基于 MapX 的局部最短路径搜索算法①

An Algorithm for Finding the Local Shortest Path Based on MapX

杨中宝　（玉溪师范学院地理系　云南玉溪　653100）
李朝艳　（玉溪师范学院图书馆　云南玉溪　653100）
吕　伟　（玉溪师范学院地理系　云南玉溪　653100）

摘要：最短路径分析是地理信息系统(GIS)网络分析的基础，拓扑关系是最短路径分析的关键。由于 MapX 不支持空间数据的拓扑结构，因此对于采用 MapX 进行二次开发的用户来说，最短路径分析就成为一个难点。为此讨论了基于 MapX 的弧段文件格式的 Dijkstra 算法，并在此基础上实现了基于 MapX 的局部最短路径搜索方法。

关键词：最短路径　邻接矩阵　拓扑关系　MapX　Dijkstra

1　引言

MapX 是 MapInfo 公司向用户提供的具有强大地图分析功能的 ActiveX 控件产品。由于 MapX 是基于 Windows 操作系统的标准控件，具有很好的易开发性和开放性，能支持绝大多数标准的可视化开发环境。因此，MapX 是应用型 GIS 二次开发的理想控件。

作为地理信息系统空间网络分析的前提和关键问题之一就是空间数据拓扑关系的建立，而 MapX 的最大不足就在于不具备拓扑关系的数据结构，空间分析能力较弱。因此基于 MapX 二次开发拓展 GIS 空间分析功能时，就必需自己构建地理空间实体之间的空间拓扑关系。

本文针对上述问题，在研究了 MapX 组件基本功能及文件组织结构的基础上，就基于 MapX 的应用型地理信息系统二次开发过程中的最短路径搜索问题作了初步的探讨。

2　MapX 的数据组织形式

2.1　地图的组织形式

拓扑空间关系是一种对空间结构进行明确定义的数学方法。GIS 中，常将空间事物抽象成点、线、面、曲面和体等几何要素，并在点、线、面之间建立拓扑关系。具有拓扑信息的矢量数据是网络分析必要的数据来源。

然而，在地图数据的组织形式上，MapX 是按图层进行组织的，相同性质的数据归为一个图层，图层中的要素称为 Feature，各 Feature 之间是独立的，不存在任何关系，更重要的是，MapX 的一个图层通常只有一种类型的地理空间对象，而一般不会把不同类型的地理空间对象置于同一图层。因此，前人为了建立点、线、面基本元素之间的空间拓扑关系，常常选择重新构建空间数据，建立自己的数据库及表结构的方法[4]。其关键在于将 MapX 中原有图层中的各弧段对象进行自动断链，从而解决弧段的相交和自相交的问题，以便于网络空间分析(如图 1)。

图 1 显示，图中断链前的弧段 1，在断链后变成了 3 条弧段 1、4、5。由此可知，弧段自动断链虽然可以解决弧段的相交和自相交问题，但是随之而来的问题是一个完整的地理空间对象将会被分割成若干个部分，从而影响地理信息系统的空间查询分析功能。

2.2　地图数据的记录形式

MapX 使用 MapInfo 的数据格式。这种数据的组织方法在形式上表现为地图—图层—Feature—Parts—Points—Point 的逐步细化过程。每一幅地图由很多图

① 教改项目：云南玉溪师范学院教改项目(基于 MapX 的地理信息系统 CAI 设计)资助

层组成，图层由 Feature（地理空间对象）组成，而 Feature 则由不同的坐标点对按顺序排列而成，其弧段的记录格式如表 1 所示。

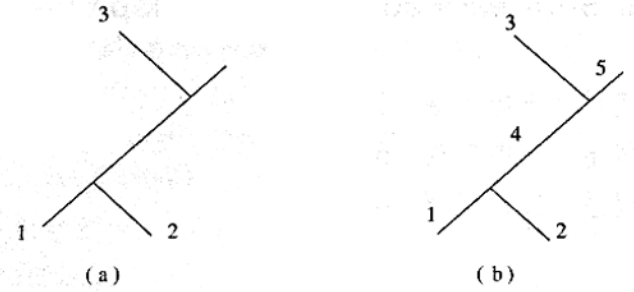

图 1 弧段断链前后对比（a 为断链前，b 为断链后）

表 1 MapInfo 数据记录格式

弧段类型码	坐标点对数目	坐标点对组合
多段线	4	起点坐标，坐标 2，坐标 3，终点坐标
直线		起点坐标，终点坐标

3 Dijkstra 算法原理与问题

求解最短路径问题的算法中，Dijkstra 算法是目前国内外一致公认的比较成功的算法。Dijkstra 算法中，网络被抽象为图论中定义的有向图或无向图，用图的遍历方法搜索最短路径。

经典 Dijkstra 算法是采用邻接结点的 Dijkstra 算法，该算法将网络结点分成 3 部分：未标记结点、临时标记结点和永久标记结点。网络中所有结点首先初始化为未标记结点，在搜索过程中凡与最短路径中的结点相连通的结点为临时标记结点，每次循环都是从临时标记结点中搜索距源点路径长度最短的结点作为永久标记结点，直至找到目标结点或者所有的结点都成为永久标记结点来结束算法。

如在一个赋权的简单连通无向图 $G=\langle V,E,W\rangle$（V：顶点集；E：边集；W：权重）中，求一源结点 a 到某结点 x 的最短路径的长度。算法思想可表述为

（1）把 V 分成两个子集起始点子集 S 和目标点子集 T，初始时，$S=\{a\}$，$T=V-S$；

（2）对于每个 $T_i(T_i\in T)$，计算 $D(T_i)$，（$D(T_i)$ 表示从 a 到 T_i 的不包含 T 中其他结点的最短路径的长度），根据 $D(T_i)$ 值找出 T 中距 a 最短的结点 x，写出 a 到 x 的最短路径的长度 D(x)；

（3）置 S 为 $S\cup\{x\}$，置 T 为 $T-\{x\}$，若 $T=\Phi$，则停止，否则再重复（2）。

在上述算法中，如果从构建的弧段数据库中直接读取数据，进行最短路径分析时，需多次遍历数据库表里的所有记录，速度相对较慢，因此直接利用邻接结点的 Dijkstra 算法无法满足实际需要。

为了能够提高 Dijkstra 算法的最短路径搜索效率，可以有两种解决途径，因为每次在临时标记结点中搜索路径最短的结点时都要遍历所有的临时标记结点。因此可以将临时标记结点进行重新组合，让每个搜索过程不必全部遍历或者较少遍历临时标记结点。另外一个解决方案是尽量减少最短路径分析过程中搜索的结点数量，从而尽快到达目标结点。本文基于第二种解决方案，并对传统 Dijkstra 方法进行了优化，提出了局部最短路径搜索方法。

4 基于 MapX 的局部最短路径搜索算法描述

在一个赋权的简单连通无向图 $G=\langle V,E,W\rangle$（V：顶点集；E：边集；W：权重）中，某一源结点 s 到某结点 t 的最短路径所必须经过的结点集合 Vst 必定是 V 的子集，即 $Vst\in V$。因此，可以通过寻找 Vst 来减少算法的邻接短阵中结点的数量，达到快速搜索最短路径的目的。

假设在网络中任意给定两点 s 和 t，显然，如果网络中存在一条边连接 s 与 t，或者网络中存在一条边链连接 s 与 t，并且组成该边链的边均在直线段 st 上，那么该条边或者边链是 s 与 t 之间的最短路径[5]。这就说明以下两点：

首先，要沿由 s 至 t 的直线段方向寻找 s 与 t 之间的最短路径，即 Vst 分布于 st 及其附近；

其次，参与计算的弧段为直线段。

据此，基于 MapX 的局部最短路径搜索算法可以描述为：

输入：选择的起止结点 s 和 t，坐标分别为 (x_s,y_s) 和 (x_t,y_t)。

输出：s 和 t 之间的最短路径及其长度。

步骤 1：获取起始点与目标点 s 和 t。

步骤 2：计算直线段 st 的长度，记为 L。

步骤 3：如果 s 与 t 之间存在一条弧段或弧段链，则为最短路径。否则转 4。

步骤 4：以 st 直线段为中线建立宽为 10 个基本单位的局部搜索矩形。

步骤 5：搜索矩形范围内的所有弧段，获取弧段在矩形范围内的所有结点，并按顺序编号。并以新的结点为依据建立临时弧段数据。

步骤 6：从 s 所在弧段开始，依次寻找与其它各弧段的交点。如果最终出现目标结点所在的弧段，转 7；否则转 8。

步骤 7：利用 Vst 建立邻接矩阵和关联矩阵及权值矩阵。并使用 Dijstra 算法进行最短路径标记并输出，算法结束。

步骤 8：以原矩形为基础，向四周扩展 10 基本单位，形成新的局部搜索矩形。转 5。

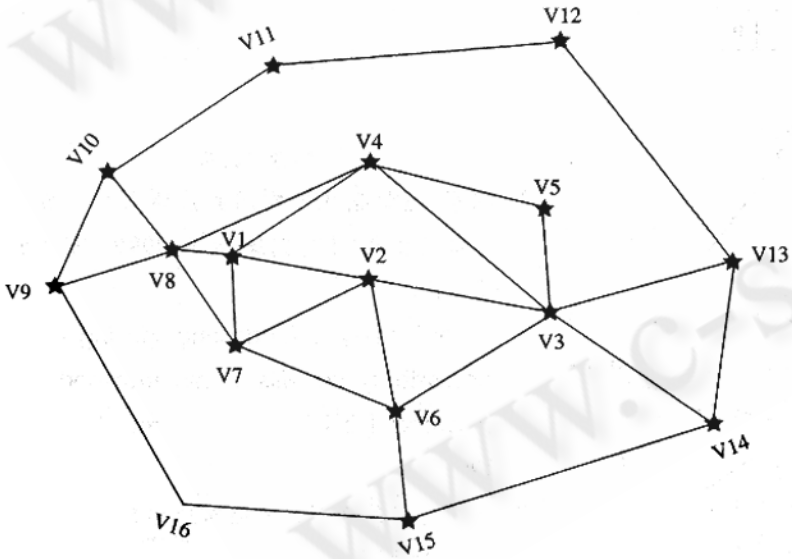

图 2　无向图

该算法是从局部开始计算并向外逐渐扩散，因此称为局部最短路径搜索 Dijkstra 算法。算法的运行的效率决定于网络图的复杂程度和各弧段之间的连通程度。连通程度越好复杂度越高，局部最短路径搜索算法的运行效率就越高，而传统的 Dijstra 算法的运行效率就越低。

5　算法分析与实例

一段时间以来，很多学者对传统的 Dijkstra 算法进行了优化与改进，并取得了很多成果，但大多数情况下都是提前把网络图中各结点之间的最短路径计算好并存储在外部数据库中，在查询时，实际上只是在数据库中进行简单的数据查询操作并把查询结果显示出来。同时，要求参与计算的空间数据具有较好的空间拓扑数据结构，并且要求参加计算的弧段必须是直线段。这就为基于 MapX 二次开发组件进行地理信息系统软件开发的用户提出了两个必须解决的问题。

首先，MapX 并不支持空间拓扑关系，其数据文件记录格式并不都是以直线段进行记录的，在很多情况下是以多段线记录的（见图 1）。

其次，现实世界的快速发展变化，使原有的各种网络图件随时都在更新数据，那么预先计算存储在数据库中的各种最短路径数据就必须重新计算更新，这就给数据库的维护带来了很大的困难。因此，如何解决这些问题就成了基于 MapX 开发地理信息系统的关键环节。

基于 MapX 的局部最短路径搜索算法是在 MapX 数据文件结构格式的基础上，以传统的 Dijkstra 算法为理论依据进行改进设计的算法。直接使用 MapX 空间数据记录中弧段对象与结点之间的组织关系，构建临时的局部邻接关联矩阵来满足局部最短路径的标记与搜索，从而较好的解决了最短路径搜索算法过程中的空间数据的拓扑关系问题。每一次计算均以上一次计算的各矩阵为基础进行扩展，从而使原计算过的路径不会重复计算。

由于局部最短路径搜索算法是以 MapX 文件为基础的临时搜索算法，它的计算结果不需要存储在数据库中，所以网络图形数据的改变不会带来数据库维护的问题，从而减少应用过程中的复杂程度。

如图 2 所示，如果求结点 V1 至 V3 的最短路径，在传统的使用邻接矩阵的的 Dijkstra 算法中，将会形成如表 2 所示的邻接矩阵。

而根据局部最短路矩搜索算法，则可以先搜索路径必须经过的顶点集合 Vst，在 Vst 中进行最短路径的标记。如图 3、表 3 所示。

表 2　全集合邻接矩阵

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16
V1	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0	0
V2		0	1	0	0	1	1	0	0	0	0	0	0	0	0	0
V3			0	1	1	1	0	0	0	0	0	0	1	1	0	0
V4				0	1	0	0	1	0	0	0	0	0	0	0	0
V5					0	0	0	0	0	0	0	0	0	0	0	0
V6						0	1	0	0	0	0	0	0	0	1	0
V7							0	1	0	0	0	0	0	0	0	0
V8								0	1	1	0	0	0	0	0	0
V9									0	1	0	0	0	0	0	1
V10										0	1	0	0	0	0	0
V11											0	1	0	0	0	0
V12												0	1	0	0	0
V13													0	1	0	0
V14														0	1	0
V15															0	1
V16																0

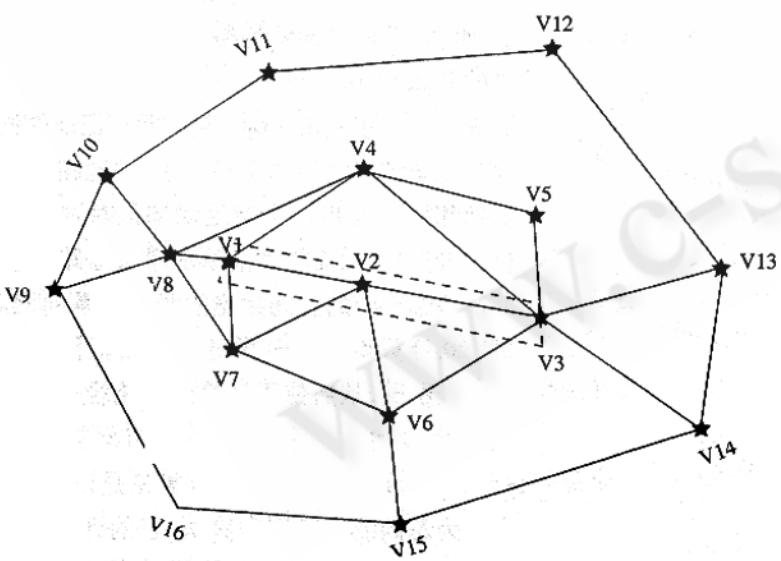

图 3　无向图及最短路径搜索矩形

从表 1 和表 2 可以看出，根据局部最短路径搜索算法可以很大程度上减少 Dijkstra 算法在最短路径搜索过程中搜索的结点数，从而达到优化算法的目的。

表 3　局部最短路径搜索矩形中的结点组成的邻接矩阵

	V1	V2	V3
V1	0	1	0
V2		0	1
V3			0

6　结束语

本文根据 MapX 开发地理信息系统的要求，在 MapX 的文件格式与传统的 Dijkstra 算法中寻找交叉点，提出的局部最短路径搜索算法比传统的 Dijkstra 算法有了较大的改进，能够较好的解决网络分析中空间数据拓扑问题，同时也在很大程度上减少了 Dijkstra 算法在搜索过程中搜索的结点数，极大地提高了算法的执行效率。

参考文献

1　陈军、赵仁亮，GIS 空间关系的基本问题与研究发展[J]，测绘学报，1999，28(2)：95～101。

2　Miller Harbey J. Measuring spacing – time accessibility benefits within transportation networks [J], Geographical Analysis, 1999.

3　严寒冰，基于 GIS 的城市道路网络最短路径算法探讨[J]，计算机学报，2000. 23(2). 210～215。

4　朱晓青、周涛、张海堂，MapInfo 中道路拓扑与最优路径的研究[J]，测绘学院学报，2001. 18(2)，133～135。

5　周培德，交通道路网中任意两点之间最短路径的快速算法[J]，计算机工程与科学，2002. 24(4). 35～37。