铁路最优路线问题

摘 要

本文研究的是在任意两站点之间没有直接到达的铁路或者直达列车票已售罄的情况下，求出两站点之间的最优路线问题。并给出从宜昌出发乘火车到上海、南京、杭州、苏州、无锡旅游最后回到宜昌的最优旅游路线。

针对问题一：将铁路网转化为一个带权有向图，把问题转化成带权有向图中的最短路径问题，建立了以换乘次数最少、乘车时间最短及乘车费用最少为目标的多目标优化模型。用分层序列法、广度优先搜索算法和改进的 Dijkstra 算法求解，编程求出优先考虑乘车时间和优先考虑乘车费用两种情况下 3 对始终站之间的最优路线，广度优先搜索算法求解换乘一次的结果如下所示：

	起点	所乘车次	换乘站	换乘车次	终点	时间（h.m）	费用/元
先考虑时间	丹东	K190/k187	镇江	D3006/D3007	宜昌	37.40	513
	天津	T5684/T5681	北京西	T27	拉萨	49.54	374
	白城	K7304	长春	K1056/k1053	青岛	28.47	208
先考虑费用	丹东	K190/k187	南京	K696/k697	宜昌	42.12	322
	天津	T5684/T5681	北京西	T27	拉萨	49.54	374
	白城	2262	天津	K1056/k1053	青岛	46.55	171

针对问题二：根据问题一中任意两城市之间的最优路线方案，建立新的带权有向图，转化为旅行商问题，以乘车时间最少和乘车费用最少为目标的双目标函数，建立 0-1 规划模型，结合分层序列法，用 LINGO 编程求出优先考虑乘车时间和优先考虑乘车费用两种情况下的最优旅游路线，如下所示：

优先考虑乘车时间最少时的最优路线：

宜昌 $\xrightarrow[D3005]{D3008}$ 苏州 $\xrightarrow[D5677]{D5680}$ 杭州 $\xrightarrow{G7362}$ 上海 $\xrightarrow{G7002}$ 南京 $\xrightarrow{G7001}$ 无锡 $\xrightarrow[D3007]{D3006}$ 宜昌

优先考虑乘车费用最少时的最优路线：

宜昌 $\xrightarrow[K695]{K698}$ 无锡 $\xrightarrow[1227]{1230}$ 上海 $\xrightarrow[1229]{1228}$ 苏州 $\xrightarrow[T7785]{T7788}$ 杭州 $\xrightarrow{K102}$ 南京 $\xrightarrow[K697]{K696}$ 宜昌

关键字：多目标最优化　广度优先搜索　改进的 Dijkstra 算法　0-1 规划

一.问题重述

1.1 问题说明

铁路既是社会经济发展的重要载体之一，同时又为社会经济发展创造了前提条件。近几年来，在全社会客运量稳步上升的同时，长期以来铁路承运了大量旅客。相对于其他的运输方式铁路具有时间准确性高、运输能力大、运行比较平稳、安全性高等优点。同时火车也成为了旅途的首选交通运输工具。

虽然目前铁路网络已经比较发达，但是仍然有很多地方之间并没有直接到达的铁路。并且在节假日期间，一些热门路线的火车票总是一票难求。在这种情况下，需要考虑换乘，即先从乘车站到换乘站，再从换乘站到目的站。

1.2 需要解决的问题

1）给出任意两个站点之间的最优铁路路线问题的一般数学模型和算法。若两个站点之间有直达列车，需要考虑直达列车票已售罄情况下最优的换乘方案。根据附录数据，利用你们的模型和算法求出一下起点到终点的最优路线：丹东→宜昌、天津→拉萨、白城→青岛。

2）假设你从打算从宜昌出发乘火车到上海、南京、杭州、苏州、无锡旅游最后回到宜昌，请建立相关数学模型，给出整个行程的最优路线。

二.模型假设

假设 1：接受列车晚点不超过一小时。

假设 2：优先考虑换乘次数。

假设 3：从起点乘车时所等车的时间忽略不计。

三.符号说明

符号	符号说明
V	站点构成的集合
A	任意两个站点之间连通的路线构成的集合
n_0	起始站点
n_k	终到站点
$C(v_i,v_j)$	时间权值集合
$W(v_i,v_j)$	费用权值集合
$f(v_i,v_j)$	任意两个站点之间能否连通构成的邻接矩阵
M_{ij}	换乘次数
$W(n_0,n)$	乘车总费用

$T(n_0,n)$	乘车总时间
x	两站之间的里程
η_0	普通火车基本票率
η_1	动车组票率
η_2	高速动车组票率
ε_i	递远递减率
σ	保险票率
w	客票发展金
ϕ	空调票率

四.问题分析

本文要解决的是铁路换乘最佳路线的选择问题。针对文中提出的问题，在综合考虑换乘次数，乘车时间和乘车费用的前提下，给出任意两个站点之间的最优换乘路线。对问题的具体分析如下：

4.1 问题一的分析

本题要求给出任意两个站点之间的最优铁路路线问题的一般数学模型和算法，并根据此算法求出题中给出的 3 对始终站之间的最优路线。在选择乘车路线时一般考虑三个因素即始终站之间的换乘次数、乘车时间和乘车费用。因为铁路列车的交通网络比较复杂，列车车次繁多，很难掌握各列车的发车时间和所经站点，为了减少换乘所带来的麻烦，把换乘次数最少作为选择最优路线的主要目标，由于主观原因，每个人的经济状况和时间空余安排不同，考虑乘车时间和乘车费用时的主次有所不同，以换乘次数最少、乘车时间最少和乘车费用最少为目标作为多目标函数，变换后两个目标函数的优先级分别求解，在转乘时考虑最大接受列车晚点一小时，不晚点情况下，乘客至少有一小时的换乘时间。将铁路网转化为一个带权有向图，将问题转化为带权有向图中的最短路径问题，运用分层序列法将目标函数分层求解，考虑到广度优先搜索算法计算两次转乘以上时算法复杂度太高，计算速度太慢，改进的 Dijkstra 算法只能输出最优解而不能输出次优解，无法解决高峰期大量缺票的问题，因此分别用广度优先搜索算法和改进的 Dijkstra 算法求解，用 MATLAB 编程求出优先考虑乘车时间和优先考虑乘车费用两种情况下的最优路线。

4.2 问题二的分析

本题要求建立数学模型，求出从宜昌出发乘火车到上海、南京、杭州、苏州、无锡最后回到宜昌的最优旅游路线。问题二是以乘车时间最少和乘车费用最少为目标的双目标最优化问题。由问题一可得到任意两城市之间的最优路线方案，将这六个城市的站点作为节点，两城市之间的最优路线作为边，构成一个新的带权有向图，将问题转化为旅行商问题。每个车站只有一条进去的路线和一条出去的路线；除起点和终点外，各个车站之间不能构成封闭的环。由于主观原因，每个

人的经济状况和时间空余安排不同，考虑乘车时间和乘车费用时的主次有所不同，因此将这两个目标函数的优先级作以变换分别求解。建立 0-1 规划模型，用分层序列法将目标函数分层求解，用 LINGO 编程求出优先考虑乘车时间和优先考虑乘车费用两种情况下的最优路线。

五.数据分析与处理

5.1 车次、站点和车次类型的统计

（1）由于附录中的数据量庞大，车次和站点的冗杂性比较高，利用 MATLAB 编程处理附录中的数据可以得到：全国共有 2867 个站点、4683 个车次，列车类型共有 11 中，其中城际高速只存在于北京到天津、上海到金山卫之间，只有少数地区可供选择，且具有速度高价格贵等特点。

（2）各种类型列车的列车车次数如下图所示：

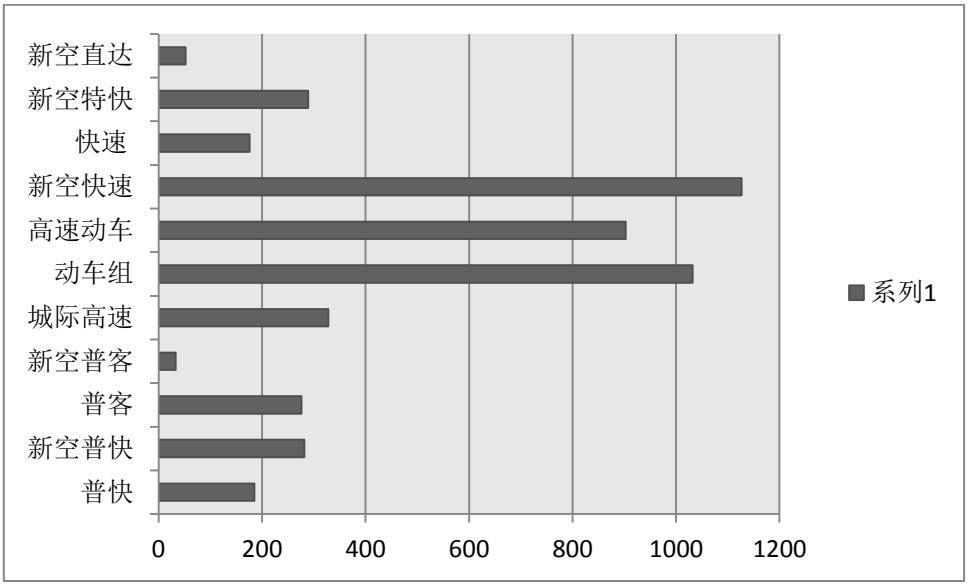

图 1 各种类型列车的列车车次数统计

从上图可以看出，新空快速、高速动车、动车组的车次数明显高于其他列车次数，我国目前运营的列车主要是快速空调列车、动车和高速动车，因此选择出游列车乘车方式时，大多数应该是选择这三种。

5.2 任意两个站点间票价的计算

附录的数据中只给出了起点到各个站点的票价，任意两个站点之间的票价并未给出，根据铁道部公布的信息，票价包括三部分：基本客票票价、附加票票价和其他（保险费、客票发展金等），根据铁道部公布的票价计算方案，可得到各车次的硬座车票价格计算公式如下：

变量的定义：两站之间的里程为 x，普通火车基本票率为 η_0，动车组票率为 η_1，高速动车组票率为 η_2，区段起点为 m_i，区段终点为 n_i，递远递减率为 ε_i，保险票率为 σ，客票发展金为 w，空调票率为 ϕ，从起始站点到第 p 个站点的票价为 w_p，从起始站点到第 q 个站点的票价为 w_q。

普客：$W_1=\eta_0(\sum_{i=1}^{k}(n_i-m_i)\varepsilon_i+(x-n_k))(1+\sigma)+w$

普快：$W_2=\eta_0(\sum_{i=1}^{k}(n_i-m_i)\varepsilon_i+(x-n_k))(1+\sigma+\lambda)+w$

快速：$W_3=\eta_0(\sum_{i=1}^{k}(n_i-m_i)\varepsilon_i+(x-n_k))(1+\sigma+2\lambda)+w$

新空普客：$W_4=\eta_0(\sum_{i=1}^{k}(n_i-m_i)\varepsilon_i+(x-n_k))(1+\sigma+\phi)+w$

新空普快：$W_5=\eta_0(\sum_{i=1}^{k}(n_i-m_i)\varepsilon_i+(x-n_k))(1+\sigma+\lambda+\phi)+w$

新空快速：$W_6=\eta_0(\sum_{i=1}^{k}(n_i-m_i)\varepsilon_i+(x-n_k))(1+\sigma+2\lambda+\phi)+w$

新空特快：$W_7=\eta_0(\sum_{i=1}^{k}(n_i-m_i)\varepsilon_i+(x-n_k))(1+\sigma+2\lambda+\phi)+w$

动车组：$W_8=\eta_1 x$

高速动车：$W_9=\eta_2 x$

新空直达：$W_{10}=y_n-y_{n-1}$

城际高速：$W_{11}=y_n-y_{n-1}$

通过编程计算得到每个车次的列车任意两点之间的票价，计算结果见附录。

六.问题一模型的建立与求解

针对问题一，要求用给出的模型和算法求出 3 对始终站之间的最优路线，首先建立衡量最优路线的三个指标，然后运用分层序列法和广度优先搜索算法求出最优路线。

6.1 模型的建立

将整个铁路线路网建立成一个带权有向图G，将站点作为节点，列车通过两个站点之间的线路作为边，节点集$V=\{v_1,v_2,\cdots,v_n\}$，边集$A=\{a_1,a_2,\cdots,a_m\}$，构成带权有向图$G=(N,A)$，记有向图中由节点v_i，v_j确定的边为$a(v_i,v_j)$。

（1）定义邻接矩阵F，若存在同一辆列车经过v_i和v_j，则v_i和v_j邻接，定义邻接矩阵如下：

$$F=(f(v_i,v_j))_{n\times n}\in\{0,1\}^{n\times n}$$

$$f(v_i,v_j)=\begin{cases}1, & (v_i,v_j)\in A\\ 0, & (v_i,v_j)\in A\end{cases}$$

（2）定义每条边$a(v_i,v_j)$都有若干个行驶时间权值、发车时间权值和乘车费用权值，行驶时间权值为列车从v_i行驶到v_j的行驶时间，发车时间权值为列车从v_i行驶到v_j的列车的出发时间，乘车费用权值为列车从v_i行驶到v_j的费用。记时间权值为$c^k(v_i,v_j)$，发车时间权值为$t^k(v_i,v_j)$，费用权值为$w^k(v_i,v_j)$，若有r个

车次经过站节点v_i和v_j，则$a(v_i,v_j)$有r个时间权值和费用权值，记$a(v_i,v_j)$得时间权值集合和费用权值集合分别为$C(v_i,v_j)$、$W(v_i,v_j)$，有

$$C(v_i,v_j)=\left\{c^1(v_i,v_j),c^2(v_i,v_j)\cdots c^r(v_i,v_j)\right\}$$
$$T(v_i,v_j)=\left\{t^1(v_i,v_j)t^2(v_i,v_j)\cdots t^r(v_i,v_j)\right\}$$
$$W(v_i,v_j)=\left\{w^1(v_i,v_j),w^2(v_i,v_j)\cdots w^r(v_i,v_j)\right\}$$

（3）将从节点n_0到节点n_k的乘车方案表示为：

$$P=\left\{n_0^{x_0},n_1^{x_1},n_2^{x_2}\cdots n_m^{x_m},n_k^{x_k}\right\}$$

其中$n_1,n_2\cdots n_m$为中转节点，x_0表示边$a(n_0,n_1)$的行驶时间权值、发车时间权值和费用权值分别取$c^{x_0}(n_0,n_1)$、$t^{x_0}(n_0,n_1)$和$w^{x_0}(n_0,n_1)$，同理x_k表示边$a(n_k,n_{k+1})$的行驶时间权值、发车时间权值和费用权值分别取$c^{x_k}(n_k,n_{k+1})$、$t^{x_k}(n_k,n_{k+1})$和$w^{x_k}(n_k,n_{k+1})$。

6.1.1 目标函数

（1）换乘次数m最小，在乘车方案P中换乘次数等于中间节点的数目：

$$\min M_{ij}=m$$

（2）乘车所用时间最少，乘车时间=所换乘列车的发车时间—从始点出发的列车的发车时间+乘坐换乘列车的行驶时间：

$$\min T(n_0,n)=\sum_{k=0}^{m}t^{x_k}(n_k,n_{k+1})=t^{x_m}(n_m,n)+c^{x_m}(n_m,n)-t^{x_m}(n_0,n_1)$$

（3）乘车所花费用最少，乘车费用等于从起始站到换乘站之间的车费加上从换乘站到终点站之间的车费：

$$\min W(n_0,n)=\sum_{k=0}^{m}w^{x_k}(n_k,n_{k+1})$$

6.1.2 约束条件

（1）换乘时从起点到达换乘站时的时间要小于换乘列车的出发时间，以保证能够顺利换乘车辆：

$$t_k^{x_k}(n_k,n_{k+1})+c_k^{k+1}(n_k+n_{k+1})\le t_{k+1}^{x_{k+1}}(n_{k+1},n_{k+2})$$

（2）换乘的站点之间要有路线存在，以保证能够换乘成功：

$$\forall\, n_k,n_{k+1}\quad f(n_k,n_{k+1})=1$$

综上所述得到问题一的多目标优化模型如下：

$$\begin{cases}\min M(n_0,n)=m\\ \min T(n_0,n)=(t^{x_m}(n_m,n)+c^{x_m}(n_m,n)-t^{x_m}(n_0,n_1))\\ \min W(n_0,n)=\sum_{k=0}^{m}w^{x_k}(n_k,n_{k+1})\end{cases}$$

$$s.t.\begin{cases}t_k^{x_k}(n_k,n_{k+1})+c_k^{k+1}(n_k+n_{k+1})\le t_{k+1}^{x_{k+1}}(n_{k+1},n_{k+2})\\ \forall\, n_k,n_{k+1}\quad f(n_k,n_{k+1})=1\end{cases}$$

6.2 模型的求解

6.2.1 算法分析

（1）分层序列法的算法求解步骤如下所示：

假设换乘次数$M(n_0,n)$的优先级最高，乘车时间次优，对其求解：

$$\min M(n_0,n)=m$$

$$s.t.\begin{cases} t_k^{x_k}(n_k,n_{k+1})+c_k^{k+1}(n_k+n_{k+1})+\Delta t \le t_{k+1}^{x_{k+1}}(n_{k+1},n_{k+2}) \\ \forall\, n_k,n_{k+1} \quad f(n_k,n_{k+1})=1 \end{cases}$$

求得其最优值为$M^*(n_0,n)$。在可行域D中，$M(n_0,n)\le M^*(n_0,n)$的区域称为最优点集合域，表示为：

$$D_1=\left\{(n_0,n)\,|\,M(n_0,n)\le M^*(n_0,n)\right\}。$$

在D_1内求第二个分目标函数的最优值：

$$\min T(n_0,n)=t^{x_m}(n_m,n)+c^{x_m}(n_m,n)-t^{x_m}(n_0,n_1)$$

$$s.t.\begin{cases} t_k^{x_k}(n_k,n_{k+1})+c_k^{k+1}(n_k,n_{k+1}) \le t_{k+1}^{x_{k+1}}(n_{k+1},n_{k+2}) \\ \forall\, n_k,n_{k+1} \quad f(n_k,n_{k+1})=1 \end{cases}$$

求得其最优值为$T^*(n_0,n)$。$T(n_0,n)$的最优点集合域为：

$$D_2=\left\{(n_0,n)\,|\,M(n_0,n)\le M^*(n_0,n),T(n_0,n)\le T^*(n_0,n)\right\}。$$

在D_2内求$W(n_0,n)$的最优值$W*(n_0,n)$。

（2）广度优先搜索算法：

① 设经过站点n_0的车次集合$H_0=\left\{h_i,h_{i+1}\cdots h_s\right\}$，经过站点$n$的车次集合为$H=\left\{h_j,h_{j+1}\cdots h_p\right\}$，$E=H_0\cap H$。

② 若$E\neq\varnothing$，则从n_0到n可以直达，通过对集合E进行搜索，得到乘车时间$M(n_0,n)$最小的有车次方案集E_1，再通过E_1搜索出费用最小的车次方案集E_2，E_2即为在此优先级下直达的最优解集合。若$E=\varnothing$，则需转乘一次。

③设H_0中所有车次包含的站点的集合为Q_0，H中所有车次包含的站点集合为Q，集合$S=Q_0\cap Q$，S即为中转站点集，通过起点n_0、终点n和中转站集合S即可得到转乘一次的的方案集S_1，通过对集合S_1进行搜索，得到乘车时间$M(n_0,n)$最小的有车次方案集S_2，再通过S_2搜索出费用最小的车次方案集S_3，S_3即为在此优先级下转乘一次的最优解集合。

（3）改进的 Dijkstra 算法：

以u_0为起始站点，$T(u)$设置计时起点为t_0。

① 令$t(u_0)=0$若$v\neq u_0$，令$t(v)=\infty,S_0=\left\{u_0\right\},i=0$。

② 对每个$v\in\overline{S}_i$，若$T(v)$用$\min\limits_{u\in S_i}\left\{t(u),\min(t(u)+c_i(u,v))\right\}$代替$t(v)$，记录$q_i(u,v)$，并存下标号。计算

$\min_{v\in S_i}\{t(v)\}$，把更新后的顶点v的记为u_{i+1}，令$S_{i+1}=S_i\cup\{u_{i+1}\}$。

③ 若$S=V$，则停止算法；若$S\subset V$，用i+1代替i，转到第②步计算。算法停止更新完成后，$t(v)$的值即为从u_0到v得最短时间，通过依次逆向寻找v的前一标号站点得到路径，并通过$q_i(u,v)$确定车次。

6.3.2 模型求解

（1）广度优先搜索算法求解：

用分层序列法和广度优先搜索算法，通过 MATLAB 编程求解得到在目标函数优先级不同的情况下，题中给出的 3 对始终站之间的最优路线如下所示：

当乘车所用时间最少的目标函数的优先级大于乘车所花费用最少的目标函数时，最优路线如下所示：

表 1.优先考虑乘车时间最少时的最优路线及次优路线

起点	所乘车次	换乘站	换乘车次	终点	时间	费用	转乘时间
丹东	K190/k187	镇江	D3006/D3007	宜昌	37.40	513	1.24
丹东	K190/K187	常州	D3006/D3007	宜昌	37.40	538	17.40
天津	T5684/T5681	北京西	T27	拉萨	49.54	374	1.40
天津	K257	郑州	T264/T265	拉萨	50.38	443	1.40
白城	K7304	长春	K1056/k1053	青岛	28.47	208	12.2
白城	K7304	长春	K704/K701	青岛	28.56	208	2.15

当乘车所花费用最少的目标函数的优先级大于乘车所用时间最少的目标函数时，最优路线如下所示：

表 2.优先考虑乘车费用最少时的最优路线及次优路线

起点	所乘车次	换乘站	换乘车次	终点	时间	费用	转乘时间
丹东	K190/k187	南京	K696/k697	宜昌	42.12	322	17.40
丹东	K190/K187	镇江	K696/k697	宜昌	66.12	333	4.2
天津	T5684/T5681	北京西	T27	拉萨	49.54	374	1.40
天津	K7728/K7725	北京西	T27	拉萨	60.42	374	1.40
白城	2262	天津	K1056/k1053	青岛	46.55	171	12.2
白城	2262	天津	K704/K701	青岛	47.4	171	2.15

（2）用改进的 Dijkstra 算法求解：

通过 MATLAB 编程求解得到在目标函数优先级不同的情况下，题中给出的 3 对始终站之间的最优路线如下所示：

表 3.优先考虑乘车时间最短的最优方案

站点	丹东	沈阳北	北京	石家庄	武汉	宜昌
车次	K7318	D606	K7744/K7741	G501	D5876/D5877	——
站点	天津	石家庄北	兰州	拉萨	——	——
车次	K548/K545	Z55	K917	——	——	——
站点	白城	长春	沈阳北	天津	北京南	青岛
车次	K7304	G8020	K1056/K1053	C2002	D333	——

此时乘车从丹东到宜昌所用的时间为 27 小时 15 分钟，乘车费用为 862 元；从天津到拉萨所用的时间为 50 小时 25 分钟，乘车费用为 512 元；从白城到青岛所用的时间为 27 小时 42 分钟，乘车费用为 675 元。每次转乘至少有一个小时的过渡时间。

表 4. 优先考虑转乘次数时乘车的最优方案

起点	所乘车次	换乘车站	换乘车次	终点	时间（h.m）	费用（元）
丹东	K190/k187	镇江	D3006/D3007	宜昌	37.40	513
天津	T5684/T5681	北京西	T27	拉萨	49.54	374
白城	K7304	长春	K1056/k1053	青岛	28.47	208

每次转乘至少有一个小时的过渡时间。对比表 1 和表 4 可以看出，此方案与用分层序列法和广度优先搜索算法求出的最优乘车路线一样。

6.4 问题一的结果分析

（1）由表 1 和表 2 可以看出，在优先考虑乘车时间和优先考虑乘车费用这两种情况下，从丹东到宜昌和从白城到青岛的最优路线是不一样的。根据不同的实际情况，旅客可以根据自己的经济情况和时间安排选择适合自己的乘车路线。

（2）在考虑转乘次数最优时,广度优先搜索算法可以快速求得直达和转乘一次的所有方案,但是当转乘次数超过 2 次时,算法复杂度很高,运算速度很慢。采用改进的 Dijkstra 算法求解时，以转乘次数最少、乘车时间最短和费用最小为目标都能求出最优解，但是只能输出最优解，无法输出次优解，当最优方案无法执行时，没有备选方案，因此应结合二者的求解方案。

七. 问题二模型的建立与求解

7.1 模型的建立

根据问题一的模型和算法，可以求得任意两个站点之间的的最优路线，将问题二中的六个站点构成新的带权有向图G，以六个城市站点作为节点，任意两个城市站点之间的线路作为边，节点集$V=\{v_1,v_2,\cdots,v_n\}$，边集$A=\{a_1,a_2,\cdots,a_m\}$，构成带权有向图$G=(N,A)$，记有向图中A中的由节点v_i，v_j确定的边为$a(v_i,v_j)$。

以问题一中的任意两站点之间的最优路线解作为边的权值，每条边$a(v_i,v_j)$都有唯一的时间权值、票价费用权值，分别用$c(v_i,v_j)$和$w(v_i,v_j)$来表示。

建立 0-1 模型，$f(v_i,v_j)=0$表示不经过从v_i节点到v_j节点的边，$f(v_i,v_j)=1$表示经过从从v_i节点到v_j节点的边。

7.1.1 目标函数

（1）乘车总时间最少，乘车时间等于各条边的时间权值之和：

$$\min T=\sum_{i\neq j}c(v_i,v_j)f(v_i,v_j)$$

（2）乘车总费用最少，乘车费用等于各条边的票价费用权值之和：

$$\min W=\sum_{i\neq j}w(v_i,v_j)f(v_i,v_j)$$

7.1.2 约束条件

（1）每个车站只有一条进去的路线：

$$\forall v_j\in V,\sum_{i=1}^{6}f(v_i,v_j)=1$$

（2）每个车站只有一条出去的路线：

$$\forall v_i \in V, \sum_{j=1}^{6} f(v_i, v_j) = 1$$

（3）除起点和终点外，各个车站之间不能构成封闭环：

$$\sum_{i,j \in s} f(v_i, v_j) \le |s| - 1, 2 \le |s| \le n-1, s \subset \{1, 2, \cdots 6\}$$

综上所述得到问题二的多目标优化模型如下所示：

$$\begin{cases} \min W = \sum_{i \ne j} w(v_i, v_j) f(v_i, v_j) \\ \min T = \sum_{i \ne j} c(v_i, v_j) f(v_i, v_j) \end{cases}$$

$$s.t. \begin{cases} \forall v_i \in V, \sum_{j=1}^{6} f(v_i, v_j) = 1 \\ \forall v_j \in V, \sum_{i=1}^{6} f(v_i, v_j) = 1 \\ \sum_{i,j \in s} f(v_i, v_j) \le |s| - 1, 2 \le |s| \le n-1, s \subset \{1, 2, \cdots 6\} \\ f(v_i, v_j) \in \{0, 1\} \end{cases}$$

7.2 模型的求解

根据问题二所建立的模型，利用 MATLAB 编程求得从宜昌出发经过上海、杭州、苏州、无锡最后回到宜昌的最优旅游路线如下：

表 5.优先考虑乘车时间最少时的最优旅游路线

站点	车次	站点
宜昌	D3008/D3005	苏州
苏州	D5680/D5677	杭州
杭州	G7362	上海
上海	G7002	南京
南京	G7001	无锡
无锡	D3006/D3007	宜昌

选择此路线的乘车总时间为 20 小时 8 分钟，总费用为 1271 元。

表 6. 优先考虑乘车费用最少时的最优旅游路线

站点	车次	站点
宜昌	K698/K695	无锡
无锡	1230/1227	上海
上海	1228/1229	苏州
苏州	T7788/T7785	杭州
杭州	K102	南京
南京	K696/K697	宜昌

选择此路线的总时间为 40 小时 20 分钟，总费用为 276 元。

7.3 问题二的结果分析

由表 5 和表 6 可以看出，当优先考虑乘车时间最少时，乘车的总费用比较高，

是优先考虑乘车费用最少方案所花费用的 4.6 倍；当优先考虑乘车费用最少时，乘车的总时间是前者方案乘车时间的 2 倍，这与实际情况是相符的，速度快的列车车票价格要比普通列车的车票价格贵，而速度快的列车，乘车时间往往比较少，两者不能兼得，旅行者可以根据自己的实际情况选择适合自己的旅游路线。

八.模型的评价

8.1 模型的优点

（1）在数据处理方面，先将每个站点和车次进行编号，并查询出每种车次的票价计算方法，方便模型的建立与求解。

（2）模型建立比较合理，充分考虑了旅客乘车时对换乘次数、乘车时间和乘车费用的不同需求，建立了三个目标函数，用分层序列法将目标函数分清主次，求出了每种情况下的最优路线。

（3）问题一用两种算法求出最优路线，通过对比可以得出每种算法的优缺点，在实际情况中，旅客可以根据自己的情况加以选择。

（4）所建模型为一般数学模型，运用此模型可以求出题中任意两个站点之间的最佳路线。

8.2 模型的不足

问题二的模型是基于问题一的结果建立的，问题一结果的准确性会影响到问题二求解结果的准确性。

十一.参考文献

[1] 司守奎，孙玺菁.数学建模算法与应用.国防工业出版社.2011.

[2] 马良河，刘言斌，廖大庆.城市公交线路网络图的最短路与乘车路线问题.数学实践与认识.2004.6.

[3] 张秀利，梁迎春，董申.多目标结构模糊化的分层序列法.哈尔滨工业大学出版社.1999 .

十二.附录

附录一：任意两个站点之间的车票价格

0	3	6	10	14	24	28	33	38	45	61	66	72	74	82	84	84	92	93	98	109	122	128	132	132	138	138	140
0	0	6	9	12	23	28	33	39	48	69	75	83	87	97	101	103	111	114	124	143	163	173	183	184	193	196	200
0	0	0	5	8	18	23	29	35	43	65	70	79	82	92	97	98	106	109	120	139	158	168	179	180	189	192	196
0	0	0	0	5	15	20	26	32	40	62	67	75	79	89	94	95	103	106	116	136	155	165	175	177	186	188	193
0	0	0	0	0	12	17	23	29	37	59	64	72	76	86	90	92	100	103	113	133	152	162	172	174	183	185	190
0	0	0	0	0	0	7	12	17	26	48	53	61	65	75	79	81	89	92	102	122	141	151	161	163	172	174	178

0	0	0	0	0	0	0	8	13	21	43	48	56	60	70	75	76	84	87	97	117	136	146	156	158	167	169	174
0	0	0	0	0	0	0	0	8	15	37	42	51	55	64	69	71	78	82	92	111	130	141	151	152	161	164	168
0	0	0	0	0	0	0	0	0	10	31	36	45	49	58	63	64	72	76	86	105	124	134	145	146	155	158	162
0	0	0	0	0	0	0	0	0	0	23	28	36	40	50	54	56	64	67	77	97	116	126	136	138	147	149	154
0	0	0	0	0	0	0	0	0	0	0	7	15	18	28	33	34	42	45	56	75	94	104	115	116	125	128	132
0	0	0	0	0	0	0	0	0	0	0	0	10	13	23	28	29	37	40	50	70	89	99	109	111	120	122	127
0	0	0	0	0	0	0	0	0	0	0	0	0	6	15	19	21	29	32	42	61	81	91	101	102	111	114	118
0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	15	17	25	28	38	58	77	87	97	99	108	110	114
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	8	15	18	28	48	67	77	87	89	98	100	105
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	11	14	24	43	63	73	83	84	93	96	100
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	12	22	42	61	71	81	83	92	94	99
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	14	34	53	63	73	75	84	86	91
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	31	50	60	70	72	81	83	87
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20	40	50	60	61	70	73	77
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20	30	41	42	51	54	58
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	21	23	32	34	39
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	13	22	24	29
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	12	14	18
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	13	17
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	9
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

附录二：程序源代码

第一问

DataAnaly.m

```
clear,home
%% 导入数据
%[~,~,data] = xlsread('附件一.xls');
[data] = readtable('Train.csv');
[n,m] = size(data);
for i=1:n
    data.A_Time{i}=data.A_Time{i}(12:end);
    data.D_Time{i}=data.D_Time{i}(12:end);
end
save data.mat

%% 转换时间
[n] = size(data,1);
for i=1:n
    index = strfind(data.A_Time{i},':');
    h = str2double(data.A_Time{i}(1:index(1)-1));
    m = str2double(data.A_Time{i}(index(1)+1:index(2)-1));
    totalM = h*60 + m;
    data.A_Time_CurrentMinute(i) = totalM;
    data.A_Time_TotalMinute(i) = totalM + (data.Day(i)-1)*24*60;
end
for i=1:n
```

```
        index = strfind(data.D_Time{i},':');
        h = str2double(data.D_Time{i}(1:index(1)-1));
        m = str2double(data.D_Time{i}(index(1)+1:index(2)-1));
        totalM = h*60 + m;
        data.D_Time_CurrentMinute(i) = totalM;
        data.D_Time_TotalMinute(i) = totalM + (data.Day(i)-1)*24*60;
    end


    save data2.mat

    %% 将每一列车的站点信息放入一行，保存成 CVS 格式
    [n,m] = size(data);
    fileID = fopen('trainLine.txt','w');
    for i=1:n
        if data.S_No(i)==1
            fprintf(fileID,'\n%s,%s',data.ID{i},data.Type{i});
        end
        fprintf(fileID,',%s',data.Station{i});
    end
    fclose(fileID);
    %% 列车类型与列车编号
    trainType = unique(data.Type);
    n = size(trainType,1);
    typeIndex = cell(n,1);
    for i = 1:n
        typeIndex{i} = find(strcmp(data.Type,trainType(i)));
    end

    trainNo = unique(data.ID);
    n = size(data,1);
    for i=1:n
        [~,No] = intersect(trainNo,data.ID{i});
        data.NumNo(i) = No;
    end
    %% 站点编号
    station = unique(data.Station);
    n = size(data,1);
    for i=1:n
        [~,No] = intersect(station,data.Station{i});
        data.StationNo(i) = No;
    end
    %% 建立连接矩阵
    m = length(station);
    startStation = find(data.S_No==1);
    adjMat = zeros(m,m);
    n = size(data,1);
    for i = 1:n-1
        endStation = startStation(find(startStation > i,1))-1;
        % [~,s_No] = intersect(station,data.Station{i});
        for j = i+1:endStation
            % [~,d_No] = intersect(station,data.Station{j});
            adjMat(data.StationNo(i),data.StationNo(j)) = 1;
        end
```

```
    end
    save data3
    %% 任意两站点之间的车次
    m = length(station);
    startStation = find(data.S_No==1);
    trainList = cell(m,m);
    n = size(data,1);
    for i = 1:n-1
        endStation = startStation(find(startStation > i,1))-1;
        for j = i+1:endStation
            trainList{data.StationNo(i),data.StationNo(j)} =
[trainList{data.StationNo(i),data.StationNo(j)},data.NumNo(i)];
        end
    end
    save data4.mat
```

directSearchFun1.m

```
    function [ path,pathTime,pathCost ] =
directSearchFun1( data,trainList,adjMat,station,s_name,d_name )

    [~,s_No] = intersect(station,s_name);
    [~,d_No] = intersect(station,d_name);

    %% 直达

    if isempty(trainList{s_No,d_No})
        path = [];
    else
        path = trainList{s_No,d_No}';
    end
    path = [ones(size(path,1),1)*s_No path ones(size(path,1),1)*d_No];

    %% 计算时间

    pathTime = zeros(size(path,1),1);
    for i=1:size(path,1)
        pathTime(i) = timeFun( data,path(i,:) );
    end

    %% 计算费用
    pathCost = zeros(size(path,1),1);
    for i=1:size(path,1)
        [pathCost(i),~] = pathCostFun( data,path(i,:) );
    end



    end
```

directSearchFun2.m

```
    function [ path,pathTime,pathCost,stopTime ] =
directSearchFun2( data,trainList,adjMat,station,s_name,d_name )

    [~,s_No] = intersect(station,s_name);
```

```
[~,d_No] = intersect(station,d_name);

%% 一次转乘
path=[];
s_set = find(adjMat(s_No,:)==1);
d_set = find(adjMat(:,d_No)==1);
stop = intersect(s_set, d_set);
for i=1:length(stop)
    firstarc = trainList{s_No,stop(i)};
    for j = 1:length(firstarc)
        secondarc = trainList{stop(i),d_No};
        for k = 1:length(secondarc)
            path = [path;[firstarc(j) stop(i) secondarc(k)]];
        end
    end
end
path = [ones(length(path),1)*s_No path ones(length(path),1)*d_No];
%% 计算时间

pathTime = zeros(size(path,1),1);
stopTime = cell(size(path,1),1);
for i=1:size(path,1)
    [pathTime(i),stopTime{i}] = timeFun( data,path(i,:) );
end

%% 计算费用

pathCost = zeros(size(path,1),1);
for i=1:size(path,1)
    [pathCost(i),~] = pathCostFun( data,path(i,:) );
end

end
```

costFun.m

```
function [ cost ] = costFun( data,s_No,d_No,trainID )
%costFun 计算两个站点之间某车次的票价
%    data 为列车数据库
%    s_No 为出发点编号
%    d_No 为目的地编号
%    trainNo 为车次编号
%    cost 为票价

% s_No = 933; d_No = 1065; trainID = 1154;
s = find(data.NumNo==trainID&data.StationNo==s_No);
d = find(data.NumNo==trainID&data.StationNo==d_No);

if data.S_No(s)==1&&(~isempty(data.P1(d)))&&(~isnan(data.P1(d)))
    cost = data.P1(d);
    return
end
if data.S_No(s)==1&&(~isempty(data.P3(d)))&&(~isnan(data.P3(d)))
    cost = data.P3(d);
```

```
        return
    end

    x = data.Distance(d) - ...
        data.Distance(s);
    index = find(data.NumNo==trainID,1);
    if strcmp(data.Type{index},'城际高速')
        cost = min(table2array(data(d,9:12))) - min(table2array(data(s,9:12)));
        cost =round(cost);
        return
    end
    if strcmp(data.Type{index},'动车组')
        cost = 0.3085*x;
        cost =round(cost);
        return
    end
    if strcmp(data.Type{index},'高速动车')
        cost = 0.485*x;
        cost =round(cost);
        return
    end
    if strcmp(data.Type{index},'新空直达')

        cost = min(table2array(data(d,9:12))) - min(table2array(data(s,9:12)));
        cost =round(cost);
        return
    end

    sigma = 1;
    if strcmp(data.Type{index},'普客')
        sigma = sigma+0.02;
    end
    if strcmp(data.Type{index},'普快')
        sigma = sigma+0.02+0.2;
    end
    if strcmp(data.Type{index},'新空普快')
        sigma = sigma+0.02+0.2+0.25;
    end
    if strcmp(data.Type{index},'新空普客')
        sigma = sigma+0.02+0.25;
    end
    if strcmp(data.Type{index},'快速')
        sigma = sigma+0.02+0.4;
    end
    if strcmp(data.Type{index},'新空快速')
        sigma = sigma+0.02+0.4+0.25;
    end
    if strcmp(data.Type{index},'新空特快')
        sigma = sigma+0.02+0.4+0.25;
    end


    eta = 0.05861;
```

```
if 0<x&&x>200
    cost = eta*x*sigma+1;
elseif 200>x&&x<=500
    cost = eta*(200+(x-200)*0.9)*sigma+1;
elseif 500>x&&x<=1000
    cost = eta*(200+300*0.9+(x-500)*0.8)*sigma+1;
elseif 1000>x&&x<=1500
    cost = eta*(200+300*0.9+500*0.8+(x-1000)*0.7)*sigma+1;
elseif 1500>x&&x<=2500
    cost = eta*(200+300*0.9+500*0.8+500*0.7+(x-1500)*0.6)*sigma+1;
else
    cost = eta*(200+300*0.9+500*0.8+500*0.7+1000*0.6+(x-2500)*0.5)*sigma+1;
end

cost =round(cost);



end
```

timeFun.m

```
function [ totalTime,stopTime,startTime ] = timeFun( data,sigpath )
%timeFun 计算路径方案的时间
%    data 列车数据
%    path 路径，格式为起点-车次-中转站-车次-终点
%    totalTime 方案的总时间
%    stopTime 方案的中间停留时间
%    startTime 方案的出发时间
n = (length(sigpath)-1)/2;
section = zeros(n,2);
sectionTime = zeros(n,1);
stopTime = zeros(n-1,1);
for i=1:n
    temp = sigpath(i*2-1:i*2+1);
    m = find(data.NumNo==temp(2)&data.StationNo==temp(1));
    n = find(data.NumNo==temp(2)&data.StationNo==temp(3));
    section(i,:) = [m n];
    sectionTime(i) = data.A_Time_TotalMinute(n)-data.D_Time_TotalMinute(m);
end
if n>1
    n = (length(sigpath)-1)/2;
    for i=1:n-1
        s_stop = data.A_Time_CurrentMinute(section(i,2));
        d_stop = data.D_Time_CurrentMinute(section(i+1,1));
        if d_stop<s_stop
            stopTime(i) = d_stop + 24*60 - s_stop;
        else
            stopTime(i) = d_stop - s_stop;
        end
    end
else
    stopTime = 0;
end
```

```
totalTime = sum(sectionTime) + sum(stopTime);
startTime = data.D_Time(section(1,1));

end
```

bestPathFun.m

```
function [ bestIndex ] = bestPathFun( pathCost,pathTime,type )
%bestPathFun 将方案按照一定的优先级进行排序
%     path 所有方案
%     type 优先级类型：1 为时间优先，2 为费用优先
%     bestPath 方案的排序

if type==1
    [~,bestIndex]=sort(pathTime);
end
if type==2
    [~,bestIndex]=sort(pathCost);
end

end
```

displayFun.m

```
function displayFun( path,pathCost,pathTime,station,trainNo )
%displayFun 输出方案

for i=1:length(path)
    if mod(i,2)==1
        fprintf('%s-',station{path(i)})
    else
        fprintf('%s-',trainNo{path(i)})
    end
end

fprintf('%d 元-',pathCost)
h = floor(pathTime/60);
m = mod(pathTime,60);
fprintf('%d 小时%d 分钟\n',h,m)

end
```

main.m

```
% 主程序

% 输出欢迎界面
fprintf('欢迎使用铁路查询系统！\n 请输入起始站与终点站\n')
% 用户输入数据
s_name = input('起始站：','s');
while ~sum(strcmp(station,s_name))
    fprintf('Sorry，没有找到此站点，请重新输入！\n')
    s_name = input('起始站：','s');
end
d_name = input('终点站：','s');
while ~sum(strcmp(station,d_name))
```

```
        fprintf('Sorry，没有找到此站点，请重新输入！\n')
        d_name = input('终点站：','s');
    end

    type = input('输入优先级，时间优先输入 1，费用优先输入 2：');



    path = cell(2,1);
    pathTime = cell(2,1);
    pathCost = cell(2,1);
    % 直达方案
    [ path{1},pathTime{1},pathCost{1} ] = directSearchFun1(...
        data,trainList,adjMat,station,s_name,d_name );
    goOn = 0;
    if isempty(path{1})
        fprintf('%s 到%s ',s_name,d_name)
        fprintf('没有直达车次 为您推荐如下转乘方案：\n')
        goOn = 1;
    else
        [ bestIndex{1} ] = bestPathFun( pathCost{1},pathTime{1},type );
        for i=1:length(path{1})
            n = bestIndex{1}(i);
            displayFun( path{1}(n,:),pathCost{1}(n),pathTime{1}(n),station,trainNo )
        end
    end
    % 转乘一次
    goOn = input('是否继续搜索一次转乘方案，若是输入 1，否则输入 0：');
    while goOn~=1&&goOn~=0
        goOn = input('是否继续搜索一次转乘方案，若是输入 1，否则输入 0：');
    end
    if goOn == 1
        [ path{2},pathTime{2},pathCost{2},stopTime ] = directSearchFun2(...
            data,trainList,adjMat,station,s_name,d_name );
        if isempty(path{2})
            fprintf('%s 到%s ',s_name,d_name)
            fprintf('没有一次转乘方案，无法搜索，Sorry\n')
        else
            [ bestIndex{2} ] = bestPathFun( pathCost{2},pathTime{2},type );
            for i=1:length(path{2})
                n = bestIndex{2}(i);
                displayFun( path{2}(n,:),pathCost{2}(n),pathTime{2}(n),station,trainNo )
                h = floor(stopTime{i}./60);
                m = mod(stopTime{i},60);
                fprintf('中转站等待时间：%d 小时%d 分钟\n',h,m)
            end
        end
    end
```

oneBestPathFun.m

```
function [ bestPath,cost,time ] = oneBestPathFun( data,trainList,adjMat,station,s_name,d_name,type )
```

```
    %oneBestPathFun  输出一条最优路线

    [                          path,pathTime,pathCost                          ]                          =
directSearchFun1( data,trainList,adjMat,station,s_name,d_name );
    if type==1
        [directMin,directIndex] = min(pathTime);

        bestPath = path(directIndex,:);
        time = pathTime(directIndex);
        cost = pathCost(directIndex);
    else
        [directMin,directIndex] = min(pathCost);

        bestPath = path(directIndex,:);
        time = pathTime(directIndex);
        cost = pathCost(directIndex);
    end

    end
```

pathCostFun.m

```
    function [ totalCost,cost ] = pathCostFun( data,sigpath )
    %pathCostFun  计算路径方案的总费用
    %     path 为路径方案
    %     totalCost 为总费用
    %     cost 为各车次票价
    n = (length(sigpath)-1)/2;
    cost = zeros(n,1);
    for i=1:n
        cost(i) = costFun( data,sigpath(i*2-1),sigpath(i*2+1),sigpath(i*2) );
    end
    totalCost = sum(cost);

    end
```

startStationMFile.m

```
    m = length(station);
    startStation = find(data.S_No==1);
    trainList = cell(m,m);
    n = size(data,1);
    for i = 1:n-1
        endStation = startStation(find(startStation > i,1))-1;
        for j = i+1:endStation
            trainList{data.StationNo(i),data.StationNo(j)}                          =
[trainList{data.StationNo(i),data.StationNo(j)},data.NumNo(i)];
        end
    end
    save data4.mat
```

dijkstraMain.m

```
    tic
    s = find(strcmp(station,'上海'));
    d = find(strcmp(station,'苏州'));
    beginH = 8;
```

```
beginM = 0;
beginTime = beginH*60 + beginM;
[ t,p,q ] = dijkstraTurnFun( adjMat,C,TS,s,d,beginTime,60 );
toc
p(s) = 0;
path = graphpred2path(p',d);
n = size(path,2)*2-1;
fullpath = zeros(1,n);
% fullpath(1) = path(1);
% fullpath(2) = trainList{path(1),path(2)}(q(path(2)));
% fullpath(3) = path(2);
% fullpath(4) = trainList{path(2),path(3)}(q(path(3)));
for i = 1:n
    if mod(i,2)==1
        fullpath(i) = path((i+1)/2);
    else
        fullpath(i) = trainList{path(i/2),path(i/2+1)}(q(path(i/2+1)));
    end
end
pathCost = pathCostFun( data,fullpath );
[ totalTime,stopTime,startTime ] = timeFun( data,fullpath );
displayFun( fullpath,pathCost,totalTime,station,trainNo )
disp(stopTime)
% s_No = 1579;
% d_No = 933;
% station(s_No)
% station(d_No)
% train = trainNo(trainList{s_No,d_No}(q(d_No)))
```

DataPossess.m

```
%% 任意两站点之间的车次
m = length(station);
startStation = find(data.S_No==1);
trainList = cell(m,m);
C = cell(m,m);
TS = cell(m,m);
r = zeros(m,m);
n = size(data,1);
h = waitbar(0,'Please wait...');
for i = 1:n-1
    endStation = startStation(find(startStation > i,1))-1;
    for j = i+1:endStation
        s_No = data.StationNo(i);
        d_No = data.StationNo(j);
        sigpath = data.NumNo(i);
        trainList{s_No,d_No} = [trainList{s_No,d_No},sigpath];
        %[ totalTime,~,startTime ] = timeFun( data,[s_No,sigpath,d_No] );
        C{s_No,d_No} =
[C{s_No,d_No},data.A_Time_TotalMinute(j)-data.D_Time_TotalMinute(i)];
        TS{s_No,d_No} = [TS{s_No,d_No},data.D_Time_CurrentMinute(i)];
        r(s_No,d_No) = r(s_No,d_No) + 1;
    end
    waitbar(i/n)
end
```

```
close(h)
save data6.mat

DG = sparse(adjMat);
minTurn = graphallshortestpaths(DG);
```

dijkstraFun.m

```
function [ t,p,q ] = dijkstraFun( adjMat,C,TS,s,d,beginTime,stopTime )
%dijkstraFun 改进的 Dijkstra 算法
%     adjMat 邻接矩阵
%     c 节点之间的行驶时间矩阵
%     TS 出发时间

N = size(adjMat,1);

Node = 1:N; % 节点集合
t = zeros(N,1); % 到达节点 u 的最短时间
%dayT = zeros(N,1); % 到达节点 u 时的当天时间
p = zeros(N,1); % s 到 u 最短路径上的 u 的前一个节点
q = zeros(N,1); % s 到 u 最短路径上 p(u)到 u 出发时间的索引
%S = [];

%s = 1; % 设置出发点
% 初始化
t(s) = beginTime;
%dayT(s) = 0;
p(s) = nan;
q(s) = nan;
S = s;

for v=setdiff(Node,s)
    t(v) = inf;
    p(v) = nan;
    q(v) = nan;
end

while ~isempty(S)
    [~,index] = min(t(S));
    u = S(index);
    S = setdiff(S,u);
    for v = find(adjMat(u,:))

        for i = 1:length(TS{u,v})
            temp = TS{u,v}(i);
            while temp<t(u)+stopTime;
                temp = temp + 24*60;
            end
            if (t(v)>temp+C{u,v}(i))
                p(v) = u;
                q(v) = i;
                t(v) = temp + C{u,v}(i);
                S = union(S,v);
            end
        end
```

```
        end
        if u == d
            return
        end
    end
    end
```

dijkstraTurnFun.m

```
function [ t,p,q,turn ] = dijkstraTurnFun( adjMat,C,TS,s,d,beginTime,stopTime )
%dijkstraFun 改进的 Dijkstra 算法
%     adjMat 邻接矩阵
%     c 节点之间的行驶时间矩阵
%     TS 出发时间

N = size(adjMat,1);

Node = 1:N; % 节点集合
t = zeros(N,1); % 到达节点 u 的最短时间
turn = zeros(N,1); % 到达节点 u 的最小转乘次数
%dayT = zeros(N,1); % 到达节点 u 时的当天时间
p = zeros(N,1); % s 到 u 最短路径上的 u 的前一个节点
q = zeros(N,1); % s 到 u 最短路径上 p(u)到 u 出发时间的索引
%S = [];

%s = 1; % 设置出发点
% 初始化
t(s) = beginTime;
turn(s) = 0;
%dayT(s) = 0;
p(s) = nan;
q(s) = nan;
S = s;

for v=setdiff(Node,s)
    t(v) = inf;
    turn(v) = inf;
    p(v) = nan;
    q(v) = nan;
end

while ~isempty(S)
    tAndTurn = [t(S),turn(S)];
    [~,index] = sortrows(tAndTurn,[2,1]);
    index = index(1);
    %[~,index] = min(t(S));
    %[~,index] = min(turn(S));
    u = S(index);
    S = setdiff(S,u);
    for v = find(adjMat(u,:))
        for i = 1:length(TS{u,v})
            temp = TS{u,v}(i);
            while temp<t(u)+stopTime;
```

```
                temp = temp + 24*60;
            end
            if
(turn(v)>turn(u)+1)||((turn(v)==turn(u)+1)&&(t(v)>temp+C{u,v}(i)))%(t(v)>temp+C{u,v}(i))
                p(v) = u;
                q(v) = i;
                t(v) = temp + C{u,v}(i);
                turn(v) = turn(u) + 1;
                S = union(S,v);
            end
        end

    end
    if u == d
        return
    end
end
end
```

第二问

main2.m

```
travel = {'宜昌东','上海','南京','杭州','苏州','无锡'};
n = length(travel);
TimeFirstBestPath=cell(n,n);
TimeFirstCost = ones(n,n)*inf;
TimeFirstTime = ones(n,n)*inf;
type = 1;
for i=1:n
    for j=1:n
        if i==j
            continue
        end
        [       TimeFirstBestPath{i,j},TimeFirstCost(i,j),TimeFirstTime(i,j)       ]       =
oneBestPathFun( data,trainList,adjMat,station,travel{i},travel{j},type );
    end
end

CostFirstBestPath=cell(n,n);
CostFirstCost = ones(n,n)*inf;
CostFirstTime = ones(n,n)*inf;
type = 2;
for i=1:n
    for j=1:n
        if i==j
            continue
        end
        [       CostFirstBestPath{i,j},CostFirstCost(i,j),CostFirstTime(i,j)       ]       =
oneBestPathFun( data,trainList,adjMat,station,travel{i},travel{j},type );
    end
end
```

WENTI2.lg4

```
MODEL:
```

```
    SETS:
    CITY / 1.. 6/: U; ! U( I) = sequence no. of city;
    LINK( CITY, CITY):
         DIST,  ! The distance matrix;
           X;  ! X( I, J) = 1 if we use link I, J;
    ENDSETS

    DATA:   !Distance matrix, it need not be symmetric;
    DIST =
              0       1064        767       1215        452
435
           1016          0         99         90         25
42
            774         99          0        228         71
55
           1159         86        216          0        106
123
            428         24         72        105          0
15
            411         41         55        123         14
0;

     ! 0  702  454  842 2396 1196
           702    0  324 1093 2136  764
           454  324    0 1137 2180  798
           842 1093 1137    0 1616 1857
          2396 2136 2180 1616    0 2900
          1196  764  798 1857 2900    0;

    ENDDATA
    !The model:Ref. Desrochers & Laporte, OR Letters,
    Feb. 91;

    N = @SIZE( CITY);
    MIN = @SUM( LINK: DIST * X);
    @FOR( CITY( K):
    !  It must be entered;
      @SUM( CITY( I)| I #NE# K: X( I, K)) = 1;
    !  It must be departed;
      @SUM( CITY( J)| J #NE# K: X( K, J)) = 1;
    ! Weak form of the subtour breaking constraints;
    ! These are not very powerful for large problems;
      @FOR( CITY( J)| J #GT# 1 #AND# J #NE# K:
          U( J) >= U( K) + X ( K, J) -
          ( N - 2) * ( 1 - X( K, J)) +
          ( N - 3) * X( J, K)
      );
    );
    ! Make the X's 0/1;
    @FOR( LINK: @BIN( X));
    ! For the first and last stop we know...;
    @FOR( CITY( K)| K #GT# 1:
      U( K) <= N - 1 - ( N - 2) * X( 1, K);
      U( K) >= 1  + ( N - 2) * X( K, 1)
    );

    END
```

附录三：丹东到宜昌优先考虑乘车时间时的路线：

始点	车次	换乘站	车次	终到站	车费	乘车时间
丹东	K190/K187	常州	D3006/D3007	宜昌东	538元	37小时40分钟
丹东	K190/K187	无锡	D3006/D3007	宜昌东	550元	37小时40分钟
丹东	K190/K187	镇江	D3006/D3007	宜昌东	513元	37小时40分钟
丹东	K190/K187	南京	K696/K697	宜昌东	322元	42小时12分钟
丹东	K190/K187	南京	K1512/K1513	宜昌东	342元	47小时9分钟
丹东	K190/K187	常州	K1512/K1513	宜昌东	362元	47小时9分钟
丹东	K190/K187	镇江	K1512/K1513	宜昌东	352元	47小时9分钟
丹东	K190/K187	常州	D3072/D3073	宜昌东	538元	54小时55分钟
丹东	K190/K187	无锡	D3072/D3073	宜昌东	550元	54小时55分钟
丹东	K190/K187	苏州	D3072/D3073	宜昌东	571元	54小时55分钟
丹东	K190/K187	镇江	D3072/D3073	宜昌东	513元	54小时55分钟
丹东	K190/K187	苏州	D3006/D3007	宜昌东	571元	61小时40分钟
丹东	K190/K187	上海	K696/K697	宜昌东	418元	66小时12分钟
丹东	K190/K187	常州	K696/K697	宜昌东	343元	66小时12分钟
丹东	K190/K187	无锡	K696/K697	宜昌东	346元	66小时12分钟
丹东	K190/K187	苏州	K696/K697	宜昌东	359元	66小时12分钟
丹东	K190/K187	镇江	K696/K697	宜昌东	333元	66小时12分钟
丹东	K190/K187	无锡	K1512/K1513	宜昌东	413元	71小时9分钟

附录四：丹东到宜昌优先考虑乘车费用时的路线：

始点	车次	换乘站	车次	终到站	车费	乘车时间
丹东	K190/K187	南京	K696/K697	宜昌东	322元	42小时12分钟
丹东	K190/K187	镇江	K696/K697	宜昌东	333元	66小时12分钟
丹东	K190/K187	南京	K1512/K1513	宜昌东	342元	47小时9分钟
丹东	K190/K187	常州	K696/K697	宜昌东	343元	66小时12分钟
丹东	K190/K187	无锡	K696/K697	宜昌东	346元	66小时12分钟
丹东	K190/K187	镇江	K1512/K1513	宜昌东	352元	47小时9分钟
丹东	K190/K187	苏州	K696/K697	宜昌东	359元	66小时12分钟
丹东	K190/K187	常州	K1512/K1513	宜昌东	362元	47小时9分钟
丹东	K190/K187	无锡	K1512/K1513	宜昌东	413元	71小时9分钟
丹东	K190/K187	上海	K696/K697	宜昌东	418元	66小时12分钟
丹东	K190/K187	镇江	D3006/D3007	宜昌东	513元	37小时40分钟
丹东	K190/K187	镇江	D3072/D3073	宜昌东	513元	54小时55分钟
丹东	K190/K187	常州	D3006/D3007	宜昌东	538元	37小时40分钟
丹东	K190/K187	常州	D3072/D3073	宜昌东	538元	54小时55分钟
丹东	K190/K187	无锡	D3006/D3007	宜昌东	550元	37小时40分钟
丹东	K190/K187	无锡	D3072/D3073	宜昌东	550元	54小时55分钟
丹东	K190/K187	苏州	D3006/D3007	宜昌东	571元	61小时40分钟
丹东	K190/K187	苏州	D3072/D3073	宜昌东	571元	54小时55分钟

附录五：天津到拉萨优先考虑乘车时间时的路线：

始点	车次	换乘站	车次	终到站	费用	乘车时间
天津	T5684/T5681	北京西	T27	拉萨	374元	49小时54分钟
天津	K257	郑州	T264/T265	拉萨	443元	50小时38分钟
天津	K548/K545	太原	T27	拉萨	381元	51小时30分钟
天津	K548/K545	石家庄北	T27	拉萨	381元	51小时30分钟
天津	K257	郑州	T164/T165	拉萨	443元	51小时33分钟
天津	T253	郑州	T264/T265	拉萨	443元	54小时5分钟
天津	K919	郑州	T264/T265	拉萨	443元	54小时48分钟
天津	T253	郑州	T164/T165	拉萨	443元	55小时0分钟
天津	K548/K545	西安	T264/T265	拉萨	418元	55小时10分钟
天津	1472/1469	徐州	T164/T165	拉萨	425元	55小时11分钟
天津	K919	郑州	T164/T165	拉萨	443元	55小时43分钟
天津	1344/1341	徐州	T164/T165	拉萨	425元	55小时59分钟
天津	1344/1341	蚌埠	T164/T165	拉萨	455元	55小时59分钟
天津	K548/K545	西安	T164/T165	拉萨	418元	56小时5分钟
天津	K928/K925	郑州	T264/T265	拉萨	418元	56小时10分钟
天津	K75/K78	徐州	T164/T165	拉萨	433元	56小时44分钟
天津	K75/K78	蚌埠	T164/T165	拉萨	465元	56小时44分钟
天津	K928/K925	郑州	T164/T165	拉萨	418元	57小时5分钟
天津	K1062/K1063	郑州	T264/T265	拉萨	418元	58小时35分钟
天津	K976/K973	郑州	T264/T265	拉萨	418元	59小时11分钟
天津	T122/T123	武昌	T264/T265	拉萨	516元	59小时27分钟
天津	T122/T123	郑州	T264/T265	拉萨	410元	59小时27分钟
天津	K1062/K1063	郑州	T164/T165	拉萨	418元	59小时30分钟

附录六：天津到拉萨优先考虑乘车费用时的路线：

始点	车次	换乘站	车次	终到站	费用	乘车时间
天津	K7728/K7725	北京西	T27	拉萨	374元	60小时42分钟
天津	T5684/T5681	北京西	T27	拉萨	374元	49小时54分钟
天津	1136/1133	北京西	T27	拉萨	378元	66小时47分钟
天津	K548/K545	太原	T27	拉萨	381元	51小时30分钟
天津	K548/K545	石家庄北	T27	拉萨	381元	51小时30分钟
天津	K522/K519	太原	T27	拉萨	384元	64小时42分钟
天津	K868/K865	太原	T27	拉萨	384元	66小时20分钟
天津	K522/K519	石家庄北	T27	拉萨	384元	64小时42分钟
天津	K868/K865	石家庄北	T27	拉萨	384元	66小时20分钟
天津	2602/2603	太原	T27	拉萨	395元	66小时8分钟
天津	K213	石家庄北	T27	拉萨	405元	70小时48分钟
天津	K213	太原	T27	拉萨	410元	70小时48分钟
天津	K388/K385	宝鸡	T22/T23	拉萨	410元	61小时44分钟
天津	K388/K385	宝鸡	T222/T223	拉萨	410元	61小时44分钟
天津	K128/K125	西安	T164/T165	拉萨	410元	65小时57分钟
天津	K128/K125	西安	T222/T223	拉萨	410元	62小时17分钟
天津	K128/K125	西安	T264/T265	拉萨	410元	65小时2分钟
天津	K388/K385	西安	T164/T165	拉萨	410元	65小时24分钟
天津	K388/K385	西安	T222/T223	拉萨	410元	61小时44分钟
天津	K388/K385	西安	T264/T265	拉萨	410元	64小时29分钟
天津	K128/K125	郑州	T164/T165	拉萨	410元	65小时57分钟
天津	K128/K125	郑州	T264/T265	拉萨	410元	65小时2分钟
天津	K369/K368	郑州	T164/T165	拉萨	410元	66小时53分钟
天津	K369/K368	郑州	T264/T265	拉萨	410元	65小时58分钟
天津	K388/K385	郑州	T164/T165	拉萨	410元	65小时24分钟
天津	K388/K385	郑州	T264/T265	拉萨	410元	64小时29分钟
天津	T122/T123	郑州	T164/T165	拉萨	410元	60小时22分

附录七：白城到青岛优先考虑乘车时间时的路线：

始点	车次	换乘站	车次	终到站	费用	乘车时间
白城	K7304	长春	K1056/K1053	青岛	208元	28小时47分钟
白城	K7304	长春	K704/K701	青岛	208元	28小时56分钟
白城	K1302	黄村	K712/K709	青岛	215元	31小时39分钟
白城	K7310	长春	K1056/K1053	青岛	224元	32小时0分钟
白城	2082/2084	四平	K1056/K1053	青岛	193元	32小时2分钟
白城	2082/2084	开原	K1056/K1053	青岛	193元	32小时2分钟
白城	2082/2084	昌图	K1056/K1053	青岛	193元	32小时2分钟
白城	2082/2084	沈阳北	K1056/K1053	青岛	191元	32小时2分钟
白城	2082/2084	铁岭	K1056/K1053	青岛	193元	32小时2分钟
白城	K7310	长春	K704/K701	青岛	224元	32小时9分钟
白城	2082/2084	四平	K704/K701	青岛	193元	32小时11分钟
白城	2082/2084	开原	K704/K701	青岛	193元	32小时11分钟
白城	2082/2084	昌图	K704/K701	青岛	193元	32小时11分钟
白城	2082/2084	沈阳北	K704/K701	青岛	191元	32小时11分钟
白城	2082/2084	铁岭	K704/K701	青岛	193元	32小时11分钟
白城	K7302	长春	K1056/K1053	青岛	224元	33小时9分钟
白城	2082/2084	沈阳	K958/K955	青岛	192元	33小时17分钟
白城	K7302	长春	K704/K701	青岛	224元	33小时18分钟
白城	2082/2084	沈阳	K972/K969	青岛	193元	34小时33分钟
白城	K1302	兴城	K1056/K1053	青岛	196元	36小时39分钟
白城	K1302	唐山	K1056/K1053	青岛	196元	36小时39分钟
白城	K1302	四平	K1056/K1053	青岛	198元	36小时39分钟
白城	K1302	塘沽	K1056/K1053	青岛	197元	36小时39分钟
白城	K1302	大虎山	K1056/K1053	青岛	198元	36小时39分钟
白城	K1302	天津	K1056/K1053	青岛	196元	36小时39分钟
白城	K1302	山海关	K1056/K1053	青岛	196元	36小时39分钟
白城	K1302	开原	K1056/K1053	青岛	198元	36小时39分钟
白城	K1302	昌图	K1056/K1053	青岛	198元	36小时39分钟

附录八：白城到青岛优先考虑乘车费用时的路线：

始点	车次	换乘站	车次	终到站	费用	乘车时间
白城	2262	天津	K1056/K1053	青岛	171元	46小时55分钟
白城	2262	天津	K704/K701	青岛	171元	47小时4分钟
白城	2262	天津	K958/K955	青岛	171元	48小时10分钟
白城	2262	天津	K972/K969	青岛	171元	49小时26分钟
白城	2262	塘沽	K1056/K1053	青岛	172元	46小时55分钟
白城	2262	塘沽	K972/K969	青岛	172元	49小时26分钟
白城	2262	唐山	K1056/K1053	青岛	173元	46小时55分钟
白城	2262	唐山	K704/K701	青岛	173元	47小时4分钟
白城	2262	唐山	K958/K955	青岛	173元	48小时10分钟
白城	2262	唐山	K972/K969	青岛	173元	49小时26分钟
白城	1468	山海关	K1056/K1053	青岛	174元	40小时19分钟
白城	1468	山海关	K704/K701	青岛	174元	40小时28分钟
白城	1468	山海关	K958/K955	青岛	174元	41小时34分钟
白城	1468	山海关	K972/K969	青岛	174元	42小时50分钟
白城	2262	山海关	K1056/K1053	青岛	174元	46小时55分钟
白城	2262	山海关	K704/K701	青岛	174元	47小时4分钟
白城	2262	山海关	K958/K955	青岛	174元	48小时10分钟
白城	2262	山海关	K972/K969	青岛	174元	49小时26分钟
白城	1468	兴城	K1056/K1053	青岛	176元	40小时19分钟
白城	1468	兴城	K958/K955	青岛	176元	41小时34分钟
白城	2262	葫芦岛	K1056/K1053	青岛	176元	46小时55分钟
白城	2262	葫芦岛	K704/K701	青岛	176元	47小时4分钟
白城	1468	锦州	K1056/K1053	青岛	176元	40小时19分钟
白城	1468	锦州	K704/K701	青岛	176元	40小时28分钟
白城	1468	锦州	K958/K955	青岛	176元	41小时34分钟
白城	1468	锦州	K972/K969	青岛	176元	42小时50分钟
白城	2262	锦州	K1056/K1053	青岛	176元	46小时55分钟
白城	2262	锦州	K704/K701	青岛	176元	47小时4分钟
白城	2262	锦州	K958/K955	青岛	176元	48小时10分钟
白城	2262	锦州	K972/K969	青岛	176元	49小时26分钟