DOI: 10.13203/j.whugis1999.03.005

第 24卷第 3期 武 汉 测 绘 科 技 大 学 学 报 Vol.24 No.3
1999年 9月 Journal of Wuhan Technical University of Surveying and Mapping Sept. 1999

Dijkstra最短路径算法的一种高效率实现*

乐 阳 龚健雅

(武汉测绘科技大学测绘遥感信息工程国家重点实验室,武汉市珞喻路 129号,430079)

摘 要 在已存在的一些最短路径算法测试总结的基础上,根据 GIS中网络计算的实际情况,从网络结构的拓扑表示以及 Dijkstra算法中快速搜索技术的实现入手,提出了一种 Dijkstra最短路径算法的高效率实现方法。

关键词 最短路径算法;网络分析;地理信息系统

分类号 P208;O22

随着计算机的普及以及地理信息科学的发展,GIS因其强大的功能得到日益广泛和深入的应用。网络分析作为 GIS最主要的功能之一,在电子导航、交通旅游、城市规划以及电力、通讯等各种管网、管线的布局设计中发挥了重要的作用,而网络分析中最基本最关键的问题是最短路径问题。最短路径不仅仅指一般地理意义上的距离最短,还可以引申到其他的度量,如时间、费用、线路容量等。相应地,最短路径问题就成为最快路径问题、最低费用问题等。由于最短路径问题在实际中常用于汽车导航系统以及各种应急系统等(如110报警、119火警以及医疗救护系统),这些系统一般要求计算出到出事地点的最佳路线的时间应该在 1 s~ 3 s内,在行车过程中还需要实时计算出车辆前方的行驶路线,这就决定了最短路径问题的实现应该是高效率的。其实,无论是距离最短、时间最快还是费用最低,它们的核心算法都是最短路径算法。经典的最短路径算法—— Dijkstra算法是目前多数系统解决最短路径问题采用的理论基础,只是不同系统对 Dijkstra算法采用了不同的实现方法。

据统计,目前提出的此类最短路径的算法大约有 17种。F. Benjamin Zhan等人对其中的 15种进行了测试,结果显示有 3种效果比较好,它们分别是: TQQ(graph growth with two queues)、DKA (the Dijkstra's algorithm implemented with approximate buckets)以及 DKD (the Dijkstra's algorithm implemented with double buckets),这些算法的具体内容可以参见文献[1]。其中 TQQ算法的基础是图增长理论,较适合于计算单源点到其他所有点间的最短距离;后两种算法则是基于 Dijkstra的算法,更适合于计算两点间的最短路径问题[1]。总体来说,这些算法采用的数据结构及其实现方法由于受到当时计算机硬件发展水平的限制,将空间存储问题放到了一个很重要的位置,以牺牲适当的时间效率来换取空间节省。目前,空间存储问题已不是要考虑的主要问题,因此有必要对已有的算法重新进行考虑并进行改进,可以用空间换时间来提高最短路径算法的效率。

1 经典 Dijkstra算法的主要思想

Dijkstra算法的基本思路是:假设每个点都有一对标号 (d_j, p_j),其中 d_j 是从起源点 s 到点 j 的最短路径的长度(从顶点到其本身的最短路径是零路(没有弧的路),其长度等于零);p_j 则是从 s 到 j 的最短路径中 j 点的前一点。求解从起源点 s 到点 j 的最短路径算法的基本过程如下:

1) 初始化。起源点设置为:① $d_s = 0$, p_s 为空;② 所有其他点: $d_i = \infty$, $p_i = ?$;③ 标记起源点 s,记 $k = s$,其他所有点设为未标记的。

2) 检验从所有已标记的点 k 到其直接连接的未标记的点 j 的距离,并设置:

$$d_j = \min[d_j,\ d_k + l_{kj}]$$

式中,l_{kj} 是从点 k 到 j 的直接连接距离。

3) 选取下一个点。从所有未标记的结点中,选取 d_j 中最小的一个 i:

$$d_i = \min[d_j,\ 所有未标记的点\ j]$$

点 i 就被选为最短路径中的一点,并设为已标记

收稿日期:1999-01-27。乐 阳,女,26岁,硕士,现从事 GIS空间分析研究。

* 国家杰出青年科学基金和国家"九五"重点科技攻关资助项目,编号 49525101及 96-B02-03

的。

4) 找到点 i 的前一点。从已标记的点中找到直接连接到点 i 的点 j^* ,作为前一点,设置:

$$i = j^*$$

5) 标记点 i。如果所有点已标记,则算法完全推出,否则,记 $k= i$,转到 2) 再继续。

2 已有的 Dijkstra算法的实现

从上面可以看出,在按标记法实现 Dijkstra算法的过程中,核心步骤就是从未标记的点中选择一个权值最小的弧段,即上面所述算法的 2)~5)步。这是一个循环比较的过程,如果不采用任何技巧,未标记点将以无序的形式存放在一个链表或数组中。那么要选择一个权值最小的弧段就必须把所有的点都扫描一遍,在大数据量的情况下,这无疑是一个制约计算速度的瓶颈。要解决这个问题,最有效的做法就是将这些要扫描的点按其所在边的权值进行顺序排列,这样每循环一次即可取到符合条件的点,可大大提高算法的执行效率。另外, GIS中的数据 (如道路、管网、线路等)要进行最短路径的计算,就必须首先将其按结点和边的关系抽象为图的结构,这在 GIS中称为构建网络的拓扑关系 (由于这里的计算与面无关,所以拓扑关系中只记录了线与结点的关系而无线与面的关系,是不完备的拓扑关系)。如果用一个矩阵来表示这个网络,不但所需空间巨大,而且效率会很低。下面主要就如何用一个简洁高效的结构表示网的拓扑关系以及快速搜索技术的实现进行讨论。

网络在数学和计算机领域中被抽象为图,所以其基础是图的存储表示。一般而言,无向图可以用邻接矩阵和邻接多重表来表示,而有向图则可以用邻接表和十字链表[4]表示,其优缺点的比较见表 1。

表 1 几种图的存储结构的比较

Tab. 1 The Comparsion of Several Graph for Storing Structures

名　称	实现方法	优　　点	缺　　点	时间复杂度
邻接矩阵	二维数组	1. 易判断两点间的关系 2. 容易求得顶点的度	占用空间大	$O(n^2+ m* n)$
邻接表	链表	1. 节省空间 2. 易得到顶点的出度	1. 不易判断两点间的关系 2. 不易得到顶点的入度	$O(n+ m)$或 $O(n* m)$
十字链表	链表	1. 空间要求较小 2. 易求得顶点的出度和入度	结构较复杂	同邻接表
邻接多重表	链表	1. 节省空间 2. 易判断两点间的关系	结构较复杂	同邻接表

目前,对于算法中快速搜索技术的实现,主要有桶结构法、队列法以及堆栈实现法。TQQ、DKA以及 DKD在这方面是比较典型的代表。TQQ虽然是基于图增长理论的,但是快速搜索技术同样是其算法实现的关键,它用两个 FIFO的队列实现了一个双端队列结构来支持搜索过程[1]。

DKA和 DKD是采用如图 1所示的桶结构来支持这个运算,其算法的命名也来源于此。在 DKA算法中,第 i 个桶内装有权值落在 $[b* i, (i+ 1)* b)$ 范围内的可供扫描的点,其中 b 是视网络中边的权值分布情况而定的一个常数。每一个桶用队列来维护,这样每个点有可能被多次扫描,但最多次数不会超过 b 次。最坏情况下, DKA的时间复杂度将会是 $O(m* b+ n(b+ C/b))$,其中, C 为图中边的最大权值。DKD将点按权值的范围大小分装在两个级别的桶内,高级别的桶保存权值较大的点,相应的权值较小的点都放在低级别的桶内,每次扫描都只针对低级别桶中的点。当然随着点的插入和删除,两个桶内的点是需要动态调整的。在 DKA算法中,给每个桶一定的范围以及 DKD中使用双桶,在一定程度上都是以空间换时间的做法,需要改进。

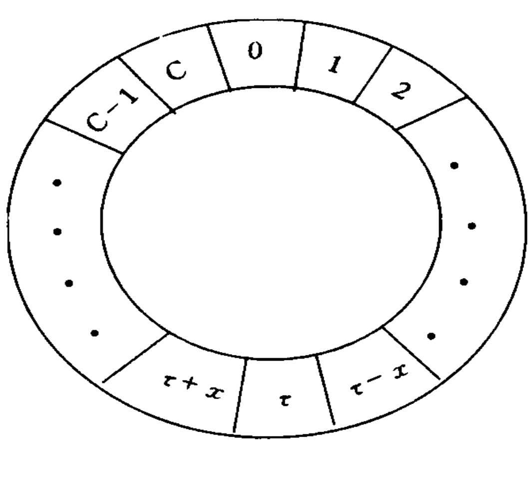

图 1 一个桶结构的示例

Fig. 1 An Example of the Bucket Data Structure

3 本文提出的 Dijkstra算法实现

3.1 网络拓扑关系的建立

上面介绍的各种图的存储结构考虑了图在理论上的各种特征,如有向、无向、带权、出度、入度等。而 GIS中的网络一般为各种道路、管网、管线等,这些网络在具有图理论中的基本特征的同时,更具有自己在实际中的一些特点。首先,在 GIS中大多数网络都是有向带权图,如道路有单双向问题,电流、水流都有方向 (如果是无向图也可归为有向图的特例),且不同的方向可能有不同的权值。更重要的一点是,根据最短路径算法的特性可以知道,顶点的出度是个重要指标,但是其入度在算法里则不必考虑。综合以上 4种存储结构的优缺点, 笔者采用了两个数组来存储网络图,一个用来存储和弧段相关的数据 (Net- Arc List),另一个则存储和顶点相关的数据 (Net- Node Index)。Net- Arc List用一个数组维护并且以以弧段起点的点号来顺序排列,同一起点的弧段可以任意排序。这个数组类似于邻接矩阵的压缩存储方式,其内容则具有邻接多重表的特点,即一条边以两顶点表示。Net- Node Index则相当于一个记录了顶点出度的索引表,通过它可以很容易地得到此顶点的出度以及与它相连的第一条弧段在弧段数组中的位置。此外,属性数据作为 GIS不可少的一部分也是必须记录的。这样,计算最佳路径所需的网络信息已经完备了。在顶点已编号的情况下,建立 Net- Arc List和 Net- Node Index两个表以及对 Net- Arc List的排序,其时间复杂度共为 $O(2n+\lg n)$,否则为 $O(m+2n+\lg n)$。这个结构所需的空间也是必要条件下最小的,记录了 m 个顶点以及 n 条边的相关信息,与邻接多重表是相同的。图 2是采用这个结构的示意图。

3.2 快速搜索技术的实现

无论何种算法,一个基本思想都是将点按权值的大小顺序排列,以节省操作时间。前面已经提到过,这两个算法都是以时间换空间的算法,所以在这里有必要讨论存储空间问题 (这部分空间的大小依赖于点的个数及其出度)。根据图中顶点和边的个数可以求出顶点的平均出度 $e=m/n$ (m 为边数, n 为顶点数),这个数值代表了图的连通程度,一般在 GIS的网络图中, $e\in[2,5]$。这样,如果当前永久标记的点为 t 个,那么,下一步需扫描点的个数就约为 $t\sim 4t$ 个。如果采用链表结构,按实际应用中的网络规模大小,所需的总存储空间一般不会超过 100 K 所以完全没有必要采用以时间换空间的做法,相反以空间换时间的做法是完全可行的。在实现这部分时,笔者采用了一个 FIFO队列,相应的操作主要是插入、排序和删除,插入和删除的时间复杂度都是 $O(1)$,所以关键问题在于选择一个合适的排序算法。一般可供选择的排序算法有快速排序、堆排序以及归并排序等,其实现的平均时间都为 $O(n\lg n)$。经过比较实验,笔者选择了快速排序法。另外, Visual C++ 提供的 run-time库也提供了现成的快速排序的函数 qsort( )可供使用。

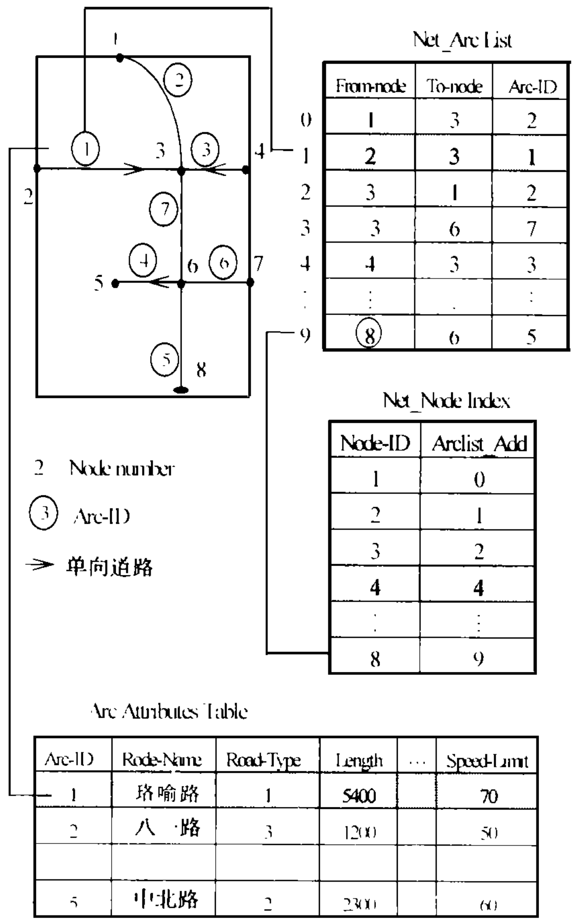

图 2 基于最佳路径计算的网络拓扑表示

Fig. 2 The Presentation of the Network Topology Used for Computing the Shortest Path

按照以上思路,笔者用 Visual C++ 实现了吉奥之星 (GeoStar)中的最佳路径模块。以北京的街道为数据 (共 6 313个结点, 9 214条弧段 (双向)),在主频为 133 硬盘为 1 G 内存为 32 M 的机器上,计算一条贯穿全城、长为 155.06 km的线路,约需 1 s~ 2 s。如图 3所示。

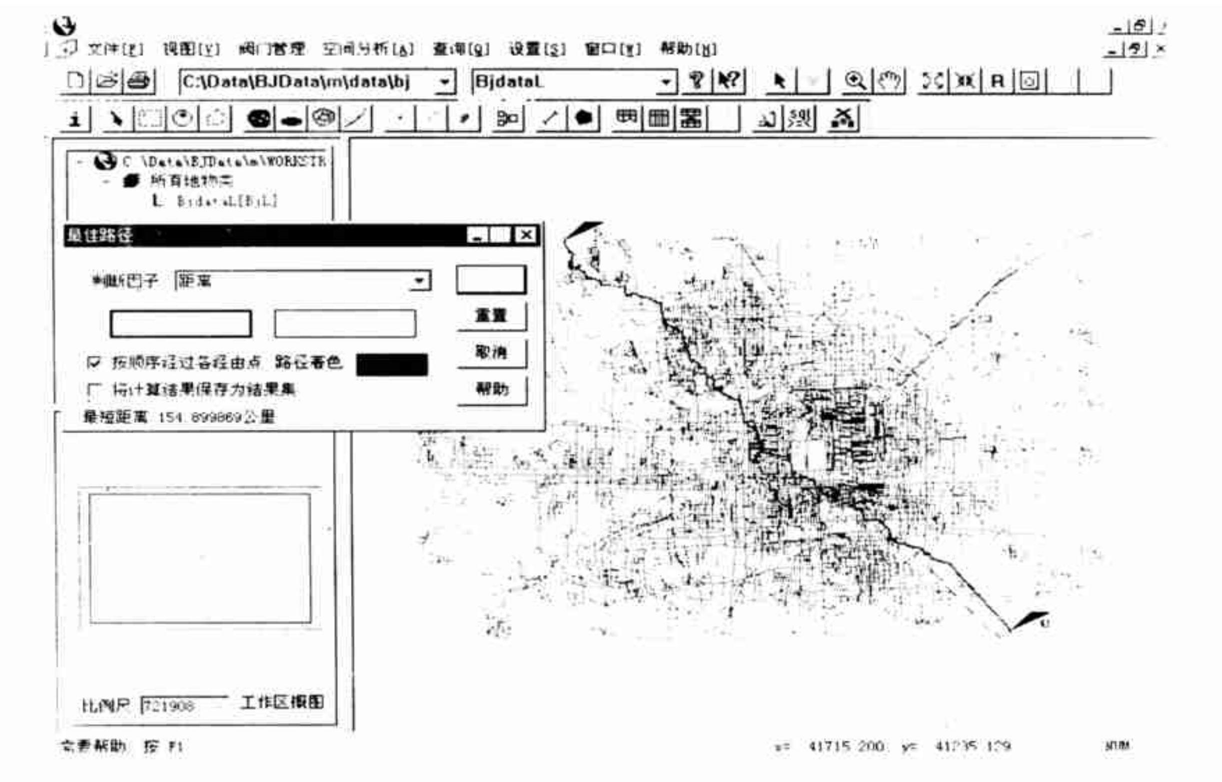

图 3 GeoStar中最佳路径实现示意图

Fig. 3 The Shortest Path in GeoStar

参 考 文 献

1 Zhan F B. Three Fastest Shortest Path Algorithms on Real Road Networks. Journal of Geographic Information and Decision Analysis, 1997, 1 (1): 69~82

2 丁跃民. GIS中实用空间算法设计的关键技术.见:地理信息系统软件工程及其相关技术高级研讨会论文集.武汉:武汉测绘科技大学出版社,1997

3 卢开澄,卢华明.图论及其应用(第二版).北京:清华大学出版社,1997

4 严蔚敏,吴伟民.数据结构.北京:清华大学出版社,1997

5 米涅卡 E.网络和图的最优化算法.李家滢,赵关旗译.北京:中国铁道出版社,1984

An Efficient Implementation of Shortest Path Algorithm Based on Dijkstra Algorithm

Yue Yang Gong Jianya

(National Laboratory for Information Engineering in Surveying, Mapping and Remote Sensing, WTUSM, 129 Luoyu Road, Wuhan, China, 430079)

Abstract With the development of geographic information science and the wide use of GIS software, more and more needs are required to the network analyses. As the key of network analyses, computing the shortest paths over a network is an important problem that scholars facus on. Start with the data structure during its computation process and combined with F. Benjamin Zhan's evaluation of a set of 15 shortest path algorithms, this paper presents an efficient method of realize the shortest path algorithm which is based on Dijkstra algorithm. Result shows that this method performs well in practice.

Key words shortest path algorithm; network analysis; GIS