# 需求分析

## 1.1程序功能：

### 1.1.1 读取文件

本程序可以读取配置文件信息，从文件中读取地铁站点网络信息和景点信息。配置文件内容可以按自己需要设计，但需要合理。

### 1.1.2 信息输入

本程序可以通过软件ui界面输入起点，终点，景点等信息。

### 1.1.3 用户互动

本程序可以通过按钮，下拉框等组件与用户互动，并在点击按钮等操作后给用户提供反馈。

### 1.1.4 绘图功能

本程序可以在ui界面将地铁线路图加载并显示出来。每条线路是参考本经地铁网站官方使用颜色用不同颜色显示。地铁站店分布按照现实中站点经纬度转换而来。加载完成的地图可以用滚轮实现放大缩小功能。

### 1.1.5 生成带权无向图功能

本程序可以根据读取到的文件信息生成无向图。此功能是为了实现计算最短用时与最少换乘所必须实现的前置功能。

### 1.1.6 计算最短用时乘车线路功能

本程序可以根据用户输入的起点终点或者起点与想去的景点（可以是复数）计算最短用时的乘车方式，包括出行时间和票价，并用文字与图像的方式向用户展现乘车方式信息。

### 1.1.7 计算最少换乘乘车线路功能

本程序可以根据用户输入的起点终点或者起点与想去的景点（可以是复数）计算最少换乘的乘车方式，包括出行时间和票价，并用文字与图像的方式向用户展现乘车方式信息。最小换乘指在保证换乘站最少的前提下用时最短的乘车方案，而且若最短用时和最少换乘计算出的线路图相同时，本程序计算出的乘车时间与费用也完全相同。

### 1.1.8 通过景点名购票拓展功能

本程序可以根据用户输入的起点与想去的景点，同样根据最短用时最少换乘两种逻辑生成乘车线路。本拓展功能分为单个景点和复数景点。

在复数景点的前提下，用户可输入多个景点，每个景点间用一个空格隔开。本程序会根据景点先后顺序生成乘车线路。

### 1.1.9 显示乘车线路图

本程序可以根据计算出的乘车线路图在地铁路线图上画出黑色加粗的路线提示。

### 1.1.10 显示乘车线路文字信息

本程序可以根据计算出的乘车线路图在ui界面的信息显示栏中将乘车站点以文字形式按顺序打印出来；在通过景点名购票拓展功能下，若输入多个景点，在每个景点的下车站会有提示本站下车是什么景点。一同显示的还有换乘站，乘车时间和票价花费。

### 1.1.11 信息查询功能

本程序支持信息查询：1，每条地铁线的所有站点；2，所有换乘站以及它们能去到的站点；3，每个景点的下车站。

### 1.1.12 计算乘车时间与乘车费用

本程序支持根据乘车路线计算出乘车时间与费用。而且在通过景点名购票拓展功能下，输入多个景点时会给予一定优惠。具体逻辑：每多一个景点票价增加0.1优惠，最多给予50%优惠。

## 1.2 需要处理的数据

### 1.2.1 文件信息

本程序总共需要读取八个文件：1，全线路以及每条线路包含的站点；2，全站点相连关系和距离，用于计算最短距离；3，全站点相连关系和距离，用于计算最少换乘；与2的区别在于2的换乘站只出现一次，所有可到达的站都写在一起；而3中换乘站会多次出现在每条属于的线路，并且每条线路的换乘站名字不同，比如十里河和十里河1；并且每条线路上的换乘站只和本条线路中与自己相连的站点相连；不同线路之间的同一个换乘站之间相连，且权值很大。4，所有换乘站相连关系和距离；5，全线路以及每条线路包含的站点，不包含换乘站；6，全站点经纬度信息；7，线路颜色信息文件；8，景点信息文件。

### 1.2.2 用户输入信息

本程序需要处理用户输入信息，包括起始站和终点站以及景点。

### 1.2.3 下拉框信息

本程序需要处理下拉框选中的信息。

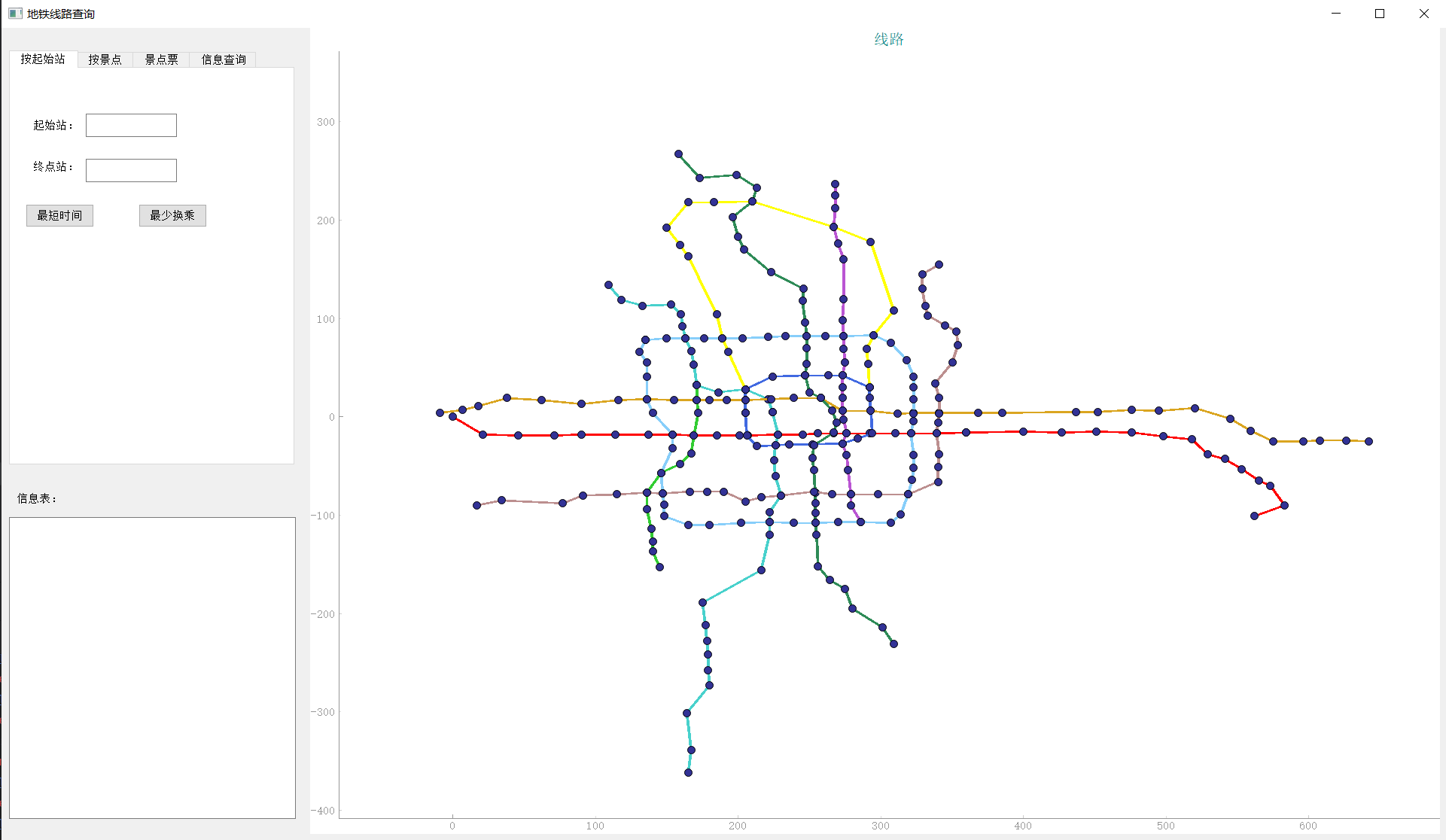
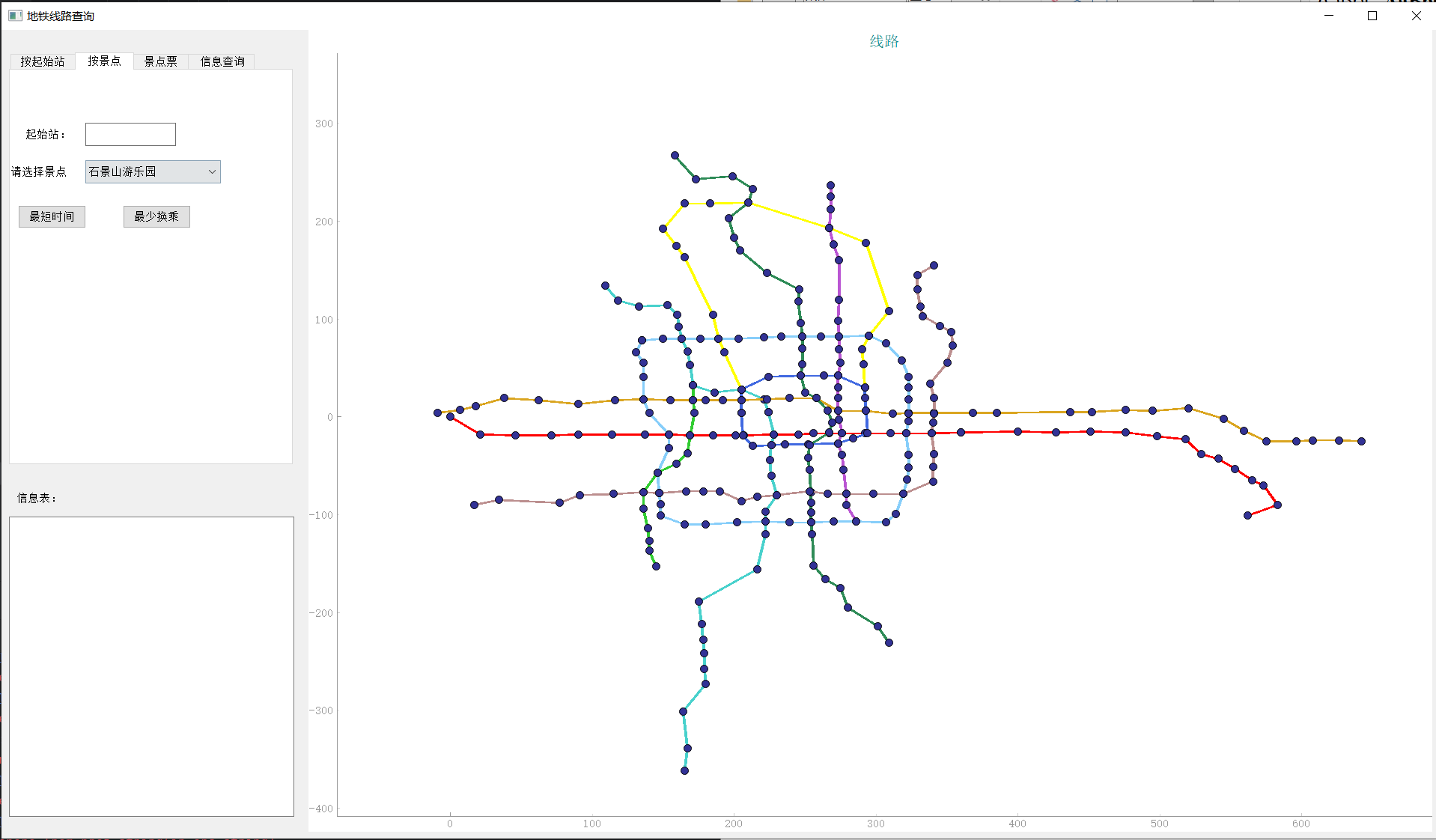
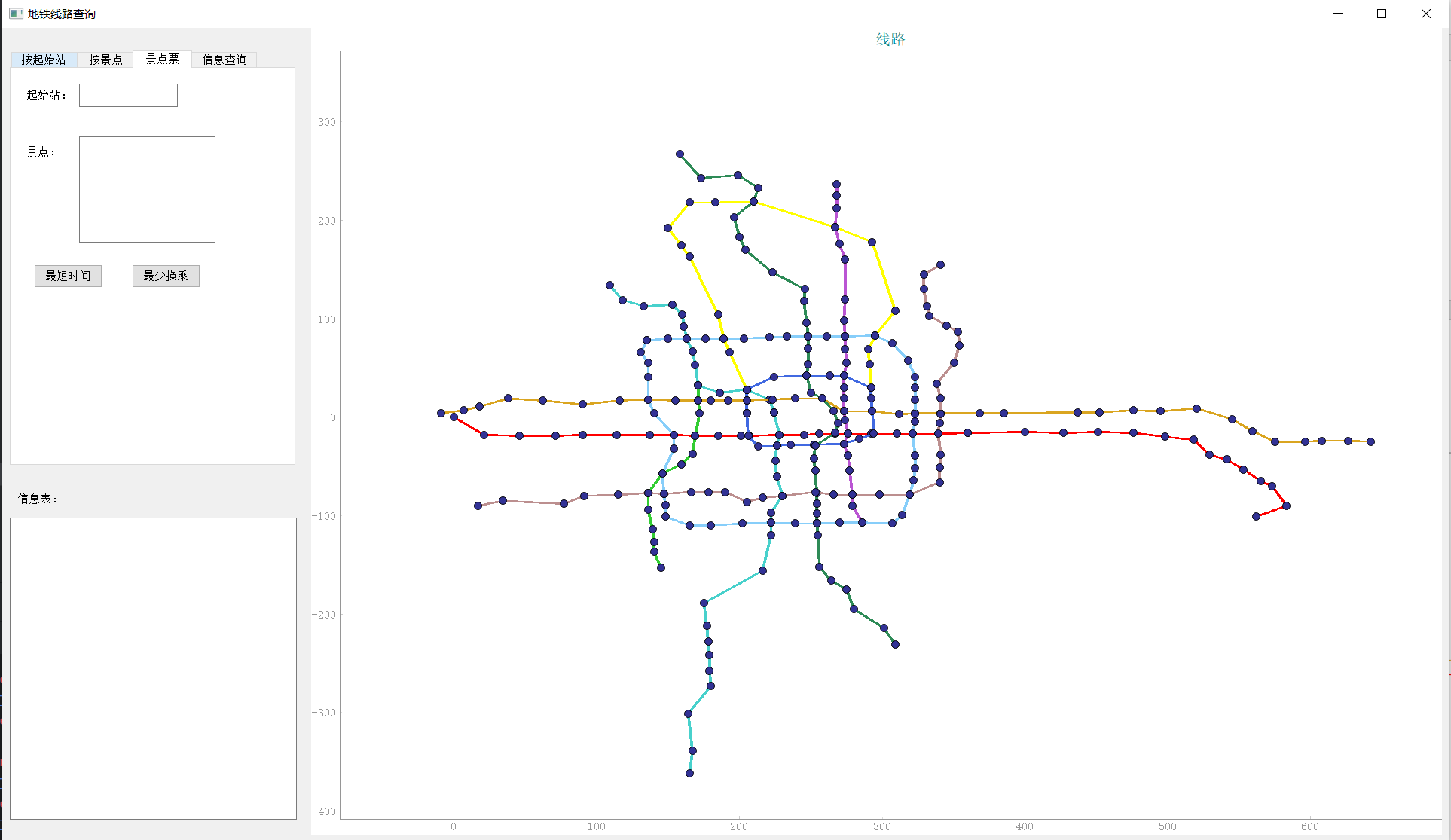
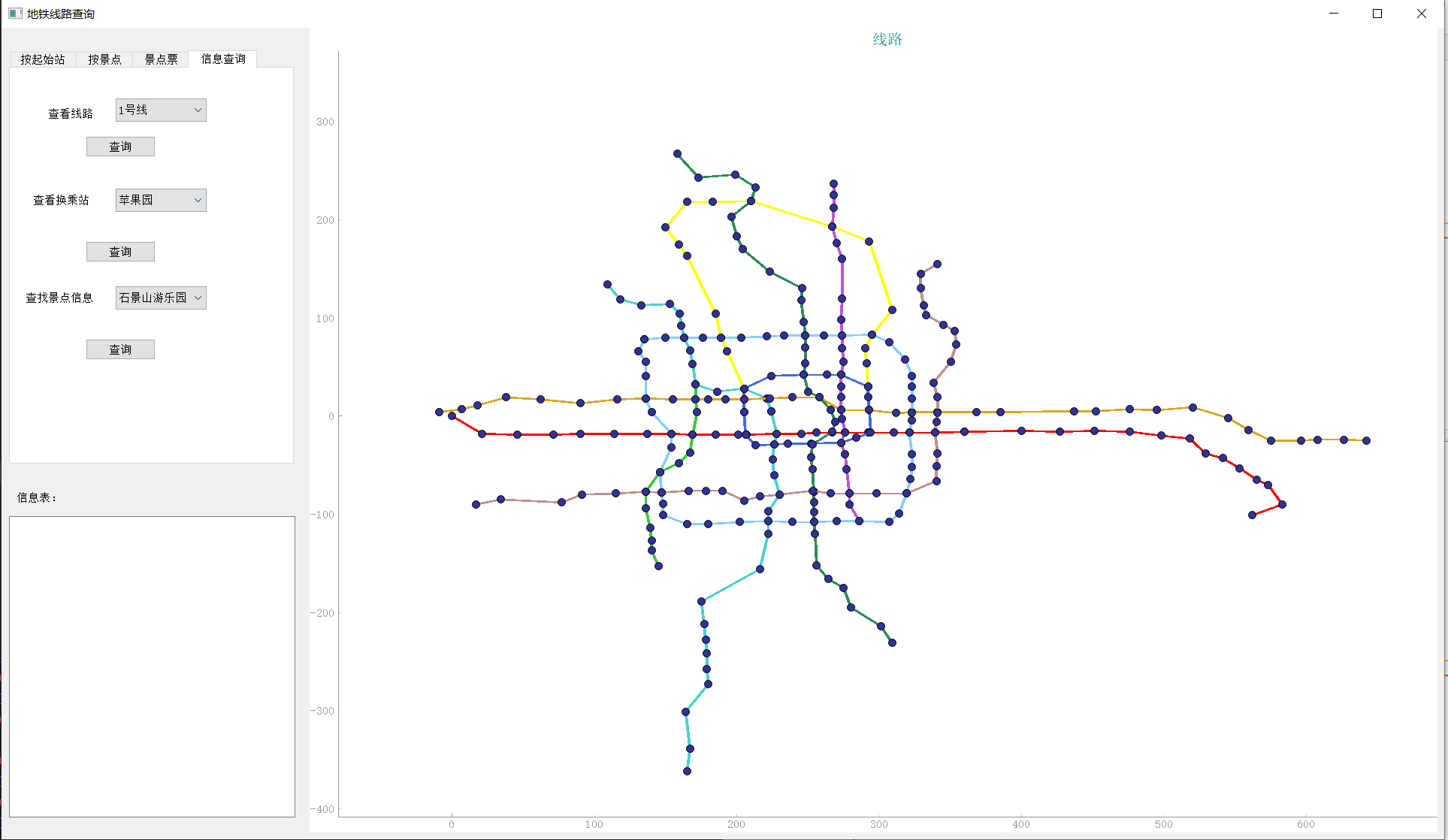
### 1.2.4 按钮

本程序要处理按钮被按下时传出的信号。

## 1.3 程序开发环境

本程序使用python语言开发，ui界面实现使用pyqt5和qt designer开发。

## 1.4 用户界面设计

如图所示，本程序由地铁线路图，信息输入区域和信息表三块部分组成。乘车线路会以黑色加粗显示在线路图上；信息输入区域可切换四个界面，分别是：1，通常查询，输入起点站和终点站并选择最少时间或最小换乘；2，按单个景点查询，输入起始站，并在下拉框里选择想去景点，最后选择最少时间或最小换乘；3，按多个景点查询，输入起始站并输入复数个景点，最后选择最少时间或最小换乘；4，信息查询页面，三个下拉框可分别选择地铁线路、换乘站、景点；点击对应查询按钮，分别显示线路所有站点、换乘站连接的所有站点、离景点最近的车站。

# 数据结构设计

本程序由两个.py文件和5个json文件，3个txt文件组成。其中test.py实现图类，计算最小换乘和最短时间，计算用时和费用，统计换乘站，按景点名查询等功能；ui.py负责实现ui界面，加载文件信息，添加按钮点击事件，绘图，调用test.py算法等功能。

## 2.1 class graph()

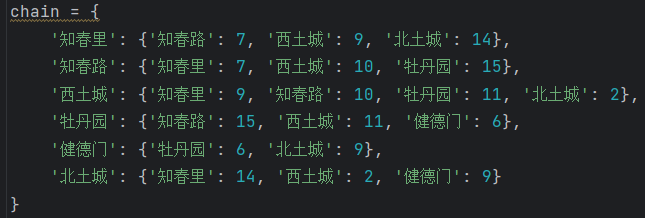
图类，用于根据传入的字典生成带权无向图。

### 2.1.1所属：

test.py

### 2.1.2参数：

chain



形如上图的字典类，其中key值是站点，value值又是一个字典，其中key为本站直接连接的站点，value为到该站点的距离权值。

### 2.1.3成员：

self.chain 用于存储传进来的chain字典

self.vertexes 存储chain的所有键值的列表

self.adjacency\_matrix 邻接矩阵

### 2.1.4成员函数：

chainTomatrix(self)

参数：无

功能：根据chain生成邻接矩阵

返回值：

邻接矩阵。

## 2.2 class Dijkstra

本程序核心算法部分。该部分实现迪杰斯特拉算法计算最短时间和最小换乘乘车线路，计算用时和费用。

### 2.2.1所属：

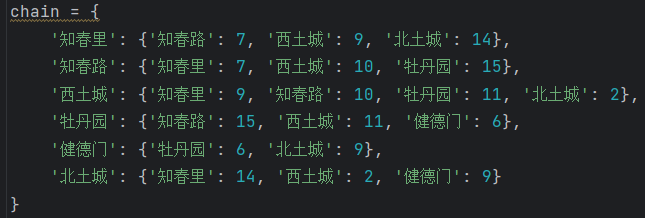
test.py

### 2.2.2参数：

Graph Graph类，根据地铁站点信息生成的图类

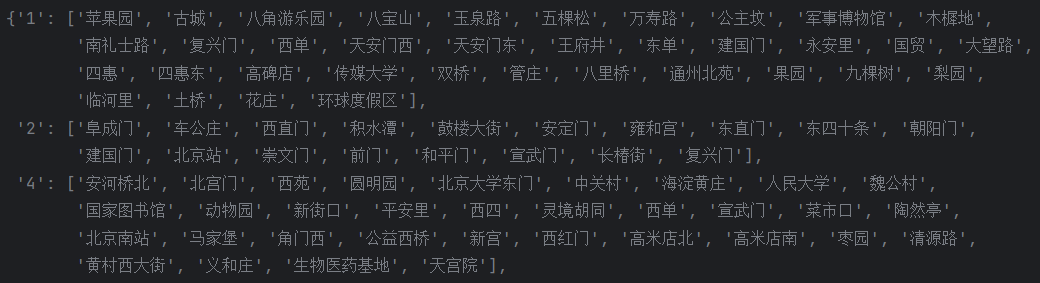
Start 字符串，起点站

Trans



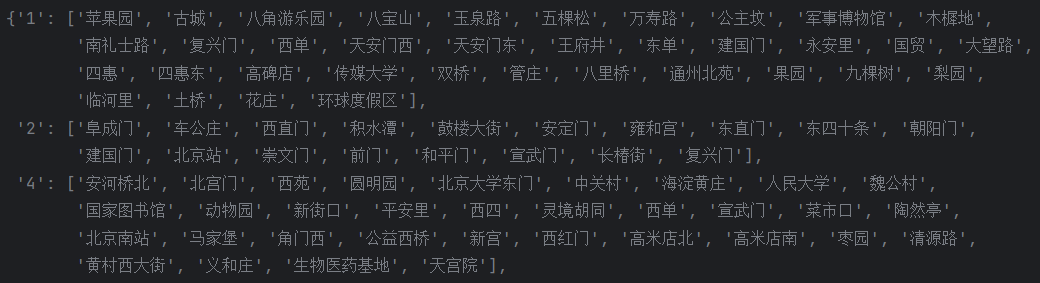
形如上图的字典类，其中key值是换乘站点，value值又是一个字典，其中key为本站直接连接的站点，value为到该站点的距离权值。

lines



型如上图的字典，其中key值是地铁线路，value是包含该线路所有站点的列表

lines\_1



型如上图的字典，其中key值是地铁线路，value是列表，存储了包含该线路除了换乘站以外的所有站点。

### 2.2.3成员：

self.start 字符串，用于存储参数start，起点站

self.graph graph类，用于存储参数graph，根据地铁站点信息生成的图类

self.trans 字符串，用于存储参数trans

self.lines 字典类，用于存储参数lines

self.lines\_1 字典类，用于存储参数lines\_1

self.set 列表类，存储已经搜索过的站点，用于迪杰斯特拉算法

self.unsearch 列表类，存储还未搜索过的站点，用于迪杰斯特拉算法

self.distance 字典类，存储各站点到其他所有站点的距离，用于迪杰斯特拉算法。

self.station 字符串，用于存储当前检索的站点

self.path 字典类，key值为每个站点，value值是列表，内容是以self.start为起点，key值为终点的前提下所计算出的最短时间乘车线路。

self.tr 列表类，用来存储转化后的最小换乘乘车线路。

self.end 字符串，终点站

### 2.2.4成员函数：

initset(self)

参数：

无

功能：

初始化成员self.set。初始时已经搜索过的站点列表中只存储起点，即self.start。

返回值：

一个只有self.start的列表。

initall(self)

参数：

无

功能：

初始化成员self.set，self.unsearch，self.distance ，self.station ，self.path，self.tr。用于查询时输入新起点计算新乘车线路时清空上条线路的存储。

返回值：

无

initunsearch(self)

参数：

无

功能：

初始化成员self.unsearch。

返回值：

一个列表，其中包含了除了起点站以外的所有站点。

getdistance(self)

参数：

无

功能：

初始化self.distance，根据graph类的无向图初始化起点站到其他所有站点的距离。

返回值：

字典类，key值是所有站点，value值是起点站到其他所有站点的距离。若起点站不能直接到达该站点，则该站点的value中存np.inf，即无限长。

initstation(self)

参数：

无

功能：

初始化参数self.station，最初检索点是初始站

返回值：

self.start

initpath(self)

参数：

无

功能：

初始化路径

返回值：

字典类，key值为所有站点，value值为列表，存储以key值为终点站，start为起点站的路径。若不联通，value值为空列表；初始化时，value值不为空列表的只有与start直接连通的站点。此时，该站value的列表append上start。

updatestation(self, index)

参数：

Index 计数用的数字

功能：

更新当前检索点。Index递增，根据distance字典中存放的各站点到起点站的距离从小到大排序，每次调用该函数就检索列表中排序第index个；若该位置站点不在self.set列表中，则将其设为当前检索点。

返回值：

无

updatepath(self)

参数：

无

功能：

更新self.path列表。设一个dis\_path临时字典。该字典结构和self.distance相同，但区别是其中存储的起点站到其他每个站点的距离是由起点站到self.station（当前检索点）的距离加上当前检索点到每个站点的距离计算得到的。若某站点dis\_path中存储的距离比self.distance中存储的距离短，则用短的距离替换self.distance中的距离。

返回值：

无

find\_shortest\_path(self, end = None)

参数：

End 字符串，终点站。

功能：

根据起点终点寻找最短路径。调用self.updatestation函数和self.updatepath函数。

返回值：

若找到最短路径则返回存储最短路径的列表，若遍历完也没找到则返回self.path

trans\_station

参数：

Path 列表类，要寻找换乘站的乘车线路。

功能：

寻找Path中的换乘站

返回值：

存储有Path线路中所有换乘站的列表。

totrans(self, path)

参数：

Path 列表类，未处理前的最小换乘乘车线路。

功能：

处理最小换乘乘车线路

未处理前的最小换乘乘车线路如下：

['和平里北街', '雍和宫1', '北新桥', '张自忠路', '东四1', '灯市口', '东单1', '崇文门1', '磁器口', '天坛东门', '蒲黄榆', '蒲黄榆1', '方庄', '十里河', '北工大西门']

本函数可以将换乘站后面的数字去掉，并统计乘车线路中的换乘站。

处理后的最小换乘乘车线路如下：

['和平里北街', '雍和宫', '北新桥', '张自忠路', '东四', '灯市口', '东单', '崇文门', '磁器口', '天坛东门', '蒲黄榆', '方庄', '十里河', '北工大西门']

返回值：

两个列表。第一个列表是处理后的最小换乘乘车线路，第二个列表是该线路中的所有换乘站。

getPrice(self, path)

参数：

Path 列表类，乘车路径。

功能：

根据乘车路径计算距离和费用

返回值：

Int型，乘车费用。

getTime(self, path)

参数：

Path 列表类，乘车路径。

功能：

根据乘车路径计算用时

返回值：

Int型，用时

## 2.3 class MyWindow

Ui类，用于生成用户友好的可视化操作界面。

### 2.3.1所属：

ui.py

### 2.3.2参数：

无

### 2.3.3成员：

self.ui ui用户界面

self.start 一般查询用起点文本输入框

self.end 一般查询用终点文本输入框

self.start1 按单个景点查询用起点文本输入框

self.start2 按多个景点查询用起点文本输入框

self.ends 按多个景点查询用终点文本输入框

self.findshortestpath 一般查询用最短用时按钮

self.findleasttrans 一般查询用最小换乘按钮

self.findshortestpath1 按单个景点查询用最短用时按钮

self.findleasttrans1 按单个景点查询用最小换乘按钮

self.findshortestpath2 按多个景点查询用最短用时按钮

self.findleasttrans2 按多个景点查询用最小换乘按钮

self.textBrowser\_2 信息显示表

self.vl 地铁线路图显示区

self.getatt 按单个景点查询用终点下拉选择框

self.linemsg 查询信息用地铁线路下拉选择框

self.transmsg 查询信息用换乘站下拉选择框

self.attrmsg 查询信息用景点下拉选择框

self.searchline 查询信息用地铁线路查询按钮

self.searchtrans 查询信息用换乘站查询按钮

self.searchattr 查询信息用景点查询按钮

self.pw pyqtgraph.PlotWidget()类，用于绘图

self.dataA 字典，存储全线路的所有站点，包含换乘站

self.info 字典，存储全站点距离关系文件，最短用时算法用

self.info\_t 字典，存储全站点距离关系文件，最小换乘算法用

self.info\_trans 字典，存储全换乘站距离关系文件

self.dataB 字典，存储全线路的所有站点，不包含换乘站

self.position 字典，存储全站点经纬度信息

self.color 字典，存储全线路颜色文件

self.attractions 字典，存储景点信息文件

self.short 列表，最短用时线路

self.s\_time 最短用时时间

self.s\_price 最短用时票价

self.trans\_stas\_1 最短用时换乘站

self. trans 列表，最小换乘线路

self. t\_time 最小换乘时间

self. t\_price 最小换乘票价

self.trans\_stas\_2 最小换乘换乘站

### 2.3.4成员函数：

init\_data(self)

参数：

无

功能：

读出各文件数据并存储

返回值：

无

dij(self, start, end)

参数：

Start 字符串，起点

End 字符串，终点

功能：

调用test.py中的算法，计算最短线路和最小换乘线路并存储信息

返回值：

无

getshort(self)

参数：

无

功能：

一般查询用查找并绘图输出最短用时线路图，并在self.textBrowser\_2 信息显示表上显示线路信息，换乘站信息，用时和费用。

返回值：

无

gettrans(self)

参数：

无

功能：

一般查询用查找并绘图输出最小换乘线路图，并在self.textBrowser\_2 信息显示表上显示线路信息，换乘站信息，用时和费用。

返回值：

无

wgs48\_to\_xy(self, lon, lat)

参数：

Lon 列表，存储所有站点经度

Lat 列表，存储所有站点纬度

功能：

把站点经纬度转换为xy坐标

返回值：

转换后的x，y坐标列表

initdraw(self)

参数：

无

功能：

根据各站点的x，y坐标绘制地铁线路图

返回值：

无

draw(self, path)

参数：

Path 列表，要绘制提示的乘车线路

功能：

根据乘车线路站点的x，y坐标绘制提示线路

返回值：

无

Getshort1(self)

参数：

无

功能：

按单个景点查询用查找并绘图输出最短用时线路图，并在self.textBrowser\_2信息显示表上显示线路信息，换乘站信息，用时和费用。

返回值：

无

Gettrans1(self)

参数：

无

功能：

按单个景点查询用查找并绘图输出最小换乘线路图，并在self.textBrowser\_2 信息显示表上显示线路信息，换乘站信息，用时和费用。

返回值：

无

Getshort2(self)

参数：

无

功能：

按多个景点查询用查找并绘图输出最短用时线路图，并在self.textBrowser\_2信息显示表上显示线路信息，换乘站信息，用时和费用。

返回值：

无

Gettrans2(self)

参数：

无

功能：

按多个景点查询用查找并绘图输出最小换乘线路图，并在self.textBrowser\_2 信息显示表上显示线路信息，换乘站信息，用时和费用。

返回值：

无

splite(self)

参数：

无

功能：

按多个景点查询用。将输入的字符串转换为列表

返回值：

存有所有目标景点的列表

loadline(self)

参数：

无

功能：

查询该线路的所有站点

返回值：

列表，存储要查询的线路的所有站点

loadtrans(self)

参数：

无

功能：

查询该换乘站的所有可到达站点

返回值：

列表，存储要查询的换乘站的所有可到达站点

getlinemsg(self)

参数：

无

功能：

将要查询的线路的所有站点的信息输出到信息显示栏

返回值：

无

gettransmsg(self)

参数：

无

功能：

将要查询的换乘站的所有可到达站点的信息输出到信息显示栏

返回值：

无

getattrmsg(self)

参数：

无

功能：

将要查询的景点的下车站的信息输出到信息显示栏

返回值：

无

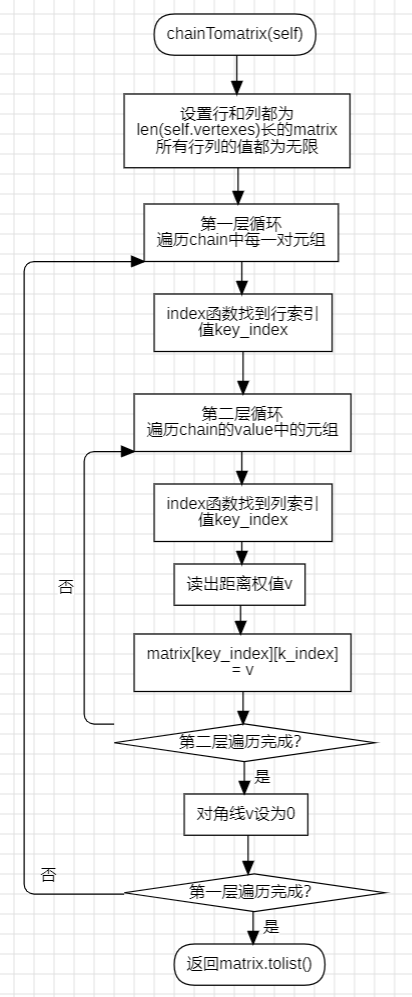
# 详细设计

## 3.1 chainTomatrix(self)

功能：

根据chain生成邻接矩阵

程序流程图

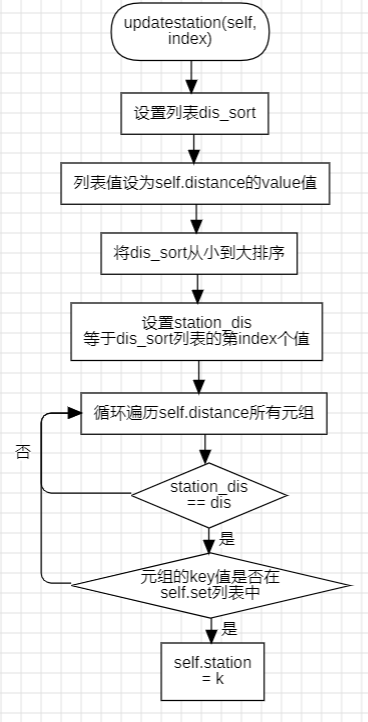


## 3.2 updatestation(self, index)

功能：

更新当前检索点。Index递增，根据distance字典中存放的各站点到起点站的距离从小到大排序，每次调用该函数就检索列表中排序第index个；若该位置站点不在self.set列表中，则将其设为当前检索点。

程序流程图

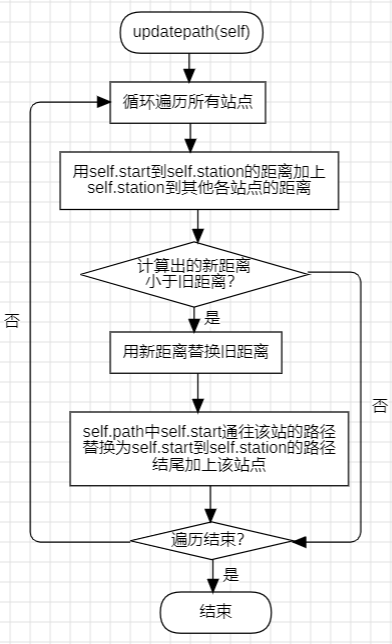


## 3.2 updatepath(self)

功能：

更新self.path列表。设一个dis\_path临时字典。该字典结构和self.distance相同，但区别是其中存储的起点站到其他每个站点的距离是由起点站到self.station（当前检索点）的距离加上当前检索点到每个站点的距离计算得到的。若某站点dis\_path中存储的距离比self.distance中存储的距离短，则用短的距离替换self.distance中的距离。

程序流程图

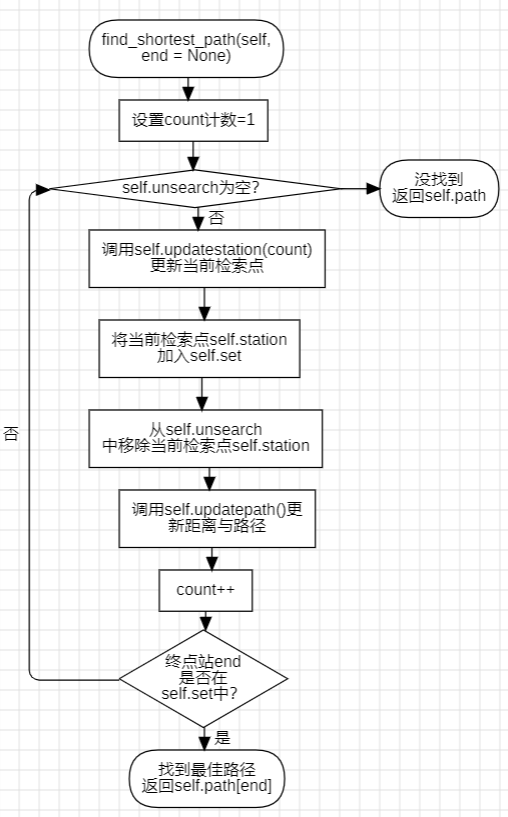


## 3.3 find\_shortest\_path(self, end = None)

功能：

根据起点终点寻找最短路径。调用self.updatestation函数和self.updatepath函数。

函数流程图

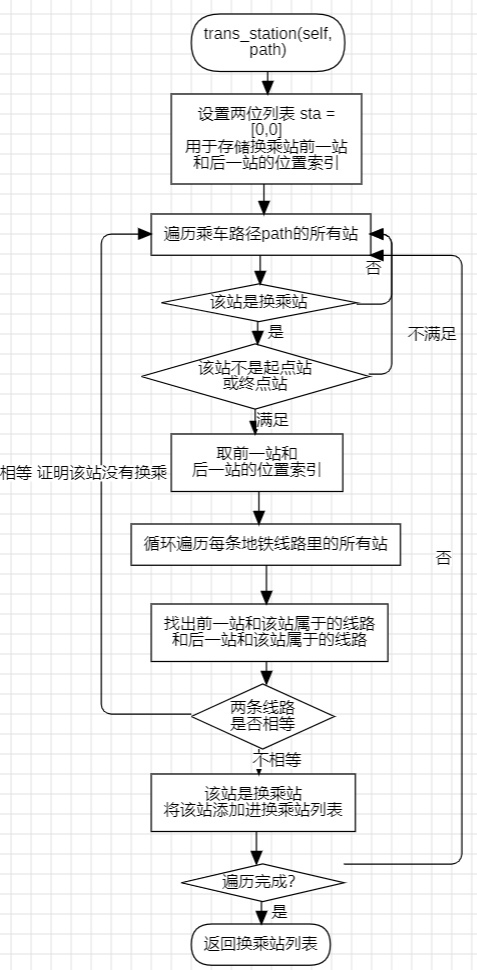


## 3.4 trans\_station(self, path)

功能：

寻找Path中的换乘站

函数流程图



## 3.5 totrans(self, path)

功能：

处理最小换乘乘车线

未处理前的最小换乘乘车线路如下：

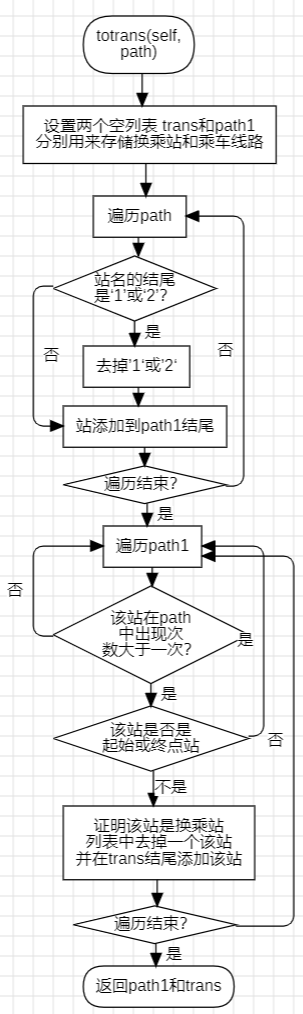
['和平里北街', '雍和宫1', '北新桥', '张自忠路', '东四1', '灯市口', '东单1', '崇文门1', '磁器口', '天坛东门', '蒲黄榆', '蒲黄榆1', '方庄', '十里河', '北工大西门']

本函数可以将换乘站后面的数字去掉，并统计乘车线路中的换乘站。

处理后的最小换乘乘车线路如下：

['和平里北街', '雍和宫', '北新桥', '张自忠路', '东四', '灯市口', '东单', '崇文门', '磁器口', '天坛东门', '蒲黄榆', '方庄', '十里河', '北工大西门']

程序流程图

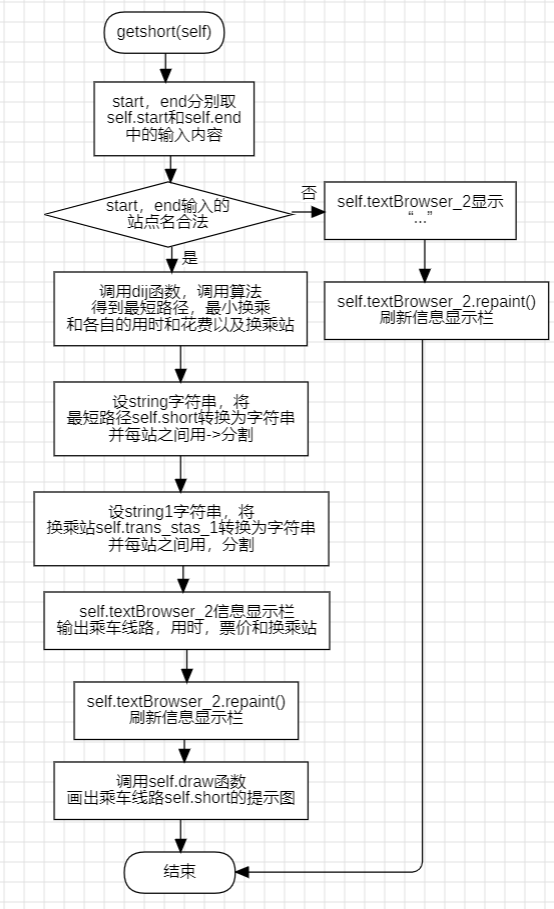


## 3.6 getshort(self)

功能：

一般查询用查找并绘图输出最短用时线路图，并在self.textBrowser\_2 信息显示表上显示线路信息，换乘站信息，用时和费用。

由于我写的所有按钮点击事件绑定的函数结构上相似，故这里只取getshort(self)作为例子

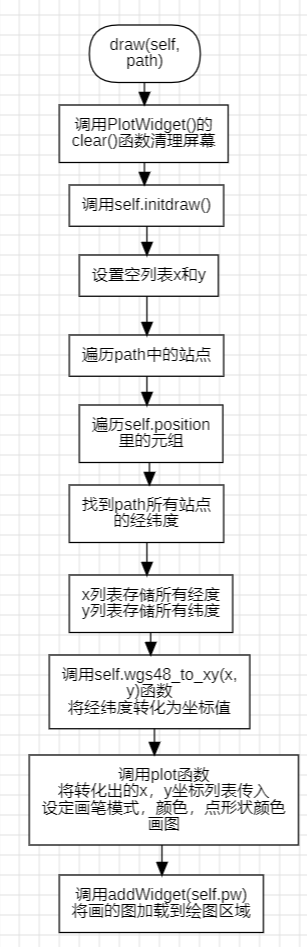


## 3.7 draw(self, path)

功能：

根据乘车线路站点的x，y坐标绘制提示线路

程序流程图

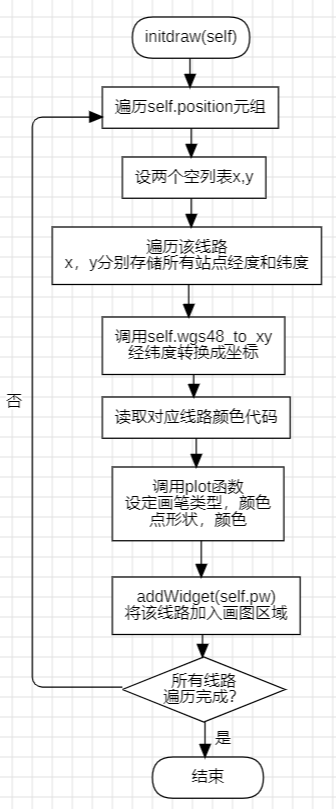


## 3.8 initdraw(self)

功能：

根据各线路站点的x，y坐标绘制提示线路，画出地铁全线路图

程序流程图

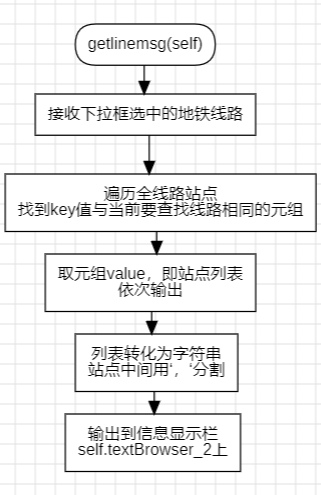


## 3.9 getlinemsg(self)

功能：

将要查询的线路的所有站点的信息输出到信息显示栏

程序流程图



## 3.10 使用文件

### 3.10.1 lines.json文件

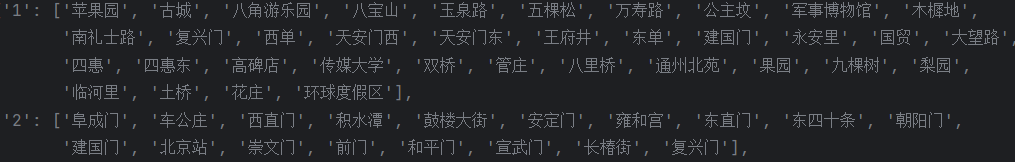
用json形式存储的字典文件。内容是地铁1，2，4，5，6，8，9，10，13，14号线路以及各线路的所有站点。

格式：

字典共有10个元组，每个元组的key值为线路号，value值为列表，列表里是对应线路的所有站点。

示例：

以一号线和二号线的元组为例：



### 3.10.2 lines\_1.json文件

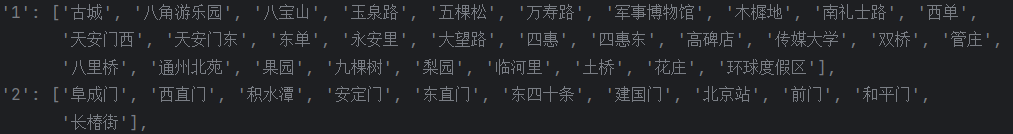
用json形式存储的字典文件。内容是地铁1，2，4，5，6，8，9，10，13，14号线路以及各线路除了换乘站以外的所有站点。

格式：

字典共有10个元组，每个元组的key值为线路号，value值为列表，列表里是对应线路的除了换乘站以外的所有站点。

示例：

以一号线和二号线的元组为例：



### 3.10.3 subway文件

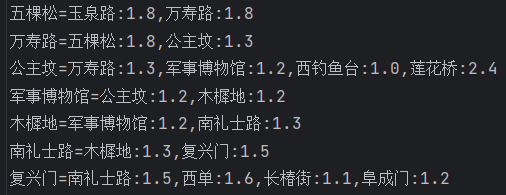
用txt形式存储的文件。内容是所有站点的联通与距离权值关系。用于计算最短时间。

格式：

站点名=可通行站点1：距离权值1，可通行站点2：距离权值2….

非换乘站的情况下，一个站点可以通往两个站；换乘站的情况下，一个站点可通往更多站。

示例：



### 3.10.4 subway1文件

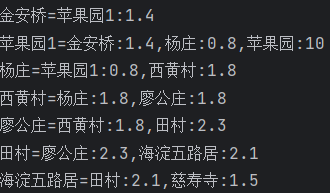
用txt形式存储的文件。内容是所有站点的联通与距离权值关系。用于计算最小换乘。

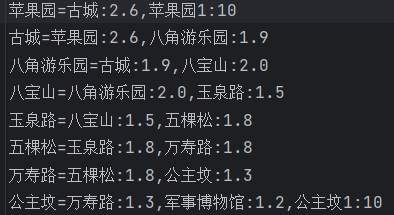
格式：

站点名=可通行站点1：距离权值1，可通行站点2：距离权值2

非换乘站的情况下，一个站点可以通往两个站；换乘站的情况下，一个站点还可以通往在其他线路上的自己，只不过权值很大。这样就可以在之改动文件不改动算法的情况下得到最小换乘。

示例：





如图所示，一苹果园站为例，苹果园站在六号线和一号线都有，所以我在一号线设一个苹果园，六号线设一个苹果园1；两站可各自通往各自线路上的前一站和后一站，两站间也互通，但如图可见权值极大。可见这样设计可以沿用最短时间的算法同时保证减少换乘次数。

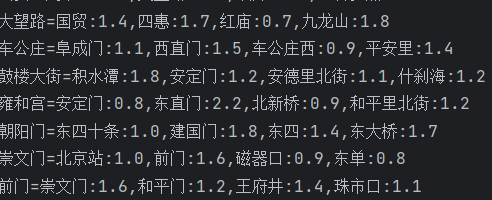
### 3.10.5 trans.txt文件

用txt形式存储的文件。内容是所有换乘站点的联通与距离权值关系。

格式：

站点名=可通行站点1：距离权值1，可通行站点2：距离权值2，可通行站点3：距离权值3….

示例：



### 3.10.6 position1.json文件

用json形式存储的字典文件。内容是地铁1，2，4，5，6，8，9，10，13，14号线路的所有站点的经纬度信息。

格式：

字典共有10个元组，每个元组的key值为线路号，value值为字典；value字典里的每个元组的key值是站点，value值是存储了该站点经纬度信息的列表。

示例：



### 3.10.7 color.json文件

用json形式存储的字典文件。内容是地铁1，2，4，5，6，8，9，10，13，14号线路的颜色代码。

格式：

字典共有10个元组，每个元组的key值为线路号，value值为该线路颜色代码的字符串。

示例：



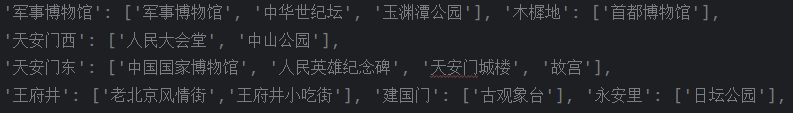
### 3.10.8 attractions.json文件

用json形式存储的字典文件。内容是地铁所有站点可到达的景点。

格式：

字典每个元组的key值是站点名，value值是列表，列表中存储从该站能到达的所有景点。

示例：



# 测试

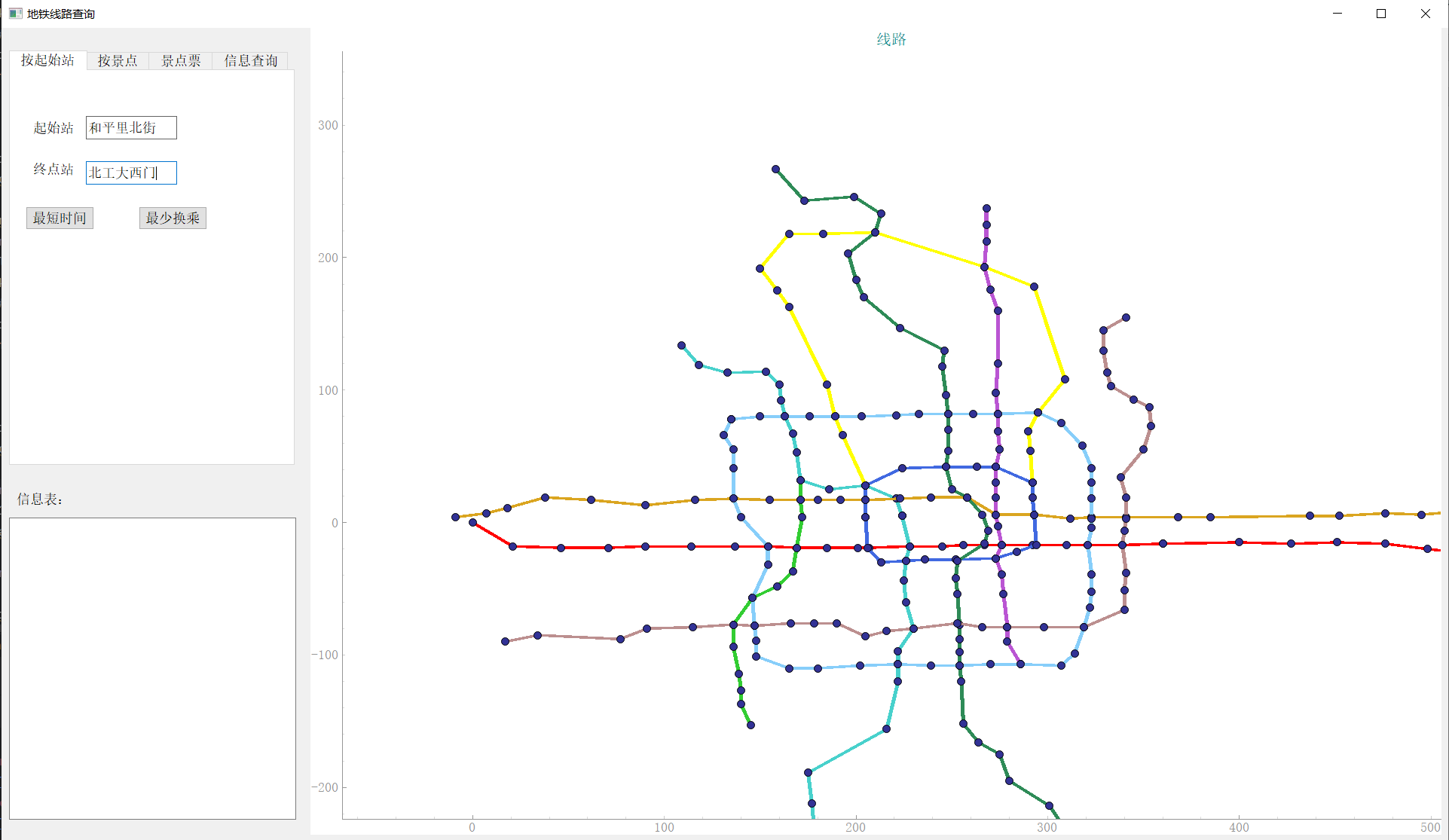
## 4.1 按起点终点查询

流程：

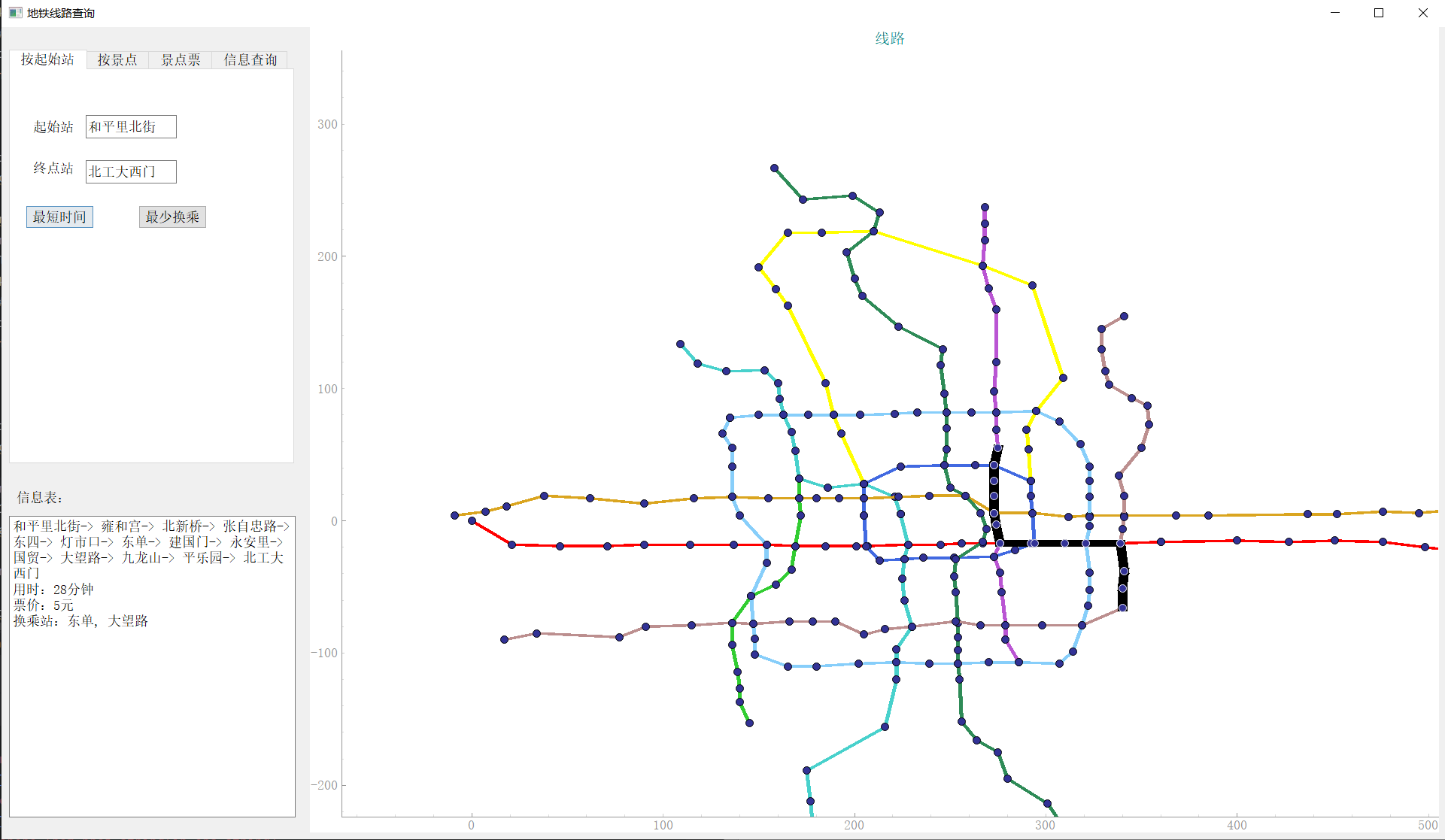
1. 输入起点站
2. 输入终点站
3. 选择乘车方案（最短时间，最小换乘）
4. 点击对应按钮，输出结果

### 4.1.1测试样例

样例1

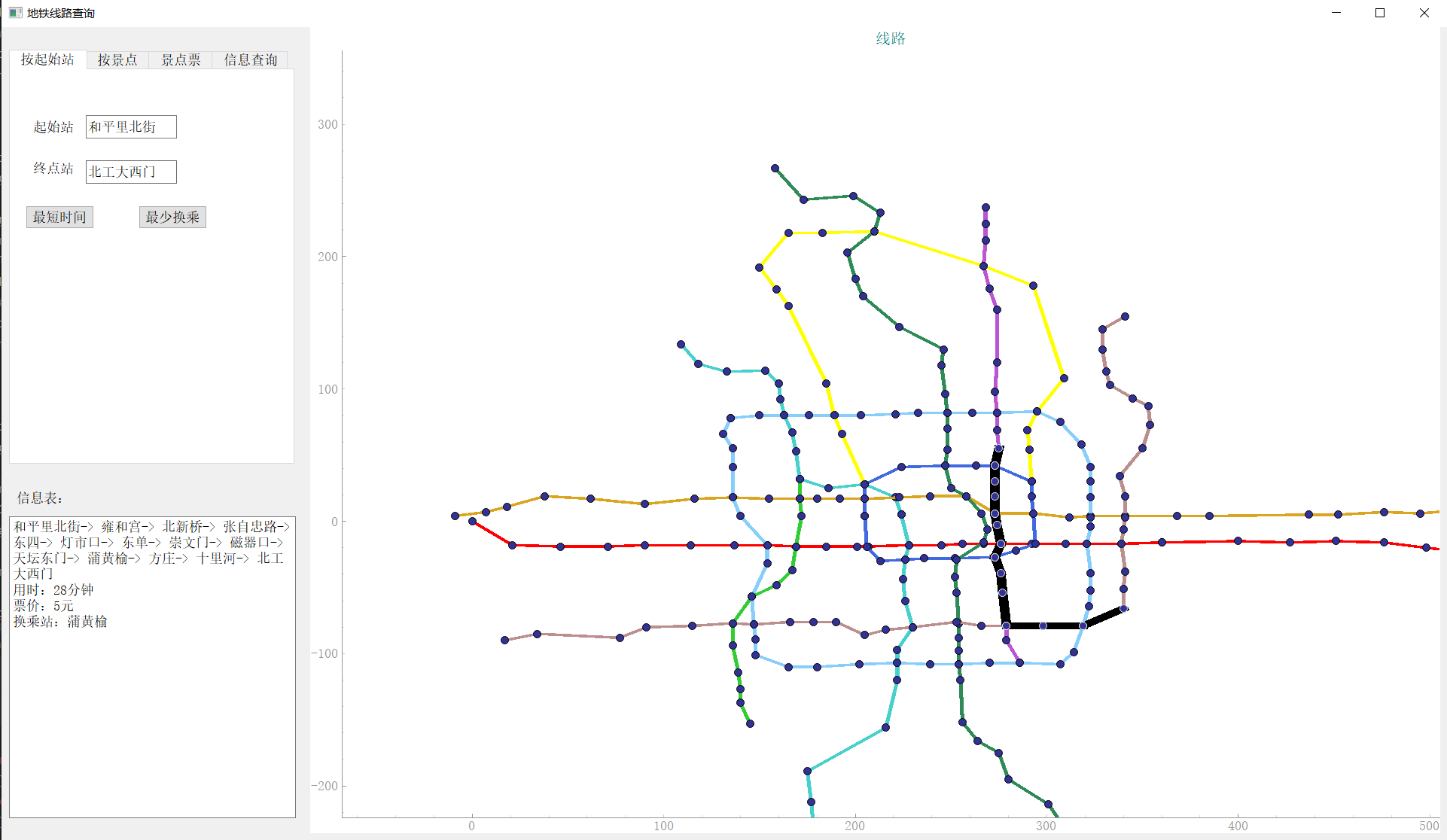


如上图，输入起点和平里北街，终点北工大西门



最短时间算法

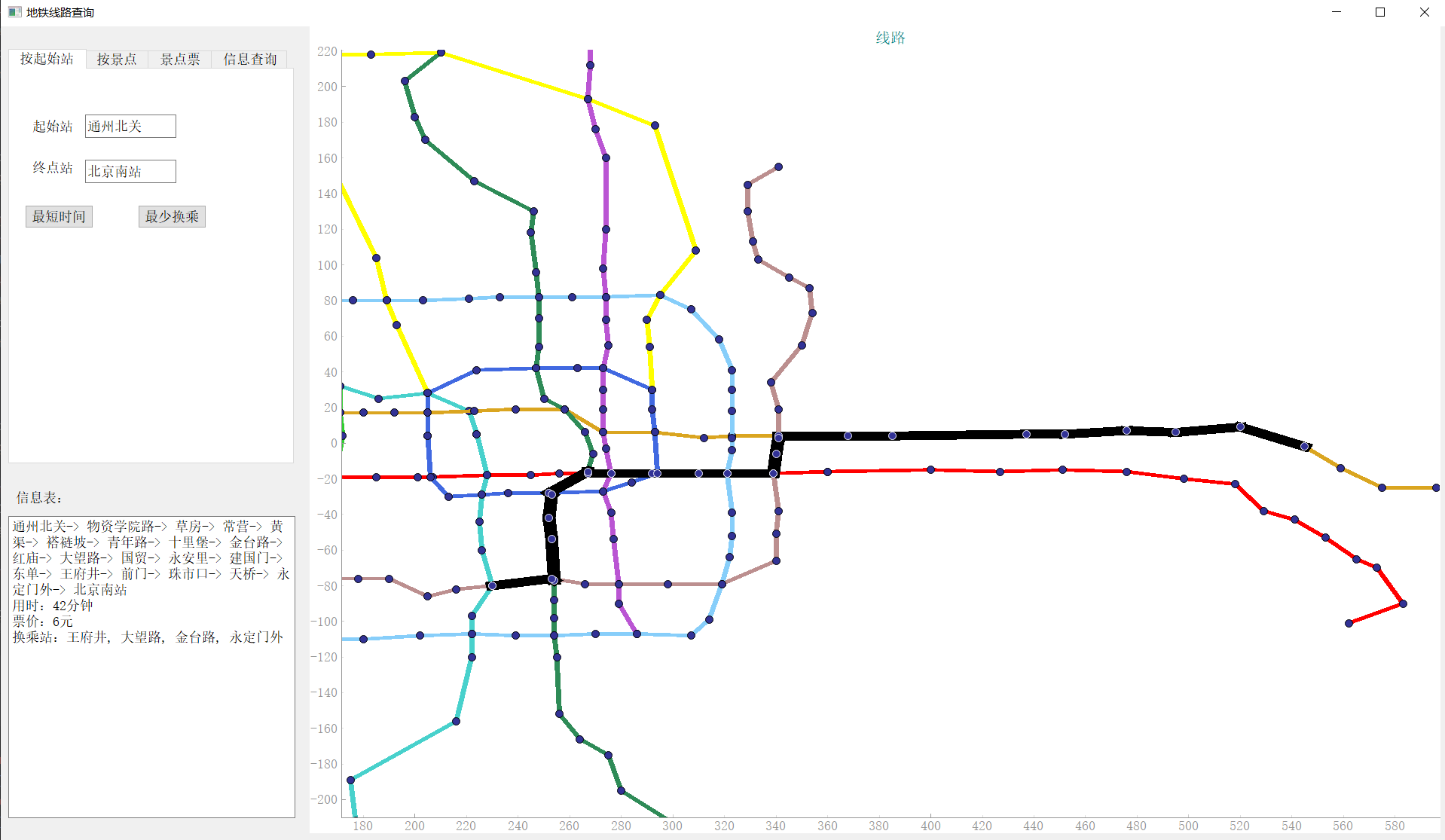
可见乘车线路在地图中被黑色加粗提示，乘车站顺序，用时，票价和换乘站也在信息显示栏标出。



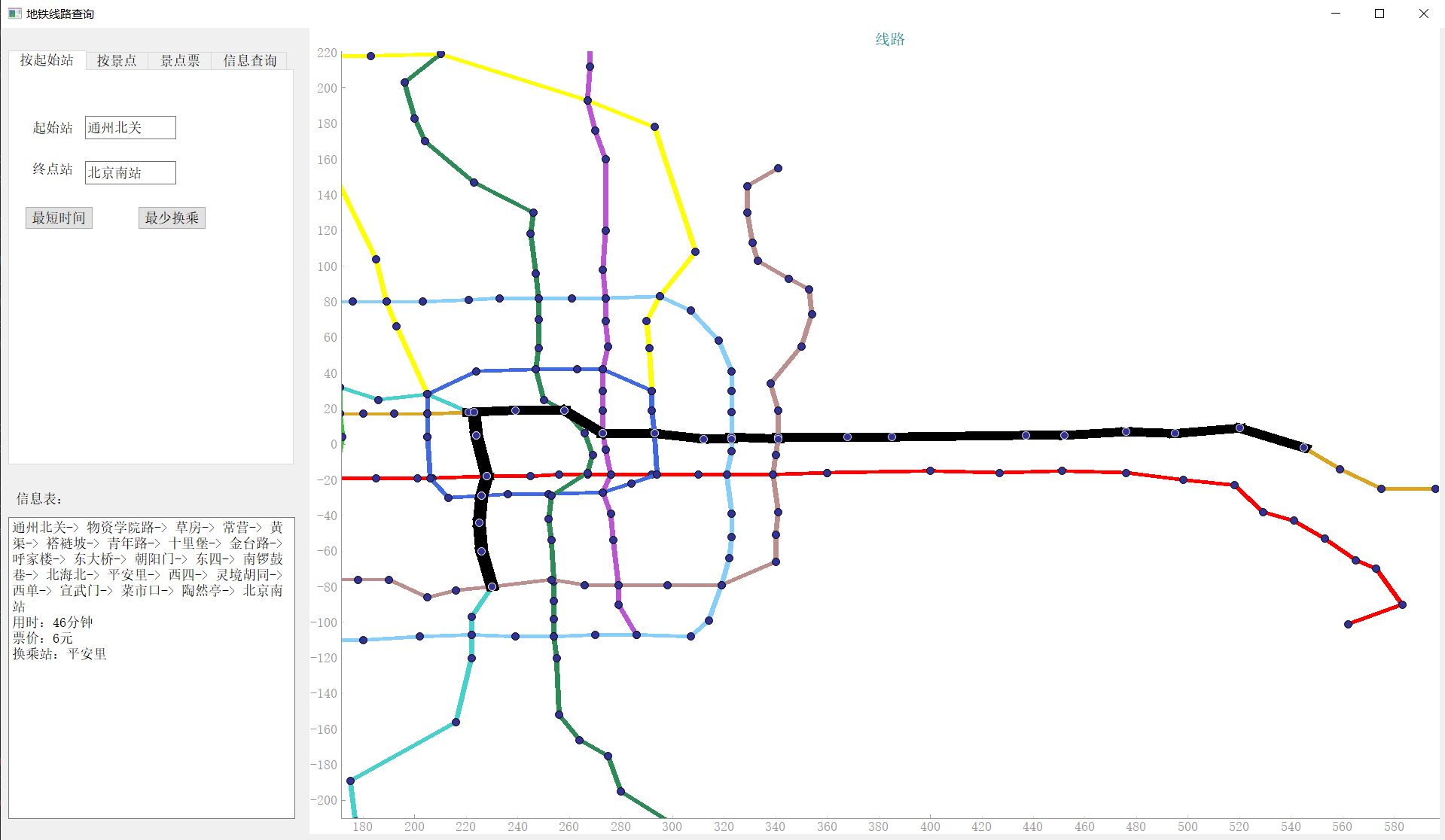
最小换乘算法

可见换乘站相比最短时间算法少了，只换乘了一次。

样例2



最短时间算法



最小换乘算法

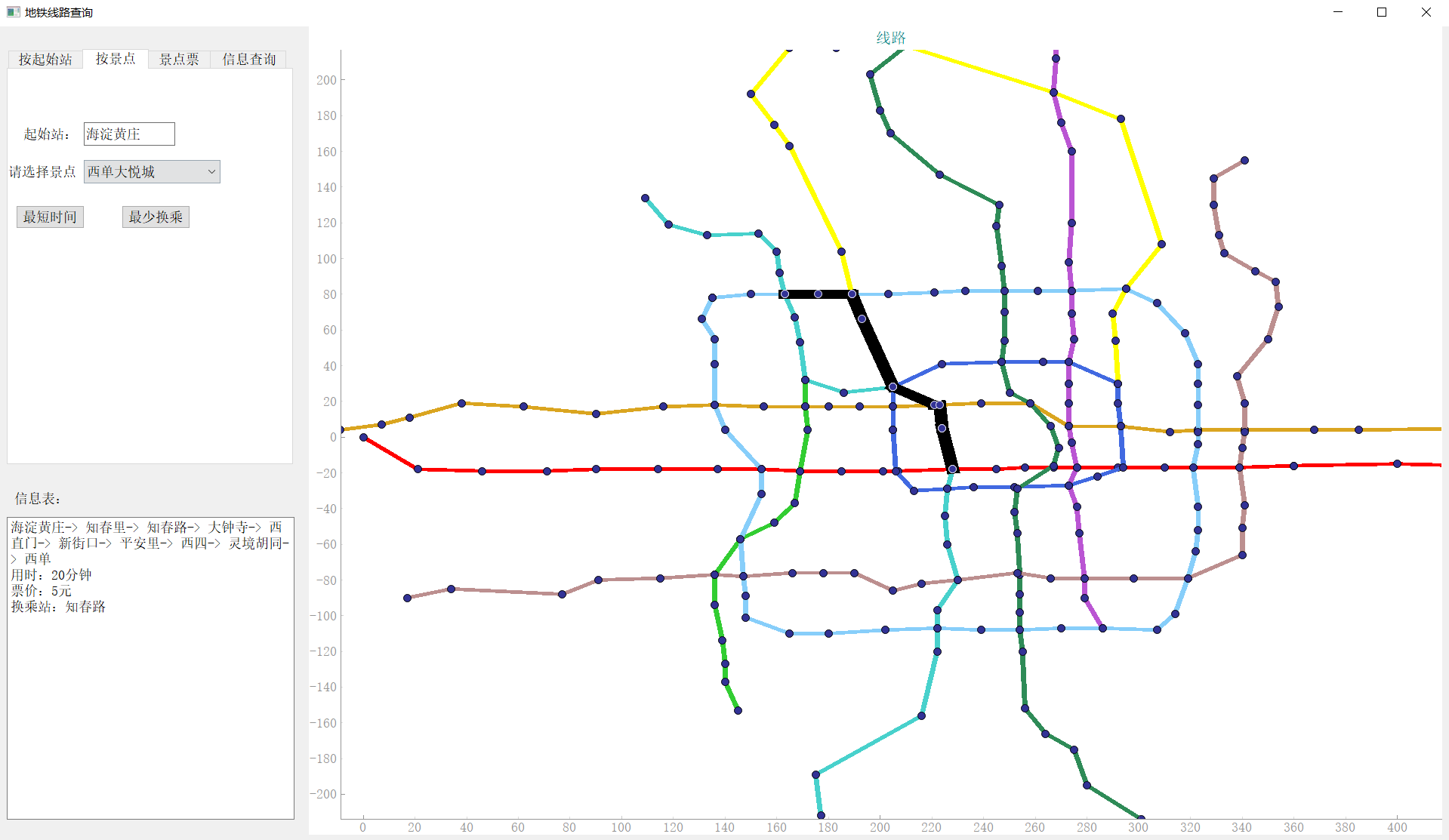
## 按单个景点查询

流程：

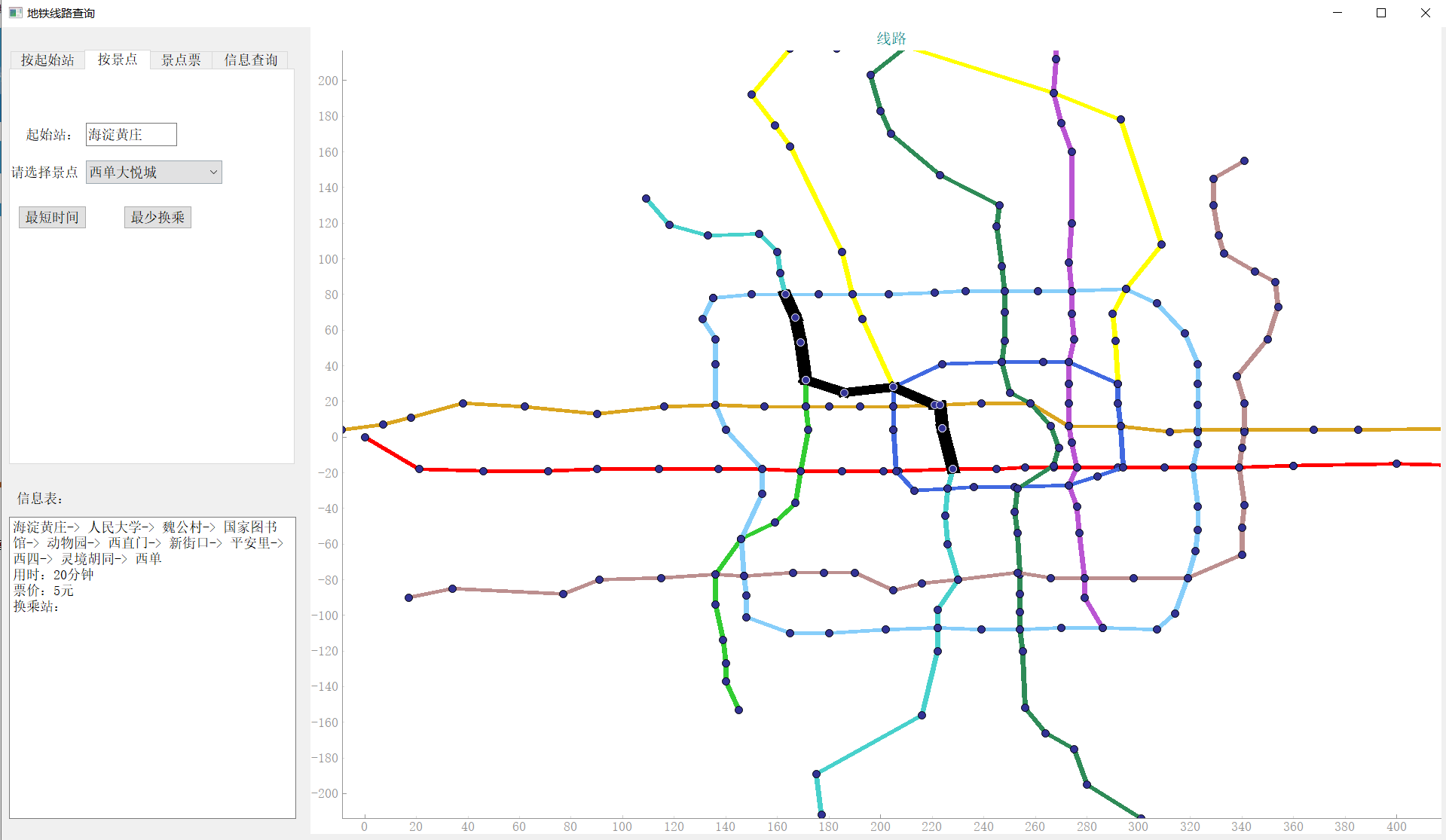
1. 输入起点站
2. 下拉框选择想去的景点
3. 选择乘车方案（最短时间，最小换乘）
4. 点击对应按钮，输出结果

### 4.2.1测试样例

样例1

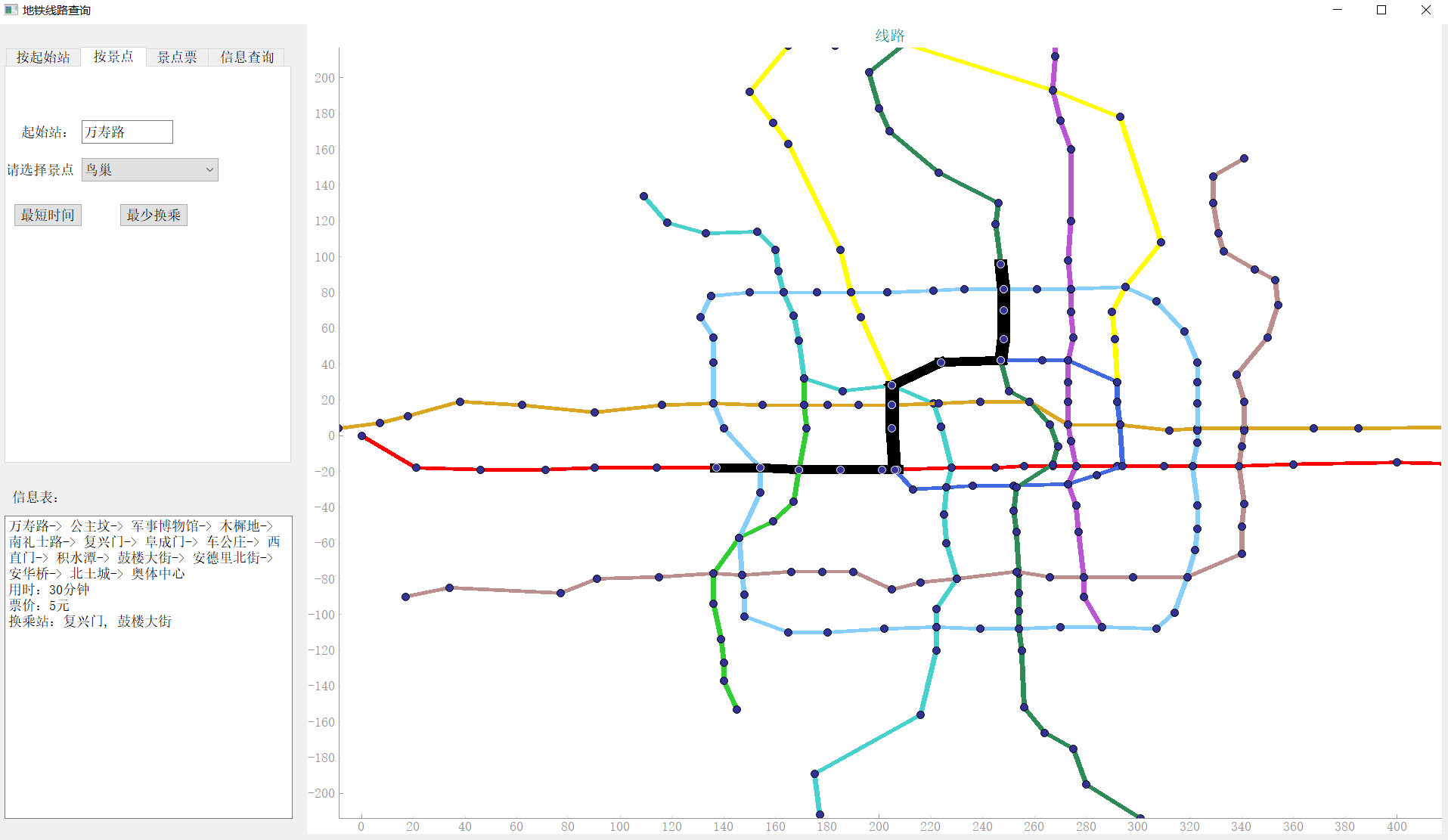


最短时间算法

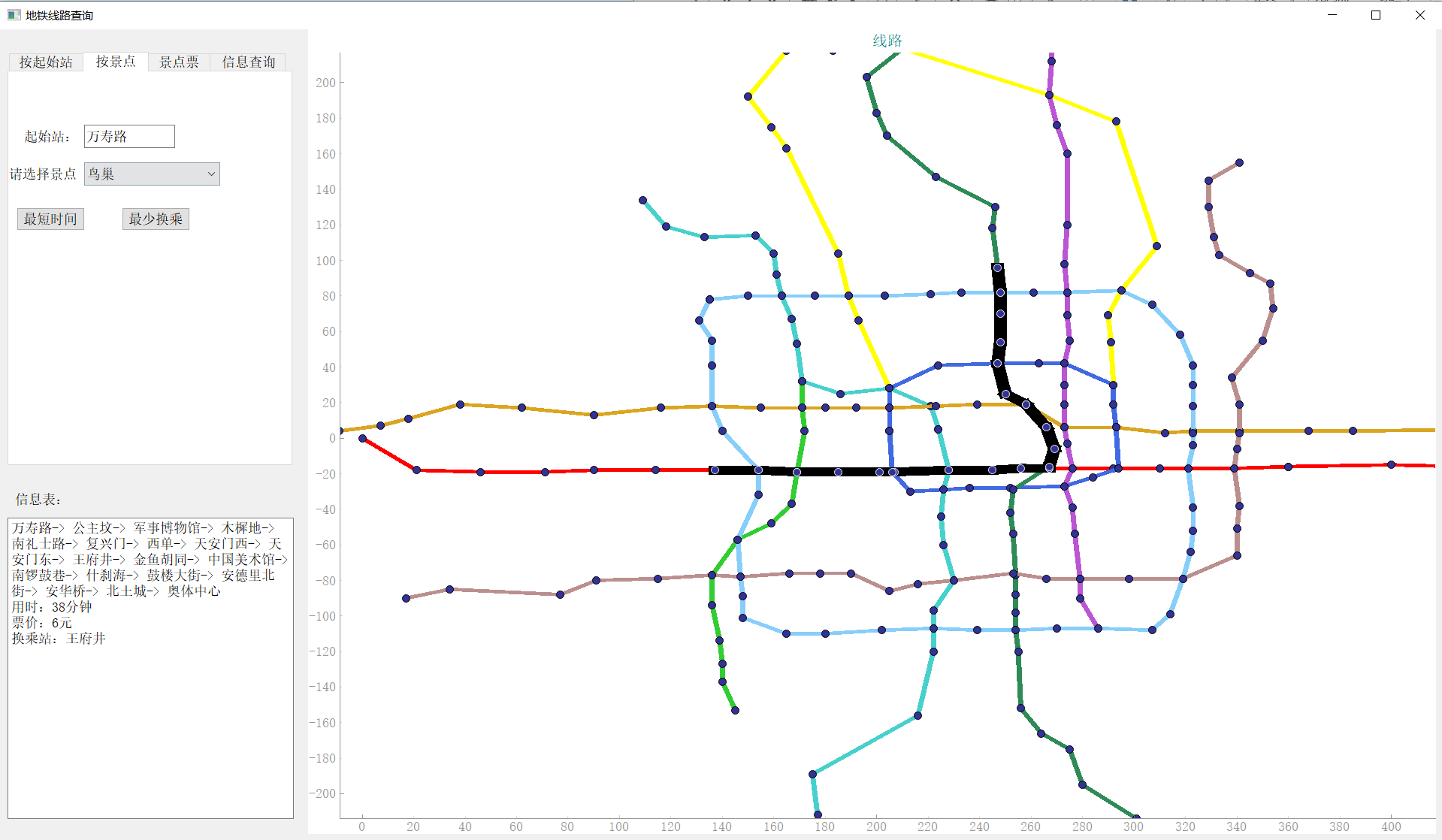


最小换乘算法

样例2



最短时间算法



最小换乘算法

## 按复数景点查询

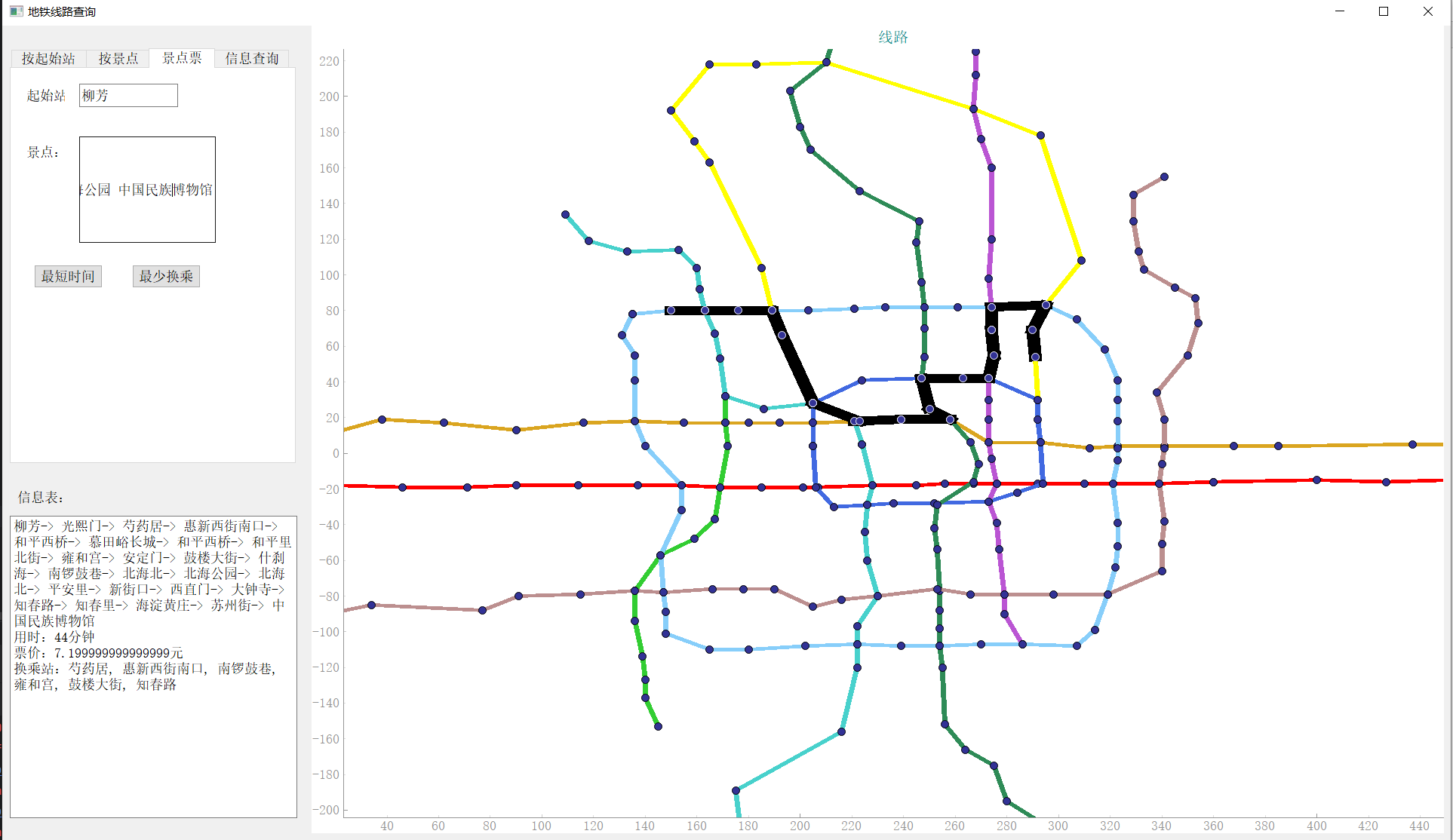
流程：

1. 输入起点站
2. 下拉框选择想去的景点
3. 选择乘车方案（最短时间，最小换乘）

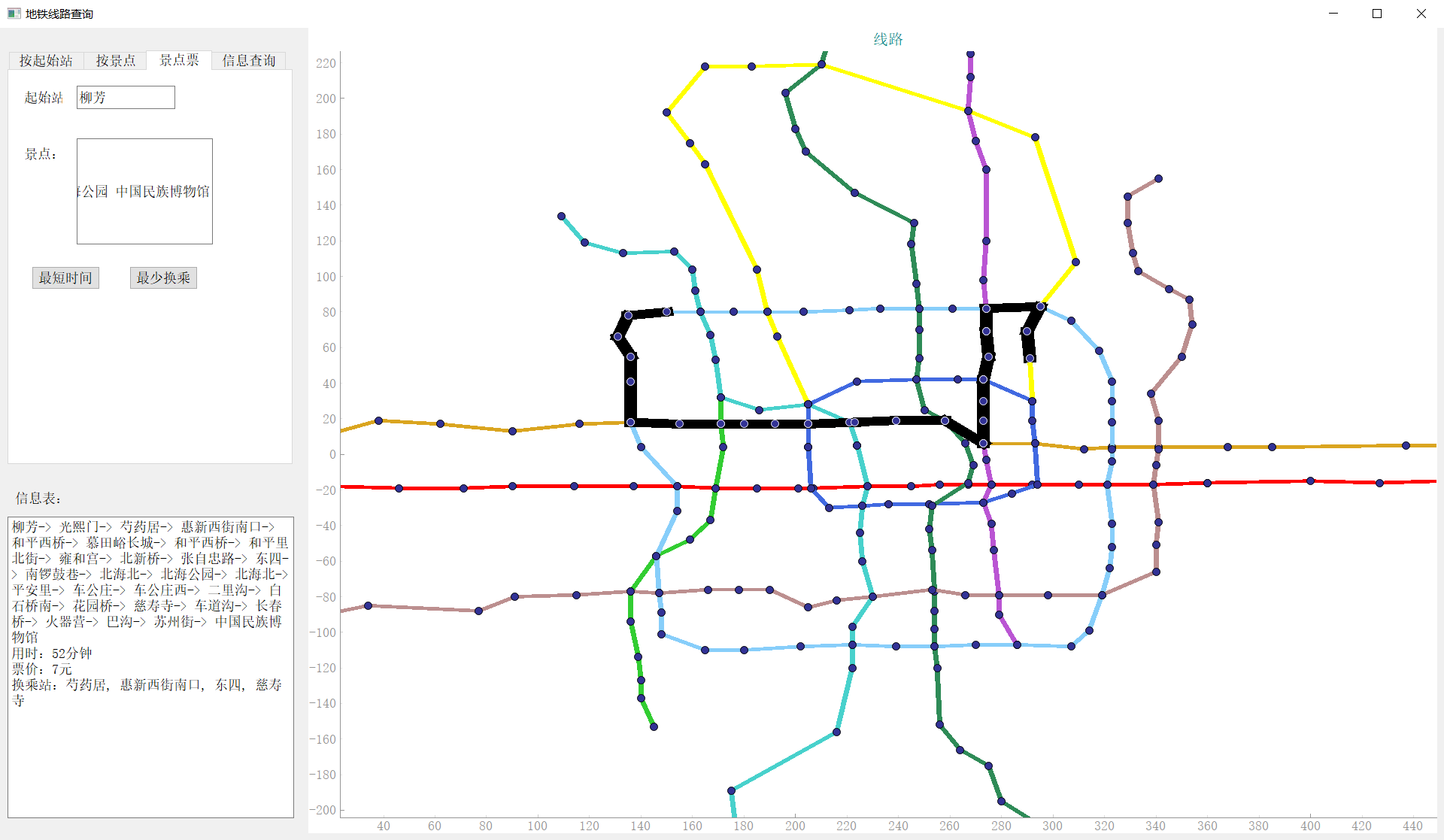
4，点击对应按钮，输出结果

样例1

输入景点：慕田峪长城 北海公园 中国民族博物馆



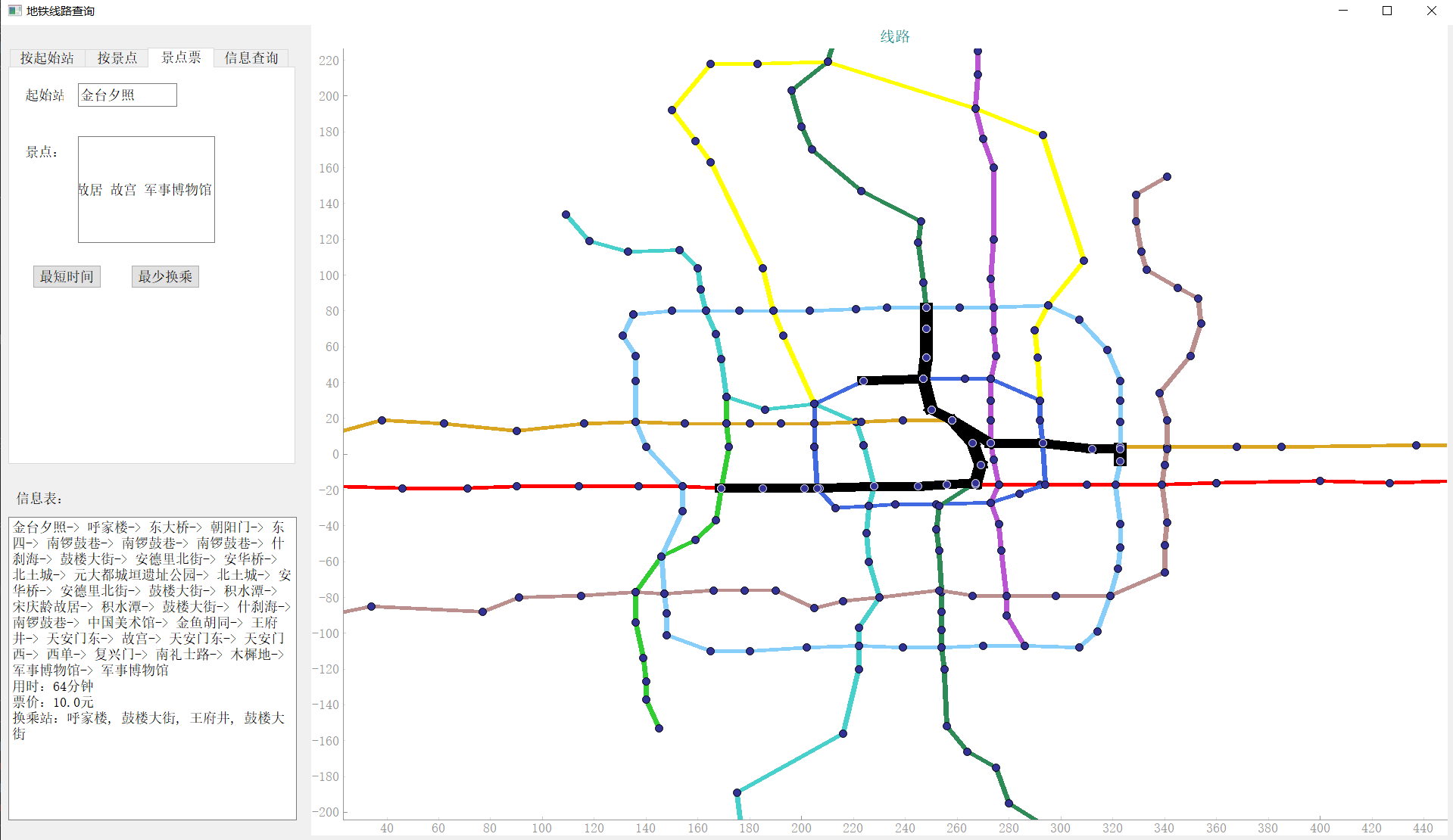
最短时间算法



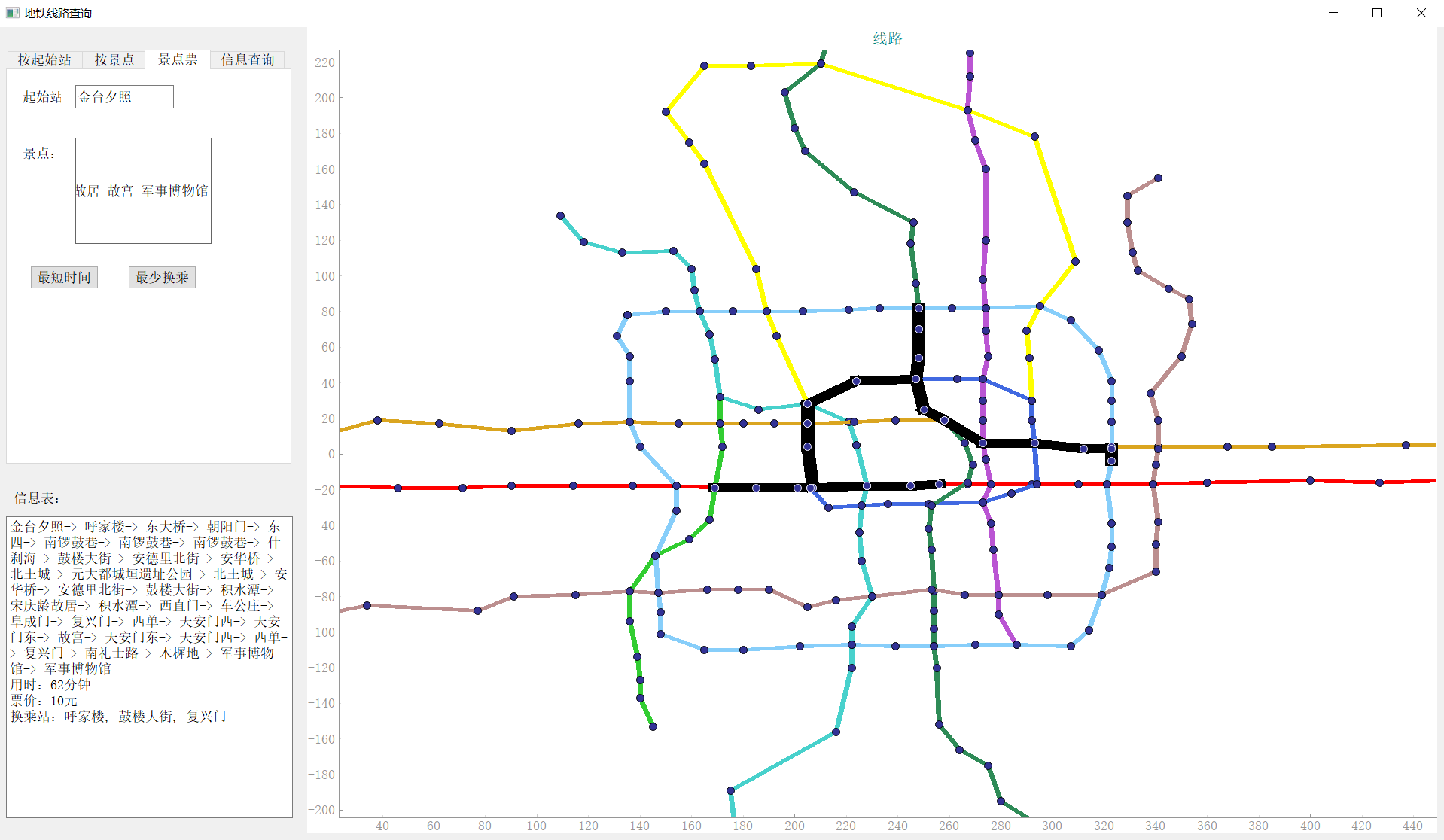
最小换乘算法

样例2

输入景点：南锣鼓巷 元大都城垣遗址公园 宋庆龄故居 故宫 军事博物馆



最短时间算法



最小换乘算法

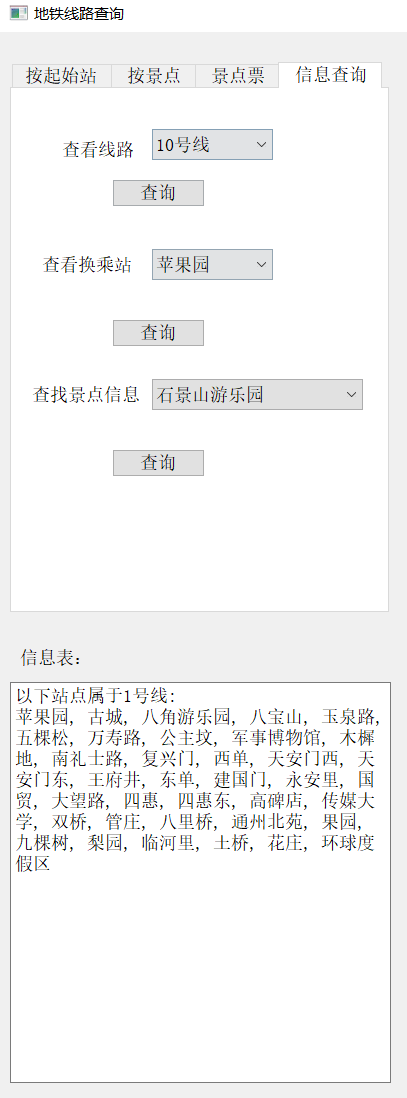
## 信息查询

流程：

1. 选择想要查看线路，换乘站还是景点
2. 下拉选择框选择具体想要查询对象
3. 点击对应查询按钮
4. 输出结果

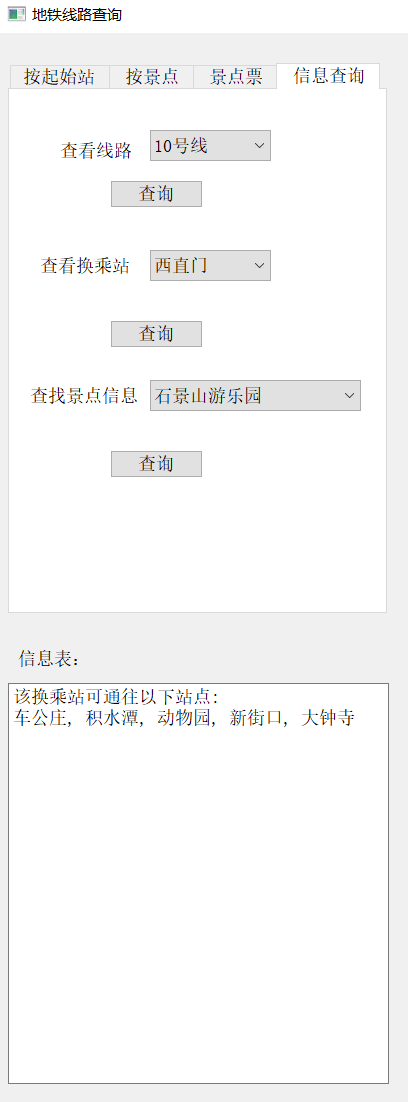
样例1

查询10号线的所有站点



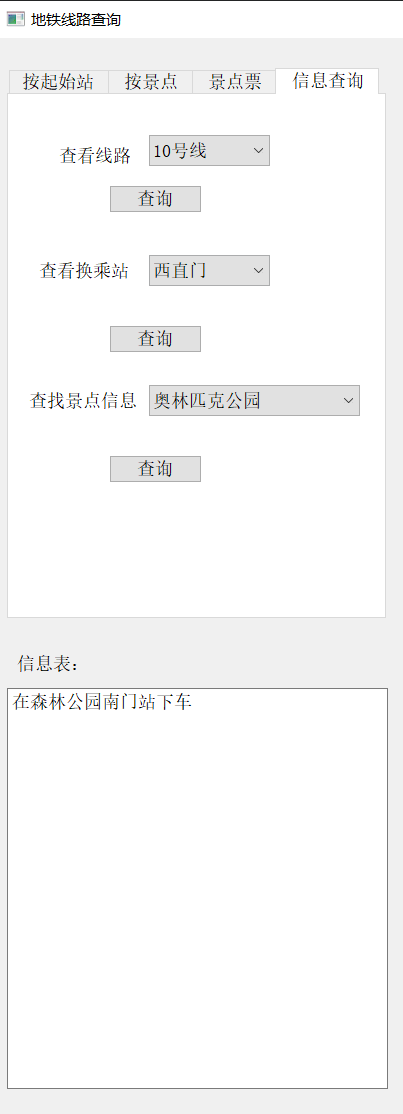
样例2

查询西直门站能到达的所有站点



样例3

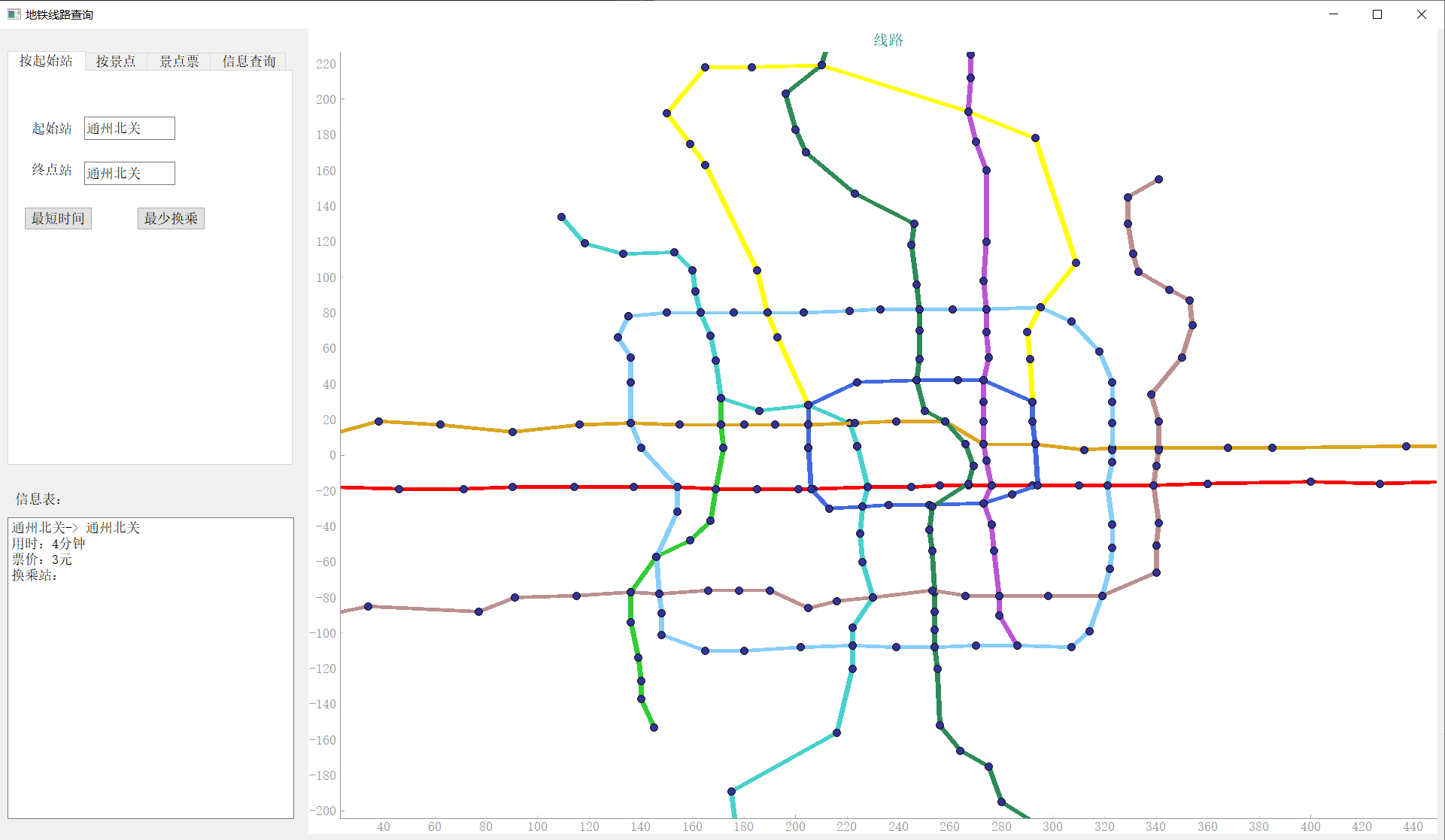
查询奥林匹克公园景点在那站下车



## 错误测试

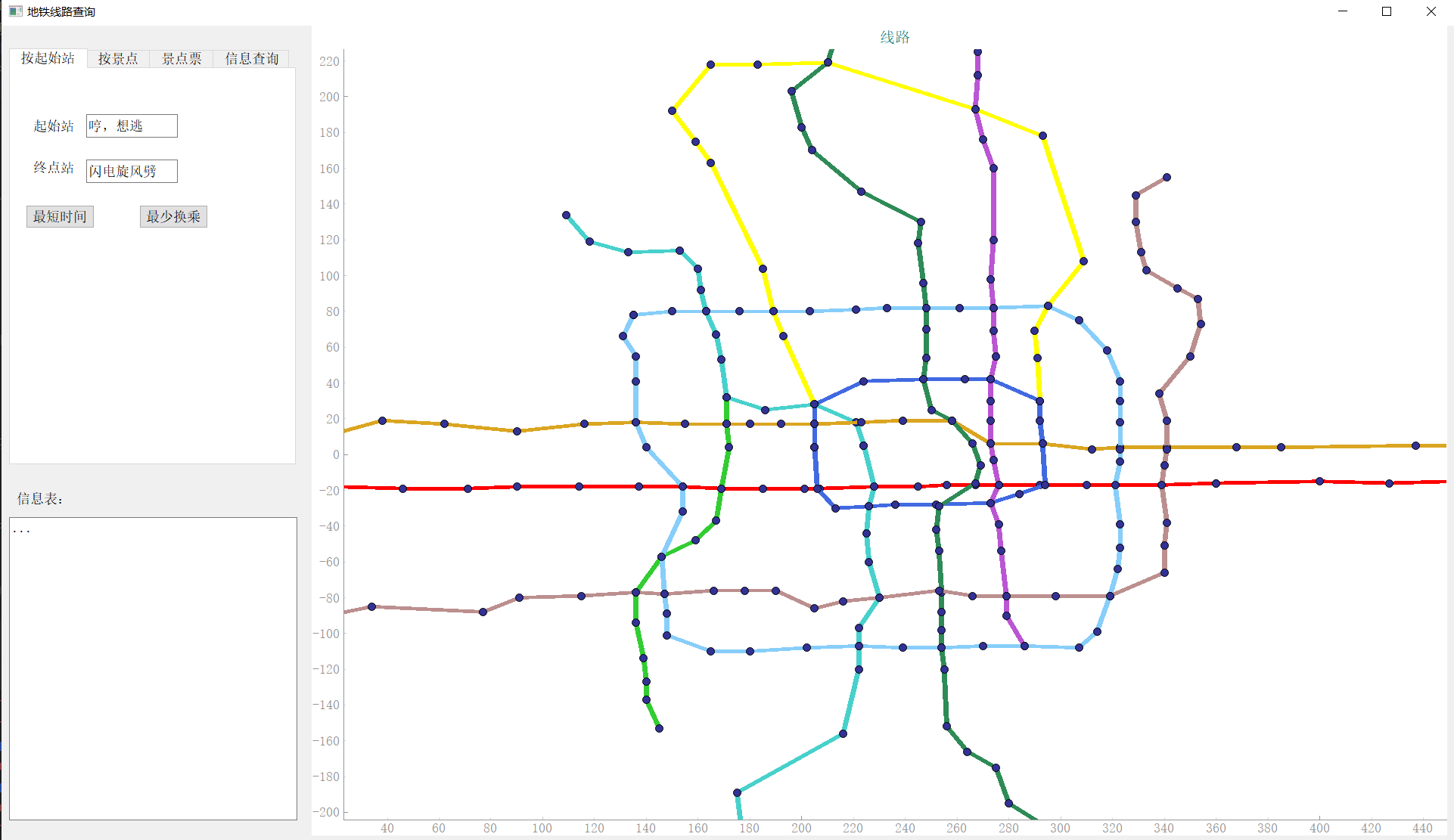
测试1

起点终点输入同样的站



测试2

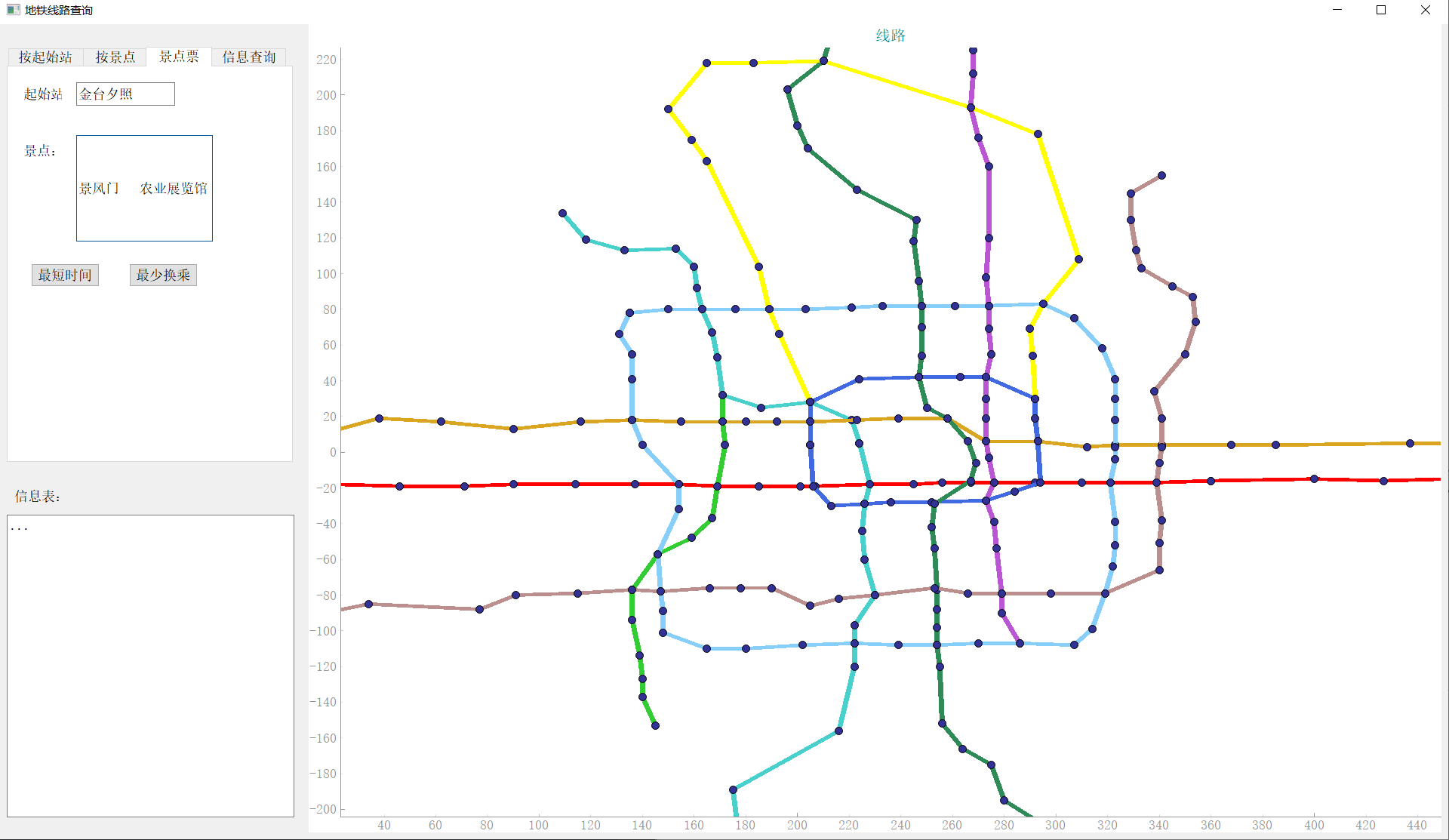
输入错误站名



可见输入错误站点并不会计算出路线，程序也不会出错若不小心打错字，只要修改后重新按下按钮就可得到正确结果。

测试3

输入多个景点时不按要求格式输入，两站之间输入多个空格隔开



可见这么做不会得到线路结果，输入时应严格按照格式要求输入。