武漢理工大學

数学建模暑期培训论文

第 1 题

基于聚类分析退火模拟的最优树水管铺设模型

第 24 组

姓名	方向
刘畅（组长）	编程
尹可汗	编程
邓米乐	建模

2020 年 7 月 22 日

摘要

合理的铺设供水站之间的管道，对减少经济成本，降低供水功率和提高供水效率具有重要意义。本文建立了基于最小生成树和聚类分析的分层模型通过 **prime 算法、模拟退火**算法和**聚类分析**算法来对模型进行求解。

针对问题一，建立基于最小生成树的分层模型，求解从中心供水站到各个二级供水站铺设管道的长度总和的最小值。综合考虑供水站之间联通与否产生的决策变量、同级供水站之间和不同级供水站之间的连通问题产生的约束条件以及总的铺设管道最短长度的目标函数，建立了基于最短生成树的分层模型。本文采用最小生成树算法的分层化处理，最终通过 prime 算法求得自来水管铺设的最短距离以及对应的 I 型和 II 型水管的距离。本文最终通过图形给出的铺设方案，求得最短长度为 **524.345816 公里**，对应的 I 型管道总长度和 II 型管道总长度分别为 **120.941215 公里**和 **403.404601 公里**。

针对问题二，建立最优树 0-1 规划模型，确定需要升级的二级供水站，通过图形给出 II 型管道长度最小的铺设方案，并求出对应条件下较问题一的方案，II 型管道长度减少的值。考虑满足铺设 II 型管道的条件，建立最优树 0-1 规划模型。为稳妥起见，本文首先采用了组合数枚举法，但是由于组合数枚举法的算法效率不高，本文又采用模拟退火算法，使算法的时间复杂度降低。本文确定了问题二需要升级的二级供水站的供水站序号为 **90** 和 **126**，用图形的方式给出了 II 型管道长度最短的铺设方案，并求得在该方案下相较问题一的方案，II 型管道长度减少了 **11.403125**。最终对结果进行了灵敏度分析，考察二级供水站的选择对于 II 型管道总长度的影响。

针对问题三，建立带优先因子的多决策目标最优树模型，求解需要升级的二级供水站数量的最小值和在该条件下的管道总里程的最小值。对于限定从一级供水站出发铺设的管道长度为 40 公里的情况下，改变问题一模型中的约束条件和决策变量，并通过引入优先因子，重新构建目标函数，建立带优先因子的多决策最优树模型。本文通过基于最短距离的聚类分析方法和动态规划的组合迭代，求得需要升级的二级供水站数量的最小值为 **2**，并求出在该配置下铺设管道的总里程最小值为 **576.678449 公里**。对求解结果进行灵敏度分析，考虑二级供水站数量和铺设管道的总里程最小值之间的关系。

本文的优点：1. 采用最小生成树算法，简化了求解模型的难度。2. 在问题二求解方面，采用模拟退火算法和聚类分析，降低了算法的时间复杂度，兼顾了局部搜索和全局搜索能力。

关键词：　最小生成树　　水管铺设　　聚类分析　　模拟退火

目录

一、 问题重述

1.1 问题背景

本题以村村通自来水工程为背景[1]，某地区需建设的 1 个中心供水站，12 个一级供水站和 168 个二级供水站的坐标位置和相邻关系已给定，根据实际的管道供应量、供水站功率等情况设计出最优自来水管道铺设方案，实现把中心供水站的自来水通过管道输送到所有的一级供水站和二级供水站，且自来水管铺设方案满足以下自来水管铺设技术要求[2]：

1. 中心供水站只能和一级供水站连接（铺设 I 型管道），不能和二级供水站直接相连，一级供水站之间可以连接（铺设 I 型管道）；
2. 一级供水站可以与二级供水站相连（铺设 II 型管道），且二级供水站之间也可以连接（铺设 II 型管道）；
3. 各级供水站之间的连接管道必须从上一级供水站或同一级供水站的位置坐标出发，不能从任意管道中间的一点进行连接;
4. 相邻两个供水站之间（如果有管道相连）所需管道长度可简化为欧氏距离。

1.2 待解决的问题

问题一：设计自来水管铺设方案使得从中心供水站出发的管道总里程最小，并且铺设方案需要以图形的形式表达，计算出 I 型管道和 II 型管道总里程数。

问题二：由于 II 型管道供给量不足，需要选取两个二级供水站升级为一级供水站以减少从一级供水站出发铺设的 II 型管道，设计使 II 型管道总里程最少的铺设方案并给出需要升级的二级供水站，计算该方案中 II 型管道的总里程相对问题 1 方案 II 型管道的总里程减少的公里。

问题三：由于功率限制，从一级供水站出发铺设的管道输送里程最多 40 公里，在满足对所有供水站都供水的条件下，计算出最少升级的二级供水站个数，并在这种配置下计算管道铺设的最少总里程数。

二、 模型假设

1. 忽略地质因素对铺设的管道长度影响，即管道长度为供水站间的直线距离；
2. 假定问题一和问题三中 I 型管道和 II 型管道供给量充足
3. 假定问题三中从一级供水站出发的管道都能供水 40 公里
4. 假定问题三中不考虑一级供水站与一级供水站联通时对功率的影响

三、 符号说明

符号	含义
m_i	第 i 个结点
(x_i, y_j)	m_i 的横坐标何纵坐标
m_1	中心供水站
$V\{m_2, m_3, ..., m_{13}\}$	I 级供水站集合
$P\{m_{14}, m15, ..., m_{181}\}$	II 级供水站集合
d_{ij}	节点 i 与节点 j 的距离

注：表中未说明的符号以首次出现处为准

四、 问题一模型的建立与求解

4.1 问题分析

在对问题提出解决方案之前，我们首先将各级供水站进行图论的抽象，以便方便问题的解决。我们将各级供水站位置图抽象为图论中的无向赋权图[3]。已知各级供水站的位置坐标及其邻接关系，可以抽象出无向赋权图 $G(V, E)$。其中 V 为该无向赋权图中的顶点集，由中心供水站 $\{m_1\}$,I 级供水站集合 V =$\{m_2, m_3, ..., m_{13}\}$,II 级供水站集合 P =$\{m_{14}, m15, ..., m_{181}\}$ 组成, 各节点的坐标用 (x_i, y_j)。E 为该无向赋权图的边集，代表着各相连供水站之间的自来水管，同时，两联通供水站之间的自来水管里程为其对应边的权值。用 d_{ij} 表示节点 i 与节点 j 的距离。

问题一要求我们设计管道铺设方案使得从中心供水点 A 出发的管道总里程最少，分别计算出 I 型管道和 II 型管道的总里程，并且铺设方案需要以图形的形式表达。

从工程铺设的角度铺设出发，本问题的要求只需要建立好一个所有顶点都连通的最小连通图即可，所以其最短的路程则主要是有最小生成树来获得，而已有一些现存文献对管道铺设的动态规划方法研究[4] 较为明确。对于本题，由于自来水管铺设技术有一定的技术要求，且中心供水站、I 级坐标、II 级坐标的坐标位置已给定，可以将供水站位置图抽象为图论中的无向赋权图，由于题目最后使得自来水管铺设总里程最小，可以转换为在一个网络规划中，从一个起点出发到所有接点，找出一条或几条路线，以使在这样一些路线中所采用的全部支线的总长度最小的规划问题，最终转换为最小生成树问题求解[5]。

4.2 分层的最小生成树模型建立

4.2.1 决策变量的确定

自来水管道从中心供水点出发，并且管道铺设的技术要求已知，只可能存在以下 4 种铺设路径：中心供水站到一级供水站、一级供水站到一级供水站、一级供水站到二级供水站、二级供水站到二级供水站。存在有的供水站联通，有的供水站则没有联通的情况。因此我们引入变量 l_{ij} 表示 m_i 之间联通与否

$$l_{ij}=\begin{cases}1, & \text{节点 i 与节点 j 联通}\\ 0, & \text{节点 i 与节点 j 不联通}\end{cases} \tag{1}$$

4.2.2 目标函数的确定

需要求出使自来水管总里程最短的管道铺设方案，因此我们以管道总里程最短为目标，由于管道都是从中心供水站出发，且每个供水站都要供水，即寻找一条从起始点 1 到个节点生成的最优树，要求各线路的权值和最小，故目标函数为：

$$\min Z=\sum_{i=1}^{181}\sum_{j=1}^{181}d_{ij}l_{ij} \tag{2}$$

其中 d_{ij} 表示为节点 i 到节点 j 的欧式距离，其计算公式为

$$d_{ij}=\sqrt{(x_i-x_j)^2+(y_i-y_j)^2} \tag{3}$$

4.2.3 约束条件的分析

在得到目标函数后，我们对目标函数的约束进行分析。

(1) 中心供水站至少与一个一级供水站相连通，即至少有一条边从中心站点出发到一级供水站，故有：

$$\sum_{\mathrm{j}=2}^{13}l_{ij}\geq 1 \tag{4}$$

(2) 一级供水站至少与一个二级供水站相联通，即至少有一条边从一级供水站出发到二级供水站

$$\sum_{\mathrm{i}=2}^{13}\sum_{j=14}^{181}l_{ij}\geq 1 \tag{5}$$

(3) 除中心供水站外的其他站点有且仅有一个供水站给其供水，即除中心供水站外其他站点有且仅有一条边进入该节点

$$\sum_{k=1,k\neq i}^{181}x_{ki}=1 \tag{6}$$

(4) 因为供水是单方向的，因此所有节点不出现圈

$$m_i - m_j + 181x_{ij} = 180 \tag{7}$$

4.2.4 最短自来水管铺设总里程模型

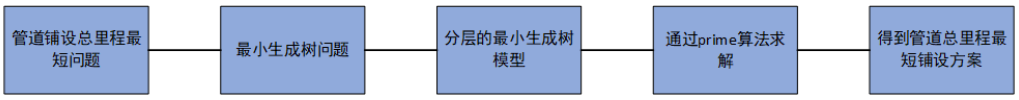

图 **1**　问题一思维导图

$$\min Z = \sum_{i=1}^{181}\sum_{j=1}^{181} d_{ij}l_{ij} \tag{8}$$

$$\begin{cases} \sum\limits_{j=2}^{13} l_{ij} \geq 1, \\ \sum\limits_{i=2}^{13}\sum\limits_{j=14}^{181} l_{ij} \geq 1, \\ \sum\limits_{k=1,k\neq i}^{181} x_{ki} = 1, \\ m_i - m_j + 181x_{ij} = 180. \end{cases} \tag{9}$$

4.3 模型求解

4.3.1 最小生成树分层化处理

针对问题一提出的模型，其实我们可以看出来该问题中除了供水中心 m_1 与 II 级区域供水站之间不能够进行连通外，其余的顶点之间都是可以互相连通的，而铺水管的问题对于我们来说，是需要得到一个最小的连通图而不需要构成一个回路，而且每个供水站不存在需求不足的情况，所以每个站点只需要有路径连通就可以了。

分析题目可知，该问题还要求 II 级的区域的管道需要由一级供水站铺设，I 级管道只能由中心和其他 I 级管道相连通，所以需要将铺设路径进行分层化处理，首先在 I 区与供水中心 m_1 之间通过最小生成树模型算法将其连接完成，然后再在 I 级连通最小生成树的基础上进行 II 级的最小生成树连通，从而实现讲一个连通图，转换成两个部分的最优树路径之和，最终获得一个完整的最优树连通图。

4.3.2 prime 算法实现

在实现最小生成树模型的常用算法上，我们有两种常用的算法——kruskal 和 prime，通过分析模型的数据可知，该图中点的数模较多，要连接和迭代过程模拟尝试的路线已

经超过了 $\frac{n(n-1)}{2}$ 的范畴，因此该问题属于稠密图的模型，所以我们需要采用 prime 算法来实现该问题的解答。

prime 算法的原理主要是利用并查集的思想，将已经连通的点和未连通的点进行分离两个集合来管理，这里我们将已经连通的点记为 A，同时我们将未连通的点集合记为 B，然后将这两个集合中所存放的元素两两进行距离运算，得到 A 集合已知点序列与 B 中每个未知点的距离矩阵 D，然后我们求出 B 中序列与之对应的最近的距离，然后添加该点的坐标加入到 A 集合中，同时在 B 中删除该元素实现更新。当 B 中不存在元素即该图已经完全连通。

这里我们将所有点构成的集合为 $G = \{A, B\}$,A 表示已经连通的点的集合 $A = \{m_i, m_{i+1}...m_j\}$，B 表示未连通的点的集合 $B = G - A$

$$D = distance(A, B), if B \neq \varnothing \tag{10}$$

$$m_k = agmin(\min(D)) \quad then \quad update \quad A, B \quad m_k \in A, B = G - A \tag{11}$$

这样我们得到的 m_k 就是我们需要新加入的与当前 A 集合最相近的点，然后我们再更新我们的 j 集合使得 $m_k \in A, B = G - A$ 直到 $B = \varnothing$。下面是分层的 prime 算法伪代码表述

通过该程序的伪代码我们可以看到该程序在并查集优化空间结构优化下，目前的算法复杂度为 $O(n^2)$

Algorithm 1 分层 prime 算法——中心与 I 供水站区域

Input: $distancematrix$ 距离矩阵，$lengthone$I 区域长度

Output: $totalroute$ 总的路径,$totaldist$ 总路程

function PRIMEONE($distancematrix$,$lengthone$)
    $totaldisone \leftarrow 0$,$routeone \leftarrow []$
    $A \leftarrow$ the point of $distancematrix$ in [1,lengthone]
    $B \leftarrow$ the point of $distancematrix$ in [lengthone,end]
    while B is not empty **do**
        $pv \leftarrow min(min(d(A, B)))$ => $p \leftarrow$ find pv in $distancematrix$
        if p **then** is in B and is not in A
            $A \leftarrow$ p, $B \leftarrow$ delete p
            $totaldisone$+= pv's dist $routeone \leftarrow pv$'s edge
        end if
    end while
    return $routeone$,$totaldisone$
end function

Algorithm 2 分层 prime 算法——I 与 II 供水站区域

function PRIMETWO($distancematrix, routeone, totaldisone$)
  $A \leftarrow$ the point of $distancematrix$ in routeone
  $B \leftarrow$ the point of $distancematrix$ setdiff in $routeone$
  $totaldistwo \leftarrow 0$
  $routetwo \leftarrow []$
  while B is not empty **do**
    $pv \leftarrow min(min(d(A, B)))$
    $p \leftarrow$ find pv in $distancematrix$
    if p **then** is in B and is not in A
      $A \leftarrow$ p
      $B \leftarrow$ delete p
      $totaldistwo$+= pv's dist
      $routetwo \leftarrow pv$'s edge
    end if
  end while
  $totaldist = totaldisone + totaldistwo$
  $totalroute \leftarrow routeone$ with $routetwo$
  return $totalroute$,$totaldist$
end function

4.4 结果分析

问题一中，自来水管的铺设实际上是求从中心供水站 A 由一级供水站 V 最终到各个二级供水站 P 的最短距离，其结果如图 2 所示。

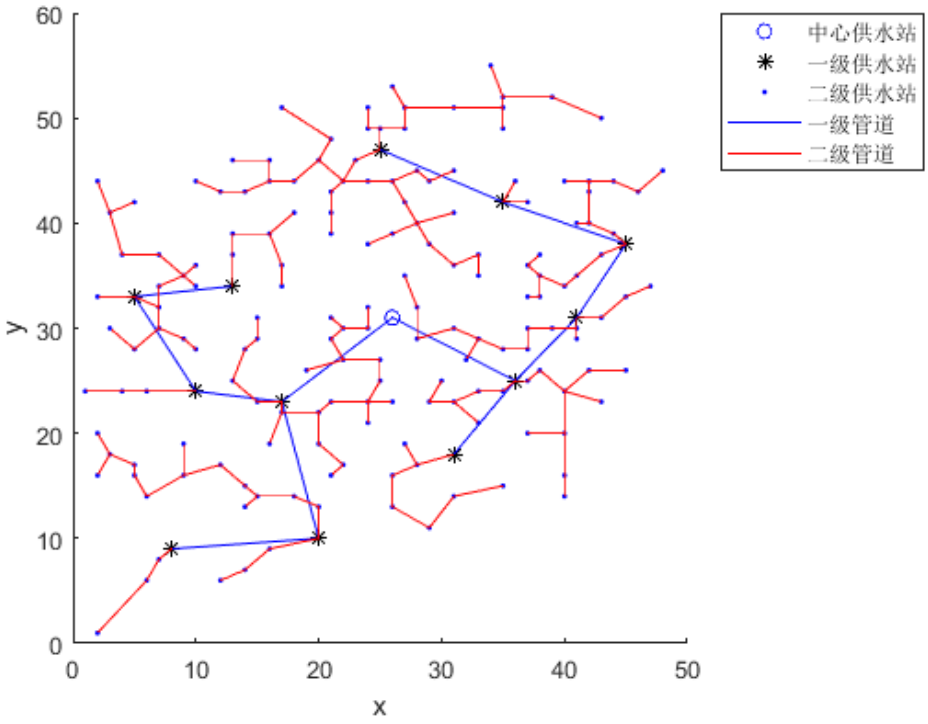

图 **2**　问题一水管铺设结果图

在这种情况下，可以求得 I 型管道总长度为 **120.941215km**，II 型管道总长度为 **403.404601km**，如表 1 所示。

表 **1**　问题一数值解

管道类型长度	管道长度 (公里)
I 型	120.941215
II 型	403.404601
总长度	524.345816

五、 问题二模型的建立与求解

5.1 问题分析

针对问题二，由于 II 型管道市场供应不足，需要选取两个二级供水站升级为一级供水站使得 II 型管程总里程最少，并且给出铺设方案以及计算减少的距离。

问题二相比于问题一，需要选取 2 个二级供水站升级为一级供水站，其次目标发生改变，第一问中目标是管道铺设的总里程最短即 I 型管道加上 II 型管道的总里程最短，而在第二问中目标是 II 型管道的总里程最小，但是这并不影响该问题的本质即最小生成树问题。因此，我们可以在第一问模型的基础上再建立水站升级最优树 0-1 规划模型。

5.2 水站升级最优树 0-1 规划模型建立

假设二级供水站 m_a,m_b($14 \leq a, b \leq 181$ 且为整数) 被升级为一级供水站。

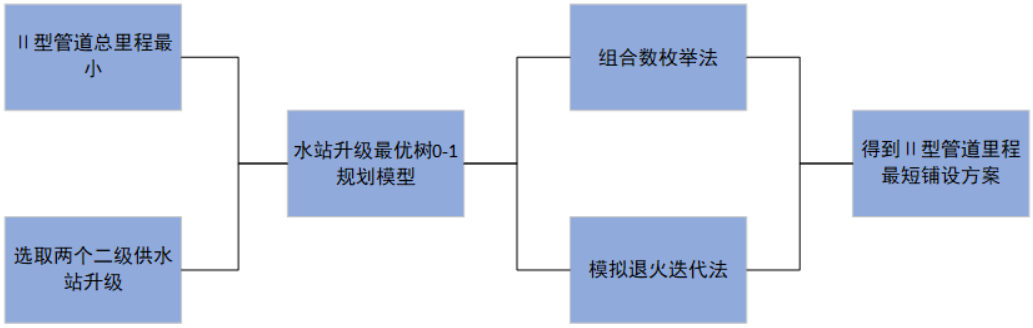

图 **3**　问题二思维导图

5.2.1 决策变量分析

问题二的目标相对于问题一而言，没有算入 I 型管道的里程，引入 $0-1$ 变量 h 使得一级供水站间联通的 I 型管道里程剖出。

$$h = \begin{cases} 0, & \text{一级供水站间联通} \\ 1, & \text{一级供水站与二级供水站或者二级供水站间联通} \end{cases} \tag{12}$$

5.2.2 目标函数的确定

题目要求出使得 II 型管道总里程最小的供水站升级和铺设方案，需要从第一问中总里程中剖去 I 型管道里程，即中心供水站与一级供水站之间的管道和一级供水站与一级供水站间的管道里程。通过引入 $0-1$ 变量 h 和控制 i,j 起始的取值，使得 I 型管道剖去，得到以下目标函数：

$$\min Y = \sum_{i=2}^{181} \sum_{j=2}^{181} h d_{ij} l_{ij} \tag{13}$$

5.2.3 约束条件分析

由于第二问本质上仍然是最小生成树问题，所以在约束条件的意义上相同，在形式上略有差异。约束条件所表示的含义分别为：至少有一条边从中心站点出发到一级供水站；至少有一条边从一级供水站出发到二级供水站；除中心供水站外其他站点有且仅有一条边进入该节点；所有节点不出现圈。

$$\begin{cases} \sum\limits_{j=2}^{13} l_{ij} + l_{1a} + l_{1b} \geq 1, \\ \sum\limits_{i=2}^{13} \sum\limits_{j=14}^{181} l_{ij} + \sum\limits_{j=14}^{181} l_{aj} + \sum\limits_{j=14}^{181} l_{bj} \geq 1, \\ \sum\limits_{k=1,k\neq i}^{181} x_{ki} = 1, \\ m_i - m_j + 181 x_{ij} = 180. \end{cases} \tag{14}$$

5.3 模型求解

5.3.1 组合数枚举法

根据我们第二问的模型和问题，我们可以明白其实该问题主要是为了降低 II 区域的水管路程进行的升级，所以为了达到该目的我们最稳妥的办法其实就是将图层中可能的结果进行统一的遍历，从而找到使得路程最小的那两个升级点，在现有的 12 个 I 级区域的基础上，我们还剩下了 168 个 II 管道区域供水站，那么所有的排列数则是有 $C_{168}^{2} = 14028$ 种情况，我们将第一问的分层算法通过微调可以实现该算法，不过由于遍历 prime 算法的复杂度已经是达到了平方级别，如果我们再加上该层算法的话，则目前的算法复杂度已经达到了 $O(n^3)$

Algorithm 3 组合枚举求最优解

Input: $distancematrix$ 距离矩阵

Output: $bestrouteonelist$ 最优 I 管道序列 $minroute$ 最短总路径,$mintotaldis2$ 最优 II 管道路程

function LOWCOSTPRIME($distancematrix$)
  $mintotaldis \leftarrow inf$,$minroute \leftarrow []$
  while $iter < C^2_{count(B)}$ **do**
    $newrouteonelist = changeroute(prerouteonelist)$
    $[routeone, distone] = primeone(distancematrix, length(newrouteonelist))$
    $[routetwo, disttwo] = primetwo(distancematrix, routeone)$
    $totaldis2 = disttwo$
    if $mintotaldis2 < totaldis2$ **then**
      update $bestrouteonelist$ $minroute$ $mintotaldis2$
    end if
    $iter++$
  end while
  return $bestrouteonelist$, $minroute$,$mintotaldis2$
end function

5.3.2 模拟退火迭代法

由于在第一种枚举方法的尝试下，我们得到了当前的最优解，不过该算法的复杂度其实比较高，运算的时间也比较长，所以我们选用模拟退火算法对问题进行迭代，从而使得回溯和时间复杂度降低。该问题与算法的对应关系如2所示 在选择好合适的初温和

表 **2**　问题二模型与模拟退火算法对应关系

模型要素	算法抽象
Y	分子集合能量
$minY$	最优解供水站序列
可行解的范围与更新	扰动函数与 metropolis 法则
迭代控制	temperature 和 coolingrate 冷却时间

可行解邻域后，我们这里使用了模拟退火常用的降温公式

$$T(t + \Delta t) = \alpha t \qquad (\alpha = 0.95) \tag{15}$$

使用 metropolis 法则迭代和更新当前最优解序列的概率

$$P(list => list') = \begin{cases} 1, & \text{Y(list')<Y(list)} \\ e^{-\frac{Y(list')-Y(list)}{T}}, & \text{Y(list')>Y(list)} \end{cases} \tag{16}$$

其中 $list$ 表示当前 I 级水站的序列 $list'$ 表示新的 I 级水站的序列，Y 表示模型二中的目标函数。

Algorithm 4 模拟退火

Input: $distancematrix$ 距离矩阵

Output: $bestrouteonelist$ 最优 I 管道序列 $minroute$ 最短总路径,$mintotaldis2$ 最短优 II 管道路程

function LOWCOSTPRIME($distancematrix$)
  $mintotaldis \leftarrow inf$,$minroute \leftarrow []$
  $Initail \quad temperature$ and $coolingrate$
  while $temperature > 0$ **do**
    $newrouteonelist = changeroute(prerouteonelist)$
    $[routeone, distone] = primeone(distancematrix, length(newrouteonelist))$
    $[routetwo, disttwo] = primetwo(distancematrix, routeone)$
    $totaldis2 = disttwo$
    if $mintotaldis2 < totaldis2$ or $rand < exp(\Delta dist/(temperature)$ **then**
      update $bestrouteonelist\ minroute\ mintotaldis2$
    end if
    $temperature- = temperature * coolingrate$
  end while
  return $bestrouteonelist$, $minroute$,$mintotaldis2$
end function

5.4 结果分析

5.4.1 模型总结和结果展示

根据目标函数对 II 型管道长度所求，我们通过随机选取两个二级供水站升级为一级供水站, 并通过模拟退火迭代法，使得目标函数的解通过不断的迭代最终收敛为一个全局最优解，最终我们可以得到管道的铺设图且

需要升级的二级供水站的坐标已经显示在坐标轴上，其结果如图 4 所示。

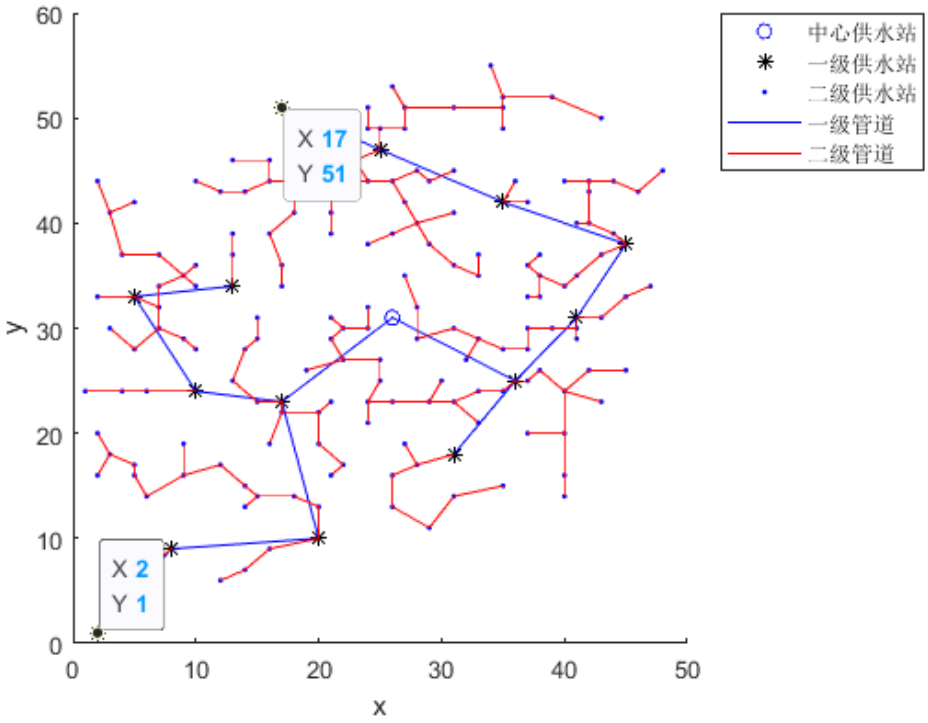

图 **4**　问题二水管铺设图 **(**含升级的二级供水站坐标**)**

则两个需要升级的二级供水站的结果可以由表 5 所示。

表 **3**　问题二升级的二级供水站

管道序号	供水站编号	X 坐标	Y 坐标
90	II 型 (P_{77})	17	51
126	II 型 (P_{113})	2	1

升级选定的两个二级供水站后，由最小生成树算法可以得到在这种情况下 II 型管

道的长度为 **392.001476km**, 具体结果如表 4 所示。

表 **4**　距离长度对比

距离类型	距离长度 (公里)
问题一中 II 型管道长度	403.404601
问题二中 II 型管道长度	392.001476
问题二较问题一减少的长度	11.403125

5.4.2 灵敏度分析

为了检验修改温度和迭代过程 I 区域对于方案变化的影响，现通过修改在不同迭代和序列下的方案结果对方案最小生成树计算的影响，我们可以得出，下面每个每隔一百次迭代的均值变化情况以及迭代过程最小代价展示。

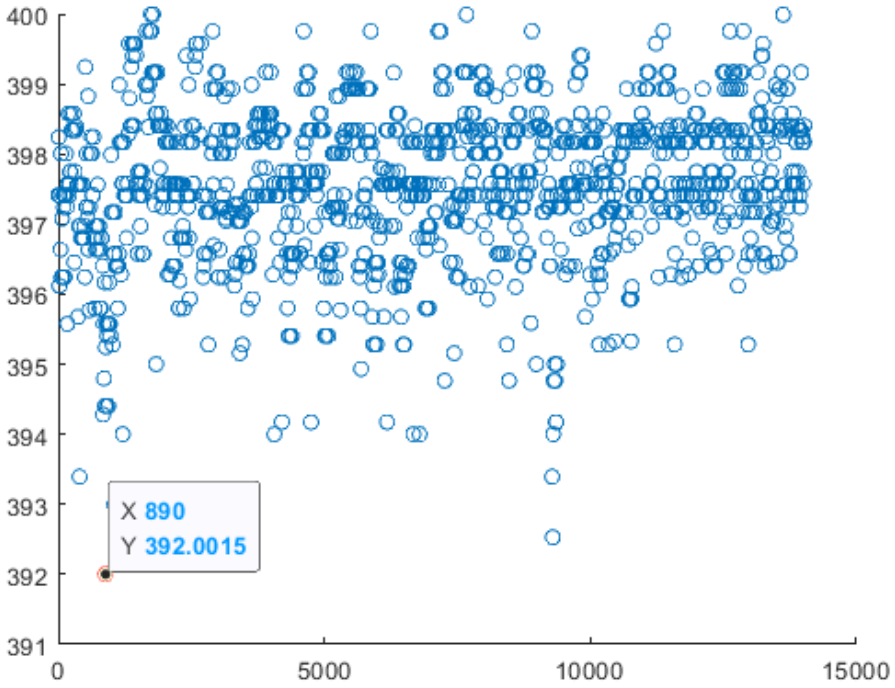

图 **5**　迭代过程最低点位置展示

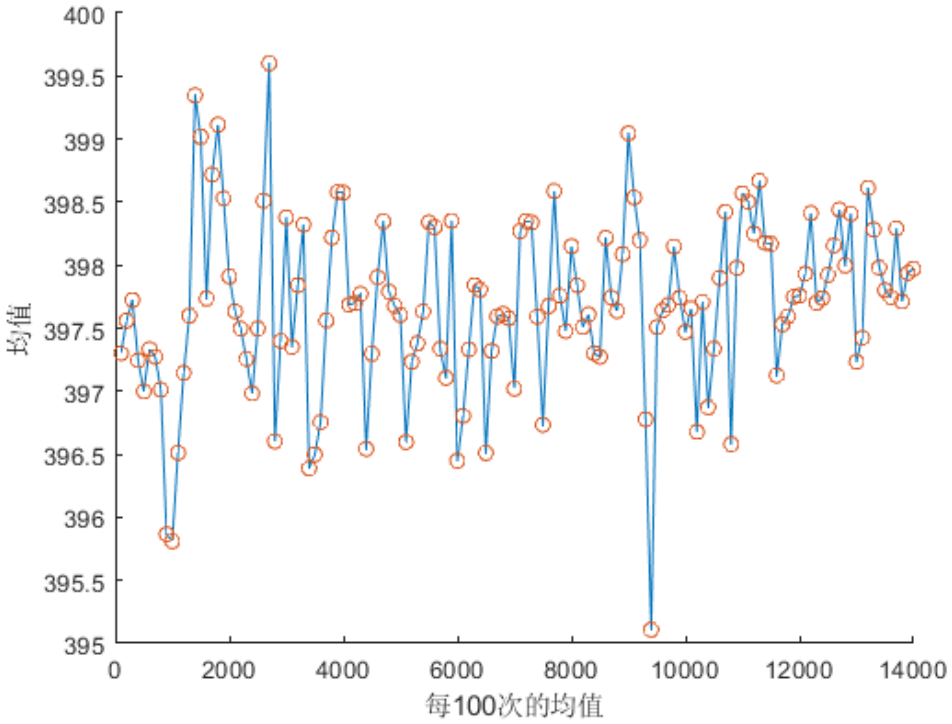

图 **6**　每隔 **100** 次方案最短距离的结果均值变化图

表 **5**　枚举与退货模拟算法比较

算法类型	运行时长	结果
组合枚举	9min	392.001476
退货模拟	2min	392.001476 395.271703

六、 问题三模型的建立与求解

6.1 问题分析

问题三在问题一的基础上增加了功率约束，导致从一级供水站出发铺设的管道最多只能供水 40 公里。正如问题假设中所阐述的，在此问题中，我们忽略掉一级供水站与一级供水站之间铺设管道对一级供水站功率的影响，计算从一级供水站出发的管道时只考虑一级供水站与二级供水站之间的管道和二级供水站两两之间的管道，即 II 型管道。在满足这个约束条件下，第一步求出最少升级的二级供水站个数，第二步再在这个配置下求出管道铺设的最小总里程，建立带优先因子的多决策目标最优树模型。

6.2 带优先因子的多决策目标最优树模型建立

假设升级 k 个二级供水站，设一级供水站集合为 $\{V_1, V_2, ..., V_{k+12}\}$, 由一级供水站 V_i 输送自来水的二级供水站集合为 $U_i\{p_{i1}, p_{i2}, ..., p_{ic}\}$，以此来对供水站进行分类。

6.2.1 决策变量分析

为了使没有联通的一级供水站与二级供水站、没有联通的二级供水站与二级供水站之间的距离不算进入，故引入 0-1 变量 h_1,h_2。

$$h_1 = \begin{cases} 1, & \text{一级供水站与二级供水站联通} \\ 0, & \text{一级供水站与二级供水站不联通} \end{cases} \tag{17}$$

$$h_2 = \begin{cases} 1, & \text{二级供水站与二级供水站联通} \\ 0, & \text{二级供水站与二级供水站不联通} \end{cases} \tag{18}$$

6.2.2 目标函数确定

题目要求是先求出最少升级的二级供水站个数，之后在此配置下求出最少管道总里程，是多层次分步目标。因此引入优先因子 P_1,P_2, 分别表示第一级优先因子、第二级优先因子，先优先算出最小升级二级供水站数量，然后算出最小管道总里程，目标函数如下：

$$\min D = P_1\mathrm{k} + P_2 d_{ij} l_{ij} \tag{19}$$

6.2.3 约束条件分析

(1) 由于从每一个一级供水站出发的管道里程最多为 40 公里，将一级供水站各自分为一类，分别计算所供水区域中 II 型管道的里程使之在 40 公里以内, 故有：

$$\sum_{p_{ij} \in U_i} h_1 d(V_i, p_{ij}) + \sum_{p_{im}, p_{in} \in U_i} h_2 d(p_{im}, p_{in}) \leq 40 \tag{20}$$

(2) 中心供水站至少与一个一级供水站相连通，即至少有一条边从中心站点出发到一级供水站，故有：

$$\sum_{\mathrm{j}=2}^{13+\mathrm{k}} l_{ij} \geq 1 \tag{21}$$

(3) 一级供水站至少与一个二级供水站相联通，即至少有一条边从一级供水站出发到二级供水站。

$$\sum_{\mathrm{i}=2}^{13+k} \sum_{j=14}^{181} l_{ij} \geq 1 \tag{22}$$

(4) 除中心供水站外的其他站点有且仅有一个供水站给其供水，即除中心供水站外其他站点有且仅有一条边进入该节点。

$$\sum_{k=1,k\neq i}^{181} x_{ki} = 1 \tag{23}$$

(5) 因为供水是单方向的，因此所有节点不出现回路。

$$m_i - m_j + 181x_{ij} = 180 \tag{24}$$

6.2.4 带优先因子的多决策目标最优树模型

$$\min D = P_1 \mathrm{k} + P_2 d_{ij} l_{ij} \tag{25}$$

$$\begin{cases} \sum\limits_{p_{ij}\in U_i} h_1 d(V_i, p_{ij}) + \sum\limits_{p_{im},p_{in}\in U_i} h_2 d(p_{im}, p_{in}) \leq 40 \\ \sum\limits_{j=2}^{13+k} l_{ij} \geq 1, \\ \sum\limits_{i=2}^{13+k} \sum\limits_{j=14}^{181} l_{ij} \geq 1, \\ \sum\limits_{k=1,k\neq i}^{181} x_{ki} = 1, \\ m_i - m_j + 181x_{ij} = 180. \end{cases} \tag{26}$$

6.3 模型求解

6.3.1 基于最短距离聚类分析

由于每个 I 供水站附加上了对 I 和 II 之间，II 和 II 之间的 40km 距离限制, 因此使得我们不能再直接使用模型一和模型二中的相关函数进行操作，但我们的目的仍然是要将使得所有的点都能在铺设在最短距离中，且满足 40km 之内，这里我们不妨在 40km 的距离限制下对当前所有设定的 I 区域进行聚类，让所有的 I 供水站点在 40km 的范围下，获取到他们可以拿到的可以拿到的所有点，从而将整个地图分成 k 个中心点的集合，从而使得铺设所有二级的地图，如下图8所示。这里我们仍然用所有点构成的集合为 $G = \{A, B\}$,A 表示已经连通的点的集合 $A = \{m_i, m_{i+1}...m_j\}$，B 表示未连通的点的集合 $B = G - A$，我们利用前面计算距离矩阵的公式 (10) 得到

$$Distmatrix = distance(A, B) \quad -> \quad C = agmin(Distmatrix) \tag{27}$$

而 C 则是我们得到的含有所有 B 区域中的点被聚类后属于的类别 m_i，它一定满足 $m_i \in A$，从而实现了将每个 II 区域点映射到 I 的点集结果。

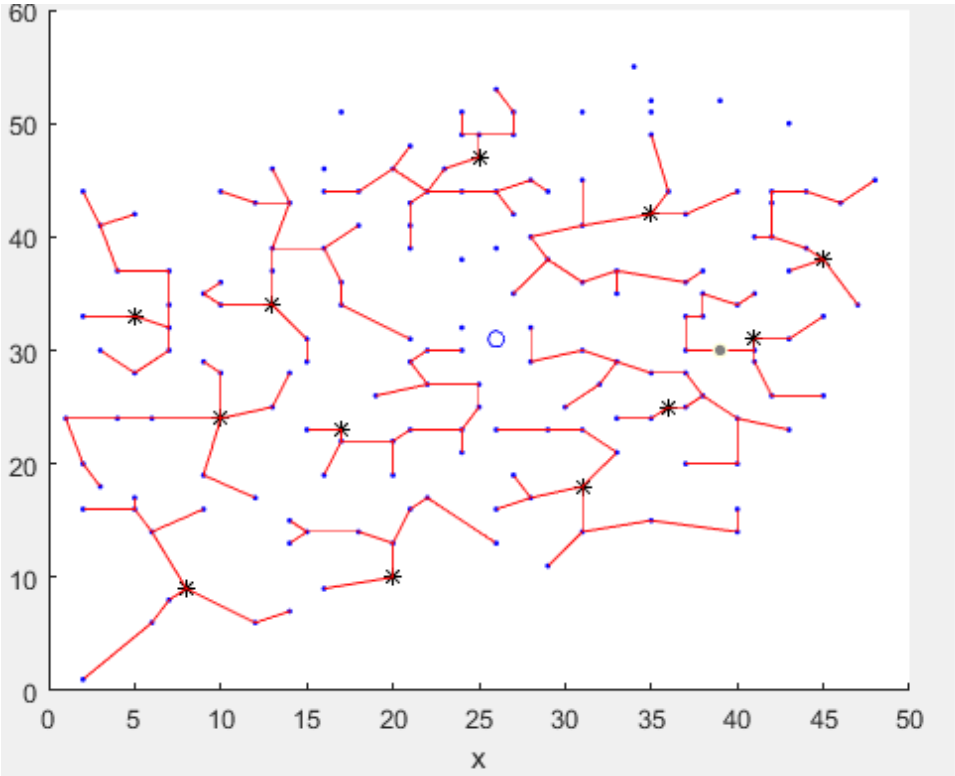

图 **7**　升级前聚类后出现部分 **II** 级点集无法铺设管道情况图

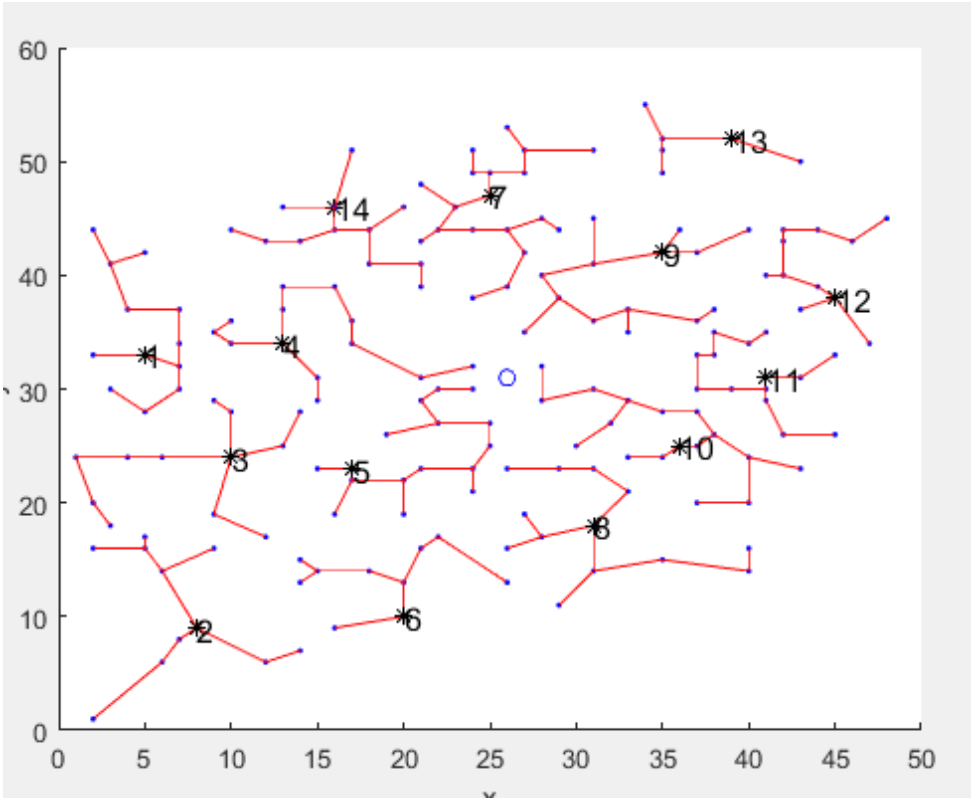

图 **8**　升级部分供水站聚类后铺设的所有 **II** 级管道连接线路

6.3.2 动态规划组合迭代

本问题最终是需要得到一个整体的最小铺设路径，这里我们采用将之前聚类的结果与中心点的最小生成树组合搭配最终得到的路线图，这里为了确定最小的路径和需要升级的个数 k_{min} 我们需要未能铺设的路段进行升级改造从而保证在最小工程量下实现结果的最优化。

在这种配置情况下，水管铺设的图如图 9 所示。

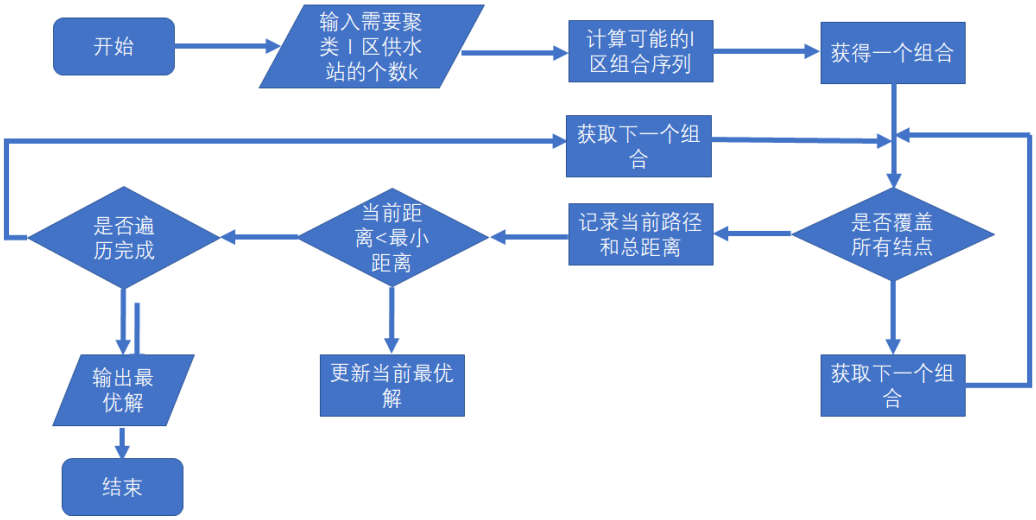

图 **9**　问题三算法流程图

6.4 结果分析

6.4.1 结果展示

由题目可知，题目所求为二级供水站需要升级为一级供水站的最少数目。因此，二级供水站对应于一级供水站的数目越多，相应的需要升级的二级供水站数目越少。所以，对问题三模型的求解首先将二级供水站根据约束条件对应到现有的一级供水站里，形成初始解的簇，然后再通过组合迭代的方法将选取新的二级供水站升级为一级供水站去聚类成新的簇，最终聚类结果如表 8 所示。

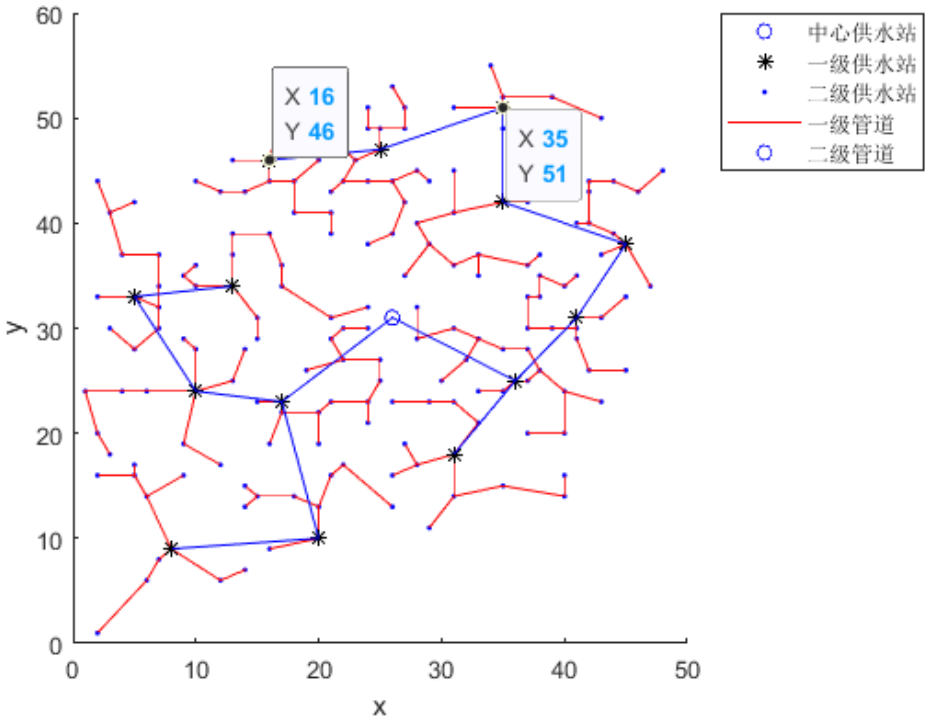

图 **10**　问题三水管铺设图 **(**含升级的二级供水站坐标**)**

表 **6**　问题三距离结果

管道长度类型	距离长度 (公里)
I 型管道	138.586590
II 型管道	438.091859
总长度	576.678449

表 **7**　需要升级的二级供水站信息

要升级的二级供水站编号	供水站序号	X 坐标	Y 坐标
P_{43}	56	16	46
P_{49}	62	35	51

表 **8**　聚类结果 **(**含升级的二级供水站**)**

供水站编号	供水站序号
V_1	93;94;95;96;97;98;104;107;108;109;110
V_2	37;39;100;119;126;127;128;129;130;131
V_3	99;101;102;103;105;106;120;123;124;125;146
V_4	23;31;43;74;75;79;91;92;111;112;144;145
V_5	22;24;28;29;30;113;114;122;132;137;138;139;140;141;143
V_6	35;36;38;40;41;115;116;121;133
V_7	45;47;49;53;54;57;65;66;70;72;73;78;85;86;87;88;89
V_8	117;118;134;135;136;142;149;155;158;161;162;163
V_9	17;18;19;20;34;46;48;50;58;60;68;69;71
V_{10}	21;25;147;148;150;153;154;159;160;164;165;166;167;169;170;171
V_{11}	14;15;16;32;33;152;156;157;168;172;173;174;175
V_{12}	26;27;51;59;151;176;177;178;179;180
P_{43}	42;44;55;76;77;80;81;82;83;84;90
P_{49}	52;61;63;64;67;181

由上面的聚类结果可以得到，需要升级的二级供水站的编号 **56** 和 **62**, 其具体的信息如表 7 所示。

可以求得如图 10 所示情况下，管道铺设的总距离为 **576.678449**,I 型管道和 II 型管道的长度分别为 **138.586590** 和 **438.091859**。具体结果汇总如 6 表所示。

6.4.2 灵敏度分析

对于问题三中的问题我们需要对问题中的升级数量 k 的变化进行分析，让 k 值在未连接区域中的点中进行组合和迭代从而寻找出我们需要的最低点标准，同时计算出该问题的均值, 可以看出尽管我们升级的个数越多，二级的消耗路程越少，但是我们从整体考虑的话还是 2 的时候最省。

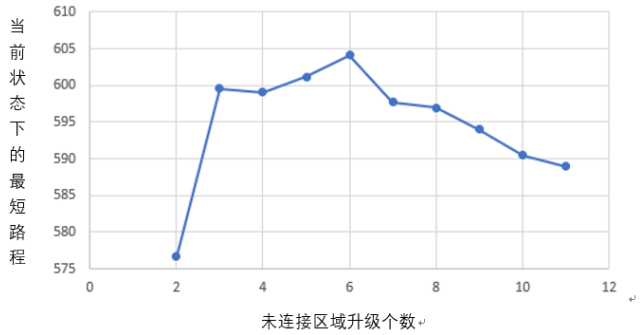

图 **11**　升级个数与当前最短路程关系

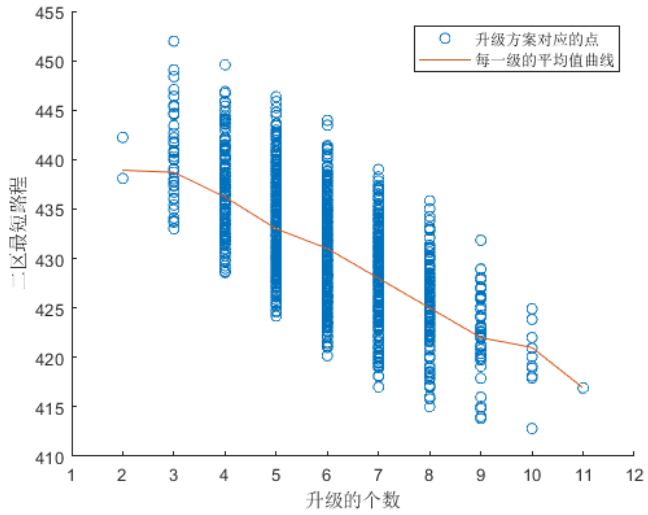

图 **12**　散点图和均值曲线

七、 模型评价

7.1 模型的优点

1. 本文充分考虑了实际中水管铺设相关的各个因素，设置的假设较为合理。

2. 本文将水管铺设的规划问题转化为了由中心供水站到各个二级供水站的最短路径问题，降低了解题难度。

3. 本文在解决问题二时，采用模拟退火迭代法，能够解决最小生成树算法在该题情况下容易陷入局部最优解的问题。

7.2 模型的缺点

1. 本文在解决问题三时，从各个一级供水站周围选取最短路径的二级供水站作为同一类，使得结果容易陷入局部最优解。

2. 本文所做的数学模型，不利于在一般情况下，分层设置不同级别的供水站，和实

际情况有一定差距。

7.3 改进与展望

1. 针对解决问题三基于最短距离的聚类分析算法易陷入局部最优解的问题，由参考文献可得，可以使用 Delaunay 三角剖分算法来对求解问题三模型进行改进。如图 1 所示，可以将所有的点通过 Delaunay 三角剖分算法将其全部转化成三角形，再将其闭合的部分删去，可以通过约束条件来约束其分割的簇的数量，其中 Delaunay 三角剖分算法的效果图如图 13 所示。

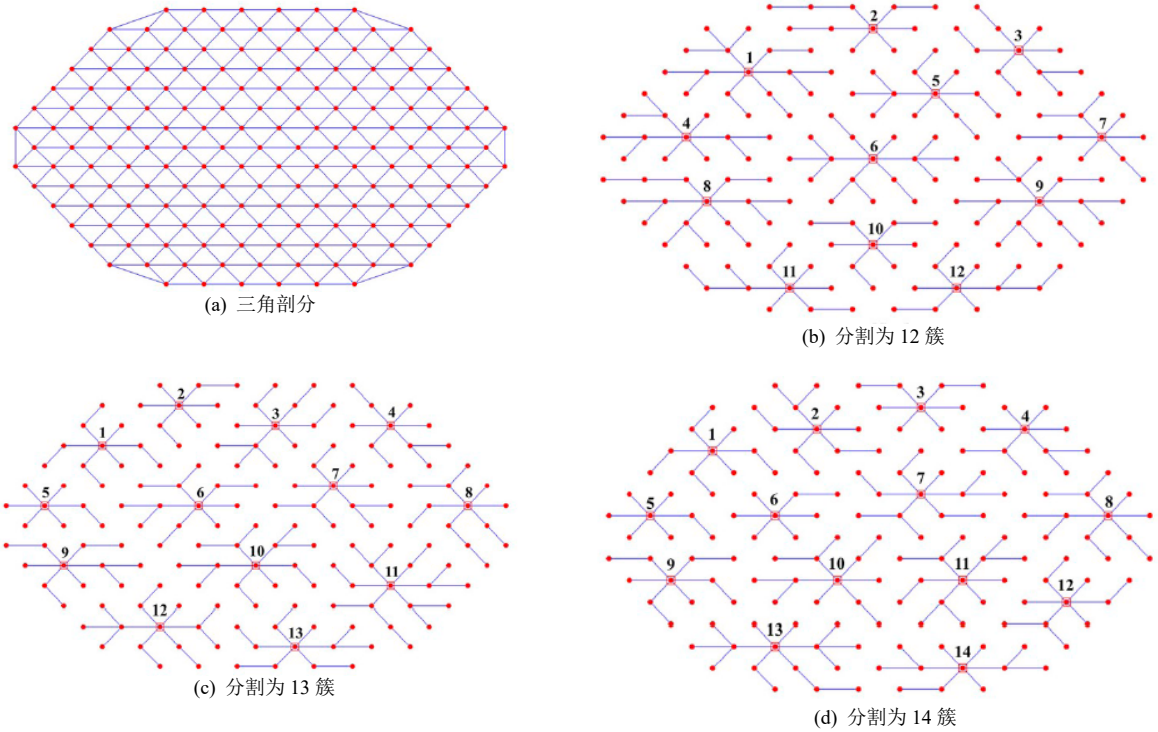

图 **13** **Delaunay** 三角剖分算法效果图

2. 本文建立的模型在实际上的供水站设置和管道分配，西部地区水井的聚类和连接等应用问题上具有较好的现实指导意义，能一定程度上帮助人们解决类似的聚类分析和连接问题。

参考文献

[1] 邓纪泽. 农村饮水安全管道铺设线路设计分析 [J]. 黑龙江水利科技. 2019. 47(11): 92-95.

[2] 于博. 浅析自来水供水管道施工技术 [J]. 百科论坛电子杂志. 2019.(22): 52.

[3] Shiono N. Suzuki H. Saruwatari Y. A dynamic programming approach for the pipe network layout problem[J]. European journal of operational research. 2019. 277(1): 52-61.

[4] 咎英飞. A numerical model for pipelaying on nonlinear soil stiffness seabed[J]. 水动力学研究与进展：英文版. 2016. 28(1): 10-22.

[5] Zhou J. Peng J. Liang G. et al. Layout optimization of tree-tree gas pipeline network[J]. Journal of Petroleum Science and Engineering. 2019. 173: 666-680.

附录 A 代码

A.1 problem1——matlab 源程序

```
clc;
%clear;
%%
[num]=xlsread('point.xlsx');

m=num(:,1);
x=num(:,3);
y=num(:,4);
gtree=zeros(181,181);
for i=1:181
for j=1:181
cell(i,:)=[x(i),y(i)];
end
end
fprintf("length(m):%d\n",length(m));
fprintf("length(x):%d\n",length(x));
fprintf("length(y):%d\n",length(y));
d=zeros(181,181);
for i=1:181
for j=1:181
d(i,j)=sqrt((x(i)-x(j))^2+(y(i)-y(j))^2);
if (i==1&&j>=14) || (j==1&&i>=14)
d(i,j)=-1;
end
end
end

% route=Prime(d(2:181,2:181));
route=Prime(d(1:13,1:13));
for i=1:13
for j=1:13
gtree(i,j)=route(i,j);
end
end

route2=Prime2(d(2:181,2:181),route(2:13,2:13));
for i=2:181
for j=2:181
gtree(i,j)=route2(i-1,j-1);
end
end
```

```
%画线路图
scatter(x(1),y(1),'bo') %中心供水站位置
hold on;
scatter(x(2:13),y(2:13),'k*') %一级供水站位置
hold on;
scatter(x(14:181),y(14:181),'b.') %二级供水站位置
hold on;

xlabel('x')
ylabel('y')
% gplot(route,cell(2:181,:),'r-');
gplot(route,cell(1:13,:),'b-')
gplot(route2,cell(2:181,:),'r-')
for i=1:181%length(cell)
%    t=cell(i:181,:);
text(x(i),y(i),[num2str(i)],'color','black','FontSize',10);
end
hleg=legend('中心供水站','一级供水站','二级供水站','一级管道','二级管道','Location','NorthEastOutside');
hold on;
```

A.2 一级区域 prime 算法——matlab 源程序

```
function [route_2,totaldistance] = Prime_one(d)
len=length(d);
dback=d;
totaldistance=0;
d(d<=0) = Inf;
P = zeros(1, len);
P(1,1) = 1;
V = 1:len;
V_P = V - P;
route = zeros(len,2);
route_2=zeros(len,len);
k=1;
while k<len
p = P(P~=0);
v = V_P(V_P~=0);
pv = min(min(d(p,v)));
[x, y] = find(d==pv);
for i=1:length(x)
if  any(P==x(i)) && any(V_P==y(i)) &&dback(x(i),y(i))~=-1 %集合判断，关键!
P(1,y(i)) = y(i);
V_P = V - P;
route(k, :) = [x(i), y(i)];
route_2(x(i),y(i))=1;
```

```
totaldistance=totaldistance+dback(x(i),y(i));
k = k+1;
break;
end
end
end
% fprintf("totaldistance1:%f\n",totaldistance);
```

A.3 二级区域 prime 算法——matlab 源程序

```
function [route_2,totaldistance] = Prime2(d,route1)
rlen=length(route1);
len=length(d);
dback=d;
totaldistance=0;
d(d<=0) = Inf;
P = zeros(1, len);
P(1,1) = 1;
for i=1:rlen
P(1,i)=i;
end
V = 1:len;
V_P = V - P;
% route = zeros(len,2);
route_2=zeros(len,len);
% for i=1:rlen
%     for j=1:rlen
%         route_2(i,j)=route1(i,j);
%     end
% end

k=rlen-1;
while k<len-1
p = P(P~=0);
v = V_P(V_P~=0);
%     if v==[]
%         break;
%     end
pv = min(min(d(p,v)));
[x, y] = find(d==pv);
for i=1:length(x)
%         if(x(i)<=12&&y(i)<=12&&route1(x(i),y(i))==1)
%             P(1,y(i)) = y(i);
%             V_P = V - P;
%             route_2(x(i),y(i))=1;
```

```
%           continue;
%       end
if  any(P==x(i)) && any(V_P==y(i)) %集合判断，关键!
P(1,y(i)) = y(i);
V_P = V - P;
%           route(k, :) = [x(i), y(i)];
route_2(x(i),y(i))=1;
totaldistance=totaldistance+dback(x(i),y(i));
k = k+1;
break;
end
end
end
% fprintf("totaldistance2:%f\n",totaldistance);
```

A.4 距离矩阵计算——matlab 源程序

```
function res = distance(x,y)
  res=zeros(181,181);
  for i=1:181
     for j=1:181
        res(i,j)=sqrt((x(i)-x(j))^2+(y(i)-y(j))^2);
        if (i==1&&j>=14) || (j==1&&i>=14)
           res(i,j)=-1;
        end
     end
  end
end
```

A.5 画图函数——matlab 源程序

```
function b = plotgraph(x,y,route,route2,cell,dlen)
%画线路图
scatter(x(1),y(1),'bo') %中心供水站位置
hold on;
scatter(x(2:dlen),y(2:dlen),'k*') %一级供水站位置
hold on;
scatter(x(dlen+1:181),y(dlen+1:181),'b.') %二级供水站位置
hold on;

xlabel('x')
ylabel('y')
% gplot(route,cell(2:181,:),'r-');
gplot(route,cell(1:dlen,:),'b-')
```

```
gplot(route2,cell(2:181,:),'r-')
t=cell(1,:);
% for i=1:13%length(cell)
% %    t=cell(i:181,:);
%    text(x(i),y(i),[num2str(i)],'color','black','FontSize',10);
% end

hleg=legend('中心供水站','一级供水站','二级供水站','一级管道','二级管道','Location','NorthEastOutside');
hold on;
end
```

A.6 problem2——matlab 源程序

```
clc;
clear;
%%
[num]=xlsread('point.xlsx');
i=0;
m=num(:,1);
x=num(:,3);
y=num(:,4);
gtree=zeros(181,181);
cell=updatecell(x,y);

fprintf("length(m):%d\n",length(m));
fprintf("length(x):%d\n",length(x));
fprintf("length(y):%d\n",length(y));
d=zeros(181,181);
d=distance(x,y);
dback=d;
% c=1;
% mintotaldis=inf;
% minc=0;
minroute1=0;
minroute2=0;
minx=0;
miny=0;
% dlen=15;
% mintotaldis2=inf;
% mintotaldis1=0;
r =randperm(168)+13;%路径格式
temperature=5000;%初始化温度
temperature_iterations=0;
cooling_rate=0.95;%温度下降比率
item=1;%用来控制降温的循环记录次数
```

```
route=[1,2,3,4,5,6,7,8,9,10,11,12,13,r];
x=x(route);
y=y(route);
d=d(route,route);
d=distance(x,y);
cell=updatecell(x,y);

dlen=15;
[minroute1,mintotaldis1]=Prime_one(d(1:dlen,1:dlen));
[minroute2,mintotaldis2]=Prime_two(d(2:181,2:181),minroute1(2:dlen,2:dlen));
mintotal=mintotaldis1+mintotaldis2;
fprintf("totaldistance1:%f,totaldistance2:%f,total=%f\n",mintotaldis1,mintotaldis2,mintotal);
plotgraph(x,y,minroute1,minroute2,cell,dlen);
% v=[1,2,3,4,5,6,7,8,9,10,11,12,13,r];
% xa=x(v);
% ya=y(v);
% kaka=d(v,v);
% kaka=distance(xa,ya);
% cell=updatecell(xa,ya);
%
% dlen=15;
% [route,totaldis1]=Prime_one(kaka(1:dlen,1:dlen));
% [route2,totaldis2]=Prime_two(kaka(2:181,2:181),route(2:dlen,2:dlen));
% total=totaldis1+totaldis2;
% fprintf("totaldistance1:%f,totaldistance2:%f,total=%f\n",totaldis1,totaldis2,total);
% plotgraph(xa,ya,route,route2,cell,dlen);


%%
i=1;
while i<14028 %循环条件
temp_route=changeroute(route,i);
%temp_route=change(route,'swap');%产生扰动。分子序列变化
xa=x(temp_route);
ya=y(temp_route);
da=d(temp_route,temp_route);
da=distance(xa,ya);
cella=updatecell(xa,ya);
[route1,totaldis1]=Prime_one(da(1:dlen,1:dlen));
[route2,totaldis2]=Prime_two(da(2:181,2:181),route1(2:dlen,2:dlen));
total=totaldis1+totaldis2;
dist=totaldis2-mintotaldis2;%两个路径的差距
if dist<0
%if(dist<0)||(rand < exp(-dist/(temperature)))
route=temp_route;
mintotaldis1=totaldis1;
mintotaldis2=totaldis2;
```

```
minroute1=route1;
minroute2=route2;
minx=xa;
miny=ya;
item=item+1;
fprintf("%d\n",i);
temperature_iterations=temperature_iterations+1;
end
i=i+1;
%    if temperature_iterations>=30
%        fprintf("temperature=%f item=%d\n\n",temperature,item);
%        temperature=cooling_rate*temperature;
%        temperature_iterations=0;
%    end
end

d=distance(minx,miny);
cell=updatecell(minx,miny);
fprintf("totaldistance1:%f,totaldistance2:%f,total=%f\n",mintotaldis1,mintotaldis2,mintotal);
plotgraph(minx,miny,minroute1,minroute2,cell,dlen);
```

A.7 退火模拟迭代——matlab 源程序

```
clc;
clear;
%%
[num]=xlsread('point.xlsx');
i=0;
m=num(:,1);
x=num(:,3);
y=num(:,4);
gtree=zeros(181,181);
cell=updatecell(x,y);

fprintf("length(m):%d\n",length(m));
fprintf("length(x):%d\n",length(x));
fprintf("length(y):%d\n",length(y));
d=zeros(181,181);
d=distance(x,y);
dback=d;
% c=1;
% mintotaldis=inf;
% minc=0;
minroute1=0;
minroute2=0;
```

```
minx=0;
miny=0;
% dlen=15;
% mintotaldis2=inf;
% mintotaldis1=0;
r =randperm(168)+13;%路径格式
temperature=5000;%初始化温度
temperature_iterations=0;
cooling_rate=0.95;%温度下降比率
item=1;%用来控制降温的循环记录次数
route=[1,2,3,4,5,6,7,8,9,10,11,12,13,r];
x=x(route);
y=y(route);
d=d(route,route);
d=distance(x,y);
cell=updatecell(x,y);

dlen=15;
[minroute1,mintotaldis1]=Prime_one(d(1:dlen,1:dlen));
[minroute2,mintotaldis2]=Prime_two(d(2:181,2:181),minroute1(2:dlen,2:dlen));
mintotal=mintotaldis1+mintotaldis2;
fprintf("totaldistance1:%f,totaldistance2:%f,total=%f\n",mintotaldis1,mintotaldis2,mintotal);
plotgraph(x,y,minroute1,minroute2,cell,dlen);
% v=[1,2,3,4,5,6,7,8,9,10,11,12,13,r];
% xa=x(v);
% ya=y(v);
% kaka=d(v,v);
% kaka=distance(xa,ya);
% cell=updatecell(xa,ya);
%
% dlen=15;
% [route,totaldis1]=Prime_one(kaka(1:dlen,1:dlen));
% [route2,totaldis2]=Prime_two(kaka(2:181,2:181),route(2:dlen,2:dlen));
% total=totaldis1+totaldis2;
% fprintf("totaldistance1:%f,totaldistance2:%f,total=%f\n",totaldis1,totaldis2,total);
% plotgraph(xa,ya,route,route2,cell,dlen);


%%
i=1;
while temperature>1.0
%while i<14028 %循环条件
%    temp_route=changeroute(route,i);
temp_route=change(route,'swap');%产生扰动。分子序列变化
xa=x(temp_route);
ya=y(temp_route);
da=d(temp_route,temp_route);
```

```
da=distance(xa,ya);
cella=updatecell(xa,ya);
[route1,totaldis1]=Prime_one(da(1:dlen,1:dlen));
[route2,totaldis2]=Prime_two(da(2:181,2:181),route1(2:dlen,2:dlen));
total=totaldis1+totaldis2;
dist=totaldis2-mintotaldis2;%两个路径的差距
%    if dist<0
if(dist<0)||(rand < exp(-dist/(temperature)))
route=temp_route;
mintotaldis1=totaldis1;
mintotaldis2=totaldis2;
minroute1=route1;
minroute2=route2;
minx=xa;
miny=ya;
item=item+1;
%        fprintf("%d\n",i);
temperature_iterations=temperature_iterations+1;
end
i=i+1;
if temperature_iterations>=30
fprintf("temperature=%f item=%d\n\n",temperature,item);
temperature=cooling_rate*temperature;
temperature_iterations=0;
end
end

d=distance(minx,miny);
cell=updatecell(minx,miny);
fprintf("totaldistance1:%f,totaldistance2:%f,total=%f\n",mintotaldis1,mintotaldis2,mintotal);
plotgraph(minx,miny,minroute1,minroute2,cell,dlen);
```

A.8 更新 1 级水站序列——matlab 源程序

```
function newroute = changeroute(pre_route,count)
  a=14:181;
  k=nchoosek(a,2);
  upone=k(count,:);
  p=setdiff( a, upone);
  newroute=[1,2,3,4,5,6,7,8,9,10,11,12,13,upone,p];
end
```

A.9 problem3——matlab 源程序

```
clc
clear;
%%
[num]=xlsread('point.xlsx');
m=num(:,1);
x=num(:,3);
y=num(:,4);
for i=1:181
for j=1:181
cell(i,:)=[x(i),y(i)];
end
end
fprintf("length(m):%d\n",length(m));
fprintf("length(x):%d\n",length(x));
fprintf("length(y):%d\n",length(y));
d=zeros(181,181);
for i=1:181
for j=1:181
d(i,j)=sqrt((x(i)-x(j))^2+(y(i)-y(j))^2);
if (i==1&&j>=14) || (j==1&&i>=14)
d(i,j)=-1;
end
end
end

count=1;
mintotaldis=inf;
min_r=0;
min_cell=0;
number=2;
min_p=0;
min_v=0;
min_xa=0;
min_ya=0;
mintotaldis=inf;
min_r=0;
mincount=0;
minnumber=0;
number=2;
% for number=1:11
julei_res=[];
max=nchoosek(11,number);
%    for count=1:max
isbad=0;
count=20;%最大55
P = zeros(1, 181);
```

```
P(1,1) = 1;
new_route=changeroute3(number,count);
for i=1:length(new_route)
P(1,new_route(i))=new_route(i);
end
dback=d;
dback(dback<=0) = Inf;
V = 1:181;
V_P = V - P;
p = P(P~=0);
v = V_P(V_P~=0);
rl=dback(p,v);
[Y,index]=min(rl);
vvp=p(index);
julei_res(:,1)=vvp';
julei_res(:,2)=v';
vx=julei_res(:,1);
vp=julei_res(:,2);
list=zeros(181,50);
for i=1:(13+number)
ml=zeros(1,50);
ml=gotlabel(new_route(i),vx,vp)';
for j=1:length(ml)
list(new_route(i),j)=ml(1,j);
end
end
totaldis=0;
for kkk=2:(13+number)
k=list(new_route(kkk),:);
k=k(k>0);
v=[new_route(kkk),k];
droute=d(v,v);
[r,totaldistance,isbad]=Prime3(droute);
fprintf("index=%d,dis=%f\n",kkk,totaldistance);
if isbad==1
%               fprintf("bad\n");
%               isbad=1;
break;
end
totaldis=totaldis+totaldistance;
% %画线路图
scatter(x(1),y(1),'bo') %中心供水站位置
hold on;
anser=p(2:length(p));
scatter(x(anser),y(anser),'k*') %一级供水站位置
hold on;
scatter(x(v),y(v),'b.') %二级供水站位置
```

```
hold on;
xlabel('x');
ylabel('y');
xa=x(v);
ya=y(v);
for i=1:length(xa)
cell(i,:)=[xa(i),ya(i)];
end

%          if(isbad==0)
%
% %               minnumber=number;
% %               mincount=count;
% %               min_p=p;
% %               min_v=v;
% %               min_xa=xa;
% %               min_ya=ya;
% % %               mintotaldis=totaldis;
% %               min_r=r;
% % %               min_cell=cell;
gplot(r,cell,'r-')

%          end
end
%        for i=1:length(anser)
%               text(x(anser(i)),y(anser(i)),[num2str(i+1)],'Fontsize',12);
%        end
if(isbad==0)%&&
if totaldis<mintotaldis&&totaldis>0
mintotaldis=totaldis;
minnumber=number;
mincount=count;
end
fprintf("count=%d\n",count);
fprintf("totaldis:%f,minnumber:%d,mincount:%d\n",mintotaldis,minnumber,mincount);
end
%    end
%    if number==2
%        break;
%    end
%    fprintf("number=%d\n",number);
%    number=number+1;
% end

% %画线路图
% scatter(x(1),y(1),'bo'); %中心供水站位置
% hold on;
```

```
% anser=p(2:length(min_p));
% scatter(x(anser),y(anser),'k*'); %一级供水站位置
% hold on;
% scatter(x(min_v),y(min_v),'b.'); %二级供水站位置
% hold on;
% xlabel('x');
% ylabel('y');
r =randperm(168)+13;
ittt=setdiff(new_route,[1,2,3,4,5,6,7,8,9,10,11,12,13]);
iikk=setdiff(r,ittt);
v=[new_route,iikk];
xa=x(v);
ya=y(v);
kaka=d(v,v);
kaka=distance(xa,ya);

caccell=updatecell(xa,ya);

dlen=13+number;
[route_as,totaldis1]=Prime_one(kaka(1:dlen,1:dlen));
gplot(route_as,caccell(1:dlen,:),'b-')
fprintf("%f,%f\n",totaldis1,totaldis1+mintotaldis);
```

A.10 带有 40km 距离限制的 prime 算法——matlab 源程序

```
function [route_2,totaldistance,isbad] = Prime3(d)
%PRIME 此处显示有关此函数的摘要
%   此处显示详细说明
len=length(d);
dback=d;
totaldistance=0;
d(d<=0) = Inf;
P = zeros(1, len);
P(1,1) = 1;
V = 1:len;
V_P = V - P;
route = zeros(len,2);
route_2=zeros(len,len);
k=1;
flag=0;
while k<len
p = P(P~=0);
v = V_P(V_P~=0);
pv = min(min(d(p,v)));
[x, y] = find(d==pv);
```

```
for i=1:length(x)
if  any(P==x(i)) && any(V_P==y(i)) &&dback(x(i),y(i))~=-1 %集合判断，关键!
P(1,y(i)) = y(i);
V_P = V - P;
route(k, :) = [x(i), y(i)];
route_2(x(i),y(i))=1;
if totaldistance+dback(x(i),y(i))>40
flag=1;
break;
end
totaldistance=totaldistance+dback(x(i),y(i));
k = k+1;
break;
end
end
if flag==1
break;
end
end
% fprintf("totaldistance1:%f\n",totaldistance);
% fprintf("if:%d\n",k~=len);
isbad=(k~=len);
```

A.11 带有 40km 距离限制的 prime 算法——matlab 源程序

```
function [route_2,totaldistance,isbad] = Prime3(d)
%PRIME 此处显示有关此函数的摘要
%   此处显示详细说明
len=length(d);
dback=d;
totaldistance=0;
d(d<=0) = Inf;
P = zeros(1, len);
P(1,1) = 1;
V = 1:len;
V_P = V - P;
route = zeros(len,2);
route_2=zeros(len,len);
k=1;
flag=0;
while k<len
p = P(P~=0);
v = V_P(V_P~=0);
pv = min(min(d(p,v)));
[x, y] = find(d==pv);
```

```
for i=1:length(x)
if  any(P==x(i)) && any(V_P==y(i)) &&dback(x(i),y(i))~=-1 %集合判断，关键！
P(1,y(i)) = y(i);
V_P = V - P;
route(k, :) = [x(i), y(i)];
route_2(x(i),y(i))=1;
if totaldistance+dback(x(i),y(i))>40
flag=1;
break;
end
totaldistance=totaldistance+dback(x(i),y(i));
k = k+1;
break;
end
end
if flag==1
break;
end
end
% fprintf("totaldistance1:%f\n",totaldistance);
% fprintf("if:%d\n",k~=len);
isbad=(k~=len);
```

A.12 按距离聚类算法——matlab 源程序

```
clc
clear;
%%
[num]=xlsread('point.xlsx');
m=num(:,1);
x=num(:,3);
y=num(:,4);
for i=1:181
for j=1:181
cell(i,:)=[x(i),y(i)];
end
end
fprintf("length(m):%d\n",length(m));
fprintf("length(x):%d\n",length(x));
fprintf("length(y):%d\n",length(y));
d=zeros(181,181);
for i=1:181
for j=1:181
d(i,j)=sqrt((x(i)-x(j))^2+(y(i)-y(j))^2);
if (i==1&&j>=14) || (j==1&&i>=14)
```

```
d(i,j)=-1;
end
end
end

count=1;
mintotaldis=inf;
min_r=0;
min_cell=0;
number=2;
min_p=0;
min_v=0;
min_xa=0;
min_ya=0;
mintotaldis=inf;
min_r=0;
mincount=0;
minnumber=0;
number=2;
for number=1:11
julei_res=[];
max=nchoosek(11,number);
for count=1:max
isbad=0;
%        count=ceil(rand*max);%最大55
P = zeros(1, 181);
P(1,1) = 1;
new_route=changeroute3(number,count);
for i=1:length(new_route)
P(1,new_route(i))=new_route(i);
end
dback=d;
dback(dback<=0) = Inf;
V = 1:181;
V_P = V - P;
p = P(P~=0);
v = V_P(V_P~=0);
rl=dback(p,v);
[Y,index]=min(rl);
vvp=p(index);
julei_res(:,1)=vvp';
julei_res(:,2)=v';
vx=julei_res(:,1);
vp=julei_res(:,2);
list=zeros(181,50);
for i=1:(13+number)
ml=zeros(1,50);
```

```
ml=gotlabel(new_route(i),vx,vp)';
for j=1:length(ml)
list(new_route(i),j)=ml(1,j);
end
end
totaldis=0;
for kkk=2:(13+number)
k=list(new_route(kkk),:);
k=k(k>0);
v=[new_route(kkk),k];
droute=d(v,v);
[r,totaldistance,isbad]=Prime3(droute);
if isbad==1
%               fprintf("bad\n");
%               isbad=1;
break;
end
totaldis=totaldis+totaldistance;

end
t=ccl(new_route,number);
totaldis=totaldis+t;
if(isbad==0)%&&
if totaldis<mintotaldis&&totaldis>0
mintotaldis=totaldis;
minnumber=number;
mincount=count;
end
fprintf("count=%d\n",count);
fprintf("totaldis:%f,minnumber:%d,mincount:%d\n",mintotaldis,minnumber,mincount);
end
end

fprintf("number=%d\n",number);
number=number+1;
end

% %画线路图
% scatter(x(1),y(1),'bo'); %中心供水站位置
% hold on;
% anser=p(2:length(min_p));
% scatter(x(anser),y(anser),'k*'); %一级供水站位置
% hold on;
% scatter(x(min_v),y(min_v),'b.'); %二级供水站位置
% hold on;
% xlabel('x');
% ylabel('y');
```

```
% xa=x(v);
% ya=y(v);
% %画线路图
% scatter(x(1),y(1),'bo') %中心供水站位置
% hold on;
% anser=p(2:length(p));
% scatter(x(anser),y(anser),'k*') %一级供水站位置
% hold on;
% scatter(x(v),y(v),'b.') %二级供水站位置
% hold on;
% xlabel('x');
% ylabel('y');
% for i=1:length(xa)
%     cell(i,:)=[xa(i),ya(i)];
% end
```

A.13 计算距离总和——matlab 源程序

```
clc;
clear;
%%
[num]=xlsread('point.xlsx');

m=num(:,1);
x=num(:,3);
y=num(:,4);
gtree=zeros(181,181);
cell=updatecell(x,y);
% for i=1:181
%     for j=1:181
%         cell(i,:)=[x(i),y(i)];
%     end
% end
fprintf("length(m):%d\n",length(m));
fprintf("length(x):%d\n",length(x));
fprintf("length(y):%d\n",length(y));
d=zeros(181,181);
d=distance(x,y);
% for i=1:181
%     for j=1:181
%         d(i,j)=sqrt((x(i)-x(j))^2+(y(i)-y(j))^2);
%         if (i==1&&j>=14) || (j==1&&i>=14)
%             d(i,j)=-1;
%         end
%     end
```

```
% end
% c=1;
% mintotaldis=inf;
% minc=0;
% minroute=0;
% minroute2=0;
% dlen=15;
% mintotaldis2=inf;
% mintotaldis1=0;
r =randperm(168)+13;%路径格式
temperature=1000;%初始化温度
cooling_rate=0.95;%温度下降比率
item=1;%用来控制降温的循环记录次数
ittt=[52,56]
iikk=setdiff(r,ittt);
v=[1,2,3,4,5,6,7,8,9,10,11,12,13,ittt,iikk];
xa=x(v);
ya=y(v);
kaka=d(v,v);
kaka=distance(xa,ya);
% for i=1:181
%     for j=1:181
%         kaka(i,j)=sqrt((xa(i)-xa(j))^2+(ya(i)-ya(j))^2);
%         if (i==1&&j>=16) || (j==1&&i>=16)
%             kaka(i,j)=-1;
%         end
%     end
% end
cell=updatecell(xa,ya);
% for i=1:181
%     for j=1:181
%         cell(i,:)=[xa(i),ya(i)];
%     end
% end
dlen=15;
[route,totaldis1]=Prime_one(kaka(1:dlen,1:dlen));
[route2,totaldis2]=Prime_two(kaka(2:181,2:181),route(2:dlen,2:dlen));
total=totaldis1+totaldis2;
fprintf("totaldistance1:%f,totaldistance2:%f,total=%f\n",totaldis1,totaldis2,total);
plotgraph(xa,ya,route,route2,cell,dlen);
```