2007 高教社杯全国大学生数学建模竞赛

承　诺　书

我们完全明白，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、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛规则的，如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们郑重承诺，严格遵守竞赛规则，以保证竞赛的公正、公平性。如有违反竞赛规则的行为，我们将受到严肃处理。

我们参赛选择的题号是（从 A/B/C/D 中选择一项填写）：B

我们的参赛报名号为（如果赛区设置报名号的话）：B 甲 1802

所属学校（请填写完整的全名）：海军航空工程学院（青岛）

参赛队员 (打印并签名) ：1. 汤志高

2. 王继利

3. 曹莹瑛

指导教师或指导教师组负责人 (打印并签名)：曹华林

日期：2007 年 9 月 24 日

赛区评阅编号（由赛区组委会评阅前进行编号）：

2007 高教社杯全国大学生数学建模竞赛

编 号 专 用 页

赛区评阅编号（由赛区组委会评阅前进行编号）：

赛区评阅记录（可供赛区评阅时使用）：

评阅人										
评分										
备注										

全国统一编号（由赛区组委会送交全国前编号）：

全国评阅编号（由全国组委会评阅前进行编号）：

公交查询系统的最佳乘车方案研究与设计

【摘要】

本文将站点实体间的线路选择抽象为图论最短路模型采用 **0-1** 整数规划表述。建立直达数据库 **Q** 作为数据基库，根据用户需求建立不同目标的 **0-1** 规划模型运用邻接算法与 ***Lingo*** 分别求解，最终方案集通过多目标分层序列排序输出到用户终端。

第一问，在数据处理阶段将直行、环行线路分别抽象为 2、4 条路线(见 5.0)。建立查询系统时考虑服务器要同时响应多个请求，计算任务繁重，采用空间换取时间的策略，先建立站点至站点直达数据库 **Q** 来描述两两可直达站点的所有线路，用户查询时，系统首先查询 **Q**，得到所有直达车方案。

在没有直达车情况下，针对不同用户需求，目标考虑：转乘次数、总耗时、总费用、转站车辆是否始发、转乘站点负载量；在 **Q** 的基础上，量化不同目标为有向赋权图的不同权矩阵(见 5.2.0)，以所求顶点 u 到顶点 v 的路径是否包含 x_{ij} 弧为决策变量，上述 5 项用户需求为目标，始、终点连通为约束建立 **0-1** 整数线性规划模型(见 5.2.3 模型Ⅰ)。

为了能够为用户提供多种备选方案，我们首先使用基于 Dijkstra 的邻接算法求解，得到不同目标下的多种优化方案；对于邻接算法不易求解的多次转乘最优方案，我们采用 ***Lingo*** 软件直接求得全局最优解；两种方法求解步骤见(5.3.1)，综合方案集见(5.3.2 表 1.1~1.6)，其中 6 条线路时间最短目标分别为 67、102、106、62、105、49(分钟)；两种求解方法的优劣在 5.4 中给出了详细评价。

第二问考虑公汽与地铁混排方案，首先把各地铁站点 D_i 和周围的公汽站点集 $R(s_i)$ 抽象为同一新站点 S_k，把已知公汽线路到达 $R(s_i)$ 都映射到 S_k，计算新直达数据库 Q^D，再结合地铁的费用与地汽换乘等待时间就可以把地铁线与公汽线结合，建立多目标 **0-1** 整数线性规划模型(见 6.2.3 模型Ⅱ)；对于转乘次数少于等于 2 次的方案仍可通过邻接算法求解；对于邻接算法不易求解的多次转乘最优方案，虽然模型规模较大但约束与目标线性程度较好，还可用 ***Lingo*** 软件求解得出 6 条线路的全局最优解；综合方案集(见 6.3.2 表 2.1~2.6)，其中 6 条线路时间最短目标分别为 65、102、98、56.5、89.5、30(分钟)；随后我们在 6.4 与 6.5 中给出了模型具体的评价与应用。

第三问综合考虑所有站点间步行与乘车情况，将其抽象为最短路问题下的叠加有向赋权图，在此基础上以换乘次数为主要约束，以总行程时间(包括步行)最短、转站车辆始发数最大、转乘站点负载量最小、费用最低为目标，建立多目标 **0-1** 整数线性规划模型(见 7.3 模型Ⅲ)，并给出了求解的一般步骤与算法。

最后本文还对实现查询系统的具体方案给出了建议，对各模型在实际中的应用价值进行了详细讨论，并提出了改进方案。

关键字：　邻接算法　有向赋权图　直达队列表　分层序列法　叠加有向赋权图

1 问题重述

我国人民翘首企盼的第 29 届奥运会明年 8 月将在北京举行，届时有大量观众到现场观看奥运比赛，其中大部分人将会乘坐公共交通工具（简称公交，包括公汽、地铁等）出行。这些年来，城市的公交系统有了很大发展，北京市的公交线路已达 800 条以上，使得公众的出行更加通畅、便利，但同时也面临多条线路的选择问题。针对市场需求，某公司准备研制开发一个解决公交线路选择问题的自主查询计算机系统。

为了设计这样一个系统，其核心是线路选择的模型与算法，应该从实际情况出发考虑，满足查询者的各种不同需求。请你们解决如下问题：

1、仅考虑公汽线路，给出任意两公汽站点之间线路选择问题的一般数学模型与算法。并根据附录数据，利用你们的模型与算法，求出以下 6 对起始站→终到站之间的最佳路线（要有清晰的评价说明）。

(1)、S3359→S1828　　(2)、S1557→S0481　　(3)、S0971→S0485

(4)、S0008→S0073　　(5)、S0148→S0485　　(6)、S0087→S3676

2、同时考虑公汽与地铁线路，解决以上问题。

3、假设又知道所有站点之间的步行时间，请你给出任意两站点之间线路选择问题的数学模型。

2 问题分析

本题主要在三种不同情况下，研究任意两站点之间的线路选择问题。联系实际，公众乘坐公交车主要考虑的因素包括转乘次数、行程时间、车站始发情况、车站的车次负载量及乘车费用等因素。为满足一般公众的乘车需求，主要按照公众对不同乘车信息的重视程度，确定出最佳的乘车路线。

仅考虑公汽线路的情况下，首先，需要根据题目给出的公交线路信息数据，对每条线路进行抽象处理，将分上下行的线路、双向行驶的线路和环行线路抽象为两条。然后，主要考虑公众最关心的乘车因素，即转乘次数。在最少转乘次数的基础上考虑共众对其他因素的需求，按照先后顺序考虑行程时间、车站始发情况、车站的车次负载量及乘车费用，给出供公众选用的多种参考方案。并考虑以时间为主要目标的情况下，建立最优化模型确定任意两站点行程时间最短的方案。

在考虑问题二的情况下，根据地铁与邻近站点可换乘的信息，可将每个地铁站点及其对应的所有公交站点抽象成一个点处理。对于两条地铁线路可按照与问题一相同的抽象方法处理。在此基础上按照相同的思路确定任意两站点间的最佳方案。

考虑公交及地铁站点的实际分布情况，有时会出现步行小段距离再转车的情况更能节省时间或转车次数。因此，研究此种情况下的出行方案对节省出行时间具有重要的实际意义。

3 模型假设

[1] 假设车站不重名；

[2] 假设各路经上公交车发车频度相同；

[3] 假设相邻站点间平均行驶时间一定；

[4] 假设不出现交通阻塞，公交运行顺畅；

[5] 假设不出现车辆故障及道路交通事故；

[6] 假设始发终点出入地铁需要步行4分钟；

[7] 假设公交准点到达，不考虑红绿灯等待时间。

4 符号系统

x_{ij} —— 弧(i,j)是否在该有向赋权图上；　　t_{ij} —— 站点$i \to j$的总乘车时间；

f_{ij} —— 第i个站点是否为$i \to j$的始发站；　　P_{ij} —— 站点$i \to j$的乘车费用。

5 公汽站点之间线路选择模型（问题一）

本节主要研究任意两公汽站点间线路选择的数学模型与算法。分别在不同行程需求下推荐最佳路线，给出更为人性化的站点负载压力及转乘点是否为始发站的提示。

- 通畅、便利目标：换乘次数最少；
- 不同的行程需求：行程耗时最少；
  行程费用最少；
- 人性化查询设计：转乘站点是否为始发站提示；
  站点负载压力提示。

基于此，本部分共分如下四小节：

5.0：数据处理，将三种公汽线路进行了图论抽象处理；

5.1：建立了可查询两站之间直达线路的“公汽直达数据库 **Q**”；

5.2：建立基于有向赋权图与最短路理论的 0-1 规划模型；

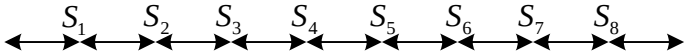
5.3：针对模型分别使用不同方法求解，得到多种适合不同用户的方案集。

5.0 数据处理——三种公汽线路抽象处理

根据题中信息，公汽线路分三种，下面将这三种线路进行数据处理：

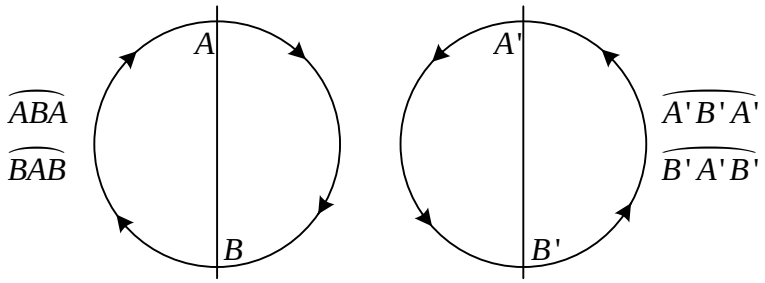
(1) 下行线、上行线原路返回

这种线路有两个端点站，在两个端点之间双向行车，而且两个方向上的行车路线相同，经过同样的站点序列。由于线路的方向不同，因此，下行线和上行线可以抽象成两条线路处理。



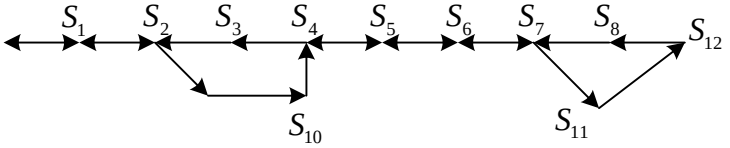
(2) 线路为环行线

实际中环形路线一般是双环，但在对这两条线路进行抽象时，为保证任意两站点距离最近，**把每条线路再抽象成 2 条**（如图所示）：



(3) 下行线与上行线经过站点不同

由于下行线与上行线经过站点不同，显然，该种线路需要抽象成两条线路处理。



5.1 “公汽直达数据库 Q”的建立

从实际出发，结合公众出行心理，公汽线路选择应优先考虑两站点之间是否有直达

车，那么在查询系统内部应设有任两站点的直达线路表，以方便查询时优先快速查询是否有直达车，若有，则直接输出所有直达车辆；若无，再搜索换乘路线。

在建立 Q 的时候，数据结构的选择非常重要，本题共有 3957 个站点,若 Q 内每个队列的每个数据都使用双精度实型储存，实际在 MATLAB 等软件内内存占用大约为 2GB，这显然超越现阶段个人电脑极限，所以根据实际情况可以采用不同数据结构，本文采用 MATLAB 内建元胞结构，当元胞内队列不存在时不占用空间，具体元胞结构设计如下：

<table>
<tr><td>Cell{1,1}</td><td>Cell{1,2}
<table><tr><th>车号</th><th>费用</th><th>耗时</th></tr><tr><td>L001</td><td>2</td><td>27</td></tr><tr><td>L076</td><td>3</td><td>18</td></tr></table></td><td>Cell{1,3}</td></tr>
<tr><td>Cell{2,1}</td><td>Cell{2,2}</td><td>Cell{2,3}</td></tr>
</table>

图 1.1 元胞结构示意图

上图中 Cell{1,2}代表 Q 中第 1 行第 2 个元胞（即从站点 S0001 到站点 S0002 的直达公交线路信息），元胞中队列的每一行代表一辆直达车信息。

5.2 模型Ⅰ分析与建立

从查询系统设计角度考虑，当输入起讫点后，系统内部通过 Q 查询无结果时，系统内部应自动搜寻换乘次数最少的路线，若换乘次数相同时有多种转乘方案，则系统应显示所有转乘路线方案（包括转乘次数、行程总时间、途径总站点数、转乘站点及路线、是否始发、行程总费用、转承站点负载压力）以供查询者自主选择。

同时，系统应向查询者推荐“时间最短”,“费用最省”,“转乘站始发站最多”,“负载压力最小”的不同目标下的最佳路线。

5.2.0 基于不同目标的有向赋权图定义

引用图论相关知识，将题中所提供的公汽网络抽象成一个有向赋权图 $\vec{G}=(V,E,W)$, $\vec{G}$ 中的每个顶点为每个不同的站点，如果从 $\vec{G}$ 中的顶点 V_i 到 V_j 有直达路线，那么这两点之间就用有向边相连，记做 $(i,j)\in E$，方向为从 i 指向 j，表示可从 i 直达 j，相应的有一个数 $w(v_i,v_j)$ 称为该有向边的权，这样公汽网络就抽象为一个有向赋权图。赋权图中的权可根据不同的目标进行定义，本模型在确定不同目标时，将其分别定义为（具体分析见 5.2.2）

时间：$W^t=\left(t_{ij}\right)_{n\times n}$ 其分量为 $t_{ij}=\begin{cases}t_{(v_i,v_j)} & \text{站点}v_i\text{至站点}v_j\text{的直达时间}\\ +\infty & \text{无直达线路}\end{cases}$

费用：$W^P=\left(P_{ij}\right)_{n\times n}$ 其分量为 $P_{ij}=\begin{cases}P_{(v_i,v_j)} & \text{站点}v_i\text{至站点}v_j\text{的直达费用}\\ +\infty & \text{无直达线路}\end{cases}$

始发：$W^f=\left(f_{ij}\right)_{n\times n}$ 其分量为 $f_{ij}=\begin{cases}f_{(v_i,v_j)} & \text{站点}v_i\text{至站点}v_j\text{的直达线路是否始发}\\ +\infty & \text{无直达线路}\end{cases}$

负载：$W^r=\left(r_i\right)_{n\times 1}$ 其分量为 $r_i=\begin{cases}r_{(v_i)} & \text{站点}v_i\text{的负载量}\\ +\infty & \text{无直达线路}\end{cases}$

5.2.1 最少换乘次数的确定

在用户输入起点与终点后，系统需要立即给出至少要转乘几次才能够到达目的地，这样就需要建立以下矩阵。

统计 Q 中各元素长度，可得任意两站点的直达线路数。由此可构造表示两两站点间直达路线数目的直达线路数矩阵 $A=\left(a_{ij}\right)_{n\times n}$，通过矩阵运算可得到任两站点间换乘线路数矩阵，进而得到任两站点间的**最少换乘次数矩阵** $C=\left(c_{ij}\right)_{n\times n}$，从而可得任两站间所需的最少换乘次数。

1）直达线路数矩阵的建立

引入直达线路数矩阵 $A=(a_{ij})$，其矩阵元素 a_{ij} 表示第 i 个站点到第 j 个站点直达线路数 n，其中，当 $i=j$ 时为无效路径，设定值为 0，即：

$$a_{ij}=\begin{cases} n & (i\neq j) \\ 0 & (i=j) \end{cases} \tag{1.1}$$

以 N 表示所有公汽所经过的的站点总数，则直达线路数矩阵可表示为：

$$A_{N\times N}=\begin{pmatrix} a_{11} & a_{12} & \cdots\cdots & a_{1N} \\ a_{21} & a_{22} & \cdots\cdots & a_{2N} \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & a_{ij} & \vdots \\ \vdots & \vdots & & \vdots \\ a_{N1} & a_{N2} & \cdots\cdots & a_{NN} \end{pmatrix}$$

2）换乘线路数矩阵的建立

直达矩阵 A 为 $N\times N$ 阶方阵，矩阵 A 的 2 次幂中元素表示任两站点间通过 1 次转乘的路线数，即：

$$A^2=A\cdot A$$

其中，A_{ij}^2 表示第 i 个站点到第 j 个站点通过 1 次转乘的路线数，下面以 A^2 第 1 行第 2 个元素 a_{12}^2 为例对 A^2 中元素意义进行举例说明：

《例》：　　$a_{12}^2=a_{11}\cdot a_{12}+a_{12}\cdot a_{22}+a_{13}\cdot a_{32}+\cdots\cdots+a_{1N}\cdot a_{N2}$

假设上式中等号右边仅 $a_{13}=1$、$a_{32}=1$，其余为 0，说明仅第 1 个站点可直达到第 3 个站点，第 3 个站点可直达到第 2 个站点，那么 $a_{12}^2=1$，即第 1 站点可通过一次换乘到达 2 站点，换乘点为站点 3。

以 A^n 表示方阵 A 的 n 次幂，A_{kj} 为站点 $k\rightarrow j$ 的直达路线数，则：

$$A_{ij}^n=\sum_{k=1}^{N}A_{ik}^{n-1}\cdot A_{kj} \tag{1.2}$$

其中，元素 A_{ij}^n 为通过 $(n-1)$ 次换乘从站点 $i\rightarrow j$ 的线路数。如 $A_{4,3}^3=1$ 表示从站点 4 到站点 3 有 1 条两次换乘路线，其换乘站点可通过运算参数记录得到。

3）最少换乘次数矩阵的建立

引入矩阵 $B=(b_{ij})$，其矩阵元素 b_{ij} 为使得 $A_{ij}^{n}\neq 0$ 的 n 的最小值，$n\in[1,\infty)$，即：

$$b_{ij}=\begin{cases}\min\quad\left\{n\mid A_{ij}^{n}\neq 0,n\in[1,\infty)\right\} & (i\neq j)\\ 0 & (i=j)\end{cases} \tag{1.3}$$

则 $b_{ij}-1$ 表示从站点 $i\rightarrow j$ 必要的最少换乘次数，以矩阵 $C=(c_{ij})$ 表示最少换乘次数矩阵，则：

$$C=B-1 \tag{1.4}$$

其中，元素 c_{ij} 表示从站点 $i\rightarrow j$ 必要的最少换乘次数。

基于最少换乘次数矩阵 C，每对起始站→终到站的最少换乘次数如下：

表 1.0 最少换乘次数表

线路编号	1	2	3	4	5	6
起始站	S3359	S1557	S0971	S0008	S0148	S0087
终到站	S1828	S0481	S0485	S0073	S0485	S3676
最少换乘	1	2	1	1	2	1

5.2.2 基于最短路理论的模型分析

我们结合实际，主要考虑用户如下几个需求因素：

目标一：换乘次数最少；
目标二：行程时间最短；
目标三：行程费用最少；
目标四：转乘车辆始发最多；
目标五：站点负载压力最小。

目标分析

目标一：换乘次数最少

基于 5.2.0 建立的有向赋权图，引入 0-1 决策变量 x_{ij} 表示弧 (i,j) 是否在起点与终点的路上，则：

$$x_{ij}=\begin{cases}1 & \text{弧}(i,j)\text{位于顶点}v_i\text{至顶点}v_j\text{的路上}\\ 0 & ELSE\end{cases}$$

若 v_i 与 v_j 之间无直接相连的弧，但可通过中间节点转跳，表明由站点 i 到 j 之间不可直达，但可通过转乘到达，则由 v_i 到 v_j 之间换乘次数为经过的总弧数减 1，即换乘次数最小可表示为：

$$Min\quad\sum_{(i,j)\in E}x_{ij}-1 \tag{1.5}$$

目标二：行程总时间最短

时间权值 $W^{t}=\left(t_{ij}\right)_{n\times n}$，则乘车总时间为：

$$\sum_{(i,j)\in E}t_{ij}x_{ij}$$

公汽换公汽时间固定是 5 分钟，则换乘时间为：

$$5\left(\sum_{(i,j)\in E} x_{ij}-1\right)$$

行程总时间包括起始站点等待的 3 分钟，行程总时间最短表示为：

$$Min \quad \sum_{(i,j)\in E} t_{ij}x_{ij}+5\left(\sum_{(i,j)\in E} x_{ij}-1\right)+3 \tag{1.6}$$

目标三：行程总费用最少

设 q_{ij} 表示 $i\to j$ 车辆属性

$$q_{ij}=\begin{cases}1 & \text{表示单一票制1元}\\ 2 & \text{分段计价}\end{cases}$$

设 s_{ij} 表示 $i\to j$ 所过站数，那么 $i\to j$ 直达费用权 $W^P=\left(P_{ij}\right)_{n\times n}$ 表示为：

$$P_{ij}=\begin{cases}1 & q_{ij}=1\\ 1 & q_{ij}=2, s_{ij}\in[1,20]\\ 2 & q_{ij}=2, s_{ij}\in[21,40]\\ 3 & q_{ij}=2, s_{ij}\in[41,+\infty]\end{cases}$$

行程总费用最少可表示为：

$$Min \quad \sum_{(i,j)\in E} P_{ij}x_{ij} \tag{1.7}$$

目标四：转乘车辆始发最多

为考虑所选路线中转乘站点是否为所需转乘车始发站，我们引入 0-1 变量 f_{ij} 表示第 i 个站点是否为 $i\to j$ 的始发站，即始发权 $W^f=\left(f_{ij}\right)_{n\times n}$：

$$f_{ij}=\begin{cases}1 & \text{第}i\text{个站点是弧}(i,j)\text{线路的始发站}\\ 0 & ELSE\end{cases}$$

从 $i\to j$ 个站点的路线中转乘点为所转乘车的始发点最多的路线可表示为：

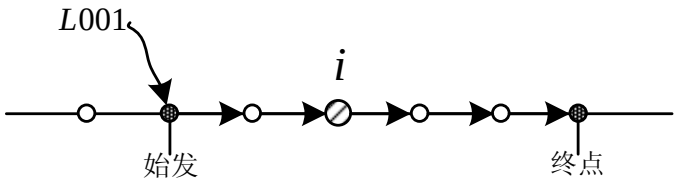
$$Max \quad \sum_{(i,j)\in E} f_{ij}x_{ij} \tag{1.8}$$

目标五：站点负载压力最小

首先假设终点站是奥运场馆，乘坐公车的人大多数都到达终点站，因此转车站点离始发站的站点数越少越好(人少)：

负载压力 = 转乘站点离始发站的站点数 — 转乘站点离终点站的站点数

注：若终点站不是奥运场馆则可以通过对负载取绝对值表示离始发或终点越近转车越方便。



如图所示，站点 i 的负载压力 = 2-3 = -1，显然负载越小越好。根据式 1.1, a_{ji} 表示进入第 i 个站点的径数，a_{ij} 表示从第 i 个站点出站的径数矩阵，以 r_i 表示第 i 个站点的负载压力权 $W^r=(r_i)_{n\times 1}$：

$$r_i=\sum_{j=1}^{N}(a_{ij}-a_{ji})$$

线路负载压力最小可表示为：

$$Min \quad \sum_{(i,j)\in E} r_i x_{ij} \tag{1.9}$$

约束分析

1） 换乘次数约束

基于对目标一的分析，可得在有向赋权图中换乘次数表达式，以 c 表示乘客所能接受的最大换乘次数，则换乘次数约束可表示为：

$$\sum_{(i,j)\in E} x_{ij}-1\le c \tag{1.10}$$

$$c\in[0,\infty) \text{ 且为整数}$$

其中，参数 c 为人为设定值，分以下三种情况：

[1] 当设定 $c=0$ 时，为严格约束不能换乘；

[2] 当设定 $c=\infty$ 时，为无乘车次数约束，即可无限次换乘；

[3] 当设定 c 为不为 0 的常数 C 时，为约束换乘次数在 C 次以内的情况；

《注》：参数 c 可根据不同的计算需求进行自由选取。仅从数学模型角度考虑，为使模型更具通用性，c 的选取可到 ∞。

从实际角度出发，查询系统中的 c 值可由查询用户自己设定，当最小换乘次数小于 b_{ij} 时，输出无解。

2）最短路起讫点约束

由于 $\vec{G}$ 为有向图，则其中顶点分为“起点”、“中间点”、“讫点”三类，对于起点只有出的边而无入的边，对于中间点既有入的也有出的边，对于讫点只有入的无出的边。

对有向图而言，若求顶点 $s\to e$ 的最短路径，以 x_{ij} 表示进入第 j 个顶点的边，以 x_{ji} 表示出第 j 个顶点的边，则 $s\to e$ 中的三类点约束可表示为：

$$\sum_{\substack{j=1\\(i,j)\in E}}^{n} x_{ij}-\sum_{\substack{j=1\\(i,j)\in E}}^{n} x_{ji}=\begin{cases}1 & (i=s)\\ -1 & (i=e)\\ 0 & (i\ne s,e)\end{cases} \tag{1.11}$$

5.2.3 模型Ⅰ建立

基于5.2.2分析，以（1.5)～(1.9)为目标，以(1.10)、(1.11)为约束，建立多目标最短路模型0-1规划表达式如下（s为起点，e为终点)：

$$Min \quad \sum_{(i,j)\in E} x_{ij} - 1$$

$$Min \quad \sum_{(i,j)\in E} t_{ij}x_{ij} + 5\left(\sum_{(i,j)\in E} x_{ij} - 1\right) + 3$$

$$Min \quad \sum_{(i,j)\in E} P_{ij}x_{ij}$$

$$Max \quad \sum_{(i,j)\in E} f_{ij}x_{ij}$$

$$Min \quad \sum_{(i,j)\in E} r_i x_{ij}$$

$$S.T.\begin{cases} \sum_{(i,j)\in E} x_{ij} - 1 \le c & (1.10) \\ \sum_{(i,j)\in E}^{n} x_{ij} - \sum_{(i,j)\in E}^{n} x_{ji} = \begin{cases} 1 & (i=s) \\ -1 & (i=e) \\ 0 & (i\ne s,e) \end{cases} & (1.11) \\ x_{ij} \in \{0,1\} \\ (i,j) \in E \end{cases}$$

模型说明：

(1.10) 换乘次数约束，表示转乘次数应在乘客所能接受的最多转乘次数。

(1.11) 最短路规划中的起讫点约束，其中s为起点，e为讫点。

x_{ij} —— 弧(i,j)是否在该路径上；

t_{ij} —— 站点$i \to j$的总乘车时间；　　f_{ij} —— 第i个站点是否为$i \to j$的始发站；

r_i —— 站点i的负载压力；　　P_{ij} —— 站点$i \to j$的乘车总费用；

c —— 人为设定参数，表示乘客可接受的最多换乘次数，详见5.2.2约束分析。

5.3 模型Ⅰ的求解（程序见附件1.1～ 1.7）

5.3.1 模型求解的2种方法

方法一、基于数据库Q与Dijkstra算法的“邻接算法”求解

Step 1：输入乘车始点i终点j，（注：$C(c_{ij})_{n\times n}$为最少换乘次数矩阵，）

若$c_{ij}=0$则有直达线路，输出Q中所有直达信息，结束程序，

若$c_{ij}=1$则有转乘1次线路，转*Step* 2，

若$c_{ij}=2$则有转乘2次线路，转*Step* 4，

若$c_{ij}>2$则存在转乘c_{ij}次线路，但全局计算时间复杂度较高，终止邻接算法，采用*Lingo*求解；

Step 2：求线路*s(i)*的直达站点*E(i,U)*,及线路*t(j)*的直达站点*F(j,V)*；

Step 3：若存在 *E(i,U)=F(j,V)*，线路 *s(i)*、*t(i)*可能不止一种，即为一次转车的线路,保存队列 U1，转 *Step6*；

Step 4：求经过*E(i,U)*的线路*r(K)*求线路*r(K)*的站点*G(K,W)*；

Step 5：若存在 *G(K,W)=F(j,V)*，线路 *s(i)*、*t(j)*、*r(K)*可能不止一种，即为两次转车的线路，保存队列 U2，转 *Step6*;

Step 6：修改队列 U1、U2 中的成员，按其属性(路过的站点数，乘坐的车辆)根据不同目标计算总行程时间、费用等；

在邻接算法内不考虑转乘 2 次以上主要原因是我们实际上并不是简单计算最短路径，而是把较优方案都记录在 U1、U2 中，这对用户多需求是吻合的，但这样对复杂度为 $O(n^2)$ 的算法来说还是不可行的，但我们权衡了二者的重要性决定用方案的丰富性取代计算的准确性，而且从实际出发用户也不一定非常关心转乘次数大于 2 次的方案。虽然在本算法内我们不考虑转乘次数大于 2 次的方案，但最后我们从规划角度使用 lingo 软件求解了全局最优值，等同于弥补了邻接算法的缺陷。

在计算机空间使用方面，我们通过对一些整形数组使用 16 位无符号整形，定义稀疏元胞数组缩小空间占用，最终求解成功。

方法二、使用 *Lingo* 软件求解无转乘次数限制的方案（针对不同目标分别求解）

邻接算法可以求出多种方案,但对于转乘次数大于 2 的情况无法在有限时间内求解。作为一个全面的查询系统必须非常全面，我们认为还是有一些用户会转乘多次且需要时间最短是他们的最重要目的（前提是在代价低于使用专车费用），而且从理论角度考虑仍需要找到求解转乘 n 次时的可行方法。

在求解上述规划模型时，通过基表 Q 建立的数量非常庞大，采用传统求解方法时不可行，但其中有大量 0 元素可以在 *Lingo* 软件内通过稀疏矩阵实现，但求解时间仍然需要 20 分钟左右，主要为数据导入时间（19.9 分钟），只要开始迭代计算后由于目标与约束线性，*Lingo* 只需要 6 秒左右即可求解出**全局最优解**（具体程序见附录 1.6）。

5.3.2 用户终端报表呈现系统：多目标分层序列排序

由于可行方案集 U1、U2 中的数据出现顺序不一定是用户所期望的，所以有必要根据不同的用户需求对其排序，本文采用多目标分层序列法对方案集排序，实质上就是按关键字顺序依次排序，即对于一个 M 个字段的表按照给定的 N 个字段排序（在数据库软件里可以通过标准 *SQL* 语句直接操作），下面给出算法的文字描述：

1）按第一关键字（即一层目标）对列表排序；
2）按第二关键字对列表每个第一关键字相同的组进行排序;
3）按第三关键字对列表每个第二关键字相同的组进行排序;
4）按第 N 关键字对列表每个第 N-1 关键字相同的组进行排序。

采用多目标分层序列法排序，输出按照用户需求的报表样式，具体方案见下面表格。（默认排序顺序是转乘次数、总时间、总费用、始发站点数、负载，可以根据不同用户需求而改变）

方案报表说明：每行代表一种方案，表中加黑字格表示该方案该项指标全局最优；
表中总时间已包括起始点等车 3 分钟。

表 1.1　一线 S3359→S1828 部分出行方案列表

求解方法	转乘	总时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	转站点始发数	总负载	总费用
Lingo/邻接	**1**	104	1784	-	436	167	-	0	-20	**3**
邻接	1	104	1784	-	436	217	-	0	-20	3
邻接	1	140	2364	-	469	217	-	**1**	-1	3
邻接	1	143	519	-	469	167	-	0	-9	4
Lingo/邻接	**2**	**67**	3697	1784	484	485	167	0	**-24**	3
邻接	2	67	3697	1784	484	485	217	0	-24	3
邻接	2	67	2027	1784	324	485	167	0	-16	3
邻接	2	67	2027	1784	324	485	217	0	-16	3
邻接	2	67	2903	1784	15	485	167	0	-9	3
邻接	2	67	2903	1784	15	485	217	0	-9	3

表 1.2　二线 S1557→S0481 部分出行方案列表

求解方法	转乘	总时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	转站点始发数	总负载	总费用
Lingo	3	**102**	1919,3186,903		L84,L189,L91,L239			-	-	4
Lingo/邻接	**2**	109	1919	3186	84	189	460	0	-113	**3**
邻接	2	109	1919	3186	363	189	460	0	-113	3
邻接	2	115	1919	2424	84	417	254	0	-112	3
邻接	2	115	1919	2424	84	417	312	0	-112	3
邻接	2	118	1919	992	84	417	460	0	**-126**	3
邻接	2	118	1919	992	363	417	460	0	-126	3
邻接	2	130	1919	417	84	497	460	**1**	-90	4
邻接	2	130	1919	417	363	497	460	1	-90	4
邻接	2	133	3389	1427	84	454	447	1	77	4

表 1.3　三线 S0971→S0485 部分出行方案列表

求解方法	转乘	总时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	转站点始发数	总负载	总费用
邻接	**1**	131	2184	-	13	417	-	0	6	**3**
邻接	1	146	2119	-	13	395	-	0	11	3
邻接	1	152	1739	-	119	417	-	0	-1	3
Lingo/邻接	2	**106**	2517	2159	13	290	469	0	-2	3
邻接	2	109	1609	2654	13	140	469	0	-33	3
邻接	2	109	1609	2654	24	140	469	0	-33	3
邻接	2	124	2324	2482	13	132	417	**1**	17	4
邻接	2	124	2324	2482	13	242	417	1	17	4
邻接	2	124	2324	2480	13	132	417	1	24	4
邻接	2	124	1520	2265	119	8	469	0	**-52**	3

表 1.4　四线 S0008→S0073 部分出行方案列表

求解方法	转乘	时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	始发数	负载	总费用
Lingo	4	**62**	3766,2085,483,525		L198,L476,L17,L328,L103			-	-	5
邻接	**1**	86	2263	-	355	345	-	0	48	**2**
邻接	1	89	2302	-	355	57	-	0	-16	2
邻接	1	113	3415	-	463	118	-	0	-46	2
邻接	1	131	3915	-	463	118	-	**1**	9	2
Lingo/邻接	2	70	1691	2184	198	290	345	0	0	3
邻接	2	70	1383	2184	43	290	345	0	37	3
邻接	2	79	630	1659	159	231	459	1	**-83**	3
邻接	2	79	630	1659	159	381	459	1	-83	3
邻接	2	79	854	1659	159	231	459	1	-60	3

表 1.5　五线 S0148→S0485 部分出行方案列表

求解方法	转乘	总时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	转站点始发数	总负载	总费用
Lingo	3	**105**	3604,2361,2210		L308,L81,L156,L417			-	-	4
Lingo/邻接	**2**	109	36	2210	308	156	417	**1**	-29	**3**
邻接	2	112	36	2482	308	157	417	1	-47	3
邻接	2	118	3604	1381	308	129	469	0	-21	3
邻接	2	118	3604	2026	308	123	469	0	-14	3
邻接	2	118	3604	1383	308	129	469	0	24	3
邻接	2	121	3604	2840	308	454	417	0	-22	3
邻接	2	121	302	2027	308	427	469	0	-13	3
邻接	2	121	3604	1321	308	129	469	0	3	3
邻接	2	124	3604	2079	308	206	417	0	**-73**	3

表 1.6　六线 S0087→S3676 部分出行方案列表

求解方法	转乘	总时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	转站点始发数	总负载	总费用
Lingo/邻接	**1**	68	3496	-	454	209	-	0	-36	**2**
Lingo/邻接	2	**49**	88	427	21	231	97	0	-52	3
邻接	2	49	88	427	206	231	97	0	-52	3
邻接	2	49	88	427	454	231	97	0	-52	3
邻接	2	52	630	427	21	381	97	0	-44	3
邻接	2	52	854	427	293	231	97	0	-21	3
邻接	2	52	1427	427	21	381	97	0	-12	3
邻接	2	55	541	2336	454	120	462	0	**-68**	3
邻接	2	61	3874	280	21	68	462	**1**	-13	3
邻接	2	73	3874	274	21	68	462	1	-12	3

用户选线指南：

上面表中已按多目标分层序列法的默认目标排序（分别是表中转乘次数、总时间、转车站点是否始发、转车站点总负载量、总费用五个字段），一般用户只需要从上到下选取即可，但如果用户希望在转站时乘坐始发车（有座位）那么可以挑选始发字段为 1、2 的方案，若希望转站时人较少的地方则可以考虑选则站点负载较小的方案。综述，本模型Ⅰ求解的方案集使用于所有用户，具有很强的实用价值。

5.4 模型Ⅰ的评价

5.4.1 邻接算法评价

1) 建立在图论基础下能够求解出转乘次数不超过两次时的所有可行方案，并可根据公众的不同需求，给出最佳需要方案，从此角度考虑，模型实用性较强；

2) 模型求解基于直达队列 Q，采用空间换取时间思想，适合查询系统设计标准能够较强的适应工程应用；

3) 在转乘次数超过两次的情况下，运用本模型求解计算过程复杂，计算量过大；故本模型存在一定的局限性。

5.4.2 0-1 规划 *Lingo* 求解方案评价

1) 在不限制最小转乘数时可以求得全局最优解，这是其他所有算法无法达到的，例如在第 2、4、5 条线路上其转车次数为 3、4、3，但是耗时相对转 2 次的要节省许多；

2)在限制最小转乘数时可以求得与邻接算法同样的方案，表明模型的通用性较强，但无法像邻接算法一样求解多种方案是用户所不能接受的；

3) 从理论角度分析，最优化模型规划角度可解具有很强的实际意义，例如从全国范围考虑求解，那么转车 3~4 次也是可以接受的，只要耗时足够短；

4) 从计算时间来分析，尽管需要 20 分钟，但大部分时间为数据导入，只有 1%的时间是真正计算耗时，如果将所需数据存放入内存不变，其求解速度将超越邻接算法；

5) 但 *Lingo* 不能求解出多种方案，实用性不如邻接算法。

6 同时考虑公汽与地铁最佳线路选择模型（问题二）

本问为综合考虑公汽与地铁线路的情况，解决查询系统中混合最佳路径选择问题的模型与算法。

6.0 数据处理——公交网简化模型

1）将可互换站点抽象处理为一个站点

题中给出了地铁换乘公汽的数据文件，由地铁与公汽互换的时间来看，可互换的两站间地理位置应非常接近且容易换乘，定义这些站点为紧邻站点，可将这些可互换的紧邻站点抽象为一个站点，使问题得到简化。

〈例〉：信息数据第一行为“$D01$：$S0567$，$S0042$，$S0025$”，则可认为这四个站点实际距离非常近，为紧邻站点，所以可看做一个点处理，示意图如下：

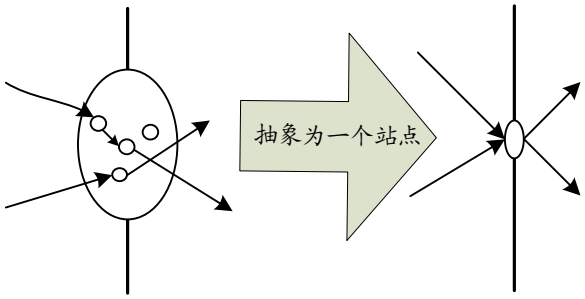


图 2.1　可互换站点抽象为一个站点示意图

基于这种思想，根据题目中给出关于地铁换乘公交的信息数据，可以将各地铁站点及其紧邻站点在整个交通网络中抽象为一个点处理。

2）两种地铁线路抽象处理

基于 5.1.0 对三种公汽线路的抽象方法，以相同的方法对两地铁线路$T1$、$T2$进行抽象处理如下：

$T1$：为双向线路，故可以根据不同的方向将其抽象为两条单向行驶线路。

$T2$：为环行线路，实际中环形路线一般是对开，故该种线路可以抽象成两条线路处理。

6.1 “公汽、地铁直达数据库Q^D”的建立

将紧邻站点处理为一个新站点，则当综合考虑公汽与地铁时，建立的Q^D实质上是新站点与新站点间的直达路线集。认为在新站点所代表的站点集中的任意站点可通过步行到达，且时间忽略。则当用户输入起、讫点后，系统内部首先自动查找这两点所属的新站点，再查找新站点间可直达的线路，并给出起点及其附近站点可直达讫点及其附近站点的路线。

采用与 5.1 相同的思路及方法，把已知公汽线路到达$R(s_i)$都映射到S_k，计算新直达数据库Q^D，再结合地铁的费用与地汽换乘等待时间就可以把地铁线与公汽线结合。

具体元胞结构设计图如下：

<table>
<tr><td>Cell{1,1}</td><td>Cell{1,2}<table><tr><th>车号</th><th>费用</th><th>耗时</th></tr><tr><td>L001</td><td>2</td><td>30</td></tr><tr><td>T001</td><td>3</td><td>22.5</td></tr></table></td><td>Cell{1,3}</td></tr>
<tr><td>Cell{2,1}</td><td>Cell{2,2}</td><td>Cell{2,3}</td></tr>
</table>

图 2.2 元胞结构示意图

上图中 Cell{1,2}代表直达队列表中第 1 行第 2 个元胞（即从站点 S0001 到站点 S0002 的直达混合线路信息），元胞中队列的每一行代表一辆直达车信息。

6.2 模型Ⅱ的分析与建立

经过数据处理后，紧邻站点被处理为一个新站点，该站点可等同看作问题一中的公汽站点，当用户输入起、讫点后，系统内部通过直达线路队列表查询无结果时，则搜寻转乘路线方案。同时，系统向查询者推荐不同目标下的最佳路线及转乘方案。

6.2.1 最少换乘次数的确定

采用与 5.2.1 相同的建模思路及方法，统计Q^D中各元素长度，可得任意两站点的直达线路数。由此可构造表示两两站点间直达路线数目的直达线路数矩阵A'，可确定换乘线路数矩阵：

$$A_{ij}'^{n}=\sum_{k=1}^{N'}A_{ik}'^{n-1}\cdot A_{kj}' \tag{2.1}$$

其中，元素$A_{ij}'^{n}$为通过$(n-1)$次换乘从站点$i\rightarrow j$的线路数。其换乘站点可通过运算参数记录得到。

进而确定最少换乘次数矩阵：

$$b_{ij}'=\begin{cases}\min\quad\left\{n\mid A_{ij}'^{n}\neq 0,n\in[1,\infty)\right\} & (i\neq j)\\ 0 & (i=j)\end{cases} \tag{2.2}$$

$$C'=B'-1 \tag{2.3}$$

其中，矩阵C'中元素c_{ij}'表示从站点$i\rightarrow j$必要的最少换乘次数。

基于最少换乘次数矩阵C'，每对起始站→终到站的最少换乘次数如下：

表 2.0 最少换乘次数表

线路编号	1	2	3	4	5	6
起始站	S3359	S1557	S0971	S0008	S0148	S0087
终到站	S1828	S0481	S0485	S0073	S0485	S3676
最少换乘	1	2	1	1	2	**0**

6.2.2 模型分析

采用与5.2.2相同的建模思路及方法,这里在考虑地铁后仍按目标的重要程度将“换乘次数最少”、“行程时间最短”、“行程费用最少”、“转乘车辆始发最多”、“站点负载压力最小”分设为第一到五层目标，基于 5.2.2 对各目标的分析与建立，这里不再复述分析，仅在模型建立时给出具体表达式，这里由于对站点的定义与第一问不同，所以对**时间**及**费用**的计算与第一问有所不同。

我们结合实际主要考虑用户如下几个因素：

目标一：换乘次数最少；
目标二：行程时间最短；
目标三：行程费用最少；
目标四：转乘车辆始发最多；
目标五：站点负载压力最小。

公汽地铁混合网络图的赋权

通过 6.0 的简化，结合图论相关知识，将第二问公汽、地铁混合网络抽象成一个有向赋权图$\overrightarrow{G'}=(V',E',W')$，$\overrightarrow{G'}$中的每个顶点为每个不同的站点，如果从$\overrightarrow{G'}$中的顶点$V'_i$到$V'_j$有直达路线，那么这两点之间就用有向边相连，记做$(i,j)\in E'$，赋权图中的权可根据不同的目标进行定义

时间：$W^{t'}=\left(t'_{ij}\right)_{n\times n}$ 其分量为 $t'_{ij}=\begin{cases}t'_{(v_i,v_j)} & \text{站点}v_i\text{至站点}v_j\text{的直达时间}\\ +\infty & \text{无直达线路}\end{cases}$

费用：$W^{P'}=\left(P'_{ij}\right)_{n\times n}$ 其分量为 $P'_{ij}=\begin{cases}P'_{(v_i,v_j)} & \text{站点}v_i\text{至站点}v_j\text{的直达费用}\\ +\infty & \text{无直达线路}\end{cases}$

始发：$W^{f'}=\left(f'_{ij}\right)_{n\times n}$ 其分量为 $f'_{ij}=\begin{cases}f'_{(v_i,v_j)} & \text{站点}v_i\text{至站点}v_j\text{的线路是否始发}\\ +\infty & \text{无直达线路}\end{cases}$

负载：$W^{r'}=\left(r'_{i}\right)_{n\times 1}$ 其分量为 $r'_{i}=\begin{cases}r'_{(v_i)} & \text{站点}v_i\text{的负载量}\\ +\infty & \text{无直达线路}\end{cases}$

目标分析

目标一：换乘次数最少

基于对混合网络的抽象，引入 0-1 决策变量 y_{ij} 表示弧 (i,j) 是否在该路径上：

$$y_{ij}=\begin{cases}1 & \text{弧(i, j)位于顶点}v_i\text{到}v_j\text{的路上}\\ 0 & ELSE\end{cases}$$

若 v_i 与 v_j 之间无直接相连的弧，但可通过中间节点转跳，表明由站点 i 到 j 之间不可直达，但可通过转乘到达，则由 v_i 到 v_j 之间换乘次数为经过的总弧数减 1，即换乘次数最小可表示为：

$$Min \sum_{(i,j)\in E'} y_{ij}-1 \tag{2.4}$$

目标二：行程总时间最短

1）乘车时间（t'_{ij} 为各站点最快直达时间，基于 Q^D，包括地铁在内）：

$$\sum_{(i,j)\in E'} t'_{ij}\, y_{ij}$$

2）总等待时间：

设 Z_{ij}=3 表示 $i\rightarrow j$ 最短直达为公汽（也表示乘始发坐公汽等待 3 分钟），等于 2 为地铁（也表示始发乘坐地铁等待 2 分钟），总等待时间表示为：

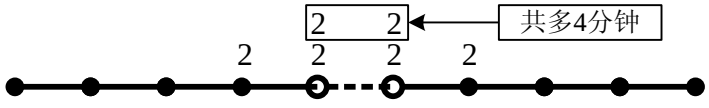
$$T1=\sum_{(i,j)\in E'} Z_{ij}y_{ij}$$

3）总步行时间：

将相同车型换乘、不同车型换乘的步行时间，一同视为 2 分钟

$$T_a=2\left(\sum_{(i,j)\in E'} y_{ij}-1\right)$$

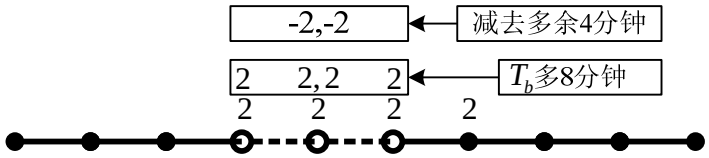
不同车型换乘多步行的 4 分钟（虚线表示地铁，空心圆表示地铁站）



表示为：

$$T_b=\sum_{\substack{Z_{ij}=2\\(i,j)\in E'}} 4y_{ij}$$

地铁转地铁是不同车型换乘的特例，且只可能在 D12 与 D18 转乘，出现这种情况在 T_b 基础上减少步行时间 4 分钟（虚线表示地铁，空心圆表示地铁站）



表示为：

$$T_c = \sum_{\substack{j=D12,D18 \\ Z_{ij},Z_{jk}=2 \\ (i,j,k)\in E'}} 4Y_{ijk}$$

在地铁直达时，需要另外加上 4 分钟出站步行时间：

$$T_d = 4y_{(s,e)} \qquad Z_{(s,e)} = 2$$

若始发乘坐地铁转公交到达终点，需要增加步行时间 2 分钟：

$$T_e = 2\sum_{\substack{i=s,\, j\neq e; Z_{ij}=2 \\ (i,j)\in E'}} y_{ij}$$

若始发乘坐公交转地铁到达终点，也需要增加步行时间 2 分钟：

$$T_f = 2\sum_{\substack{i\neq s,\, j=e; Z_{ij}=2 \\ (i,j)\in E'}} y_{ij}$$

总步行时间表示为：　$T2 = T_a + T_b - T_c + T_d + T_e + T_f$

行程总时间最短表示为（总等待时间+总步行时间+乘车时间）：

$$Min \quad T1 + T2 + \sum_{(i,j)\in E'} t'_{ij}\, y_{ij} \tag{2.5}$$

目标三：行程总费用最少

设 q'_{ij} 表示 $i \to j$ 的车辆属性

$$q'_{ij} = \begin{cases} 1 & \text{表示单一票制1元} \\ 2 & \text{表示分段计价} \\ 3 & \text{表示地铁线路} \end{cases}$$

设 s'_{ij} 表示 $i \to j$ 所过站数，那么 $i \to j$ 直达费用权 $W^{P'} = \left(P'_{ij}\right)_{n\times n}$ 表示为：

$$P'_{ij} = \begin{cases} 1 & q'_{ij} = 1 \\ 1 & q'_{ij} = 2, s'_{ij} \in [1,20] \\ 2 & q'_{ij} = 2, s'_{ij} \in [21,40] \\ 3 & q'_{ij} = 2, s'_{ij} \in [41,+\infty] \\ 3 & q'_{ij} = 3 \end{cases}$$

行程总费用最少可表示为（正常票价 － 地铁换乘免费）：

$$Min \quad \sum_{(i,j)\in E'} P'_{ij}\, y_{ij} - \sum_{\substack{j=D12,D18 \\ q'_{ij},q'_{jk}=3 \\ (i,j,k)\in E'}} 3Y_{ijk} \tag{2.6}$$

目标四：转乘车辆始发最多（地铁不考虑）

为考虑所选路线中转乘站点是否为所需转乘车始发站，我们引入 0-1 变量 f'_{ij} 表示

第i个站点是否为$i \to j$的始发站，即始发权$W^{f'}=\left(f'_{ij}\right)_{n\times n}$：

$$f'_{ij}=\begin{cases}1 & \text{第}i\text{个站点是弧}(i,j)\text{线路的始发站}\\ 0 & ELSE\end{cases}$$

从第i个站点到第j个站点的路线中转乘点为所转乘车的始发点最多的路线可表示为：

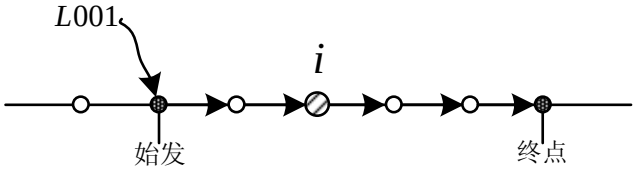
$$Max \quad \sum_{(i,j)\in E'} f'_{ij}\, y_{ij} \tag{2.7}$$

目标五：站点负载压力最小

首先假设终点站是奥运场馆，乘坐公车的人大多数都到达终点站，因此转车站点离始发站的站点数越少越好(人少)：

负载压力 = 转乘站点离始发站的站点数 — 转乘站点离终点站的站点数

注：若终点站不是奥运场馆则可以通过对负载取绝对值表示离始发或终点越近转车越方便。



如图所示，站点i的负载压力 = 2-3 = -1，显然负载越小越好。根据式 2.1, a'_{ji}表示进入第i个站点的径数，a'_{ij}表示从第i个站点出站的径数矩阵，以r'_i表示第i个站点的负载压力权$W^{r'}=\left(r'_i\right)_{n\times 1}$：　　$r'_i=\sum_{j=1}^{N}(a'_{ij}-a'_{ji})$

线路负载压力最小可表示为：

$$Min \quad \sum_{(i,j)\in E'} r'_i\, y_{ij} \tag{2.8}$$

约束分析

1）换乘次数约束

基于对目标一的分析，可得在有向赋权图中换乘次数表达式，以c表示乘客所能接受的最大换乘次数，则换乘次数约束可表示为：

$$\sum_{(i,j)\in E'} y_{ij}-1\le c \tag{2.9}$$

$$c\in[0,\infty) \text{ 且为整数}$$

其中，参数 c 为人为设定值，分以下三种情况：

[1] 当设定$c=0$时，为严格约束不能换乘；

[2] 当设定$c=\infty$时，为无乘车次数约束，即可无限次换乘；

[3] 当设定c为不为 0 的常数C时，为约束换乘次数在C次以内的情况；

《注》：参数c的可根据不同的计算需求进行自由选取。仅从数学模型角度考虑，为使模型更具通用性，c的选取可到∞。

从实际出发，查询系统中的c值可由查询用户自己设定，当最小换乘次数小于C_{ij}时，输出无解。

2）最短路起讫点约束

由于$\vec{G}$为有向图，则其中顶点分为“起点”、“中间点”、“讫点”三类，对于起点只有出的边而无入的边，对于中间点既有入的也有出的边，对于讫点只有入的无出的边。

对有向图而言，若求顶点 $s \to e$ 的最短路径，以 x_{ij} 表示进入第 j 个顶点的边，以 x_{ji} 表示出第 j 个顶点的边，则 $s \to e$ 中的三类点约束可表示为：

$$\sum_{\substack{j=1 \\ (i,j)\in E'}}^{n} y_{ij} - \sum_{\substack{j=1 \\ (i,j)\in E'}}^{n} y_{ji} = \begin{cases} 1 & (i=s) \\ -1 & (i=e) \\ 0 & (i \neq s,e) \end{cases} \tag{2.10}$$

3）地铁间换乘约束

站点 $i \to j \to k$ 间是否地铁换乘地铁，使用 Y_{ijk} 表示，那么 Y_{ijk} 与走的路径 y_{ij} 需要满足：

$$y_{ij} + y_{jk} \geq 2Y_{ijk} \qquad (i,j,k) \in E'; Z_{ij}, Z_{jk} = 2 \tag{2.11}$$

地铁转地铁情况只可能在 D12 与 D18 转乘，故换乘总次数不能够大于 2：

$$\sum_{(i,j,k)\in E'} Y_{ijk} \leq 2 \qquad Z_{ij}, Z_{jk} = 2 \tag{2.12}$$

6.2.3 模型Ⅱ建立

基于 6.2.2 分析，以 (2.4)～(2.8)为目标，以(2.9)、(2.13)为约束，建立多目标图论模型 0-1 规划表达式如下（ s 为起点， e 为终点）：

$$Min \quad \sum_{(i,j)\in E'} y_{ij} - 1$$

$$Min \quad T1 + T2 + \sum_{(i,j)\in E'} t'_{ij}\, y_{ij}$$

$$Max \quad \sum_{(i,j)\in E'} f'_{ij}\, y_{ij}$$

$$Min \quad \sum_{(i,j)\in E'} r'_{i}\, y_{ij}$$

$$Min \quad \sum_{(i,j)\in E'} P'_{ij}\, y_{ij} - \sum_{\substack{j=D12,D18 \\ q'_{ij},q'_{jk}=3 \\ (i,j,k)\in E'}} 3Y_{ijk}$$

$$s.t. \begin{cases} \sum_{(i,j)\in E'} y_{ij} - 1 \leq c & (2.9) \\ \sum_{\substack{j=1 \\ (i,j)\in E'}}^{n} y_{ij} - \sum_{\substack{j=1 \\ (i,j)\in E'}}^{n} y_{ji} = \begin{cases} 1 & (i=s) \\ -1 & (i=e) \\ 0 & (i \neq s,e) \end{cases} & (2.10) \\ \left.\begin{array}{l} y_{ij} \geq Y_{ijk} \\ y_{jk} \geq Y_{ijk} \end{array}\right\} \begin{array}{l} (i,j,k) \in E' \\ Z_{ij}, Z_{jk} = 2 \end{array} & (2.11) \\ \sum_{(i,j,k)\in E'} Y_{ijk} \leq 2 \qquad Z_{ij}, Z_{jk} = 2 & (2.12) \\ y_{ij} \in \{0,1\} \qquad (i,j) \in E' \\ Y_{ijk} \in \{0,1\} \qquad \begin{cases} (i,j,k) \in E' \\ Z_{ij}, Z_{jk} = 2 \end{cases} \end{cases}$$

模型说明：

(2.9) 换乘次数约束，表示转乘次数应在乘客所能接受的最多转乘次数。

(2.10) 最短路规划中的起讫点约束，其中 s 为起点，e 为讫点。

y_{ij} —— 弧 (i,j) 是否在该路径上；

Y_{ijk} —— 站点 $i\rightarrow j\rightarrow k$ 间是否地铁换乘地铁；

t'_{ij} —— 站点 $i\rightarrow j$ 的乘车时间；

f'_{ij} —— 第 i 个站点是否为 $i\rightarrow j$ 的始发站；

r'_{i} —— 站点 i 的负载压力；

P'_{ij} —— 站点 $i\rightarrow j$ 的乘车总费用；

c —— 人为设定参数，表示乘客可接受的最多换乘次数；

$T1,T2$ —— 总等待时间，总步行时间；

Z_{ij}=3 表示 $i\rightarrow j$ 最短直达为公汽（也表示乘始发坐公汽等待 3 分钟），等于 2 为地铁（也表示始发乘坐地铁等待 2 分钟）。

6.3 模型Ⅱ求解（程序见附录 1.1～1.9）

6.3.1 模型求解的 2 种方法

方法一、基于数据库 Q 与 Dijkstra 算法的“邻接算法”求解

在计算初始化 Q^D 后具体算法同模型Ⅰ“邻接算法”，在计算乘坐不同车型费用与换乘等待、步行时间时可以通过增加简易的判断语句实现，求解结果见表 2.1～2.6。

方法二、使用 *Lingo* 软件求解无转乘次数限制的方案（针对不同目标分别求解）

上述最优化模型规模较大，但是通过模型分析章节抽象，模型所有约束与目标都已经线性化，所以采用 ***Lingo*** 软件严格按照 **0-1** 模型求解，可求得 6 条线路**全局最优解**，具体方案分别见表 2.1 到 2.6，其中第一字段为 ***Lingo***。

求解软件环境是*Lingo10.0*并使用了CALC段编程功能，能够一次求解6个模型，故调试时不能使用低版本调试。

6.3.2 用户终端报表呈现系统：多目标分层序列排序

采用多目标分层序列法排序(排序方法见 5.3.2)，输出按照用户需求的报表样式，具体方案见下面表格。(默认排序顺序是转乘次数、总时间、总费用、始发站点数、负载，可以根据不同用户需求而改变)

方案报表说明： 每行代表一种方案，表中加黑字格表示该方案该项指标最优；
表中总时间已包括起始点公汽、地铁等车 3、2 分钟。

表 2.1　一线 S3359→S1828 部分出行方案列表

求解方法	转乘	总时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	转站点始发数	总负载	总费用
Lingo/邻接	**1**	104	'S1784'	-	'L436'	'L167'	-	0	-20	**3**
邻接	1	104	'S1784'	-	'L436'	'L217'	-	0	-20	3
邻接	1	110	'S1241'	-	'L436'	'L167'	-	0	21	3
邻接	1	140	'S2364'	-	'L469'	'L217'	-	**1**	-1	3
Lingo	4	**65**	S2903,D12,D37,S1671		L15,L201,T02,L428,L41			-	-	7
邻接	2	67	'S3697'	'S1784'	'L484'	'L485'	'L167'	0	-24	3
邻接	2	67	'S3697'	'S1784'	'L484'	'L485'	'L217'	0	-24	3
邻接	2	67	'S2027'	'S1784'	'L324'	'L485'	'L167'	0	-16	3
邻接	2	73	'D06'	'S1784'	'L484'	'L485'	'L167'	0	**-28**	3
邻接	2	79	'S1842'	'S1671'	'L123'	'L475'	'L041'	1	87	3

表 2.2　二线 S1557→S0481 部分出行方案列表

求解方法	转乘	总时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	转站点始发数	总负载	总费用
Lingo	3	**102**	1919,3186,903		L84,L189,L91,L239			-	-	4
邻接	**2**	109	'D20'	'S3186'	'L084'	'L189'	'L460'	0	**-79**	**3**
邻接	2	109	'D20'	'S3186'	'L363'	'L189'	'L460'	0	-79	3
邻接	2	112	'D20'	'S3186'	'L084'	'L189'	'L460'	0	-79	3
邻接	2	112	'D20'	'S3186'	'L363'	'L189'	'L460'	0	-79	3
邻接	2	112	'D20'	'S2424'	'L084'	'L417'	'L254'	0	-78	3
邻接	2	112	'D20'	'S2424'	'L084'	'L417'	'L312'	0	-78	3
邻接	2	112	'D20'	'S2424'	'L084'	'L417'	'L447'	0	-78	3
邻接	2	121	'D20'	'S1402'	'L084'	'L030'	'L460'	**1**	-65	3
邻接	2	121	'D20'	'S1402'	'L363'	'L030'	'L460'	1	-65	3

表 2.3　三线 S0971→S0485 部分出行方案列表

求解方法	转乘	总时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	转站点始发数	总负载	总费用
邻接	**1**	131	'S2184'	-	'L013'	'L417'	-	0	6	**3**
邻接	1	146	'S2119'	-	'L013'	'L395'	-	0	11	3
Lingo	3	**98**	**D01,D15,3351**		**L094,T001,L156,L417**			**-**	**-**	**6**
Lingo/邻接	2	99	'D01'	'D21'	'L094'	'T001'	'L051'	0	**-120**	5
邻接	2	99	'D01'	'D21'	'L094'	'T001'	'L104'	0	-120	5
邻接	2	99	'D01'	'D21'	'L094'	'T001'	'L395'	0	-120	5
邻接	2	99	'D01'	'D21'	'L094'	'T001'	'L450'	0	-120	5
邻接	2	124	'S2324'	'S2482'	'L013'	'L132'	'L417'	**1**	17	4
邻接	2	124	'S2324'	'S2482'	'L013'	'L242'	'L417'	1	17	4
邻接	2	127	'S3405'	'S2515'	'L013'	'L079'	'L417'	1	19	4

表 2.4　四线 S0008→S0073 部分出行方案列表

求解方法	转乘	时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	始发数	负载	总费用
Lingo	3	**56.5**	D15,D12,D25		L200,T001,T002,L103			-	-	5
邻接	**1**	83	'D13'	-	'L159'	'L474'	-	0	**-58**	**2**
邻接	1	86	'S3614'	-	'L159'	'L058'	-	0	-35	2
邻接	1	86	'S2083'	-	'L463'	'L057'	-	0	-11	2
邻接	1	86	'S3315'	-	'L159'	'L058'	-	0	-8	3
邻接	1	86	'S2303'	-	'L355'	'L345'	-	0	-4	2
邻接	2	58	'D30'	'D25'	'L150'	'T002'	'L103'	0	39	5
邻接	2	58	'D30'	'D25'	'L159'	'T002'	'L103'	0	39	5
邻接	2	63.5	'D30'	'D24'	'L150'	'T002'	'L103'	**1**	53	5
邻接	2	63.5	'D30'	'D24'	'L259'	'T002'	'L103'	1	53	5

表 2.5　五线 S0148→S0485 部分出行方案列表

求解方法	转乘	总时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	转站点始发数	总负载	总费用
Lingo	3	**89.5**	D02,D15,3351		L024,T001,L156,L417			-	-	6
Lingo/邻接	**2**	90.5	'D02'	'D21'	'L024'	'T001'	'L051'	0	**-112**	5
邻接	2	90.5	'D02'	'D21'	'L024'	'T001'	'L104'	0	-112	5
邻接	2	90.5	'D02'	'D21'	'L024'	'T001'	'L395'	0	-112	5
邻接	2	90.5	'D02'	'D21'	'L024'	'T001'	'L450'	0	-112	5
邻接	2	90.5	'D02'	'D21'	'L024'	'T001'	'L469'	0	-112	5
邻接	2	109	'S0036'	'S2210'	'L308'	'L156'	'L417'	**1**	-29	**3**
邻接	2	112	'S0036'	'D20'	'L308'	'L157'	'L417'	1	-89	3
邻接	2	124	'S0036'	'S1406'	'L308'	'L157'	'L045'	1	-51	3
邻接	2	124	'S0036'	'S2082'	'L308'	'L156'	'L045'	1	-12	3

表 2.6　六线 S0087→S3676 部分出行方案列表

求解方法	转乘	总时间	转站点 1	转站点 2	车辆 1	车辆 2	车辆 3	转站点始发数	总负载	总费用
Lingo	0	**30**	-	-	T002	-	-	-	-	3
邻接	0	33	-	-	L231	-	-	-	-	**1**
邻接	0	42	-	-	L381	-	-	-	-	1

用户选线指南：

上表已按多目标分层序列法的目标排序（分别是表中转乘次数、总时间、转车站点是否始发、转车站点总负载量、总费用五个字段），一般用户只需要从上到下选取即可，

但如果用户希望在转站时乘坐始发车（有座位）那么可以挑选是否始发字段为 1、2 的方案，若希望转站时人较少的地方则可以考虑选则站点负载较小的方案;另外，如果线路比较长而且换乘站点负载非常大时用户可以考虑乘坐地铁线路。综述，本模型求解的方案集使用于所有用户，具有很强的实用价值。

6.4 模型评价与应用

6.4.1 邻接算法评价

1) 在本模型下能够求解出转乘次数不超过两次时的所有可行方案,并可根据公众的不同需求，给出最佳需要方案；从此角度考虑，模型实用性较强；

2) 模型求解基于直达数据库Q^D，采用空间换取时间思想，适合查询系统设计标准能够较强的适应工程应用；

3) 在转乘次数超过两次的情况下，运用本模型求解计算过程复杂，计算量过大；故本模型存在一定的局限性。

6.4.2 0-1 规划 *Lingo* 求解方案评价：

1) 在限制最小转乘数时可以求得与邻接算法同样的方案，表明模型的通用性较强，但无法像邻接算法一样求解多种方案是用户所不能接受的；

2) 在不限制最小转乘数时可以求得全局最优解，这是其他所有算法无法达到的，例如在第 2、3、4、5 条线路上其转车次数为 3，但是总耗时相对转车 2 次的要节省许多；

3) 从理论角度分析，最优化模型规划角度可解具有很强的实际意义，例如从全国范围考虑求解，那么转车 3~4 次也是可以接受的，只要耗时足够短，而且查询系统一般事先在大型计算机中计算完毕最短路储存进查询数据库；

4) 从计算时间来分析，尽管需要 20 分钟，但大部分时间为数据导入，只有 1%的时间是真正计算耗时，如果将所需数据存放入内存不变，其求解速度将超越邻接算法；

5) 但 *Lingo* 不能求解出多种方案，实用性不如邻接算法。

6.5 模型应用

——本文模型在数据库查询系统中的实际评价与应用

基于模型建立的思想，可将模型直接应用在查询系统的建立中，具体应用如下：

直达快表 Q 建立的实际应用：在实际应用中块表应用的主要目的为缩减系统搜寻样本空间，以减少搜寻时间及系统搜寻负担。具体搜寻方案为：将模型求解出的直达队列表储存在查询系统的第一层数据库中。查询时，乘客输入起讫点后，系统首先在直达列表中搜寻，若有结果，则优先输出直达路线以及相关信息；若无，则系统自动将数据导入下一层进行处理。

邻接算法建立的实际应用：当第一层搜寻无结果时系统自动将数据转入本层邻接算法模型中。本层在处理时将数据带入本模型，确定出任意两站点之间的最少转乘次数，同时记录对应的多条转乘方案，当转乘次数小于等于 2 时，数据在本层进一步处理。即将得到的不同方案导入分层序列排序模型中，针对不同需求对各方案进行优劣排序，最终将最优方案输出到用户终端。但当转乘次数大于 2 时，由于算法时间限制，为提高查询效率，系统自动将数据导入下一层数据库进行处。

0-1 规划模型建立的实际应用：当转乘次数大于 2 次时，数据进入本层处理。首先，系统将提示用户输入可接受的最大转乘次数，系统将其值带入本模型中的c参数中，利用本模型规划求解出在该转乘次数约束下的最佳路线；若在该约束下无可行解，则系统提示无公交可乘，必须步行适当距离才可能乘坐到车。

7 已知站点间步行时间的线路选择模型（问题三）

本问假设在知道所有站点之间的步行时间的基础上，建立任意两站点之间线路选择问题的数学模型,及其算法实现。

7.1 问题分析

前两问为不考虑站点间步行时间问题，事实上在实际中会出现这样一种情况：人们步行一小段距离再转车，选择这种方式通常可以减少出行总时间。因此，研究此种情况下的出行方案对节省出行时间具有重要的实际意义。根据前面的分析公众出行时除了出行时间最短外，需要考虑的因素仍然包括转乘次数尽量少，行程费用最少，转乘点始发站最多，站点负载压力最小行。

在只靠虑同一站点转乘时，针对公交和地铁网络节点图，可以建立最短路规划模型。假设在只考虑步行的情况下，同样，根据任意两点间的步行时间可以个建立关于两点之间步行时间最短的最短路规划模型。

根据上面的情况，对于同一有向赋权图，当对于任意两个点之间对应两个时间权值时，因此，对于步行和乘车的情况，我们建立最短路问题的推广模型。

7.2 模型Ⅲ分析

基于前面问题的分析，本模型主要以出行总时间最省为主要目标，同时考虑转乘次数尽量少，行程费用最少，转乘点始发站最多，站点负载压力最小行。根据目标从前到后，的重要程度，根据分层序列法，建立最短路问题的0-1整数规划模型。建立本模型首先要创建邻接点矩阵 E'，需要考虑的两个赋权值为乘车时间和步行时间，即赋权图中任意两点之间的权值有两个，即乘车时间 t_{1ij} 和步行时间 t_{2ij}，且都为已知量。令乘车时间对应的决策变量为 x_{ij} (0-1变量)，步行时间对应的决策变量为 y_{ij} (0-1变量)。

目标分析

目标一：行程总时间最短

公众出行是否会选择第 i 到第 j 个节点之间的路，有决策变量 x_{ij} 和 y_{ij} 共同决定。根据 6.2.2 分析，可以得出其出行总时间最少的目标函数为：

$$Min \quad T1+T2+\sum_{(i,j)\in E'} t_{2ij}y_{ij}+\sum_{(i,j)\in E'} t_{1ij}x_{ij} \tag{3.1}$$

目标二：行程总费用最少

当满足前面所有目标后，再求解所有可行方案中费用最小的路线，在这本层目标分析时，将图中的权值赋为途中所需的费用。

基于模型 6.2 中关于任两站点费用的计算方法，这里可直接得到任一弧的所代表的行程的费用，所以总费用的计算为在将弧的权值赋为费用后，线路上代表该线路的各弧长之和。以 P_{ij} 表示站点 i 到 j 的行程费用，则行程总费用最小可表示为：

$$Min \quad \sum_{(i,j)\in E'} P_{ij}x_{ij}-\sum_{\substack{j=D12,D18\\ q'_{ij},q'_{jk}=3\\ (i,j,k)\in E'}} 3Y_{ijk} \tag{3.2}$$

目标三：转乘点始发站最多

当满足第一层目标后，再考虑所选路线中转乘站点是否为所需转乘车始发站，引入

0-1 变量 f_{ij} 表示第 i 个站点是否为 $i \to j$ 的始发站，即：

$$f_{ij}=\begin{cases}1 & \text{第}i\text{个站点是弧}(i,j)\text{线路的始发站}\\ 0 & ELSE\end{cases}$$

则从第 i 到 j 个站点的路线中转乘点为所转乘车的始发点最多的路线可表示为：

$$Max \quad \sum_{(i,j)\in E'} f_{ij}x_{ij} \tag{3.3}$$

目标四：站点负载压力最小

在满足以上三目标的前提下，从更为人性化的角度及城市交通规划角度考虑，应使乘客尽量到负载压力小的站点转乘。在本层目标分析时，将图中的权值赋为各弧起点处的站点负载压力。基于模型Ⅰ中关于任一站点负载压力的定义及计算方法，这里可得到任一弧起点处负载压力，以 r_i 表示起点处负载压力，则线路负载压力最小可表示为：

$$Min \quad \sum_{(i,j)\in E'} r_i x_{ij} \tag{3.4}$$

主要约束分析

1）最短路约束

由于行走路线中任意两点间只会选择一种出行方式，故：

$$x_{ij}+y_{ij}\le 1 \tag{3.5}$$

同时，决策变变量还要满足最短路问题中的主要限制条件，如下所示：

$$\sum_{\substack{j=1\\(i,j)\in E'}}^{n}(x_{ij}+y_{ij})-\sum_{\substack{j=1\\(i,j)\in E'}}^{n}(x_{ji}+y_{ji})=\begin{cases}1 & (i=s)\\ -1 & (i=e)\\ 0 & (i\ne s,e)\end{cases} \tag{3.6}$$

2）换乘次数约束

公众在考虑出行时间尽量短的同时，也会考虑到换乘次数给出行带来的不便。以 c 表示乘客所能接受的最大换乘次数，根据乘车次数确定换乘次数约束可表示为：

$$\sum_{(i,j)\in E'} x_{ij}-1\le c \tag{3.7}$$

3）地铁间换乘约束

站点 $i \to j \to k$ 间是否地铁换乘地铁，使用 Y_{ijk} 表示，那么 Y_{ijk} 与走的路径 y_{ij} 需要满足：

$$y_{ij}+y_{jk}\ge 2Y_{ijk} \qquad (i,j,k)\in E';Z_{ij},Z_{jk}=2 \tag{3.8}$$

地铁转地铁情况只可能在 D12 与 D18 转乘，故换乘总次数不能够大于 2：

$$\sum_{(i,j,k)\in E'} Y_{ijk}\le 2 \qquad Z_{ij},Z_{jk}=2 \tag{3.9}$$

7.3 模型 III 建立

根据问题分析中的目标分析和主要约束分析可建立多目标最短路模型，0-1规划表达式（s 为起点，e 为终点）：

$$Min \quad T1+T2+\sum_{(i,j)\in E'} t_{2ij}y_{ij}+\sum_{(i,j)\in E'} t_{1ij}x_{ij}$$

$$Min \quad \sum_{(i,j)\in E'} P_{ij}x_{ij}-\sum_{\substack{j=D12,D18\\ q'_{ij},q'_{jk}=3\\ (i,j,k)\in E'}} 3Y_{ijk}$$

$$Max \quad \sum_{(i,j)\in E'} f_{ij}\cdot x_{ij}$$

$$Min \quad \sum_{(i,j)\in E'} r_i\cdot x_{ij}$$

$$s.t.\begin{cases} \sum_{(i,j)\in E'} x_{ij}-1\le c & (3.6)\\ \sum_{(i,j)\in E'}^{n}(x_{ij}+y_{ij})-\sum_{(i,j)\in E'}^{n}(x_{ji}+y_{ji})=\begin{cases}1 & (i=s)\\ -1 & (i=e)\\ 0 & (i\neq s,e)\end{cases} & (3.7)\\ \left.\begin{aligned}x_{ij}&\ge Y_{ijk}\\ x_{jk}&\ge Y_{ijk}\end{aligned}\right\}\begin{aligned}&(i,j,k)\in E'\\ &Z_{ij},Z_{jk}=2\end{aligned} & (3.8)\\ \sum_{(i,j,k)\in E'} Y_{ijk}\le 2 \quad Z_{ij},Z_{jk}=2 & (3.9)\\ x_{ij}+y_{ij}\le 1 \quad (i,j)\in E' & (3.5)\\ x_{ij}\in\{0,1\} \quad (i,j)\in E' & \\ Y_{ijk}\in\{0,1\} \quad \begin{cases}(i,j,k)\in E'\\ Z_{ij},Z_{jk}=2\end{cases} & \end{cases}$$

符号说明：

x_{ij} —— 弧(i,j)是否在该路径上；　y_{ij} —— 弧(i,j)是否在该路径上；

t_{1ij} —— 站点$i\to j$的乘车时间；　t_{2ij} —— 站点$i\to j$的步行时间；

f_{ij} —— 第i个站是否为$i\to j$的始发站；　r_i —— 站点i的负载压力；

P_{ij} —— 站点$i\to j$的乘车总费用；　$T1,T2$ —— 总等待时间，总步行时间；

c —— 乘客可接受的最多换乘次数；　Y_{ijk} —— 站点$i\to j\to k$是否地铁换地铁；

Z_{ij}=3 表示 $i\to j$ 最短直达为公汽（也表示乘始发坐公汽等待 3 分钟），等于 2 为地铁（也表示始发乘坐地铁等待 2 分钟）

7.4 模型求解方法

在公交和地铁交通网络系统对应的最短路权值确定的情况下，本模型线性可以考虑运用 *Lingo* 软件编程求解，针对不同目标分别求解可能比较容易。另外，针对本模型我们给出一种近似求解的算法。

在所有站点之间的步行时间确定的情况下，公众出行时可以考虑步行小段距离再换乘车次比较符合实际。基于这种思想，可以考虑将位置比较接近的站点抽象为一个点。根据人的心理分析，一般人对步行时间有一个心理承受值，令该值为 ω，此时可以根据问题二站点的抽象方法，将这种点抽象为一个点处理。为方便描述，将公交和地铁整个系统描述为公交，算法思想如下：

算法思想：

Step 1：输入起始站点A和目的站点B；根据输入的站点进行优化，映射入紧邻站点集 $A=D(a,\omega)$， $B=D(b,\omega)$ (ω 表示乘客在乘车时对步行时间的最大心理承受值)。

Step 2：查询直达队列表是否有 A 到 B 的乘车方案，若有则直接输出结果；若无，继续。

Step 3：在公交站点数据库中查出过站点 A 及紧邻站点 $A=D(a,\omega)$ 的站点 $L(i)(i=1,2,3,\cdots,n)$，及经过站点 B 及紧邻站点 $D(B,\omega)$ 的公交线路 $S(j)(j=1,2,3,\cdots,n)$；

Step 4：判断是否有 $L(i)=S(j)$，若有一条线路满足要求，则该公交线路即为最优线路，输出结果；若没有，继续。

Step 5：从公交线路数据库中查出经过站点 A 的公交线路 $L(i)$ 的站点 $E(i,g)(i=1,2,3,\cdots,m;g=1,2,3,\cdots,n)$，以及经过站点B的公交线路 $S(j)$ 的站点 $F(j,h)(j=1,2,3,\cdots,q;h=1,2,3,\cdots,p)$。

Step 6：判断是否有 $E(i,g)=F(g,h)$。若有一个站点满足要求，该站点即为一次换乘的站点。从A站点出发，在该站点换乘即可以到达 站点。可能有一对或多对公交线路满足要求，从中选择一对距离最短的公交线路即为最优线路，输出结果。若有几个站点满足要求，则先分别求出每一个站点的距离最短的换乘方案，然后选择所有方案中距离最短的换乘方案为最优线路，输出结果；若没有，继续。

Step 7：从公交站点数据库中查得经过 $E(i,g)$ 的公交线路 $T(k)(k=1,2,3,\cdots,m)$，从公交线路数据库中查得线路 $T(k)$ 的站点 $G(k,w)(k=1,2,3,\cdots,m;w=1,2,3\cdots,n)$。

Step 8：判断是否有 $G(k,w)=F(j,h)$。若有某个站点E满足要求。则站点 E 为第二个换乘站点。从起始站点A经过一次换乘(假设换乘点为站点 D)，可以到达站点 E，从站点 E 可以换乘公交车直达目的站点 。按照步骤4)～6)的方法求出从起始站点A到站点E的一次换乘的最优线路，在按照2)～3)的方法求出从站点 E 到目的站点的最优线路。两个换乘站点和两段最优线路即组成了从起始站点A到目的站点 的最优线路。若有多个站点满足，$G(k,w)=F(j,h)$，则分别求出各站点的最优换乘方案，比较各方案的线路距离，选择一种距离最短的换乘方案作为最后的结果，输出结果。

8 模型改进方向

由于题中信息有限，所以本文模型在实际应用时仍存在改进空间，若信息充足，则为使模型更具实用性，可在如下四方面进行改进：

一、考虑通过站点周围建筑物进行查询

查询用户一般为对城市道路交通情况不熟悉的外地乘客，所以在查询时可能并不知道具体的站台名称，而只知道站台所在的大概位置，如对于观看奥运会比赛的用户可能仅知道奥运场馆的名称而非其邻近站点的名称，因而在抽象公交网络时要考虑站台周围的建筑、大的单位及景点。根据实际情况可以把这些相关因素与其周围最近的站台名联系在一起，从而抽象成一个节点。

如可以以站台为圆点，可以接受的步行距离为半径所画的圆形区，抽象为一个站点

进行记录，用户在查询时可仅输入所要到达的地点标志，系统会自动提示最近站点及最佳乘车方案。

二、考虑提示观光路线

对于许多乘客而言，更希望乘车路线沿途可观赏到北京的特色景观及建筑，所以从宣扬首都文化角度考虑，应在系统内部将沿途有特色景观（如奥运场馆、名胜古迹等）的路径段进行特别标注或分区存放数据，在用户查询时，系统应在给出常规最佳路线的同时，提示一条观光路线供乘客自由选择。

三、考虑步行对公众感受的影响程度

由于题中未给出步行时间及乘车时间对公众感受的影响程度，所以本模型在建立时采取保守做法，即认为其影响程度相等。事实上存在这样一种情况，即乘客宁愿多坐10分钟的车，也不愿走3分钟的路，这样在计算总时间时应将走路时间设的系数设置的更大才更符合乘客出行心理。

四、考虑乘客特殊乘车嗜好

当同时考虑公汽及地铁时，本模型并为考虑公众出行的特殊嗜好，比如有的乘客不喜欢坐地铁或公汽。则查询系统内部应分三区，分别存放仅公汽、仅地铁、两者混合三区，用户查询时可根据系统提示自主进行选择。

参考文献

[1] 谢金星、薛毅，优化建模与 LINDO/LINGO 软件，北京：清华大学出版社，2005 年 7 月第一版

[2] Duane Hanselman、Bruce Littlefield 著，朱仁峰译，Matlab 7，北京：清华大学出版社，2006 年 5 月第一版

[3] 王莉，李文权，公共交通系统最佳路径算法，东南大学学报，第 34 卷：第 3 期，2004 年 3 月

[4] 徐多勇，李志蜀，梅林，基于 GSM 短信息的公交查询系统的最优化转乘方案研究与设计，计算机应用， 27 卷：2007 年 6 月

附录

1.0 排序函数

T_SORT.m

```
function [ A ] = T_SORT( A ,n , p )
%  T_SORT( A ,n , p )
%  A 根据第n行排序
%  p=1升序 , 2 降
%  powered_BY_*
SIZE=size(A);
if p==1
    [xx,idx]=sort(A(n,:));
    for i=1:SIZE(1)
        A(i,:)=A(i,idx);
    end
elseif p==2
    [xx,idx]=sort(A(n,:),'descend');
    for i=1:SIZE(1)
        A(i,:)=A(i,idx);
    end
end
```

1.1 数据读取程序段

ReadData.m%第一问

```
%读取数据
clear L a
fid = fopen('B2007data/1.1 公汽线路信
息.txt','r');
i=1;
while 1
   tline = fgetl(fid);
   if ~ischar(tline),   break,   end
   if strcmp(tline,'')
       continue
   end
   if strcmp(tline(1),'L')
       str=tline;
      continue
   elseif strcmp(tline,'END')
       break
   end

   if strcmp(tline,'单一票制1元。')
       P=1;
       continue
   elseif strcmp(tline,'分段计价。')
       P=2;
       continue
   end

   if strcmp(tline(1:2),'上行')
       L{i,1}=str;
       L{i,2}=P;
       L{i,3}='上行';
       L{i,4}=tline(4:end);
       i=i+1;
       continue
   elseif strcmp(tline(1:2),'下行')
       L{i,1}=str;
       L{i,2}=P;
       L{i,3}='下行';
       L{i,4}=tline(4:end);
       i=i+1;
       continue
   elseif strcmp(tline(1:2),'环行')
       L{i,1}=str;
       L{i,2}=P;
       L{i,3}='环行1';

L{i,4}=strcat(tline(4:end),tline(10:end));
       i=i+1;
       %计算来回
       L{i,1}=str;
       L{i,2}=P;
       L{i,3}='环行2';

L{i,4}=strcat(tline(4:end),tline(10:end));
       i=i+1;
       continue
   elseif strcmp(tline(1),'S')
       L{i,1}=str;
       L{i,2}=P;
       L{i,3}='来回1';
       L{i,4}=tline;
       i=i+1;
       %计算来回
       L{i,1}=str;
       L{i,2}=P;
       L{i,3}='来回2';
       L{i,4}=tline;
       i=i+1;
       continue
```

ChangeData.m 第二问

```
fid = fopen('B2007data/2.1 地铁T1线换乘公汽信息.txt','r');
i=1;
while 1
    tline = fgetl(fid);
    if ~ischar(tline),   break,   end
    if strcmp(tline,'')
        continue
    end
    if strcmp(tline(1),'D')
        D{i,1}=tline(5:end);
        i=i+1;
    end
end
fclose(fid);
fid = fopen('B2007data/2.2 地铁T2线换乘公汽信息.txt','r');
while 1
    tline = fgetl(fid);
    if ~ischar(tline),   break,   end
    if strcmp(tline,'')
        continue
    end
    if strcmp(tline(1),'D')
        D{i,1}=tline(5:end);
        i=i+1;
    end
end
fclose(fid);
tx=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,
23,24,25,...
    26,12,27,28,29,30,31,32,18,33,34,35,36,37,38,39]+3957;
DL{1,1}='';
DL{2,1}='';
for j=1:23
    DL{1,1}=strcat(DL{1,1},'S',num2str(tx(j)));
end
for j=24:41
    DL{2,1}=strcat(DL{2,1},'S',num2str(tx(j)));
end
DL{2,1}=strcat(DL{2,1},'S',num2str(tx(24)));
for i=1:1040
    for j=1:41
        tline=D{j,1};
        t=findstr(tline,'S');
        for k=1:length(t)
            L{i,4}=regexprep(L{i,4},
strcat('S',tline(t(k)+1:t(k)+4)),...
                    strcat('S',num2str(tx(j))));
        end
```

```
    end
end    fclose(fid);
for i=1:size(L,1)
    tline=L{i,4};
    t=findstr(tline,'S');
    temp=zeros(1,length(t));
    if strcmp(L{i,3},'来回2') ||
strcmp(L{i,3},'环行2')
        for j=length(t):-1:1

temp(length(t)-j+1)=str2double(tline(t(j)+1:
t(j)+4));
        end
    else
        for j=1:length(t)

temp(j)=str2double(tline(t(j)+1:t(j)+4));
        end
    end
    L2{i,1}=temp;
end
for i=1:3957
    if floor(i/10)==0
        Cit{i}=strcat('S000',num2str(i));
    elseif floor(i/100)==0
        Cit{i}=strcat('S00',num2str(i));
    elseif floor(i/1000)==0
        Cit{i}=strcat('S0',num2str(i));
    else
        Cit{i}=strcat('S',num2str(i));
    end
end
Cit{3958}='D01: S0567, S0042, S0025';
Cit{3959}='D02: S1487';
Cit{3960}='D03: S0303, S0302';
Cit{3961}='D04: S0566';
Cit{3962}='D05: S0436, S0438, S0437, S0435';
Cit{3963}='D06: S0392, S0394, S0393, S0391';
Cit{3964}='D07: S0386, S0388, S0387, S0385';
Cit{3965}='D08: S3068, S0617, S0619, S0618,
S0616';
Cit{3966}='D09: S1279';
Cit{3967}='D10: S2057, S0721, S0722, S0720';
Cit{3968}='D11: S0070, S2361, S3721';
Cit{3969}='D12: S0609, S0608';
Cit{3970}='D13: S2633, S0399, S0401, S0400';
Cit{3971}='D14: S3321, S2535, S2464';
Cit{3972}='D15: S3329, S2534';
Cit{3973}='D16: S3506, S0167, S0168';
Cit{3974}='D17: S0237, S0239, S0238, S0236,
S0540';
Cit{3975}='D18: S0668';
Cit{3976}='D19: S0180, S0181';
Cit{3977}='D20: S2079, S2933, S1919, S1921,
S1920';
Cit{3978}='D21: S0465, S0467, S0466, S0464';
Cit{3979}='D22: S3457';
Cit{3980}='D23: S2512';
Cit{3981}='D24: S0537, S3580';
Cit{3982}='D25: S0526, S0528, S0527, S0525';
Cit{3983}='D26: S3045, S0605, S0607';
Cit{3984}='D27: S0087, S0088, S0086';
Cit{3985}='D28: S0855, S0856, S0854, S0857';
Cit{3986}='D29: S0631, S0632, S0630';
Cit{3987}='D30: S3874, S1426, S1427';
Cit{3988}='D31: S0211, S0539, S0541, S0540';
Cit{3989}='D32: S0978, S0497, S0498';
Cit{3990}='D33: S1894, S1896, S1895';
Cit{3991}='D34: S1104, S0576, S0578, S0577';
Cit{3992}='D35: S3010, S0583, S0582';
Cit{3993}='D36: S3676, S0427, S0061, S0060';
Cit{3994}='D37: S1961, S2817, S0455, S0456';
Cit{3995}='D38: S3262, S0622';
Cit{3996}='D39: S1956, S0289, S0291';
```

```
    end
end
Lt1=L;
Lt1{1041,1}='T001';
Lt1{1041,2}=3;
Lt1{1041,3}='来回1';
Lt1{1041,4}=DL{1,1};

Lt1{1042,1}='T001';
Lt1{1042,2}=3;
Lt1{1042,3}='来回2';
Lt1{1042,4}=DL{1,1};

Lt1{1043,1}='T002';
Lt1{1043,2}=3;
Lt1{1043,3}='环行1';
Lt1{1043,4}=strcat(DL{2,1},DL{2,1}(6:end));

Lt1{1044,1}='T002';
Lt1{1044,2}=3;
Lt1{1044,3}='环行2';
Lt1{1044,4}=strcat(DL{2,1},DL{2,1}(6:end));
for i=1:1044
    tline=Lt1{i,4};
    t=findstr(tline,'S');
    temp=zeros(1,length(t));
    if strcmp(Lt1{i,3},'来回2') || strcmp(Lt1{i,3},'环行2')
        for j=length(t):-1:1

temp(length(t)-j+1)=str2double(tline(t(j)+1:t(j)+4));
        end
    else
        for j=1:length(t)
            temp(j)=str2double(tline(t(j)+1:t(j)+4));
        end
    end
    Lt2{i,1}=temp;
end
```

1.2 直达队列表 Q 与 Q^D 的建立

MakeS1.m 第一问

```
clear S
S(1:3957,1:3957)={zeros(1,1,'uint16')};
for i=1:1040
    t=L2{i,1};
    for j=1:length(t)-1
        for k=j+1:length(t)
            temp=S{t(j),t(k)};
            str=L{i,1};
            [n,m]=size(temp);
            if n==1 && temp(1,1)==0

temp(n,1)=str2double(str(2:end));
                if L{i,2}==2
                    if (k-j)>40
                        temp(n,2)=3;
                    elseif (k-j)>20
                        temp(n,2)=2;
```

MakeS2.m 第二问

```
clear S
S(1:3996,1:3996)={zeros(1,1,'uint16')};
for i=1:1044
    t=Lt2{i,1};
    for j=1:length(t)-1
        for k=j+1:length(t)
            temp=S{t(j),t(k)};
            [n,m]=size(temp);
            if n==1 && temp(1,1)==0
                temp(n,1)=i;
                if Lt1{i,2}==2
                    if (k-j)>40
                        temp(n,2)=3;
                    elseif (k-j)>20
                        temp(n,2)=2;
                    else
                        temp(n,2)=1;
```

```
                    else
                        temp(n,2)=1;
                    end
                else
                    temp(n,2)=1;
                end
                temp(n,3)=k-j;
            else

temp(n+1,1)=str2double(str(2:end));
                if L{i,2}==2
                    if (k-j)>40
                        temp(n+1,2)=3;
                    elseif (k-j)>20
                        temp(n+1,2)=2;
                    else
                        temp(n+1,2)=1;
                    end
                else
                    temp(n+1,2)=1;
                end
                temp(n+1,3)=k-j;
            end
            S{t(j),t(k)}=temp;
        end
    end
end


for i=1:3957
    for j=1:3957
        if length(S{i,j})~=1
            S{i,j}=T_SORT(S{i,j}',3,1)';
        end
    end
end
Time=zeros(3957,3957,'uint8');
for i=1:3957
    for j=1:3957
        if length(S{i,j})~=1
            Time(i,j)=size(S{i,j},1);
        end
    end
end
TT=zeros(3957,3957,'uint8');
for i=1:3957
    for j=1:3957
        temp=S{i,j};
        if temp(1,1)~=0
            TT(i,j)=temp(1,3);
        end
    end
end
```

```
                end
            elseif Lt1{i,2}==1
                temp(n,2)=1;
            else
                temp(n,2)=3;
            end
            temp(n,3)=k-j;
        else
            temp(n+1,1)=i;
            if Lt1{i,2}==2
                if (k-j)>40
                    temp(n+1,2)=3;
                elseif (k-j)>20
                    temp(n+1,2)=2;
                else
                    temp(n+1,2)=1;
                end
            elseif Lt1{i,2}==1
                temp(n+1,2)=1;
            else
                temp(n+1,2)=3;
            end
            temp(n+1,3)=k-j;
        end
        S{t(j),t(k)}=temp;
    end
  end
end
for i=1:3996
    for j=1:3996
        if length(S{i,j})~=1
            temp=double(S{i,j});
            for k=1:size(temp,1)
                if temp(k,1)>1040
                    temp(k,3)=temp(k,3)*2.5;
                else
                    temp(k,3)=temp(k,3)*3;
                end
            end
            temp=T_SORT(temp',3,1)';
            for k=1:size(temp,1)
                if temp(k,1)>1040
                    temp(k,3)=temp(k,3)/2.5;
                else
                    temp(k,3)=temp(k,3)/3;
                end
            end
            S{i,j}=uint16(temp);
         end
    end
end
Time=zeros(3996,3996,'uint16');
for i=1:3996
    for j=1:3996
        if length(S{i,j})~=1
            Time(i,j)=size(S{i,j},1);
        end
    end
end
TT=zeros(3996,3996,'uint8');
for i=1:3996
    for j=1:3996
        temp=S{i,j};
        if temp(1,1)~=0
            for k=1:size(temp,1)
                if temp(k,1)>1040
                    temp(k,3)=temp(k,3)*2.5;
                else
                    temp(k,3)=temp(k,3)*3;
                end
            end
            TT(i,j)=min(temp(:,3));
        end
    end
end
```

1.3 邻接搜索程序段问题一求解

Search1.m

```
N=87;
M=3676;
% 直达
if Time(N,M)~=0
    temp=S{N,M};
    for i=1:size(temp,1)
        disp(strcat('直达车为:L',num2str(temp(i,1))))
    end
    return
end
clear U U2
% 一次转车
t=1:3957;
T2=Time(N,:).*Time(:,M)';
if sum(T2)~=0
    x=1;
    t=t(T2~=0);
    for i=1:length(t)
        t1=S{N,t(i)};
        t2=S{t(i),M};
               t1=double(t1);
               t2=double(t2);
        for j=1:size(t1,1)
            for k=1:size(t2,1)
                U(x,1)=1;%转站次数
                U(x,2)=(t1(j,3)+t2(k,3))*3+5+3;%总时间
                U(x,3)=t(i);%转站点
                U(x,4:5)=[t1(j,1) t2(k,1)];%车辆
                %是否始发站
                temp=0;
                for k1=1:1040
                    if str2double(L{k1,1}(2:end))==U(x,5)
                        if L2{k1,1}(1,1)==t(i)
                            temp=1;
                            break
                        end
                    end
                end
                U(x,6)=temp;
                U(x,7)=-sum(Time(:,t(i)))+...
                    sum(Time(t(i),:));%人流负载
                U(x,8)=t1(j,2)+t2(k,2);%费用
                x=x+1;
            end
        end
    end
    return
end
%二次转车
t=1:3957;
x=1;
for i=1:3957
    if Time(N,i)~=0
        for j=1:3957
            if Time(j,M)~=0  && Time(i,j)~=0
                t1=S{N,i};
                t2=S{i,j};
                t3=S{j,M};
                t1=double(t1);
                t2=double(t2);
                t3=double(t3);
                for k1=1:size(t1,1)
                    for k2=1:size(t2,1)
                        for k3=1:size(t3,1)
                            U2(x,1)=2;%转站次数
                            %总时间
                            U2(x,2)=(t1(k1,3)+t2(k2,3)+t3(k3,3))*3+10+3;
                            U2(x,3)=t(i);%转站点
                            U2(x,4)=t(j);%转站点
                            U2(x,5:6)=[t1(k1,1) t2(k2,1)];%车辆1,2
                            %是否始发
```

```
                        temp=0;
                        for kk=1:1040
                            if str2double(L{kk,1}(2:end))==U2(x,6)
                                if L2{kk,1}(1,1)==t(i)
                                    temp=1;
                                    break
                                end
                            end
                        end
                        U2(x,7)=t3(k3,1);%车辆3
                        %始发站数
                        for kk=1:1040
                            if str2double(L{kk,1}(2:end))==U2(x,7)
                                if L2{kk,1}(1,1)==t(j)
                                    temp=temp+1;
                                    break
                                end
                            end
                        end
                        U2(x,8)=temp;
                        %人流量
                        U2(x,9)=sum(Time(t(j),:))-...
                            sum(Time(:,t(j)))-...
                            sum(Time(:,t(i)))+...
                            sum(Time(t(i),:));
                        %费用
                        U2(x,10)=t1(k1,2)+t2(k2,2)+t3(k3,2);
                        x=x+1;
                    end
                end
            end
        end
    end
end
```

1.4 邻接搜索程序段问题二求解

Search2.m

```
N='S0087';
M='S3676';
% 映射
if strcmp(N(1),'S')
    for i=1:41
        t=findstr(D{i,1},N);
        if ~isempty(t)
            N=tx(i);
            break
        end
    end
    if ischar(N)
        N=str2double(N(2:end));
    end
end
if strcmp(M(1),'S')
    for i=1:41
        t=findstr(D{i,1},M);
        if ~isempty(t)
            M=tx(i);
            break
        end
    end
    if ischar(M)
        M=str2double(M(2:end));
    end
end
% 直达
if Time(N,M)~=0
    temp=S{N,M};
    for i=1:size(temp,1)
        if temp(i,1)>1040
                disp(strcat('直达地铁为:',Lt1{temp(i,1),1}))
        else
                disp(strcat('直达汽车为:',Lt1{temp(i,1),1}))
        end
    end
```

```
end
clear U U2
% 一次转车
t=1:3996;
T2=Time(N,:).*Time(:,M)';
if sum(T2)~=0
    x=1;
    t=t(T2~=0);
    for i=1:length(t)
        t1=S{N,t(i)};
        t2=S{t(i),M};
              t1=double(t1);
              t2=double(t2);
        for j=1:size(t1,1)
            for k=1:size(t2,1)
                U{x,1}=1;%转站次数
                U{x,8}=t1(j,2)+t2(k,2);%费用
                if t1(j,1)>1040
                    if t2(k,1)>1040
                        U{x,2}=(t1(j,3)+t2(k,3))*2.5+4+2;%d d
                        U{x,8}=3;%2次地铁费用3
                    else
                        U{x,2}=t1(j,3)*2.5+t2(k,3)*3+7+2;%d g
                    end
                else
                    if t2(k,1)>1040
                        U{x,2}=t1(j,3)*3+t2(k,3)*2.5+6+3;%g d
                    else
                        U{x,2}=t1(j,3)*3+t2(k,3)*3+5+3;%g g
                    end
                end
                U{x,3}=Cit{t(i)};%转站点
                U{x,4}=Lt1{t1(j,1),1};%车辆
                U{x,5}=Lt1{t2(k,1),1};
                %是否始发站
                temp=0;
                if t2(k,1)<=1040
                    for k1=1:1040
                        if strcmp(Lt1{k1,1},U{x,5})
                            if Lt2{k1,1}(1,1)==t(i)
                                temp=1;
                                break
                            end
                        end
                    end
                end
                U{x,6}=temp;
                U{x,7}=-sum(Time(:,t(i)))+...
                    sum(Time(t(i),:));%人流量
                x=x+1;
            end
        end
    end
    return
end
%二次转车
t=1:3996;
x=1;
for i=1:3996
    if Time(N,i)~=0
        for j=1:3996
            if Time(j,M)~=0  && Time(i,j)~=0
                t1=S{N,i};
                t2=S{i,j};
                t3=S{j,M};
                t1=double(t1);
                t2=double(t2);
                t3=double(t3);
                for k1=1:size(t1,1)
                    for k2=1:size(t2,1)
                        for k3=1:size(t3,1)
                            U2{x,1}=2;%转站次数
                            %总时间
                if t1(k1,1)>1040
                    if t2(k2,1)>1040
```

```
                if t3(k3,1)>1040
                    %d d d
                    U2{x,2}=(t1(k1,3)+t2(k2,3)+t3(k3,3))*2.5+8+2;
                    U2{x,10}=3;
                else
                    %d d g
                    U2{x,2}=2.5*t1(k1,3)+2.5*t2(k2,3)+3*t3(k3,3)+11+2;
                    U2{x,10}=3+t3(k3,2);
                end
            else
                if t3(k3,1)>1040
                    %d g d
                    U2{x,2}=2.5*t1(k1,3)+3*t2(k2,3)+2.5*t3(k3,3)+13+2;
                    U2{x,10}=6+t2(k2,2);
                else
                    %d g g
                    U2{x,2}=2.5*t1(k1,3)+3*t2(k2,3)+3*t3(k3,3)+12+2;
                    U2{x,10}=3+t2(k2,2)+t3(k3,2);
                end
            end
        else
            if t2(k2,1)>1040
                if t3(k3,1)>1040
                    %g d d
                    U2{x,2}=3*t1(k1,3)+2.5*t2(k2,3)+2.5*t3(k3,3)+11+3;
                    U2{x,10}=t1(k1,2)+3;
                else
                    %g d g
                    U2{x,2}=3*t1(k1,3)+2.5*t2(k2,3)+3*t3(k3,3)+13+3;
                    U2{x,10}=3+t1(k1,2)+t3(k3,2);
                end
            else
                if t3(k3,1)>1040
                    %g g d
                    U2{x,2}=3*t1(k1,3)+3*t2(k2,3)+2.5*t3(k3,3)+11+3;
                    U2{x,10}=3+t1(k1,2)+t2(k2,2);
                else
                    %g g g
                    U2{x,2}=3*t1(k1,3)+3*t2(k2,3)+3*t3(k3,3)+10+3;
                    U2{x,10}=t1(k1,2)+t2(k2,2)+t3(k3,2);
                end
            end
        end
                U2{x,3}=Cit{t(i)};%转站点
                U2{x,4}=Cit{t(j)};%转站点
                U2{x,5}=Lt1{t1(k1,1),1};%车辆1
                U2{x,6}=Lt1{t2(k2,1),1};
                %是否始发
                temp=0;
                if t2(k2,1)<=1040
                    for kk=1:1040
                    if strcmp(Lt1{kk,1},U2{x,6})
                        if Lt2{kk,1}(1,1)==t(i)
                            temp=1;
                            break
                        end
                    end
                    end
                end
                U2{x,7}=Lt1{t3(k3,1),1};%车辆3
                %始发站数
                if t3(k3,1)<=1040
                    for kk=1:1040
                    if strcmp(Lt1{kk,1},U2{x,7})
                        if Lt2{kk,1}(1,1)==t(j)
                            temp=temp+1;
                            break
                        end
                    end
                    end
                end
                U2{x,8}=temp;
                %人流量
                U2{x,9}=sum(Time(t(j),:))-...
                    sum(Time(:,t(j)))-...
                    sum(Time(:,t(i)))+...
```

```
                        sum(Time(t(i),:));
                    x=x+1;
                end
            end
        end
    end
end
```

1.5 分层序列法排序程序段

Fang_Sort1.m 第一问转乘 1 次

```
clear Ux
[n,m]=size(U);
t=zeros(1,n);
t1=[10^(5*5) 10^(5*4) 10^(5*3) 10^(5*2)
10^(5*1)];
for i=1:n
    t(i)=t(i)+U(i,1)*t1(1);
    t(i)=t(i)+U(i,2)*t1(2);
    t(i)=t(i)+t1(2)-U(i,6)*t1(3);
    t(i)=t(i)+U(i,7)*t1(4);
    t(i)=t(i)+U(i,8)*t1(5);
end
[t2,ix]=sort(t);
t3=1;
for i=1:n
    if t3>=2  && Ux(t3-1,4)==U(ix(i),4)
&& Ux(t3-1,5)==U(ix(i),5)
        continue
    end
    Ux(t3,:)=U(ix(i),:);
    t3=t3+1;
end
```

Fang_Sort2.m 第一问转乘 2 次

```
clear Ux
[n,m]=size(U2);
t=zeros(1,n);
t1=[10^(5*5) 10^(5*4) 10^(5*3) 10^(5*2) 10^(5*1)];
for i=1:n
    t(i)=t(i)+U2(i,1)*t1(1);
    t(i)=t(i)+U2(i,2)*t1(2);
    t(i)=t(i)+t1(2)-U2(i,8)*t1(3);
    t(i)=t(i)+U2(i,9)*t1(4);
    t(i)=t(i)+U2(i,10)*t1(5);
end
[t2,ix]=sort(t);
t3=1;
for i=1:n
    if t3>=2  && Ux(t3-1,5)==U2(ix(i),5) && ...
           Ux(t3-1,6)==U2(ix(i),6) && ...
           Ux(t3-1,7)==U2(ix(i),7)
        continue
    end
    Ux(t3,:)=U2(ix(i),:);
    t3=t3+1;
end
```

Fang_Sort3.m第二问转乘1次

```
clear Ux
[n,m]=size(U);
t=zeros(1,n,'double');
t1=[10^(5*5) 10^(5*4) 10^(5*3) 10^(5*2)
10^(5*1)];
for i=1:n
    t(i)=t(i)+U{i,1}*t1(1);
    t(i)=t(i)+double(U{i,2})*t1(2);
    t(i)=t(i)+t1(2)-U{i,6}*t1(3);
    t(i)=t(i)+U{i,7}*t1(4);
    t(i)=t(i)+double(U{i,8})*t1(5);
end
[t2,ix]=sort(t);
t3=1;
for i=1:n
    if t3>=2  &&...

strcmp(Ux{t3-1,4},U{ix(i),4}) &&...

strcmp(Ux{t3-1,5},U{ix(i),5})
        continue
    end
    for j=1:m
        Ux{t3,j}=U{ix(i),j};
    end
    t3=t3+1;
end
```

Fang_Sort4.m 第二问转乘 2 次

```
clear Ux
[n,m]=size(U2);
t=zeros(1,n);
t1=[10^(5*5) 10^(5*4) 10^(5*3) 10^(5*2) 10^(5*1)];
for i=1:n
    t(i)=t(i)+U2{i,1}*t1(1);
    t(i)=t(i)+double(U2{i,2})*t1(2);
    t(i)=t(i)+t1(2)-U2{i,8}*t1(3);
    t(i)=t(i)+U2{i,9}*t1(4);
    t(i)=t(i)+double(U2{i,10})*t1(5);
end
[t2,ix]=sort(t);
t3=1;
for i=1:n
    if t3>=2  &&...
           strcmp(Ux{t3-1,5},U2{ix(i),5}) &&...
           strcmp(Ux{t3-1,6},U2{ix(i),6}) &&...
           strcmp(Ux{t3-1,7},U2{ix(i),7})
        continue
    end
    for j=1:m
        Ux{t3,j}=U2{ix(i),j};
    end
    t3=t3+1;
end
```

1.6 问题一 LINGO 求解程序（Lingo 版本 10.0）

数据文件通过附录 1.7MATLAB 程序创建（运行顺序：附录 1.1~1.4）

```
MODEL:
sets:
C/1..3957/:;
roads(C,C)/@file('roads.txt')/:W,X;
endsets
data:
```

```
w=@file('Time.txt');
enddata
!目标段;
SUBMODEL SUB_LosT:
[OBJ_LosT]min=3+ @sum(roads:W*X)*3+5*(@sum(roads:X)-1);
ENDSUBMODEL
!约束段;
SUBMODEL SUB_CON:
@for(C(i)|i#ne#N #and# i#ne#M:
 @sum(roads(i,j):x(i,j))=@sum(roads(j,i):x(j,i)));
@for(C(i)|i#eq#N:
 @sum(roads(i,j):x(i,j))-@sum(roads(j,i):x(j,i))=1);
@for(C(i)|i#eq#M:
 @sum(roads(i,j):x(i,j))-@sum(roads(j,i):x(j,i))=-1);
ENDSUBMODEL
SUBMODEL SUB_CON2:
@for(C(i)|i#ne#N #and# i#ne#M:
 @sum(roads(i,j):x(i,j))=@sum(roads(j,i):x(j,i)));
@for(C(i)|i#eq#N:
 @sum(roads(i,j):x(i,j))-@sum(roads(j,i):x(j,i))=1);
@for(C(i)|i#eq#M:
 @sum(roads(i,j):x(i,j))-@sum(roads(j,i):x(j,i))=-1);
@sum(roads:x)-1<2;
ENDSUBMODEL
CALC:
@SET('TERSEO',1);!简洁输出报告;
!@set('stawin',0);
@divert('第一问结果.log');
!一线;
N=3359;M=1828;
@solve(sub_lost,sub_con2);
@write('一线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
!二线;
N=1557;M=481;
@solve(sub_lost,sub_con);
@write('二线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
!三线;
N=971;M=485;
@solve(sub_lost,sub_con);
@write('三线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
!四线;
N=8;M=73;
@solve(sub_lost,sub_con);
@write('四线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
```

```
!五线;
N=148;M=485;
@solve(sub_lost,sub_con);
@write('五线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
!六线;
N=87;M=3676;
@solve(sub_lost,sub_con);
@write('六线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
endcalc
END
```

1.7 LINGO 程序数据创建程序

数据文件创建（运行顺序）（运行顺序：附录 1.1~1.4）

```
%write txt
fid1 =fopen('Time.txt','w');
fid2 =fopen('roads.txt','w');
for i=1:3957
    for j=1:3957
        if TT(i,j)~=0 && i~=j
            fprintf(fid1,'%5.1f\n',TT(i,j));
            fprintf(fid2,'%d,%d\n',i,j);
        end
    end
  %  fprintf(fid2,'\n');
  %  fprintf(fid1,'\n');
end
fclose(fid1);
fclose(fid2);
```

1.8 问题二 LINGO 求解程序（Lingo 版本 10.0）

数据文件通过附录 1.9MATLAB 程序创建（运行顺序：附录 1.1~1.4）

```
MODEL:
sets:
C/1..3996/:;
roads(C,C)/@file('roads2.txt')/:W,Z,X;
H1(C,C,C)/@file('dd.txt')/:dd;
endsets
data:
w=@file('Time2.txt');
Z=@file('Z.txt');
enddata
!目标段;
SUBMODEL OBJ_1:
[OBJ_LosT]min=
    @sum(roads:w*x)!乘车时间;
   +@sum(roads:Z*x)!等待时间T1;
 +2*(@sum(roads:x)-1)!Ta;
  +4*@sum(roads|z#eq#2:x)!Tb;
  -4*@sum(H1:dd)!Tc;
  +4*@sum(roads(i,j)|i#eq#N #and# j#eq#M #and# Z(i,j)#eq#2:x(i,j))!Td;
;
ENDSUBMODEL
!约束段;
SUBMODEL SUB_CON:
@for(C(i)|i#ne#N #and# i#ne#M:
```

```
 @sum(roads(i,j):x(i,j))=@sum(roads(j,i):x(j,i)));
@for(C(i)|i#eq#N:
 @sum(roads(i,j):x(i,j))-@sum(roads(j,i):x(j,i))=1);
@for(C(i)|i#eq#M:
 @sum(roads(i,j):x(i,j))-@sum(roads(j,i):x(j,i))=-1);
@for(H1(i,j,k):x(i,j)+x(j,k)>=2*dd(i,j,k));
@sum(H1:dd)<=2;
@for(H1:@bin(dd));
ENDSUBMODEL
!模型求解流程;
CALC:
@SET('TERSEO',1);!简洁输出报告;
@divert('第二问结果.log');
!一线;
N=3359;M=1828;
@solve(OBJ_1,sub_con);
@write('一线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
!二线;
N=1557;M=481;
@solve(OBJ_1,sub_con);
@write('二线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
!三线;
N=971;M=485;
@solve(OBJ_1,sub_con);
@write('三线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
!四线;
N=8;M=73;
@solve(OBJ_1,sub_con);
@write('四线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
!五线;
N=148;M=485;
@solve(OBJ_1,sub_con);
@write('五线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
!六线;
N=3984;M=3993;
@solve(OBJ_1,sub_con);
@write('六线：',OBJ_LosT,@newline(1));
@for(roads(i,j):
```

```
 @ifc(x(i,j)#eq#1:
   @write(i,',',j,';',@newline(1));
  );
);
endcalc
END
```

1.9 LINGO 程序,数据创建程序 2

数据文件创建（运行顺序：附录 1.1~1.4）

```
%write txt
fid1 =fopen('Time2.txt','w');
fid2 =fopen('roads2.txt','w');
fid3 =fopen('Z.txt','w');
YYY=[0,0,0;0,4,7;0,6,5;];
for i=1:3996
    for j=1:3996
        if TT(i,j)~=0 && i~=j
            fprintf(fid2,'%d,%d\n',i,j);
            temp=zeros(2,Time(i,j));
            for k1=1:Time(i,j)
                if S{i,j}(k1,1)>1040
                    temp(1,k1)=2.5*double(S{i,j}(k1,3));
                    temp(2,k1)=2;%地铁
                else
                    temp(1,k1)=3*S{i,j}(k1,3);
                    temp(2,k1)=3;%公交
                end
            end
            temp=T_SORT(temp,1,1);
            fprintf(fid1,'%5.1f\n',temp(1,1));
            fprintf(fid3,'%d\n',temp(2,1));
        end
    end
end
fclose(fid1);
fclose(fid2);
fclose(fid3);
fid1 =fopen('dd.txt','w');
YYY=[0,0,0;0,4,7;0,6,5;];
for i=3958:3996
    for j=3958:3996
        if j==3969||j==3975
        if TT(i,j)~=0 && i~=j
            temp=zeros(2,Time(i,j));
            for k1=1:Time(i,j)
                if S{i,j}(k1,1)>1040
                    temp(1,k1)=2.5*double(S{i,j}(k1,3));
                    temp(2,k1)=2;%地铁
                else
                    temp(1,k1)=3*S{i,j}(k1,3);
                    temp(2,k1)=3;%公交
                end
            end
            temp=T_SORT(temp,1,1);
            for k=3958:3996
                if (i<=3980&&k>=3981)||(i>=3981&&k<=3980)
                if TT(j,k)~=0 && k~=j && k~=i
                    temp2=zeros(2,Time(j,k));
                    for k1=1:Time(j,k)
                        if S{j,k}(k1,1)>1040
                            temp2(1,k1)=2.5*double(S{j,k}(k1,3));
                            temp2(2,k1)=2;%地铁
                        else
                            temp2(1,k1)=3*S{j,k}(k1,3);
                            temp2(2,k1)=3;%公交
                        end
                    end
                    temp2=T_SORT(temp2,1,1);
                    if YYY(temp(2,1),temp2(2,1))==4
                        fprintf(fid1,'%d,%d,%d\n',i,j,k);
                    end
                end
                end
            end
```

```
        end
        end
    end
end
fclose(fid1);
```