

# 西安邮电大学

## 《数学建模 B》

### 课程论文

学生班级：对抗 1602

学生姓名：郝希烜

学 号：03166063

班内序号：27

# 校车安排问题

## 摘要

本文研究了让教师和工作人员满意度最高为目标的校车安排问题。

在问题 1 中，利用 *Dijkstra* 算法求出了任意两区直接或间接到达的最短距离，然后以各区域到最近乘车点的距离和最小为目标对 50 个区进行遍历分析，从 50 个地区中选出  $n$  个站点，共有  $C_{50}^n$  种取法，用 MATLAB 算出各个地区到这些站点的路程和的最小值，再取出这  $C_{50}^n$  个最小值中的最小者及其对应的  $n$  个站点即可。从而得到当  $n=2$  时，乘车点设立在 18 区和 31 区，各个区域到各自最近乘车点的最短距离之和为  $D=24492$  米。当  $n=3$  时，乘车点设立在 15 区、21 区和 31 区，各个区域到各自最近乘车点的最短距离之和  $D=19660$  米。

在问题 2 中，因为只有路程一个量，所以直接可以用人们到站点的总路程来刻画人们的满意程度，总路程越大人们满意度越小。据此建立满意度函数求解用 MATLAB 求得当建立 2 个乘车点时，乘车点为区域 24 和区域 32，平均满意度为 0.7239；当建立 3 个乘车点时，乘车点为区域 16、区域 23 和区域 32。平均满意度为 0.7811。

对于问题 3，本文在模型二的基础上，设立满意度最低标准，添加满意度的约束条件  $P_k > h$ ，建立车辆数模型。求得满意度最大的情况下的 3 个乘车点车辆使用情况，确定车辆最少需要 54 辆，三个站点所在的区域分别为 2、26、31，对应的车辆数分别为 12、19、23。

对于问题 4，我们根据第三问的结果发现每个站点都存在空座的情况，这样造成了一定的资源的浪费。所以我们建议在站点校车空座率较高的情况下时，在其他站点进行一次巡游。以节约了成本同时满足满意度。

**关键词：** *Dijkstra* 算法 最短距离 满意度函数 双目标函数 节约资源 MATLAB

## 1 问题的提出

许多学校都建有新校区，因此需要将老校区的教师和工作人员用校车送到新校区。有效的安排车辆并让教师和工作人员尽量满意是个十分重要的值得研究的问题。现有如下四个问题需要设计解决。

假设老校区的教室和工作人员分布在 50 个区。

问题 1：如果建立  $n$  个乘车点，为使各区人员到最近乘车点的距离最小，建立模型，并分别给出  $n=2, 3$  时的结果。

问题 2：考虑每个区的乘车人数，使工作人员和教室的满意度最大，建立模型，并分别建立两个和三个乘车点的校车安排方案。（假定车只在起始点载人）

问题 3：若建立 3 个乘车点，为使教师和工作人员尽量满意，至少需要安排多少辆车。假设每辆车最多载客 47 人（假设车只在起始站点载人）。

问题 4：关于校车安排问题，你还有什么好的建议和考虑。可以提高乘车人员的满意度，又可节省运行成本。

## 2 问题的分析

### 2.1 问题 1 的分析

问题 1 要求建立  $n$  个乘车点，使各区人员到最近乘车点的距离最小，首先利用 *Dijkstra* 算法求得任意两点之间最短距离；其次在 50 个区域中任意选取  $n$  个区域作为乘车点，找出每个区域所对应的最近乘车点，最后以 50 个区域到各自最近乘车点的最短距离和的最小值为目标函数建立模型，并对设立 2 个和 3 个乘车点时的校车安排问题进行求解。

### 2.2 问题 2 的分析

问题 2 要求在教师和工作人员的满意度最大为前提条件下选出最佳乘车点，为此需要建立关于满意度的函数，然后以平均满意度最高为目标函数建立模型，并对设立 2 个和 3 个乘车点时的校车安排问题进行求解。

### 2.3 问题 3 的分析

问题 3 要求建立 3 个乘车点，在尽量使教师和工作人员满意的前提下，所需的车辆最少，我们利用总车辆数最少函数的双目标函数进行优化求解，得出最优解。

### 2.4 问题 4 的分析

问题 4 中结合第 3 问的结果对车辆的安排情况提出了建议。

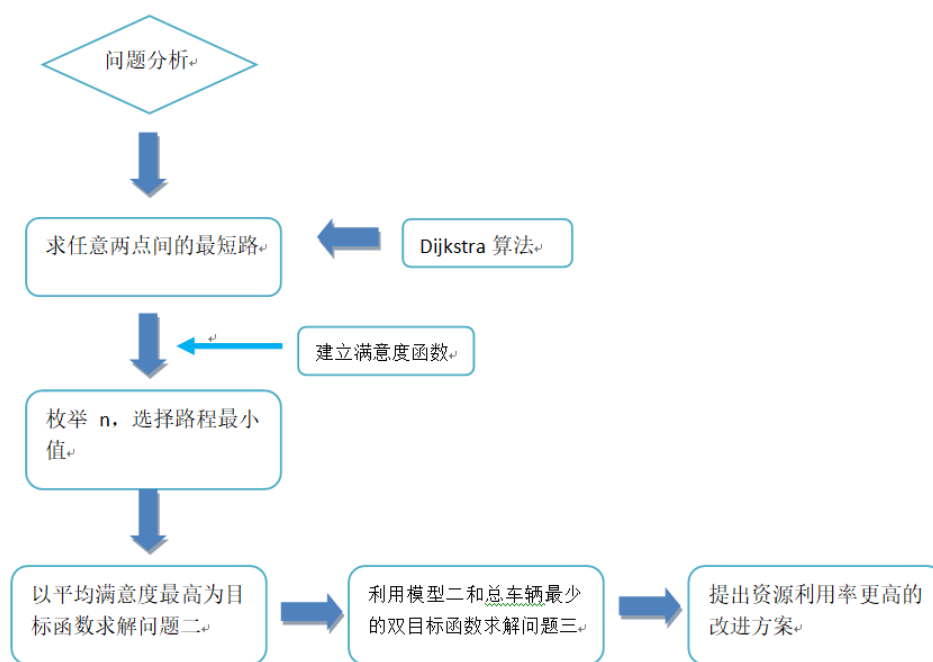


图 2-1 建模思路流程图

### 3 基本假设

- 1、每位教师及工作人员均选择最短路径乘车。
- 2、假设不考虑教师及工作人员的等车时间。
- 3、假设每个乘车点的乘车人数固定不变
- 4、教师及工作人员到各站点乘车的满意度与到该站点的距离有关系，距离近则满意度高，距离远则满意度低。
- 5、假设所设置的乘车点数不大于 50。
- 6、假设所有人员均乘车。
- 7、假设每辆车的型号一致

### 4 符号说明

符号	代表含义
$A(i, j)$	各个区通路的邻接矩阵
$P_i$	第 $i$ 乘车点所在的区

$l_k$	第 k 个区到最近乘车点的距离
$D$	50 个区到各自最近乘车点的距离之和
$P_k$	第 k 区乘客的满意度
$\bar{P}$	所有乘客的平均满意度
$q_i$	第 i 个乘车点的车辆数
$r_k$	第 k 区的人数
$h$	每个区满意度的下限 ( $0 < h < 1$ )
$n$	共要建的站点数

## 5 模型的建立

### 5.1、问题 1 的模型建立

#### 5.1.1 Dijkstra 算法

*Dijkstra* 算法是一种求单源最短路的算法,但对每个点进行遍历后可以求任意两点之间的最短路,效率高于 *Floyed* 算法。

*Dijkstra* 算法的基本思路是:先将与起点有边直接相连的节点到起点的距离记为对应的边的权重值,将与起点无边直接相连的节点到起点的距离记为 $\infty$ 。然后以起点为中心向外层层扩展,计算所有节点到起点的最短距离。每次新扩展到一个距离最短的点后,更新与它有边直接相邻的节点到起点的最短距离。当所有点都扩展进来后,所有节点到起点的最短距离将不会再被改变,因而保证了算法的正确性。

1. *Dijkstra* 算法求解最短路径问题的基本步骤如下:

①设立  $Y$  和  $N$  两个集合,  $Y$  用于保存所有等待访问的节点,  $N$  记录所有已经访问过的节点。

②访问网络节点中距离起始节点最近且没有被访问过的节点,把这个节点放入  $Y$  中等待访问。

③从  $Y$  中找出距离起点最近的节点,放入  $N$  中,更新与这个节点有边直接相连的相邻节点到起始节点的最短距离,同时把这些相邻节点加入  $Y$

④重复步骤（2）和（3），直到 Y 集合为空， N 集合为网络中所有节点为止。

2. 由于题中图节点较多，下面用该图来演示一下 Dijkstra 算法

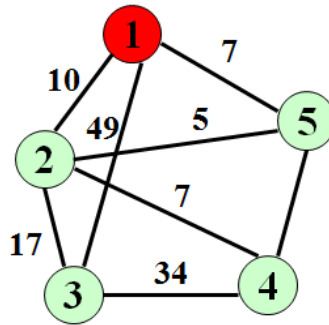


图 5-1 Dijkstra 算法演示样图

开始点（源点）：start

$D[i]$ : 顶点  $i$  到 start 的最短距离。

初始:  $D[start]=0$ ;  $D[i]=a[start][i]$  （无边设为 INF）

集合 1: 已求点

集合 2: 未求点



算法步骤:

①在集合 2 中找一个到 start 距离最近的顶点  $k$ ，距离= $d[k]$

②把顶点  $k$  加到集合 1 中，同时修改集合 2 中的剩余顶点  $j$  的  $d[j]$  是否经过  $k$  后变短。如果变短修改  $d[j]$

$if (d[k] + a[k][j] < d[j]) \rightarrow d[j] = d[k] + a[k][j]$

③重复 1，直至集合 2 空为止。

关于本文具体问题的算法如下：

1. 先根据题目所给的各个连通区域之间距离的数据为初始矩阵 $A(i, j)$  赋值，其中没有给出距离的赋为 $\infty$ ，其中  $A(i, j) = 0 \quad (i = j)$

2. 对于每一个点，进行一次 *Dijkstra* 算法，依次从与当前点相连的边中选取最短的边去更新其他点与当前点的距离

3. 对所有点均进行 *Dijkstra* 算法遍历后，任意两个点之间的距离就是它们的最短距离，可以得到最短距离矩阵  $A(i, j), (i, j = 1, 2, \dots, 50)$

### 5.1.2 模型一的建立

在上述最短路距离矩阵 $A(i, j)$ 的基础上，分析建立  $n$  个乘车点的情况：

首先，在 50 个区域中任意选取  $n$  个区域作为乘车点

$$\{p_1, p_2, \dots, p_n\} \in \{1, 2, \dots, 50\}$$

其次，由于每个区的乘客都选距离本区最近的乘车点乘车，引入变量  $l_k$ ，表示第  $k$  区域到最近乘车点的距离

$$l_k = \min \{A(k, p_1), A(k, p_2), \dots, A(k, p_n)\} \quad (k = 1, 2, \dots, 50) \quad (1)$$

然后，求出 50 个区域到各自最近乘车点的最短距离之和

$$D = \sum_{k=1}^{50} l_k \quad (2)$$

最后，建立针对问题 1 所述的数学模型。最佳乘车点是使得 50 个区域到各自最近乘车点的距离之和最小的点，基于此建立目标函数

$$\min D = \sum_{k=1}^{50} l_k \quad (3)$$

## 5.2、问题 2 模型的建立

### 5.2.1 根据题意建立顾客满意度函数

如果车站就建在自己的区，则乘客就非常的满意，如果离自己区最近的车站比较远，则乘客就不满意，乘客对车站点的满意度取决于自己区到最近乘车点的距离，为此我们建立满意度函数

$$P_k = \frac{d_{\max} - d_k}{d_{\max} - d_{\min}} \quad (4)$$

其中,  $d_{\max}$  为第  $k$  个区离本区最远区的距离,  $d_{\min}$  为第  $k$  个区离本区最近区的距离, 当然离自己区的距离最近, 即  $d_{\min} = 0$ , 化简得:

$$P_k = 1 - \frac{d_k}{d_{\max}} \quad (5)$$

其中  $P_k$  的值越大, 满意度就越大。如果乘车点就建在自己的区, 则  $d = 0$ ,  $P_k = 1$ , 该区的乘客非常满意; 如果让乘客去距离本区最远的区乘车, 则  $P_k = 0$ , 为极度不满意。

### 5.2.2 模型二的建立

由于每个区乘客的满意度不同, 每个区的人数也不同, 我们不可能使每个区乘客的满意度都最大, 因此我们关注的是全体乘客的平均满意度  $\bar{P}$

$$\bar{P} = \frac{\sum_{k=1}^{50} P_k \times r_k}{\sum_{k=1}^{50} r_k} \quad (6)$$

为使教师和工作人员的满意度最大, 为此我们将全体人的平均满意度作为目标函数

$$\max \bar{P} = \frac{\sum_{k=1}^{50} P_k \times r_k}{\sum_{k=1}^{50} r_k} \quad (7)$$

## 5.3、问题3的模型建立

### 5.3.1 模型三的建立

对所需车辆数  $q_k$  的分析, 设到第  $i$  个乘车点的区域的子集合为  $A_i$

$$q = \left\lceil \frac{\sum_{k \in A_i} z_k}{47} \right\rceil \quad (\lceil \rceil \text{表示向上取整}) \quad (8)$$

$$\min Q = q_1 + q_2 + q_3 \quad (9)$$

由于每个站点的人数不恰好是车辆满载乘客数的整数倍, 每个站点就有可能有一辆车不能满载, 所以当站点数越多, 不能满载的车辆数就越多, 从而导致所需车辆总数的增加。当  $n=1$  时,  $q=54$ , 这也是所需车辆数的最小值。



关于模型二当  $n=3$  时结出的结果，其中平均满意度是在建立 3 个站点的请况下最大的结果，经运算得需车辆数为 56，但车辆数不是最小。

在模型二中，虽然使得平均满意度最大，但个别区的满意度却相当的小，比如第三个区的满意度仅为 0.4434。

为了兼顾平均满意度尽可能的大、车辆数尽可能小，建立以下模型：在每个区的满意度都大于最低满意度标准的情况下，即  $P_k > h$ ，其中  $h$  可人为地设定且  $0 < h < 1$ ，求出  $\bar{P}$  的最大值，即

$$\max \bar{P} = \frac{\sum_{k=1}^{50} P_k \times r_k}{\sum_{k=1}^{50} r_k} \quad (0 < h < 1, P_k > h) \quad (10)$$

## 6 模型的求解

### 6.1、问题 1 模型的求解

依据模型一，由题目给出的各区距离表 1 及假设 1、2 运用 C++语言实现 *Dijkstra* 算法（算法程序参见附录程序 1），求出任意两点之间的最短路径矩阵。

1	0	400	450	700	910	1140	1110	1200	1400	1614	1764	1904	2104	1644	1454	1284	1424	1154	950	810	630	930	900	1000	1170	1400	1320	1130	1110	1100	1300	1530	1720	1700	1670	1560	1830	1870	1740	1700	1840	1320	1310	1890	1200	1300	540	1110	1310	1510
2	400	0	850	500	510	740	710	800	1000	1214	1364	1504	1704	1244	1054	884	1024	754	550	410	230	500	600	770	1060	920	730	710	790	900	1100	1220	1160	1430	1470	1140	1300	1440	930	810	650	800	990	540	710	910	1110			
3	450	850	0	600	810	1040	1010	1100	1300	1500	1710	1910	2000	1604	1414	1244	1384	1114	910	800	600	1000	1100	1320	1500	1320	1100	1080	1260	1460	1580	1600	1870	1860	1600	1920	1910	1510	1660	1800	1240	1370	1470	1690	1560	1760	1070	1470	1670	
4	700	500	600	0	210	440	410	500	700	900	1110	1250	1450	1004	814	644	704	514	310	450	500	630	830	725	485	655	485	615	875	1015	1065	1205	1475	1265	1435	1325	1595	1635	1505	1405	1695	1145	1135	875	1100	1200	440	1010	1210	1275
5	910	510	810	210	0	230	200	270	570	750	900	1040	1240	845	655	490	630	500	300	600	740	1040	955	605	540	1010	870	825	1005	1225	1275	1505	1540	1330	1445	1535	1385	1845	1715	1675	1815	1355	1345	1005	1210	1420	1445			
6	1140	740	1040	440	230	0	320	340	540	720	870	1010	1210	815	625	610	750	480	684	824	970	1270	1070	830	660	1005	900	820	1220	1300	1410	1640	1535	1325	1700	1670	1540	1890	1810	1950	1490	1480	1220	1540	1730	880	1450	1650	1620	
7	1110	710	1010	410	200	320	0	170	370	550	700	840	1040	645	455	290	400	180	364	504	654	804	704	510	340	810	670	640	900	1040	1090	1320	1540	1130	1400	1390	1660	1530	1490	1630	1170	1160	900	1254	1444	830	1164	1364	1300	
8	1200	800	1100	500	370	540	170	0	200	380	530	670	870	475	285	455	535	330	534	674	824	1154	920	680	510	665	775	810	1070	1210	1260	1385	1195	855	1525	1520	1605	1820	1700	1640	1800	1340	1330	1870	1424	1624	1020	1334	1534	1470
9	1400	1000	1300	700	570	540	370	200	0	180	330	470	670	460	340	510	590	530	734	874	1024	1354	1120	880	710	650	790	880	1240	1410	1430	1370	1180	970	1510	1610	1670	1805	1740	1800	1440	1530	1270	1624	1814	1220	1534	1734	1640	
10	1614	1214	1500	900	720	720	570	200	180	0	150	290	490	280	160	330	410	400	664	804	954	1284	1050	810	640	470	610	800	1000	1340	1230	1100	1000	700	1330	1430	1490	1615	1610	1610	1760	1470	1460	1200	1554	1744	1334	1464	1644	1460
11	1764	1364	1710	1110	900	870	700	530	330	150	0	140	340	130	310	400	500	610	814	954	1314	1330	1020	790	700	320	460	650	910	1310	1180	1040	850	640	1180	1340	1475	1460	1470	1610	1440	1430	1170	1600	1790	1454	1510	1710	1310	
12	1904	1504	1850	1250	1040	810	640	470	290	140	0	200	270	450	620	700	750	954	1094	1274	1470	1110	910	730	600	400	790	1050	1450	1320	1000	810	600	1140	1240	1300	1435	1420	1430	1570	1500	1710	1540	1630	1624	1650	1850	1440		
13	2104	1704	2050	1450	1240	1110	840	670	490	340	200	0	470	630	820	900	950	1154	1294	1474	1670	1360	1110	910	660	400	900	1220	1270	1030	800	610	400	1440	1600	1100	1235	1220	1210	1370	1400	1480	1510	1600	1930	1824	1850	2050	1440	
14	1644	1244	1604	1004	845	615	645	475	460	280	130	270	470	0	100	360	440	490	694	834	1014	1200	950	650	670	330	520	730	1100	1180	970	910	720	510	1050	1150	1210	1345	1330	1340	1400	1440	1470	1600	1364	1380	1500	1180		
15	1454	1054	1414	814	655	625	455	285	340	160	110	450	650	190	0	170	250	300	504	644	824	1124	890	650	400	300	490	640	1100	1130	1100	910	700	1240	1340	1400	1535	1520	1530	1670	1310	1300	1840	1394	1204	1174	1304	1504	1240	
16	1284	884	1244	644	490	610	200	455	510	330	400	620	820	360	170	0	140	130	334	474	654	954	720	480	330	520	300	570	830	1010	1020	1240	1050	840	1300	1280	1540	1590	1400	1420	1220	1120	950	1364	1254	1144	1274	1474	1400	
17	1424	1024	1384	704	630	750	430	535	590	410	540	700	900	440	250	140	0	270	474	614	794	1094	800	560	450	380	240	430	690	1090	880	1100	910	700	1240	1140	1400	1450	1320	1200	1420	1220	1120	950	1364	1254	1144	1274	1474	1400
18	1154	754	1114	514	500	460	160	330	530	460	610	750	950	490	300	130	270	0	204	344	524	814	500	580	680	630	430	480	740	800	930	1160	1160	970	1000	1100	1460	1500	1370	1310	1470	1810	1000	1480	1204	874	1004	1204	1140	
19	950	550	910	310	520	684	364	534	734	664	814	954	1154	694	504	334	474	204	0	140	320	620	415	175	345	635	495	365	565	755	905	1165	955	1125	1015	1285	1325	1155	1155	1295	835	825	565	890	1000	670	800	1000	955	
20	810	410	1050	490	660	824	674	874	804	954	1094	1294	834	644	474	414	344	140	0	180	480	430	190	560	650	510	320	580	720	770	1000	1180	970	1140	1050	1300	1340	1140	1210	1170	1510	850	840	500	750	940	530	660	800	
21	630	230	1080	530	740	970	684	854	1054	984	1134	1274	1474	814	824	654	594	524	320	180	0	300	270	370	540	500	600	580	480	560	670	900	1090	1150	1040	930	1200	1240	1110	1070	1210	800	680	420	570	760	350	480	680	680
22	950	550	1350	650	1040	1270	984	1154	1354	1254	1370	1470	1670	1200	1124	954	1054	824	480	300	0	310	550	720	1010	880	620	520	680	710	940	1130	1330	1080	970	1240	1200	1150	1110	1250	730	420	160	270	460	650	100	300	820	
23	800	500	1325	725	935	1070	750	920	1120	1050	1020	1160	1360	890	800	720	800	590	415	430	270	310	0	240	410	700	560	570	210	200	400	630	820	1020	770	660	930	970	840	800	940	410	410	550	780	620	400	610		
24	1000	600	1085	485	695	830	510	680	800	810	780	920	1120	650	650	480	350	175	190	370	550	240	0	170	460	320	130	330	550	580	810	990	700	950	840	1110	1150	1020	980	1120	660	800	820	1010	720	730	930	790		
25	1170	770	1255	655	540	660	340	510	710	640	790	920	1130	670	480	310	450	180	345	300	540	720	410	170	0	630	490	380	560	700	750	980	1160	950	1120	1010	1220	1100	1150	1220	630	820	500	990	1120	800	1100	900		
26	1460	1060	1545	945	1010	1005	810	665	650	470	320	460	660	190	300	520	380	560	635	650	930	760	460	330	0	140	330	590	990	780	720	530	320	680	960	1020	1155	1140	1150	1290	1120	1110	880	1200	1470	1100	1350	900		
27	1320	920	1405	805	870	990	670	770	790	610	460	680	800	330	490	380	240	510	455	510	680	870	580	320	480	140	0	190	450	850	640	860	670	460	800	1100	1210	1180	1004	1180	980	770	710	1140	1330	1040	1050	1250	950	
28	1130	730	1215	615	825	960	640	810	580	560	560	560	560	570	430	480	350	480	350	520	680	730	130	330	330	190	0	260	660	450	660	650	820	710	1080	1020	890	850	990	790	720	920	1540	1410	800	1600	660			
29	1210	820	1565	1015	1140	1200	1010	1100	1250	1400	1610	1760	1900	2000	1600	1400	1240	1380	1110	910	800	600	1000	1100	1320	1500	1320	1100	1080	1260	1460</																			

见附录程序 3)

求得结果如下:

① 当  $n = 2$  时:

乘车点设立在 18 区和 31 区, 各个区域到各自最近乘车点的最短距离之和为  $D=24492$  米。

选 18 站点乘车的区域	选 31 站点乘车的区域
1、2、3、4、5、6、7、8、9、10、11、 12、13、14、15、16、17、18、19、20、 21、24、25、22、26、27、47	23、28、29、30、31、32、33、34、35、 36、37、38、38、40、41、42、43、44、 45、46、48、49、50

② 当  $n = 3$  时:

乘车点设立在 15 区、21 区和 31 区, 各个区域到各自最近乘车点的最短距离之和  $D=19660$  米。

选 15 站点乘车的区域	选 21 站点乘车的区域	选 31 站点乘车的区域
5、6、7、8、9、10、11、 12、13、14、15、16、17、 18、24、25、26、27	1、2、3、4、19、20、21、 22、23、24、44、45、46、 47、48、49	28、29、30、31、32、33、 34、35、36、37、38、39、 40、41、42、43、50

## 6. 2、问题 2 模型的求解

利用 MATLAB 软件求得结果如下 (程序参见附录中程序 4):

①当  $n = 2$  时:

选择的 2 个乘车点为区域 24 和区域 32, 平均满意度  $\bar{P}$  为 0.7239。

选 24 站点乘车的区域	选 32 站点乘车的区域
1、2、3、4、5、6、7、8、9、10、11、12、 13、15、16、17、18、19、20、21、22、 23、24、25、26、27、28、29、43、44、 45、46、47、48、49、50	14、30、31、32、33、34、35、36、37、 38、39、40、41、42

选 24 站乘车的区域、的有 36 个，选 32 站乘车的区域有 14 个。

②当  $n=3$  时：

选择的三个乘车点为区域 16、区域 23 和区域 32。平均满意度为 0.7811。

选 16 站点乘车的区域	选 23 站点乘车的区域	选 32 站点乘车的区域
1、2、25、26、27	3、4、5、6、7、8、9、10、 11、12、13、14、15、16、 17、18、19、42、43、44、 45、46、47、48、49	20、21、22、23、24、28、 29、30、31、32、33、34、 35、36、37、38、39、40、 41、50

选 16 站乘车的区域有 5 个，选 23 站乘车的区域有 25 个，选 32 站乘车的区域有 20 个。

### 6.3、问题 3 模型的求解

依据此模型利用 MATLAB 软件求得结果如下（程序参见附录中程序 5）：

当  $n=3$  时, 取不同的值时，算得在平均满意度较高的几种情况下，站点、平均满意度及车辆数的情况如表 1 所示：

表 5.1 站点、平均满意度及车辆数

H	平均满意度	总车辆数	站点 p1	站点 p2	站点 p3	p1 的车辆数	p2 的车辆数	p3 的车辆数
0.500	0.7809	55	18	23	32	20	28	17
0.533	0.7690	54	2	16	31	12	19	23
0.547	0.7565	54	7	23	33	15	23	16
0.550	0.7469	55	2	14	23	10	17	28
0.570	0.7424	55	19	23	34	16	21	18

## 6.4、问题 4 的求解

通过对第三问的结果的分析可知，每个站点都存在空座的情况，这样造成了一定的资源的浪费。所以我们建议在站点校车空座率较高的情况下时，在其他站点进行一次巡游。当校车型号单一时，很容易造成某些站点乘客难以乘车而其他某些站点又大量空座的情况，这种方案最大限度的节省了成本，相当于所有乘客集中乘车，同时因为乘客依然可以在对自己满意度高的站点候车，也达到了使满意度逼近甚至达到最大的效果。

## 7 结果分析

### 7.1 问题 1 的结果分析

由结果可看出当乘车点越多时， $D$  值越小。

### 7.2 问题 2 的结果分析

由计算结果可看出，建立车站数越多，乘客的平均满意度越高。

### 7.3 问题 3 的结果分析

可取得在  $h = 0.533$ ，车辆数达到了最小值 54，平均满意度  $\bar{P}$  为 0.769，相对比较高。三个站点所在的区域分别为 2、26、31，对应站点的车辆数分别为 12、19、23。

### 7.4 问题 4 的结果分析

在站点校车空座率较高的情况下时，在其他站点进行一次巡游的方案最大限度的节省了成本，相当于所有乘客集中乘车，同时因为乘客依然可以在对自己满意度高的站点候车，也达到了使满意度逼近甚至达到最大的效果。

## 8 模型的评价与改进

### 8.1 模型评价

①优点：模型结构简单，巧妙地将教师和工作者满意度量化为函数问题，而且便于计算，当数据量很大时，此优势更加突出，并且在求解时，结合 C++语言和 MATLAB，在一定程度上提高了计算速度。

②缺点：模型的影响因素过于单一化，假设条件过于理想化，使得结果与实际情况有些误差。用 *Dijkstra* 算法计算的也是理想化的情况。比如存在车载量未开走或车辆等候教师及工作人员而停滞的现象。未考虑到天气（阴雨天）、时间（节假日）及每个

人的具体情况。

## 8.2 模型改进

本文模型适合于区域较少的情况，当区域量十分庞大的时候，模型的误差变大，所以我们考虑到，对于区域量很大的情况，以区域密集度为决策量，选出密集度高的区域作为乘车点被选区，在对乘车点被选区利用本文模型进行求解，这样使得问题变得简单化。

## 参考文献

- [1]姜启源，数学模型[M]，高等教育出版社，2003。
- [2] Thomas H.Cormen、Charles E.Leiserson，算法导论[M]，机械工业出版社，2006。
- [3]刘浩，韩晶，MATLAB 完全自学一本通[M]，电子工业出版社，2016。
- [4]郭学军，数学建模竞赛辅导教程[M]，浙江大学出版社，2009。
- [5]刘汝佳，算法竞赛入门经典[M]，清华大学出版社，2009。

## 附录

### 1. 程序 1 (Dijkstra 算法求出最短路矩阵)

```
#include <iostream>
#include <vector>
#include <utility>
#include <queue>
#include <functional>
#include <algorithm>
#include <cstdio>
using namespace std;
const int maxn = 1010 + 20;
const int INF = 9999999;
int V, E;
int Prev[maxn];           //最短路上的前驱顶点
int d[maxn];
```

```

int a[100][100];      //i->j 上的权值
int al[100][100];
int used[maxn];

//求从起点 s 出发到各个顶点的最短距离
void dijkstra(int s);

//从 s 点出发到各个顶点的最短距离
void dijkstra(int s)
{
    fill(d, d + V, INF);
    fill(used, used + V, false);
    fill(Prev, Prev + V, -1);
    d[s] = 0;

    while (true) {
        int v = -1;
        for (int u = 0; u < V; u++) {
            if (!used[u] && (v == -1 || d[u] < d[v]))
                v = u;  //找出到下一条尝试的顶点中距离最短的点
        }

        if (v == -1) break;
        used[v] = true;
        for (int u = 0; u < V; u++) {
            if (d[u] > d[v] + a[v][u]) {
                d[u] = d[v] + a[v][u];  //从 v 到各个临边 u 中最短的路—>存放
                //到 d[u], 用于下一次计算
                Prev[u] = v;              //u 的前驱是 v
            }
        }
    }
}

```

```

    }
}
}
int main()
{
    for(int i=0;i<50;i++)
        for(int j=0;j<50;j++)
            if(i==j)
                a[i][j] = 0;
            else
                a[i][j] = INF;

a[0][1]=400;a[0][2]=450;a[1][3]=300;a[1][20]=230;a[1][46]=140;a[2][3]=600;

a[3][4]=210;a[3][18]=310;a[4][5]=230;a[4][6]=200;a[5][6]=320;a[5][7]=340;

a[6][7]=170;a[6][17]=160;a[7][8]=200;a[7][14]=285;a[8][9]=180;a[9][10]=150;
    a[9][14]=160;a[10][11]=140;a[10][13]=130;a[11][12]=200;a[12][33]=400;
    a[13][14]=190;a[13][25]=190;a[14][15]=170;a[14][16]=250;a[15][16]=140;
    a[15][17]=130;a[16][26]=240;a[17][18]=204;a[17][24]=180;a[18][19]=140;
    a[18][23]=175;a[19][20]=180;a[19][23]=190;a[20][21]=300;a[20][22]=270;
    a[20][46]=350;a[21][43]=160;a[21][44]=270;a[21][47]=180;a[22][23]=240;
    a[22][28]=210;a[22][29]=290;a[22][43]=150;a[23][27]=130;a[23][24]=170;
    a[25][26]=140;a[25][33]=320;a[26][27]=190;a[27][28]=260;a[28][30]=190;
    a[29][30]=240;a[29][41]=130;a[29][42]=210;a[30][31]=230;a[30][35]=260;
    a[30][49]=210;a[31][32]=190;a[31][34]=140;a[31][35]=240;a[34][36]=160;
    a[35][38]=180;a[35][39]=190;a[36][37]=135;a[37][38]=130;a[38][40]=310;
    a[39][40]=140;a[39][49]=190;a[41][49]=200;a[42][43]=260;a[42][44]=210;
    a[32][33]=210;a[44][45]=240;a[45][47]=280;a[47][48]=200;

```

```

for(int i=0;i<50;i++)
    for(int j=0;j<50;j++)
        if(a[i][j]!=0 && a[i][j]!=INF)
            a[j][i]=a[i][j];

```

V=50,E=77;

```

for(int i=0;i<50;i++)
{
    dijkstra(i);
    for(int j=0;j<50;j++)
    {
        a1[i][j]=d[j];
    }
}

for(int i=0;i<50;i++)
{
    for(int j=0;j<50;j++)
        cout<<a1[i][j]<<' ';
    cout << endl;
}

return 0;
}

```

2. 程序 2: (求出  $n=2$  时应选择的站点, 及各区所应选的乘车点)

```

sl=inf;
for b=1:n
    for c=1:n
        for d=1:n
            if a(b,d)<a(c,d)
                l(d)=a(b,d);
            else l(d)=a(c,d);

```



```

        end
    end
    L=sum(1);
    if s1>L
        s1=L;p1=b;p2=c;
    end
end
end
s1,p1,p2
for i=1:n
    if a(i,p1)<=a(i,p2)
        qulu(1,i)=p1;
    else qulu(1,i)=p2;
    end
end
qulu

```

3. 程序 3: (求出  $n=3$  时应选择的站点, 及各区所应选的乘车点)

```

s1=inf;
for b=1:n
    for c=1:n
        for d=1:n
            for e=1:n
                if a(b,e)<a(c,e)&a(b,e)<a(d,e)
                    l(e)=a(b,e);
                elseif a(c,e)<a(b,e)&a(c,e)<a(d,e)
                    l(e)=a(c,e);
                else l(e)=a(d,e);
                end
            end
            L=sum(1);
            if s1>L
                s1=L;p1=b;p2=c;p3=d;
            end
        end
    end
end
s1,p1,p2,p3
for i=1:n
    if a(i,p1)<=a(i,p2)&a(i,p1)<=a(i,p3)
        qulu(1,i)=p1;
    elseif a(i,p2)<=a(i,p1)&a(i,p2)<a(i,p3)
        qulu(1,i)=p2;
    else qulu(1,i)=p3;
    end
end

```

```

        end
    end
qulu

```

4. 程序 4 : (求 n=3 时怎样达到最大满意度)

```

ren=[65 67 42 34 38 29 17 64 39 20 61 47 66 21 70 85 12 35 48 54 49 12 54 46
76 16 94 18 29 75 10 86 70 56 65 26 80 90 47 40 57 40 69 67 20 18 68 72 76 62]
q=sum(ren);
s1=0;
A=max(a);
for b=1:n
    for c=1:n
        for d=1:n
            for e=1:n
                mm=[a(b,e),a(c,e),a(d,e)];
                l(e)=min(mm);
                lren(e)=(A(e)-l(e))/A(e)*ren(e);
            end
            L=sum(lren);
            if s1<L
                s1=L;p1=b;p2=c;p3=d;
            end
        end
    end
end
manyidu=s1/q,p1,p2,p3
for i=1:n
    if a(i,p1)<=a(i,p2)&a(i,p1)<=a(i,p3)
        qulu(1,i)=p1;
    elseif a(i,p2)<=a(i,p1)&a(i,p2)<a(i,p3)
        qulu(1,i)=p2;
    else qulu(1,i)=p3;
    end
end
qulu

```

5. 程序 5:

```

ren=[65 67 42 34 38 29 17 64 39 20 61 47 66 21 70 85 12 35 48 54 49 12 54 46
76 16 94 18 29 75 10 86 70 56 65 26 80 90 47 40 57 40 69 67 20 18 68 72 76 62]
q=sum(ren);
s1=0;
A=max(a);

```

```

for b=1:n
    for c=1:n
        for d=1:n
            for e=1:n
                mm=[a(b,e),a(c,e),a(d,e)];
                l(e)=min(mm);
                lren(e)=(A(e)-l(e))/A(e)*ren(e);
            end
            L=sum(lren);
            if s1<L
                s1=L;p1=b;p2=c;p3=d;
            end
        end
    end
end
manyidu=s1/q,p1,p2,p3
for i=1:n
    if a(i,p1)<=a(i,p2)&a(i,p1)<=a(i,p3)
        qulu(1,i)=p1;
    elseif a(i,p2)<=a(i,p1)&a(i,p2)<a(i,p3)
        qulu(1,i)=p2;
    else qulu(1,i)=p3;
    end
end
qulu
for i=1:n
    if a(i,p1)<=a(i,p2)&a(i,p1)<=a(i,p3)&a(i,p1)<=a(i,p4)
        qulu(1,i)=p1;
    elseif a(i,p2)<=a(i,p1)&a(i,p2)<a(i,p3)&a(i,p2)<=a(i,p4)
        qulu(1,i)=p2;
    elseif a(i,p3)<=a(i,p1)&a(i,p3)<=a(i,p2)&a(i,p3)<=a(i,p4)
        qulu(1,i)=p3;
    else qulu(1,i)=p4;
    end
end
plrenshu=0;p2renshu=0;p3renshu=0;p4renshu=0;
for i=1:n
    if qulu(1,i)==p1
        plrenshu=plrenshu+ren(1,i);
    elseif qulu(1,i)==p2
        p2renshu=p2renshu+ren(1,i);
    elseif qulu(1,i)==p3
        p3renshu=p3renshu+ren(1,i);
    else p4renshu=p4renshu+ren(1,i)

```

```
        end
    end
    plrenshu, p2renshu, p3renshu, p4renshu
    ch1=plrenshu/47, ch2=p2renshu/47, ch3=p3renshu/47, ch4=p4renshu/47
    che=ceil(ch1)+ceil(ch2)+ceil(ch3)+ceil(ch4)
```