doi:10.11887/j.cn.202101015 http://journal.nudt.edu.cn

改进 Dijkstra 算法在大型城市轨道交通网计价系统中的应用*

谢建平[1,2],陈治亚[1],邓连波[1],谢宜斌[2],杨 坤[2]
(1. 中南大学 交通运输工程学院,湖南 长沙 410075;2. 长沙市轨道交通集团有限公司,湖南 长沙 410133)

摘 要:随着城市地铁建设的迅猛发展,国内众多城市地铁线路均已实现网络化运营。考虑到地铁票价制定的公益性,目前国内大部分城市均以最短路径或最少车站的方式计算线网两车站间的票价。以传统的Dijkstra算法为基准,对传统的Dijkstra算法进行改进,分别采用传统Dijkstra算法和改进Dijkstra算法对长沙地铁1~5号线线网最短距离行走路线进行测算。结果表明,改进后的Dijkstra算法不仅有效地提高了算法的时效性,克服了传统算法时间冗长的缺陷,在一定程度上消除了线路之间累积的误差且提高了线网的可延展性。

关键词:长沙地铁1~5号线;城市地铁票价;Dijkstra算法;最短路径算法
中图分类号:TN95 **文献标志码**:A **文章编号**:1001-2486(2021)01-109-08

Application of improved Dijkstra algorithm in large urban rail transit network valuation system

XIE Jianping[1,2], *CHEN Zhiya*[1], *DENG Lianbo*[1], *XIE Yibin*[2], *YANG Kun*[2]
(1. School of Traffic and Transportation Engineering, Central South University, Changsha 410075, China;
2. Changsha Metro Group Co., Ltd, Changsha 410133, China)

Abstract: With the rapid development of urban subway construction, many city subway lines have been networked. Considering the public welfare when making ticket price, most cities in China calculate the ticket price between the two stations of the line network by the shortest path or the least stations now. the traditional Dijkstra algorithm was improved based on the traditional Dijkstra algorithm. The traditional Dijkstra algorithm and the improved Dijkstra algorithm were respectively used to calculate the shortest walking route of Changsha metro line 1~5 network. Results show that the improved Dijkstra algorithm not only effectively improves the efficiency of the algorithm and overcomes the long-time defect of the traditional algorithm, but also partly eliminate the accumulated errors between lines and improve the ductility of the network.

Keywords: Changsha metro line 1~5; urban subway fare; Dijkstra algorithm; shortest path algorithm

随着城市地铁建设的迅猛发展,国内外众多城市地铁线路均已达到网络化运营规模。这将使得两个站点间存在多条行走路径,以哪条路径作为计价路径,才能确保其计价的科学性、合理性,是当前轨道交通行业研究的热点课题之一。

针对地铁两个站点间距离的长短对轨道交通计价的影响,国内外许多专家学者对此进行了深入的研究。文献[1]通过构建城市客运系统整体出行时间最小的双层规划模型,研究了城市轨道交通系统在城市客运结构中的作用。文献[2-6]通过建立模型对票价策略进行了优化,但未对轨道交通成网后各车站间票价计算方式进行深入探讨。文献[7-18]分别就节点约束型、改进的Dijkstra算法的最短路程计算,基于结合Dijkstra算法的信息素递减蚁群(Dijkstra and Ant Colony Optimization based on Pheromone Declining, Dijkstra-PD-ACO)算法的大城市公交线路优化等方面进行研究和探讨,而对于Dijkstra算法优化,并将其应用于城市轨道交通各车站票价计算方面未做详细且深入的研究。文献[19-20]通过建立两种定价方式的双层规划模型对影响票价的因素进行研究,研究结果表明对价格的敏感度是影响两种定价的主要因素。

尽管有众多的专家学者针对Dijkstra算法在大型城市轨道交通线网计价系统中的应用进行了深入研究,但是到目前为止,对Dijkstra算法进行改进并将其应用在大型城市轨道交通线网计价系统中的研究很少。因此,对Dijkstra算法进行改进并将其应用在大型城市轨道交通线网计价系统中具有非常重要的意义。本文对传统的Dijkstra

* **收稿日期**:2020-05-09
基金项目:国家自然科学基金资助项目(71471179)
作者简介:谢建平(1980—),男,湖南邵阳人,高级工程师,博士研究生,E-mail:744833658@qq.com

算法进行了改进,使用传统的 Dijkstra 算法和改进的 Dijkstra 算法计算分析了长沙地铁罐子岭站至毛竹塘站间的最短距离,并对结果进行了分析。为进一步研究 Dijkstra 算法在大型城市轨道交通线网计价系统中的应用提供了参考。

1 改进 Dijkstra 算法的实现

1.1 改进 Dijkstra 算法的主要思想

本算法主要根据地铁线网规划、建设规划、线路里程表等相关资料,提取线网各车站的里程标,绘制出地铁线网示意图,再筛选出线网所有换乘站,且对于同一线路中两换乘站之间无其他换乘站可认定为相邻换乘站。根据换乘站之间的关系及换乘站的中心线里程标,使用传统的 Dijkstra 算法计算各换乘站之间的最短距离。算法详细思路如图 1 所示。

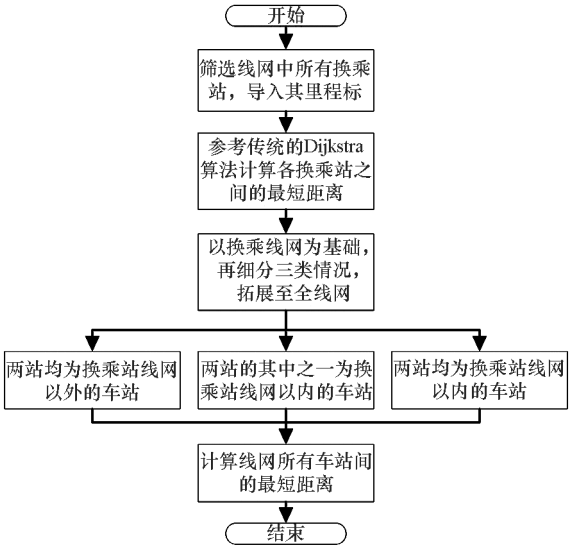


图 1 改进 Dijkstra 算法流程图

Fig. 1 Improved Dijkstra algorithm flow chart

1.2 传统的 Dijkstra 算法计算换乘站间最短距离

假定地铁线网有 n 个换乘站,首先参照传统的 Dijkstra 算法计算出线网所有换乘站之间的最短距离,算法实现如图 2 所示。

算法详细描述如下:

Step 1:筛选出线网中所有换乘站,导入其里程标。

Step 2:计算所有相邻换乘站之间的距离 $l_{i,j}$,令本站的 $l_{i,i}=0$ 且不相邻换乘站之间的距离 $l_{i,j}=+\infty$,其中 $i=1,2,3,\cdots,n$。

Step 3:初始化,令 $i=1$。

Step 4:初始化,令 $d_{i,j}=l_{i,j}$,$m_{i,j}=+\infty$,其中 $j=1,2,3,\cdots,n$ 且 $j\neq i$。

Step 5:比较所有新解且不为站名的 $d_{i,j}$ 大小,即令 $m_{i,j}=\min\{d_{i,j}\}$,其中 $j=1,2,3,\cdots,n$ 且 $d_{i,j}\neq j$ 站名。

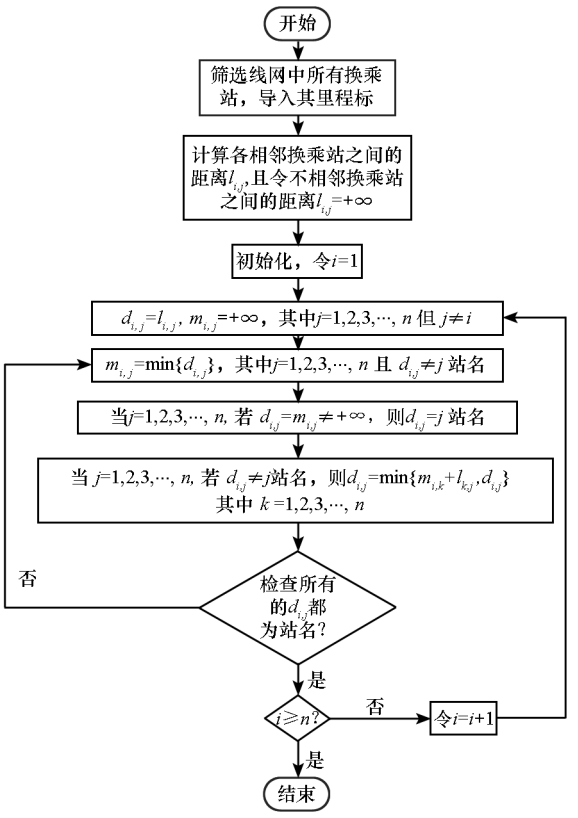


图 2 传统的 Dijkstra 算法计算换乘站间最短距离流程图

Fig. 2 Traditional Dijkstra algorithm to calculate the shortest distance flow chart between transfer stations

Step 6:对于所有 $j=1,2,3,\cdots,n$,当 $d_{i,j}=m_{i,j}\neq+\infty$ 时,则令 $d_{i,j}=j$ 站名。

Step 7:对于所有 $j=1,2,3,\cdots,n$,若 $d_{i,j}\neq j$ 站名,则令 $d_{i,j}=\min\{m_{i,k}+l_{k,j},d_{i,j}\}$,其中 $k=1,2,3,\cdots,n$。

Step 8:检查是否所有 j 的 $d_{i,j}=j$ 站名,其中 $j=1,2,3,\cdots,n$,若为否,则跳至 Step 5,若为是,则跳至下一步。

Step 9:检查 i 是否大于等于 n,若为否,则令 $i=i+1$ 且跳至 Step 4,若为是,则该算法结束。

其中:i,j 表示换乘站线网中第 i 或 j 个换乘站;n 表示换乘站线网总计有 n 个换乘站;$l_{i,j}$ 表示两相邻换乘站 i 和 j 之间的距离,若不相邻,其取值为 $+\infty$;$m_{i,j}$ 表示换乘站 i 与换乘站 j 之间的最短距离;$d_{i,j}$ 表示换乘站 i 与换乘站 j 之间的最短距离求值过程中经历的中间值。

1.3 改进的 Dijkstra 算法计算线网各车站间最短距离

参考如上算法,计算出各换乘站之间最短距离表,再参考图 2 流程图中算法的主要思想,细分三个部分对线路各车站间最短距离进行计算,其具体的算法实现如图 3 所示。

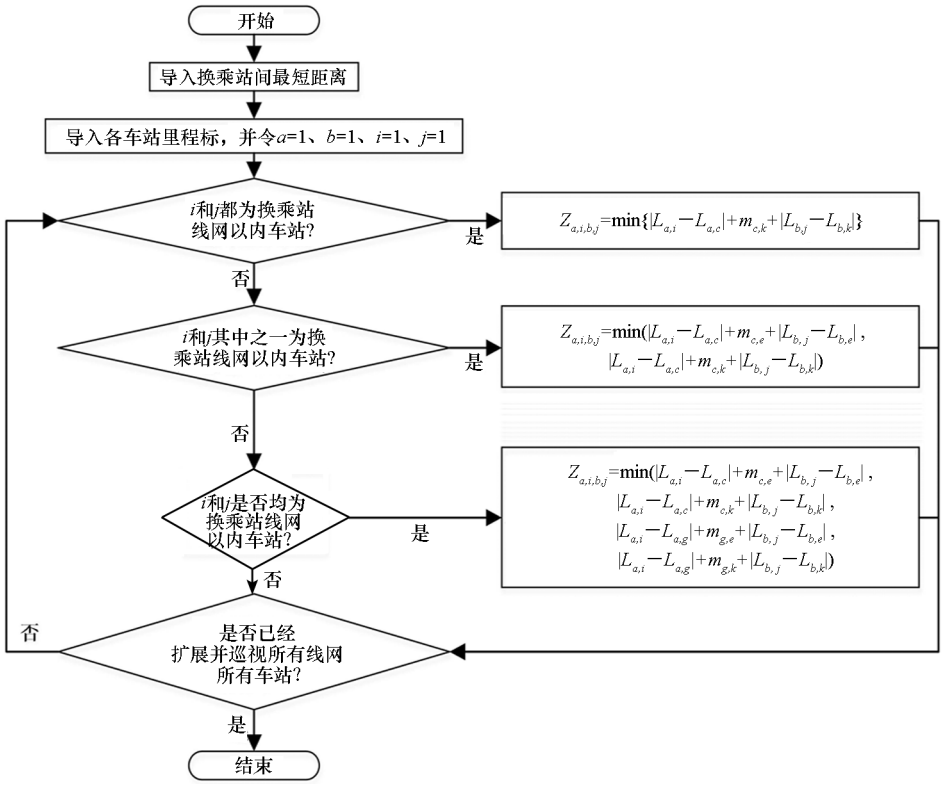


图 3　改进的 Dijkstra 算法计算换乘站间最短距离流程图

Fig. 3　Flow chart which used the improved Dijkstra algorithm to calculate the shortest distance between the stations

算法详细描述如下:

Step 1:导入换乘站间的最短距离即图 2 算法的结果,并参考换乘站在线网中的实际位置对算法结果进行适当的调整。

Step 2:导入线网中各车站的线路里程标,并令 $a=1$、$b=1$、$i=1$、$j=1$。

Step 3:检查 i 和 j 是否为换乘站线网以内的车站,若为否,则跳至下一步,若为是,则令:

$$Z_{a,i,b,j}=\min\{|L_{a,i}-L_{a,c}|+m_{c,k}+|L_{b,j}-L_{b,k}|\} \tag{1}$$

Step 4:检查 i 和 j 其中之一是否为换乘站线网以内的车站,若为否,则跳至下一步,若为是,则令:

$$\begin{aligned}Z_{a,i,b,j}=\min(&|L_{a,i}-L_{a,c}|+m_{c,e}+|L_{b,j}-L_{b,e}|,\\&|L_{a,i}-L_{a,c}|+m_{c,k}+|L_{b,j}-L_{b,k}|)\end{aligned} \tag{2}$$

Step 5:检查 i 和 j 是否均为换乘站线网以内的车站,若为否,则跳至下一步,若为是,则令:

$$\begin{aligned}Z_{a,i,b,j}=\min(&|L_{a,i}-L_{a,c}|+m_{c,e}+|L_{b,j}-L_{b,e}|,\\&|L_{a,i}-L_{a,c}|+m_{c,k}+|L_{b,j}-L_{b,k}|,\\&|L_{a,i}-L_{a,g}|+m_{g,e}+|L_{b,j}-L_{b,e}|,\\&|L_{a,i}-L_{a,g}|+m_{g,k}+|L_{b,j}-L_{b,k}|)\end{aligned} \tag{3}$$

Step 6:判断是否已经扩展并巡视所有线网所有车站,若为否,则跳至 Step 3,若为是,则该算法结束。

其中:a、b 表示线路编号,当算法巡视所有车站时,a 和 b 均等于线路总数;i 表示第 a 条线路中第 i 个车站,i 不大于第 a 条线路车站总数;j 表示第 b 条线路中第 j 个车站,j 不大于第 b 条线路车站总数;$L_{a,i}$ 表示线路 a 中第 i 个车站的中心线里程标;$L_{b,j}$ 表示线路 b 中第 j 个车站的中心线里程标;$Z_{a,i,b,j}$ 表示第 a 条线路中第 i 个车站与第 b 条线路中第 j 个车站的最短距离;c、g 表示为图 3 中离线路 a 第 i 个车站最近的换乘站,是线路 a 上的第 c 和 g 个车站;e、k 表示为图 3中离线路 b 第 j 个车站最近的换乘站,是线路 b 上的第 e 和 k 个车站;$m_{c,e}$、$m_{c,k}$ 表示换乘站 c 与换乘站 e 或 k 的最短距离,是根据图 2 算法求出的结果。

1.4　线网各车站间票价计算算法

根据图 3 中算法求出线网各车站之间最短距离 $Z_{a,i,b,j}$,再参考如图 4 所示算法,核算线网各车站之间的票价。

算法详细描述如下:

Step 1:导入线网各车站间最短距离 $Z_{a,i,b,j}$,即图 3 的计算结果。

Step 2:初始化,令 $p_1=0$,F、$P_{a,i,b,j}$ 均为 $+\infty$,$f_1=0$。

Step 3:初始化,令 $m=1$。

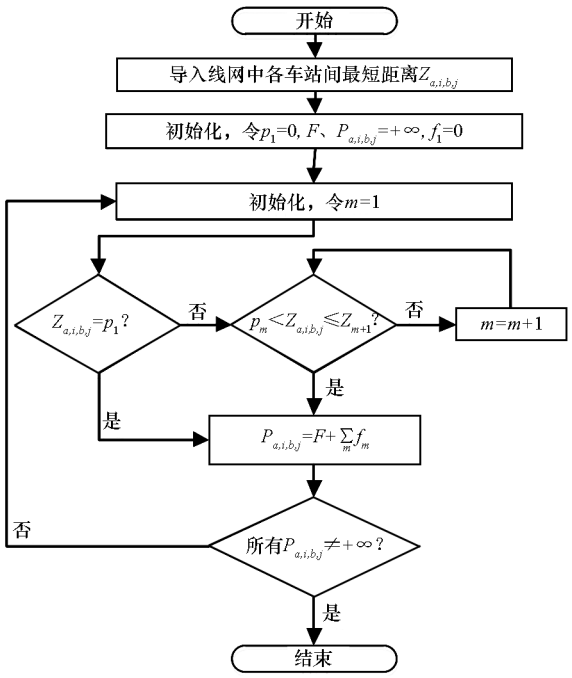


图 4 线网各车站间票价计算算法

Fig. 4 Algorithm flow chart of ticket price between stations in line network

Step 4:判断第 a 条线路第 i 个车站与第 b 条线路第 j 个车站之间的最短距离 $Z_{a,i,b,j}$ 是否等于 p_1,即 $Z_{a,i,b,j}=p_1$,若为是,则跳至 Step 6,若为否,则跳至 Step 5。

Step 5:判断第 a 条线路第 i 个车站与第 b 条线路第 j 个车站之间的最短距离 $Z_{a,i,b,j}$ 属于哪个计价区间,即检查 $p_m < Z_{a,i,b,j} \leqslant p_{m+1}$,若为否,则令 $m=m+1$,并再次执行该检查,若为是,则跳至 Step 6。

Step 6:令 $P_{a,i,b,j} = F + \sum_m f_m$。

Step 7:检查所有 $P_{a,i,b,j}$ 是否不等于 $+\infty$,即 $P_{a,i,b,j} \neq +\infty$,若为否,则跳至 Step 3,若为是,则该算法结束。

其中:m 表示城市地铁第 m 个计价区间;p_m 表示在第 m 个计价区间乘客可乘坐的距离,F 表示城市地铁起步票价;$P_{a,i,b,j}$ 表示第 a 条线路中第 i 个车站与第 b 条线路中第 j 个车站的票价,f_m 表示第 m 个计价区间较第 $m-1$ 个计价区间需要调整的票价。

2 改进 Dijkstra 算法实例与分析

2.1 算法实例

以长沙地铁罐子岭站至毛竹塘站、罐子岭站至赤岗岭站、湖南师大站至赤岗岭站的最短距离计算为例。对于传统算法需要根据长沙地铁 1 ~ 5 号线线网布局及各车站的中心线里程标,计算线网中各相邻车站之间的距离,以罐子岭站(或湖南师大站)为目标节点,并建立集合 1,开始遍历长沙地铁 1 ~ 5 号线线网各车站,逐步将计算出最短距离的车站纳入集合 1,算法在完成长沙地铁线网 99 个车站间最短距离的计算后结束,并根据要求提取出罐子岭站至毛竹塘站(或罐子岭站至赤岗岭站,或湖南师大站至赤岗岭站)的最短距离。改进的 Dijkstra 算法需根据长沙地铁 1 ~ 5 号线线网布局及各车站里程标,将线路中换乘站筛选出来,根据长沙地铁 1 ~ 5 号线线网布局,共计有 12 个换乘站,若任意两个换乘站在同一线路且其之间无其他换乘站,则认定其为相邻换乘站,将相邻换乘站连接起来,形成换乘站线网(如图 5 所示),并计算相邻换乘站之间的距离;再根据传统 Dijkstra 算法,计算各换乘站之间最短距离,而长沙地铁的换乘站线网图只有 12 个换乘站,较线网 99 个车站,计算难度及消耗时间大大缩减;再详细按照图 3 将线网各车站细分为三类,计算罐子岭站至毛竹塘站、罐子岭站至赤岗岭站,湖南师大站至赤岗岭站的最短距离,如罐子岭站和毛竹塘站均为换乘站线网以外的车站,则罐子岭站至毛竹塘站的计算方法可参考图 3 中第一种情况,对于罐子岭站至赤岗岭站而言,其中赤岗岭站为换乘站线网以内车站,则罐子岭站至赤岗岭站的计算方法可参考图 3 中第二种情况,湖南师大站和赤岗岭站均为换乘站线网以内车站,则湖南师大站至赤岗岭站的计算方法可参考图 3 中第三种情况。图 6 和图 7 以罐子岭站至毛竹塘站为例,使用两种算法时所涉及的站点及最短距离行走路径标记如下所示。

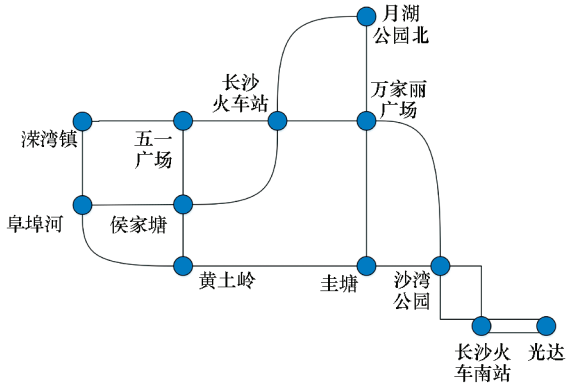


图 5 长沙地铁 1 ~ 5 号线换乘站线网图

Fig. 5 Transfer station network of Changsha metro line 1 ~ 5

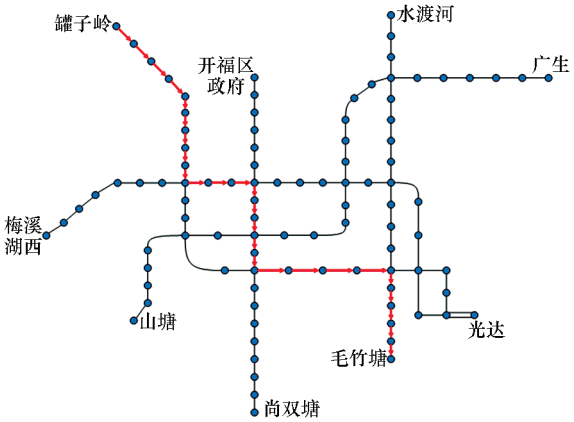


图 6　传统 Dijkstra 算法计算罐子岭站至毛竹塘站最短距离时行走路径

Fig. 6　Traditional Dijkstra algorithm to calculate the shortest walking path between Guanziling station and Maozhutang station

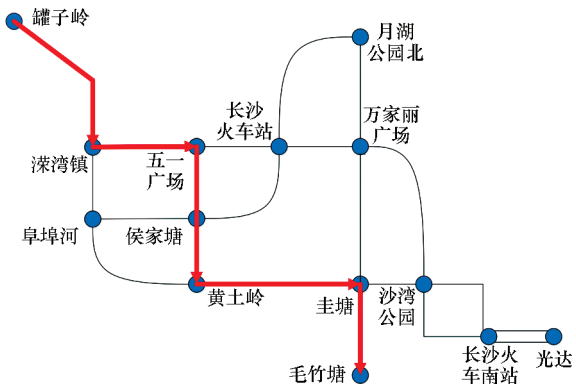


图 7　改进 Dijkstra 算法计算罐子岭站至毛竹塘站最短距离时行走路径

Fig. 7　Improved Dijkstra algorithm to calculate the shortest walking path between Guanziling station and Maozhutang station

由图 6 和图 7 可知,使用传统 Dijkstra 算法计算罐子岭站至毛竹塘站间最短距离时,参与计算的站点数为 99,使用改进的 Dijkstra 算法计算罐子岭站至毛竹塘站间最短距离时,参与计算的站点数为 14。具体情况如表 1 和图 8 ~ 10 所示,图中①~⑥与表 1 中标号相对应。

由表 1 可知,传统 Dijkstra 算法与改进的 Dijkstra 算法的行走路径和最短距离大致相同,但在参与计算的站点数、计算次数、数据存储及运行时间方面,改进的 Dijkstra 算法都进行了大量缩减。

表 1　传统 Dijkstra 算法和改进 Dijkstra 算法的算例比较

Tab. 1　Example compared with traditional Dijkstra algorithm and improved Dijkstra algorithm

算法	起始站	终点站	最短距离/km	票价/元	参与计算的站点数	计算次数	数据存储	程序运行时间/s
①传统 Dijkstra 算法	罐子岭	毛竹塘	31.349	7	99	333 102	9 999	53
②改进 Dijkstra 算法					14	705	170	5
③传统 Dijkstra 算法	罐子岭	赤岗岭	21.673	5	99	333 102	9 999	53
④改进 Dijkstra 算法					14	707	170	5
⑤传统 Dijkstra 算法	湖南师大	赤岗岭	8.607	3	99	333 102	9 999	53
⑥改进 Dijkstra 算法					14	709	170	5

注:目前长沙地铁票价政策是“起步价 2 元可乘坐 6 km(含 6 km),超过 6 km 采用‘递远递减’的计价原则,6 ~ 16 km(含 16 km)范围每递增 5 km 加 1 元,16 ~ 30 km(含 30 km)范围内每递增 7 km 加 1 元,30 km 以上每递增 9 km 加 1 元”,参考表 1 可知,罐子岭站至毛竹塘站的最短距离为 31.349 km,属于 30 km 以上范围,票价为 7 元,罐子岭站至赤岗岭站的最短距离为 21.673 km,属于 16 ~ 23 km(含 23 km)范围,票价为 5 元,湖南师大站至赤岗岭站的最短距离为 8.607 km,属于 6 ~ 11 km(含 11 km)范围,票价为 3 元。

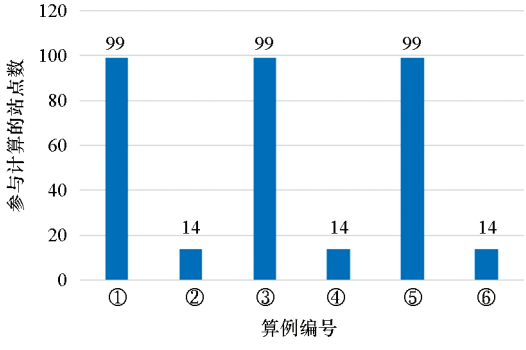


图 8　参与计算的站点数比较图

Fig. 8　Comparison chart of the number of stations participating in calculation

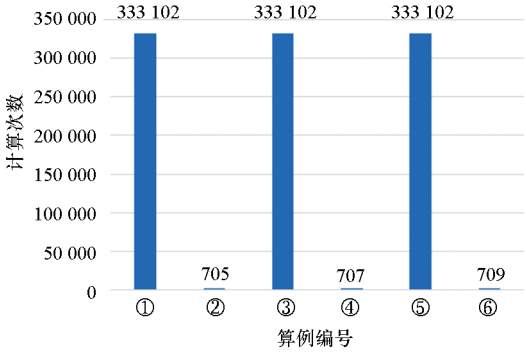


图 9　计算次数比较图

Fig. 9　Comparison chart of calculation times

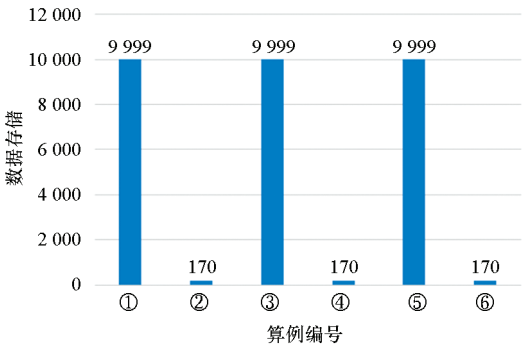


图 10 数据存储比较图

Fig. 10 Data storage comparison chart

2.2 比较分析

当使用传统 Dijkstra 算法和改进的 Dijkstra 算法求任意两车站之间最短距离时，所需参与计算的站点数、计算次数及数据存储比较如表 2 所示。

表 2 传统 Dijkstra 算法和改进 Dijkstra 算法比较

Tab. 2 Comparison between original Dijkstra algorithm and improved Dijkstra algorithm

算法	参与计算的站点数	计算次数	数据存储
传统 Dijkstra 算法	S	$S^2+\sum_{S}[2x(S-x)-1]$	S^2+2S
改进 Dijkstra 算法	$n+2$	$n^2+\sum_{n}[2x(n-x)-1]+t$	n^2+2n+2

表 2 中，S 为地铁线网站点总数；n 为地铁线网中换乘站总数；x 为已经寻找出最短路径的站点，初始值为 1；t 为固定参数值，当两车站均为换乘站线网（含换乘站）以外车站时 t 取 1，其中之一为换乘站线网以内车站时 t 取 3，两个均为换乘站线网（含换乘站）以内车站时 t 取 5。

在计算过程中，使用传统 Dijkstra 算法计算某两个站点的最短距离时，就需要先根据里程表计算各站点之间的距离初始值，相邻站点之间的距离为 $|L_{a,i}-L_{a,i+1}|$，非相邻车站之间的距离为 $+\infty$，其计算次数为 S^2。而改进的算法中，需要先筛选相应换乘站，并根据换乘站之间的关系，求出各换乘站之间的距离初始值，其计算次数为 n^2。

在计算过程中，使用传统 Dijkstra 算法计算任意两个站点之间的最短距离时，都需要以其中一个站点往外延，延伸至线网所有车站（不含重复车站），先使用前一步计算的结果，寻找起点至某一节点最短距离的值，再比较起点至所有未寻找出最短距离的节点（下文为未纳入的节点）距离，并找出距离最小值。其中寻找起点至未纳入的节点最小值计算次数为 $x(S-x)$，需要比较 $x(S-x)-1$ 次才能求得该阶段的最小值，当所有阶段完成，计算次数为 $\sum_{S}[2x(S-x)-1]$，而使用改进算法的计算次数为 $\sum_{n}[2x(n-x)-1]+t$。

在计算过程中，传统的 Dijkstra 算法需要的第一个数组，大小为 S^2，存储所有车站的初始距离值，并可以用来存储计算中间值；第一个数组，大小为 S，存储起点至所有车站的最短距离；第三个数组，大小为 S，存储所有节点里程标。而改进的 Dijkstra 算法，需要存储的换乘站数组及参与计算车站里程表的数组总数为 n^2+2n+2，此外，还需要一个变量存储起点至要求节点的最短距离。

2.3 算法延展

以长沙地铁 1 号线北延线一期为例，假定 $L_{a,i}$ 表示线路 a 中第 i 个车站的中心线里程标，$L_{b,j}$ 表示线路 b 中第 j 个车站的中心线里程标，使用改进 Dijkstra 算法求解如下：

Step 1：当 i,j 均为 1 号线北延线一期车站，则最短距离 $Z_{1,i,1,j}=|L_{1,i}-L_{1,j}|$。

Step 2：当 i 或 j 之一为 1 号线北延线一期车站，假定 i 为 1 号线北延线一期车站，则最短距离 $Z_{1,i,b,j}=|L_{1,i}-L_{1,1}|+Z_{1,1,b,j}$，由于 i 车站为 1 号线北延线一期车站，离其最近的车站为 1 号线第 1 个车站。

Step 3：当 i 和 j 均非 1 号线北延线一期车站，则最短距离 $Z_{a,i,b,j}=Z_{a,i,b,j}$。

算法实例如表 3 所示。

由于长沙地铁 1 号线北延线为 1 号线新增车站，若按照传统的 Dijkstra 算法的原理计算线网中所有车站（含新增车站）间最短距离，需遍历线网所有车站进行求解，计算烦琐性增加，显然其延展性较改进的 Dijkstra 算法差。

表 3　改进的 Dijkstra 算法的延展性算例

Tab. 3　Example of extension of improved Dijkstra algorithm

起始站	终点站	计算线路	计算公式	最短距离/km	票价/元
青竹湖路（1 号线北延线）	周南中学（1 号线北延线）	青竹湖路→周南中学	$\lvert L_{1,青} - L_{1,周} \rvert$	4.01	2
青竹湖路（1 号线北延线）	广生（3 号线一期）	青竹湖路→开福区政府→广生	$\lvert L_{1,青} - L_{1,开} \rvert + Z_{1,开,b,广}$	36.49	7
北辰三角洲（1 号线一期）	广生（3 号线一期）	北辰三角洲→广生	$Z_{1,北辰,b,广}$	25.30	6

3　结论

1)改进后的 Dijkstra 算法克服了传统算法时间冗长的缺陷,创新地将换乘站从线网中独立出来,再通过换乘站线网拓展至线网各站,实际中,换乘站数量远远小于线网车站总数,这就减少了节点遍历数和查询时间,增强了设计的可行性;

2)若需知道线网各车站间最短距离,无须逐一核算各相邻车站之间的距离,减少了产生误差的可能性;

3)在换乘站不变,但有新线开通时,只需将车站里程标导入,便可直接计算出该站至其他车站的最短距离,可延展性更强。

参考文献(References)

[1] 刘美银, 王建伟. 考虑城市客运结构优化的城市轨道交通定价方法研究[J]. 交通运输系统工程与信息, 2017, 17(3): 53 - 59.
LIU Meiyin, WANG Jianwei. Pricing method of urban rail transit considering the optimization of passenger transport structure[J]. Journal of Transportation Systems Engineering and Information Technology, 2017, 17(3): 53 - 59. (in Chinese)

[2] 许旺土, 何世伟, 宋瑞, 等. 整合运营下的轨道交通发车间隔及票价模型[J]. 系统工程学报, 2011, 26(3): 330 - 339.
XU Wangtu, HE Shiwei, SONG Rui, et al. Headway and fare model of urban rail transit for integrated operation[J]. Journal of Systems Engineering, 2011, 26(3): 330 - 339. (in Chinese)

[3] 邓连波, 王锋, 张钊, 等. 城市轨道交通线路票价策略优化[J]. 系统工程理论与实践, 2016, 36(6): 1568 - 1575.
DENG Lianbo, WANG Feng, ZHANG Zhao, et al. Fare optimization for urban rail line[J]. System Engineering—Theory & Practice, 2016, 36(6): 1568 - 1575. (in Chinese)

[4] 林晓言, 冯颖. 参照点依赖的北京市轨道交通票价弹性研究[J]. 中国工业经济, 2013, 7: 108 - 120.
LIN Xiaoyan, FENG Ying. Rail transit ticket price elasticity in Beijing based on reference-dependent principle[J]. China Industrial Economics, 2013, 7: 108 - 120. (in Chinese)

[5] 田贵超, 张建同, 胡一竑. 基于乘客行为的城市轨道交通分时定价策略[J]. 上海管理科学, 2013, 35(5): 77 - 83.
TIAN Guichao, ZHANG Jiantong, HU Yihong. Research on urban rail transit's time differential pricing strategy based on passenger behavior[J]. Shanghai Management Science, 2013, 35(5): 77 - 83. (in Chinese)

[6] 陈治亚, 文邦彦, 晏秀娟, 等. 基于社会福利最大化的 PPP 模式下城市轨道交通票价优化[J]. 铁道科学与工程学报, 2018, 15(9): 2423 - 2431.
CHEN Zhiya, WEN Bangyan, YAN Xiujuan, et al. Optimal ticket pricing of urban railway transit with public-private partnership aiming at maximum welfare[J]. Journal of Railway Science and Engineering, 2018, 15(9): 2423 - 2431. (in Chinese)

[7] 康文雄, 许耀钊. 节点约束型最短路径的分层 Dijkstra 算法[J]. 华南理工大学学报(自然科学版), 2017, 45(1): 66 - 73.
KANG Wenxiong, XU Yaozhao. A hierarchical Dijkstra algorithm for solving shortest path from constrained nodes[J]. Journal of South China University of Technology (Natural Science Edition), 2017, 45(1): 66 - 73. (in Chinese)

[8] 刘建美, 马寿峰, 马帅奇. 基于改进的 Dijkstra 算法的动态最短路计算方法[J]. 系统工程理论与实践, 2011, 31(6): 1153 - 1157.
LIU Jianmei, MA Shoufeng, MA Shuaiqi. Computation method of the dynamic shortest path based on improved-Dijkstra algorithm[J]. System Engineering—Theory & Practice, 2011, 31(6): 1153 - 1157. (in Chinese)

[9] SZETO W Y, WU Y Z. A simultaneous bus route design and frequency setting problem for Tin Shui Wai, Hong Kong[J]. European Journal of Operational Research, 2011, 209(2): 141 - 155.

[10] NIKOLIC M, TEODOROVIC D. Transit network design by bee colony optimization[J]. Expert Systems with Applications, 2013, 40(15): 5945 - 5955.

[11] MIRHASSANI S A, ABOLGHASEMI N. A particle swarm optimization algorithm for open vehicle routing problem[J]. Expert Systems with Applications, 2011, 38(9): 11547 - 11551.

[12] YU Bin, YANG Zhongzhen, JIN Penghuan, et al. Transit route network design-maximizing direct and transfer demand density[J]. Transportation Research Part C: Emerging Technologies, 2012(22): 58 - 75.

[13] 潘若愚, 褚伟, 杨善林. 基于 Dijkstra-PD-ACO 算法的大城

市公交线路优化与评价方法研究[J]. 中国管理科学, 2015, 23(9): 106 – 115.
PAN Ruoyu, CHU Wei, YANG Shanlin. Metropolitan bus lines optimization and evaluation method based on Dijkstra-PD-ACO algorithm [J]. Chinese Journal of Management Science, 2015, 23(9): 106 – 115. (in Chinese)

[14] 李玉鑑, 李厚君. 近似最小树的哈希 Dijkstra 算法[J]. 北京工业大学学报, 2011, 37(12): 1915 – 1920.
LI Yujian, LI Houjun. Hash Dijkstra algorithm for approximate minimal spanning tree [J]. Journal of Beijing University of Technology, 2011, 37(12): 1915 – 1920. (in Chinese)

[15] XU M H, LIU Y Q, HUANG Q L, et al. An improved Dijkstra's shortest path algorithm for sparse network [J]. Applied Mathematics and Computation, 2007, 185(1): 247 – 254.

[16] 王志坚, 韩伟一, 李一军. 具有多条最短路径的最短路问题[J]. 哈尔滨工业大学学报, 2010, 42(9): 1428 – 1431.
WANG Zhijian, HAN Weiyi, LI Yijun. Shortest path problem with multiple shortest paths[J]. Journal of Harbin Institute of Technology, 2010, 42(9): 1428 – 1431. (in Chinese)

[17] 王世明, 邢建平, 张玉婷, 等. 典型城市路网中的椭圆最短路径算法[J]. 系统工程理论与实践, 2011, 31(6): 1158 – 1164.
WANG Shiming, XING Jianping, ZHANG Yuting, et al. Ellipse-based shortest path algorithm for typical urban road networks[J]. System Engineering—Theory & Practice, 2011, 31(6): 1158 – 1164. (in Chinese)

[18] SUNG K, BELL M G H, SEONG M, et al. Shortest paths in a network with time-dependent flow speeds [J]. European Journal of Operational Research, 2000, 121(1): 32 – 39.

[19] 李雪岩, 李雪梅, 李学伟. 基于元胞自动机的城市轨道交通定价双层规划[J]. 管理工程学报, 2017, 31(1): 155 – 161.
LI Xueyan, LI Xuemei, LI Xuewei. Bi-level programming for urban rail transit ticket pricing based on cellular automata[J]. Journal of Industrial Engineering and Engineering Management, 2017, 31(1): 155 – 161. (in Chinese)

[20] NASSI C D, COSTA FC C. Use of the analytic hierarchy process to evaluate transit fare system [J]. Research in Transportation Economics, 2012, 36: 50 – 62.