1970 1240 3720 2608 2543 2510 2507 2384 1526

781 1186 844 1148 1773 769 995 1461 1267 726 2294 3230

120 0 370 580 660 919 1249 1479 1200 1034 1313 1144 2058 1600 360 790 1330 1710 2460 2679

3040 2120 2130 2610 3120 4010 1210 1840 2070 1340 3820 2548 2623 2590 2587 2324 1466

721 1306 964 1268 1893 709 1075 1541 1387 846 2374 3310

270 370 0 210 740 1069 1399 1629 1151 985 1264 1090 1930 1250 311 440 980 1360 2110 2329

2690 1750 1760 2240 2750 3640 840 1470 1700 970 3450 2420 2273 2240 2237 2196 1417 672 1456

1114 1418 2043 660 725 1191 1144 926 2024 2960

480 580 210 0 530 1279 1609 1839 1361 1195 1474 1080 1920 1240 521 430 970 1350 2100

2319 2680 1540 1550 2030 2540 3430 630 1260 1490 760 3240 2410 2263 2230 2227 2186 1627

882 1666 1324 1628 2253 870 715 1181 934 716 2014 2950

540 660 740 530 0 1339 1669 1899 1800 1634 1913 1610 2450 1770 960 960 1500 1880 2630

2849 3210 1941 1951 2431 2941 3390 1031 1220 1450 720 3200 2940 2793 2760 2757 2716 2066

1321 1726 1384 1688 2313 1309 1245 1711 727 186 2544 3480

799 919 1069 1279 1339 0 330 560 2059 1893 2172 2003 2917 2319 1219 1509 2049 2429 3179

3398 3759 2819 2829 3309 3819 4709 1909 2539 2769 2039 4519 3407 3342 3309 3306 3183 2325

1580 387 727 677 1346 1568 1794 2260 2066 1525 3093 4029

1129 1249 1399 1609 1669 330 0 230 2389 2223 2502 2333 3247 2649 1549 1839 2379 2759 3509

3728 4089 3149 3159 3639 4149 5039 2239 2869 3099 2369 4849 3737 3672 3639 3636 3513 2655
 1910 717 429 347 1016 1898 2124 2590 2396 1855 3423 4359 1359 1479 1629 1839 1899 560
 230 0 2619 2453 2732 2563 3477 2879 1779 2069 2609 2989 3739 3958 4319 3379 3389 3869
 4379 5269 2469 3099 3329 2599 5079 3967 3902 3869 3866 3743 2885 2140 947 659 577 81 2128
 2354 2820 2626 2085 3653 4589
 1260 1200 1151 1361 1800 2059 2389 2619 0 280 190 440 935 830 840 1078 980 1360 1690 2329
 2270 2137 2477 2370 3020 4498 1628 2328 2558 2121 4308 1425 1828 1820 1792 1201 343 878
 2446 2104 2408 3033 605 1363 1342 1932 1986 2024 2540
 1094 1034 985 1195 1634 1893 2223 2453 280 0 279 160 1024 660 674 798 700 1080 1520 2049
 2100 1857 2197 2090 2740 4218 1348 2048 2278 1955 4028 1514 1683 1650 1647 1290 432 598
 2280 1938 2242 2867 325 1083 1062 1652 1820 1744 2370
 1373 1313 1264 1474 1913 2172 2502 2732 190 279 0 439 745 640 953 1077 910 1280 1500 2210
 2080 2067 2407 2290 2940 4497 1627 2327 2557 2234 4307 1235 1638 1630 1602 1011 153 877
 2559 2217 2521 3146 604 1316 1272 1931 2099 1944 2350
 1204 1144 1090 1080 1610 2003 2333 2563 440 160 439 0 1184 610 784 650 540 920 1470 1889
 2050 1697 2037 1930 2580 4070 1200 1900 2130 1840 3880 1674 1633 1600 1597 1450 592 758
 2390 2048 2352 2977 435 935 902 1504 1796 1584 2320
 2118 2058 1930 1920 2450 2917 3247 3477 935 1024 745 1184 0 680 1698 1490 950 1320 954 1664
 1534 2107 2447 2314 2484 4617 1993 2693 2923 2633 4673 490 893 1084 857 266 592 1622 3304
 2962 3266 3891 1349 1356 1312 2297 2636 1969 1804
 1520 1600 1250 1240 1770 2319 2649 2879 830 660 640 610 680 0 1240 810 270 640 860 1570
 1440 1427 1767 1650 2300 3937 1313 2013 2243 1953 3993 1170 1023 990 987 946 793 1258
 2706 2364 2668 3293 985 676 632 1617 1956 1304 1710
 420 360 311 521 960 1219 1549 1779 840 674 953 784 1698 1240 0 430 970 1350 2100 2319
 2680 1890 1900 2360 2890 3850 980 1680 1910 1281 3660 2188 2263 2230 2227 1964 1106
 361 1606 1264 1568 2193 349 715 1181 1284 1146 2014 2950
 710 790 440 430 960 1509 1839 2069 1078 798 1077 650 1490 810 430 0 540 920 1670 1889
 2250 1460 1470 1930 2460 3420 550 1250 1480 1190 3230 1980 1833 1800 1797 1756 1230
 791 1896 1554 1858 2483 473 285 751 854 1146 1584 2520
 1250 1330 980 970 1500 2049 2379 2609 980 700 910 540 950 270

970	540	0	380	1130	1349	1710	1157	1497	1390	2040	3667
1043	1743	1973	1683	3723	1440	1293	1260	1257	1216		
1063	1298	2436	2094	2398	3023	975	406	362	1347	1686	1044
1980											
1630	1710	1360	1350	1880	2429	2759	2989	1360	1080		
1280	920	1320	640	1350	920	380	0	780	969	1360	1303
1660	3813	1423	2123	2353	2063	4103	1810	943	910	907	1468
1433	1678	2816	2474	2778	3403	1355	786	508	1727	2066	
664	1630										
2380	2460	2110	2100	2630	3179	3509	3739	1690	1520		
1500	1470	1020	860	2100	1670	1130	780	0	710	580	1850
1360	1530	3850	2173	2873	3103	2813	4853	1510	163	130	
127	688	1612	2118	3566	3224	3528	4153	1845	1536	1288	
2477	2816	1015	850								
2599	2679	2329	2319	2849	3398	3728	3958	2329	2049		
2210	1889	1730	1570	2319	1889	1349	969	710	0	560	1140
1480	650	820	3140	2050	2750	2980	2690	4730	2220	873	840
837	1398	2322	2647	3785	3443	3747	4372	2324	1755	1477	
2354	2895	305	830								
2960	3040	2690	2680	3210	3759	4089	4319	2270	2100		
2080	2050	1600	1440	2680	2250	1710	1360	580	560	0	1700
2040	1210	1380	3700	2610	3310	3540	3250	5290	2090		
743	710	707	1268	2192	2698	4146	3804	4108	4733	2425	2116
1868	2914	3396	865	270							
2020	2120	1750	1540	1941	2819	3149	3379	2137	1857		
2067	1697	2107	1427	1890	1460	1157	1303	1850	1140		
1700	0	340	490	1090	2510	910	1610	1840	1550	3590	2597
2013	1980	1977	2373	2220	2251	3206	2864	3168	3793		
1933	1261	795	1214	1755	1050	1970					
2030	2130	1760	1550	1951	2829	3159	3389	2477	2197		

2407	2037	2447	1767	1900	1470	1497	1643	2190	1480
2040	340 0	830 990	2170	920 1620	1850	1560	3600	2937	2353
2320	2317	2713	2560	2261	3216	2874	3178	3803	1943
1557	1135	1224	1765	1390	2310				
2510	2610	2240	2030	2431	3309	3639	3869	2370	2090
2290	1930	2330	1650	2360	1930	1390	1010	1360	650 1210
490 830 0	650	2970	1400	2100	2330	2040	4080	2820	1523
1490	1487	2048	2443	2688	3696	3354	3658	4283	2365
1751	1285	1704	2245	560 1480					
3020	3120	2750	2540	2941	3819	4149	4379	3020	2740
2940	2580	2550	2300	2890	2460	2040	1660	1530	820 1380
1090	990 650 0	2320	1910	2610	2840	2550	4590	3040	1693
1660	1657	2218	3093	3251	4206	3864	4168	4793	2933
2351	1885	2214	2755	1125	1650				
3910	4010	3640	3430	3390	4709	5039	5269	4498	4218
4497	4070	4617	3937	3850	3420	3667	3813	3850	3140
3700	2510	2170	2970	2320	0 2870	2170	1940	2670	2672
5107	4013	3980	3977	4538	4650	4211	5096	4754	5058
5683	3893	3507	3305	3174	3224	3445	3970		
1110	1210	840 630	1031	1909	2239	2469	1628	1348	1627
1200	1993	1313	980 550	1043	1423	2173	2050	2610	910 920
1400	1910	2870	0 700	930 640	2680	2483	2336	2303	2300
2259	1780	1341	2296	1954	2258	2883	1023	637 1103	304
845 1960	2880								
1740	1840	1470	1260	1220	2539	2869	3099	2328	2048
2327	1900	2693	2013	1680	1250	1743	2123	2873	2750
3310	1610	1620	2100	2610	2170	700 0	230 500	1980	3183
3036	3003	3000	2959	2480	2041	2926	2584	2888	3513
1723	1337	1803	1004	1054	2660	3580			
1970	2070	1700	1490	1450	2769	3099	3329	2558	2278

2557	2130	2923	2243	1910	1480	1973	2353	3103	2980	
3540	1840	1850	2330	2840	1940	930 230 0	730 2210	3413		
3266	3233	3230	3189	2710	2271	3156	2814	3118	3743	
1953	1567	2033	1234	1284	2890	3810				
1240	1340	970 760	720 2039	2369	2599	2121	1955	2234	1840	
2633	1953	1281	1190	1683	2063	2813	2690	3250	1550	
1560	2040	2550	2670	640 500	730 0	2480	3123	2976	2943	
2940	2899	2387	1642	2426	2084	2388	3013	1630	1277	
1743	944 554	2600	3520							
3720	3820	3450	3240	3200	4519	4849	5079	4308	4028	
4307	3880	4673	3993	3660	3230	3723	4103	4853	4730	
5290	3590	3600	4080	4590	2672	2680	1980	2210	2480	0
5163	5016	4983	4980	4939	4460	4021	4906	4564	4868	
5493	3703	3317	3783	2984	3034	4640	5560			
2608	2548	2420	2410	2940	3407	3737	3967	1425	1514	
1235	1674	490 1170	2188	1980	1440	1810	1444	2154	2024	
2597	2937	2804	2974	5107	2483	3183	3413	3123	5163	0
1383	1574	1347	756 1082	2112	3794	3452	3756	4381	1839	
1846	1802	2787	3126	2459	2294					
2543	2623	2273	2263	2793	3342	3672	3902	1853	1683	
1663	1633	1183	1023	2263	1833	1293	943 163	873 743	2013	
2353	1523	1693	4013	2336	3036	3266	2976	5016	1673	0
293 36	627 1775	2281	3729	3387	3691	4316	2008	1699	1451	
2640	2979	1178	1013							
2510	2590	2240	2230	2760	3309	3639	3869	1820	1650	
1630	1600	1150	990 2230	1800	1260	910 130	840 710	1980	2320	
1490	1660	3980	2303	3003	3233	2943	4983	1640	293 0	
257 818	1742	2248	3696	3354	3658	4283	1975	1666	1418	
2607	2946	1145	980							
2507	2587	2237	2227	2757	3306	3636	3866	1817	1647	

1627	1597	1147	987	2227	1797	1257	907	127	837	707	1977	2317	
1487	1657	3977	2300	3000	3230	2940	4980	1637	36	257	0		
591	1739	2245	3693	3351	3655	4280	1972	1663	1415	2604			
2943	1142	977											
3068	3148	2798	2788	3318	3867	4197	4427	2378	2208				
2188	2158	1708	1548	2788	2358	1818	1468	688	1398	1268			
2538	2878	2048	2218	4538	2861	3561	3791	3501	5541				
2198	627	818	591	0	2300	2806	4254	3912	4216	4841	2533		
2224	1976	3165	3504	1703	1538								
1526	1466	1417	1627	2066	2325	2655	2885	343	432	153	592		
592	793	1106	1230	1063	1433	1546	2256	2126	2220	2560			
2443	3076	4650	1780	2480	2710	2387	4460	1082	1485				
1676	1449	858	0	1030	2712	2370	2674	3299	757	1469	1425		
2084	2252	2097	2396										
781	721	672	882	1321	1580	1910	2140	878	598	877	758	1622	1258
361	791	1298	1678	2118	2647	2698	2251	2261	2688	3251			
4211	1341	2041	2271	1642	4021	2112	2281	2248	2245				
1888	1030	0	1967	1625	1929	2554	368	1076	1542	1645			
1507	2342	2968											
1186	1306	1456	1666	1726	387	717	947	2446	2280	2559	2390		
3304	2706	1606	1896	2436	2816	3566	3785	4146	3206				
3216	3696	4206	5096	2296	2926	3156	2426	4906	3794				
3729	3696	3693	3570	2712	1967	0	1114	1064	1733	1955			
2181	2647	2453	1912	3480	4416								
844	964	1114	1324	1384	727	429	659	2104	1938	2217	2048	2962	
2364	1264	1554	2094	2474	3224	3443	3804	2864	2874				
3354	3864	4754	1954	2584	2814	2084	4564	3452	3387				
3354	3351	3228	2370	1625	1114	0	304	929	1613	1839	2305		
2111	1570	3138	4074										
1148	1268	1418	1628	1688	677	347	577	2408	2242	2521	2352		

3266	2668	1568	1858	2398	2778	3528	3747	4108	3168
3178	3658	4168	5058	2258	2888	3118	2388	4868	3756
3691	3658	3655	3532	2674	1929	1064	304 0	669 1917	2143
2609	2415	1874	3442	4378					
1773	1893	2043	2253	2313	1346	1016	819 3033	2867	3146
2977	3891	3293	2193	2483	3023	3403	4153	4372	4733
3793	3803	4283	4793	5683	2883	3513	3743	3013	5493
4381	4316	4283	4280	4157	3299	2554	1733	929 669 0	2542
2768	3234	3040	2499	4067	5003				
769 709 660 870 1309	1568	1898	2128	605 325 604 435 1349	985 349				
473 975 1355	1845	2324	2425	1933	1943	2365	2933	3893	
1023	1723	1953	1630	3703	1839	2008	1975	1972	1615
757 368 1955	1613	1917	2542	0	758 1224	1327	1495	2019	
2695									
995 1075	725 715 1245	1794	2124	2354	1363	1083	1316	935	
1356	676 715 285 406 786 1536	1755	2116	1261	1557	1751	2351		
3507	637 1337	1567	1277	3317	1846	1699	1666	1663	1622
1469	1076	2181	1839	2143	2768	758 0	466 941 1431	1450	
2386									
1461	1541	1191	1181	1711	2260	2590	2820	1342	1062
1272	902 1312	632 1181	751 362 508 1288	1477	1868	795 1135			
1285	1885	3305	1103	1803	2033	1743	3783	1802	1451
1418	1415	1578	1425	1542	2647	2305	2609	3234	1224
466 0	1407	1897	1172	2138					
1267	1387	1144	934 727 2066	2396	2626	1932	1652	1931	
1504	2297	1617	1284	854 1347	1727	2477	2354	2914	1214
1224	1704	2214	3174	304 1004	1234	944 2984	2787	2640	
2607	2604	2563	2084	1645	2453	2111	2415	3040	1327
941 1407	0	541 2264	3184						
726 846 926 716 186 1525	1855	2085	1986	1820	2099	1796	2636		

1956	1146	1146	1686	2066	2816	2895	3396	1755	1765		
2245	2755	3224	845	1054	1284	554	3034	3126	2979	2946	
2943	2902	2252	1507	1912	1570	1874	2499	1495	1431		
1897	541	0	2730	3666							
2294	2374	2024	2014	2544	3093	3423	3653	2024	1744		
1944	1584	1984	1304	2014	1584	1044	664	1015	305	865	1050
1390	560	1125	3445	1960	2660	2890	2600	4640	2474	1178	
1145	1142	1703	2097	2342	3480	3138	3442	4067	2019		
1450	1172	2264	2730	0	1135						
3230	3310	2960	2950	3480	4029	4359	4589	2540	2370		
2350	2320	1870	1710	2950	2520	1980	1630	850	830	270	1970
2310	1480	1650	3970	2880	3580	3810	3520	5560	2360		
1013	980	977	1538	2462	2968	4416	4074	4378	5003	2695	
2386	2138	3184	3666	1135	0;						

enddata

min=@sum(weizhi(k):x(k)*gufei(k))+@sum(weizhi(j):@sum(weizhi(i):xuqiu(j)*C(i,j)*y(i,j)))*0.5;

@for(weizhi(i):@bin(x(i)));

@for(assign(i,j):@bin(y(i,j)));

@for(weizhi(j):@sum(weizhi(i):y(i,j))=1);

@for(weizhi(i):@for(weizhi(j):y(i,j)<=x(i)));

end

代码 4:

Demand_point = [

3639 2685

3712 2601

3488 2465

3326 2444

3238 2771

4196 2956

4312 3210

4386	3430
4177	1756
3918	1821
4061	1630
3780	1788
4029	1162
3676	1422
3715	2322
3429	2092
3507	1624
3394	1357
3439	799
2935	760
3140	450
2769	1508
2545	1643
2778	1174
2370	1025
1304	1688
3007	2030
2562	2244
2381	2324
2788	2509
1332	3305
4263	1069
3538	702
3470	696
3526	737
3928	971
4201	1603

```

4016    2285
4089    2613
4296    2920
4095    3374
4512    2710
3751    2055
3334    1893
3229    1633
3054    2290
3089    2749
3044    919
3053    261

```

```
];
```

```
Sending_point = [4, 7, 11, 20, 23, 26, 28, 45];
```

```
figure(1)
```

```
title('供应点到需求点的连线图')
```

```
coordinatex = Demand_point(Sending_point, 1);
```

```
coordinatey = Demand_point(Sending_point, 2);
```

```
% 供应点
```

```
plot(coordinatex, coordinatey, '^','LineWidth',2,...
```

```
    'MarkerEdgeColor','k',...
```

```
    'MarkerFaceColor','y',...
```

```
    'MarkerSize', 10)
```

```
hold on
```

```
% 需求点
```

```
plot(Demand_point(:,1),Demand_point(:,2),'o','LineWidth',1,...
```

```
    'MarkerEdgeColor','k',...
```

```
    'MarkerFaceColor','r',...
```

```
    'MarkerSize',5)
```

```
% 连线
```

```
A=[3 3 4 4 4 7 7 7 11 11 11 10 11 17 3 4 45 45 20 20 20 23 23 20 20 26 4 28 28 28 28 13 35 19 19 13 11 10
6 7 7 40 10 45 45 27 5 20 21];
```

```
for i=1:49
```

```
    x=[Demand_point(i,1),Demand_point(A(i),1)];
```

```
    y=[Demand_point(i,2),Demand_point(A(i),2)];
```

```
    plot(x,y,'k');hold on
```

```
end
```

```
hold on
```

```
% 标点
```

```
for i=1:49
```

```
    b=num2str(i);
```

```
    b=[' ',b];
```

```
    text(Demand_point(i,1),Demand_point(i,2),b);
```

```
end
```

代码 5

```
#include <stdio.h>
```

```
#define M 99999999
```

```
#pragma warning(disable:4996)
```

```
#define max 49
```

```
#define n 49
```

```
void printPath(int u, int v, int path[][max])
```

```
{
```

```
    if (path[u][v] == -1)
```

```
    {
```

```
        printf("<%d, %d>", u, v);
```

```
    }// 直接输出
```

```
    else
```

```
    {
```

```
        int mid = path[u][v];
```

```
        printPath(u, mid, path);
```

```

        printPath(mid, v, path);
    }
}

void Floyd(int m, int MGraph[][n], int path[][n])
{
    int i, j, v;

    int A[n][n];

    for (i = 0; i < m; i++)
    {
        for (j = 0; j < m; j++)
        {
            A[i][j] = MGraph[i][j];

            path[i][j] = -1;
        }
    }

    for (v = 0; v < n; ++v)
    {
        for (i = 0; i < n; ++i)
        {
            for (j = 0; j < n; ++j)
            {
                if (A[i][j] > A[i][v] + A[v][j])
                {
                    A[i][j] = A[i][v] + A[v][j];
                    path[i][j] = v;
                }
            }
        }
    }
}

```


{ 420, 360, 311, M, M, M, M, M, 840, 680, M, M, M, M, 0, 430, M, M, M, M, M, M, M, M, M, M, M, M,
M, M, M, M, M, M, M, M, M, 361, M, M, M, M, 349, M, M, M, M, M, M } ,

{ M, M, M, M, M, M, M, M, M, M, M, 540, M, 270, M, 540, 0, 380, M, M, M, M, M, M, M, M, M, M,
M, M, M, M, M, M, M, M, M, M, M, M, M, M, 406, 362, M, M, M, M },

{ M, M, M, M, M, M, M, M, M, M, M, M, M, 1020, 860, M, M, M, 780, 0, 710, 580, M, M, M, M, M, M, M, M, M, M, M, 130, 127, 688, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M },

$$\{ M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 580, 560, 0, M, M, M, M, M, M, M, M, M, \\ M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 270 \},$$
$$\{ M, 340, 0, M, 990, 2170, 920, M, \\ M, M \},$$
$$\{M, 820, M, 1090, 990, 650, 0, 2320, M, \\ M, M\},$$
$$\{ M, M, M, 630, M, M, M, M, M, M, M, M, M, M, M, 550, M, M, M, M, M, 910, 920, M, M, M, 0, 700, \\ M, 640, M, M, M, M, M, M, M, M, M, M, M, M, M, 637, M, 304, M, M, M \},$$
[illegible]


```

M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 0, 466, M, M, M, M },

    { M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 362, 508, M, M, M, 795, M, M, M, M, M, M,
M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 466, 0, M, M, M, M },

    { M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 304, M, M, M,
M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 0, 541, M, M },

    { M, M, M, M, 186, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M,
554, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 541, 0, M, M },

    { M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 664, M, 305, M, M, M, 560, M, M, M, M, M,
M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 0, M },

    { M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 270, M, M, M, M, M, M, M, M, M,
M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, M, 0 },

};

int main()
{
    int u, v;

    scanf("%d %d", &u, &v);

    Floyd(49, MGraph, path);

    printPath(u, v, path);

    return 0;
}

```