文章编号：1001－9081(2009)02－0406－03

一种处理障碍约束的聚类算法

王小乐，刘青宝，陆昌辉，陈文凯
(国防科学技术大学 信息系统与管理学院，长沙 410073)
(showlf@163.com)

摘 要：根据障碍约束空间聚类问题的特点，利用图论的相关知识，提出了一种分阶段的基于图的聚类的算法。首先，通过最小生成树聚类算法，在不考虑障碍约束的情况下对空间对象进行聚类；然后，引入障碍物对上一步的聚类结果进行分割；最后，根据被障碍物分割后形成的各个类之间的障碍距离，将距离较近的两个类合并，形成最终的聚类结果。最后通过实验验证了算法的效果，而且输入参数少，时间复杂度低。

关键词：聚类；障碍约束；最小生成树；障碍距离
中图分类号：TP391　　文献标志码：A

Clustering arithmetic with obstacle constraints

WANG Xiao-le, LIU Qing-bao, LU Chang-hui, CHEN Wen-kai
(College of Information System and Management, National University of Defense Technology, Changsha Hunan 410073, China)

Abstract: According to the characteristics of clustering with obstacle constraints, using the knowledge of graph theory, a multi-step Arithmetic was proposed. Firstly, it clustered the objects without obstacles by minimum spanning tree clustering method. Then it took obstacles to divide the generated clusters. Lastly, it merged the clusters whose obstruct distance was little enough. The algorithm need only one parameter, it is of good performance and can find clusters with arbitrary shapes and varying densities. At last its effectiveness was demonstrated through experiment.

Key words: clustering; obstacle constraints; minimum spanning tree; obstruct distance

0 引言

传统的聚类分析[1]是在没有任何条件约束的情况下对实体集进行分组，而在实际问题中往往会有领域知识和用户要求对聚类过程进行指导和限制，这样就提出了基于约束的聚类分析[2]（Constraint Clustering, CC）。在对空间对象进行聚类时会面临这样的问题：有些对象之间的欧氏距离很小，但是由于河流或高速公路等障碍物的隔离，使得两个看起来很近的对象不能聚到同一类，如图 1所示。

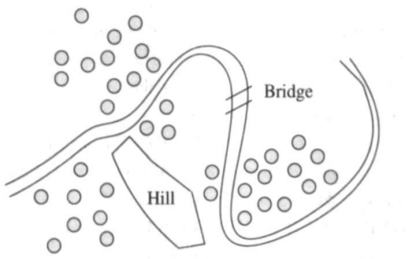

图 1 空间对象的分布

基于障碍约束的聚类分析具有现实应用价值，因为在现实问题中存在河流、山脉等物理障碍，它们的存在会影响聚类结果的合理性[3]。基于障碍约束的聚类分析应用广泛，如公共设施及商业场所的选址，空间对象的分组管理以及图像分割等。文献[4]中首先提出基于障碍约束的聚类问题（Clustering with Obstacles Entities, COE）以来，在聚类的研究领域开辟了一个新的研究方向，受到广泛关注。

1 相关研究

在解决障碍约束的空间数据聚类问题时，虽然障碍物的形状各异但是都可以通过多边形对其进行模拟，多边形方便在计算机中存储。目前障碍约束的聚类问题在国内外都受到了广泛研究，国外比较著名的算法有：COD_CLARANS[5]、AUTOCLUST+[6]、DBCluC[7]和 DBRS+[8]。国内的研究者大都是在原有的算法上结合遗传算法、粒子群优化和蚁群优化等启发式算法对障碍约束的聚类算法进行优化[3,9－12]。

COD_CLARANS算法的核心思想[9]：在障碍物约束的条件下，计算任意两点的距离，无障碍的情况下两个对象的距离为直接欧氏距离，但存在障碍时就要计算两个对象之间的障碍距离[4]，它将采样技术和 PAM相结合，通过迭代的方法来完成在障碍物约束下的聚类问题。AUTOCLUST+算法最大的好处是不需要用户输入参数，算法效率较低，没有说清楚如何处理障碍物多边形。DBCluC算法采用障碍线的方法来保证可视空间的不变，降低了障碍对象的处理时间，但输入参数难以确定。

障碍约束的聚类问题可以描述为：用 o_1, o_2, …, o_n表示空间分布的 n个对象，用 p_1, p_2, …, p_m表示 m个障碍物，该问题就是将 n个对象，在有障碍物约束的条件下进行聚类。其中对障碍物都可以用多边形进行模拟。

本文根据障碍约束空间聚类问题的特点，应用图论的相关知识，提出了一种处理障碍约束的基于图的聚类算法（Graph-based Clustering arithmetic with OBstacle constraints, GCOB）。该方法首先在不考虑障碍物的情况下通过最小生

收稿日期：2008－08－19；修回日期：2008－10－10。　基金项目：国家自然科学基金资助项目（70771110）。

作者简介：王小乐（1983－），男，陕西西安人，硕士，主要研究方向：数据挖掘、智能决策；刘青宝（1967－），男，江西永新人，副教授，博士，主要研究方向：数据仓库、数据挖掘；陆昌辉（1976－），男，湖南邵阳人，讲师，博士，主要研究方向：数据仓库、数据挖掘；陈文凯（1984－），男，河北保定人，硕士研究生，主要研究方向：地理信息系统、数据仓库。

成树聚类算法[13]对数据对象进行聚类；然后引入障碍物，以障碍物的边为分割线，将同一类的最小生成树分割成子树；然后根据障碍环境下的最近邻关系合并子树，得出最终聚类。

2　算法的原理

2.1　基本概念

定义 1　根据对象之间的距离矩阵构建无向加权图 $G(V, E)$（其中 V表示节点的集合，每个节点对应一个对象，E表示边的集合）然后再计算出图的最小生成树，该最小生成树称作对象集的最小生成树，用 MST_{objSet}表示，其中 $objSet$为对象集。虽然可能最小生成树并不唯一，但是对本文中的算法并无影响，所以文中仅用对象集的任意一个最小生成树。

$$MST_{objSet} = \{(V, E^{MST}) E^{MST} = e_1^{MST}, e_2^{MST}, \cdots, e_{n-1}^{MST}\}$$

其中 E^{MST}是最小生成树中的边集。

定义 2　最小生成树中每条边 e的两个端点分别记为 v_l^e和 v_r^e表示左端点和右端点（此处左右并非左边右边，仅用以区别两个端点），与边 e共用顶点 v_l^e的边的集合称为 e的左邻域边集，记为 $N(v_l^e)$；同样与 e共用顶点 v_r^e的边的集合称为 e的右邻域边集，记为 $N(v_r^e)$。

推论　如果 e为叶子节点那么其左邻域边集和右邻域边集有且只有一个为空。

定义 3　不一致边。边 e只要满足下列条件，则为不一致边：

$$\frac{|e|}{m} > \lambda$$

$$m = \min \begin{cases} \dfrac{\sum |e_l|}{|N(v_l^e)|}, & e_l \in N(v_l^e), \text{ if } |N(v_l^e)| \neq 0 \\ \dfrac{\sum |e_r|}{|N(v_r^e)|}, & e_r \in N(v_r^e), \text{ if } |N(v_r^e)| \neq 0 \end{cases}$$

其中 $|e|$为边 e的边长，$|N(v_l^e)|$和 $|N(v_r^e)|$分别为 e左邻域和右邻域边个数，它们至少有一个不为 0。

如图 2所示当 $\lambda = 4$时边 a为不一致边。

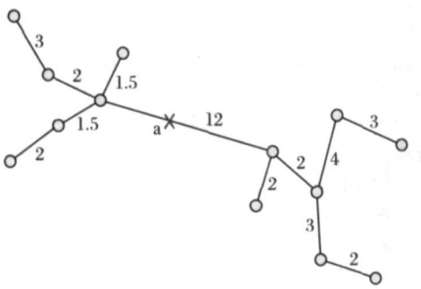

图 2　不一致边示意图

定义 4　可视。存在障碍物的情况下，空间两个对象 O_1，O_2的连线不穿过任何障碍物则称这两个对象可视，记为：$O_1 \leftrightarrow O_2$。

定义 5　障碍距离。两个对象在障碍环境下的最短距离。当两个对象可视时为直接欧氏距离，否则如图 3所示为绕过障碍物的最短路径，可用 Dijkstra算法求解。则障碍距离记为：$OD(A, B)$，其中 A，B为两个对象。则：

$$OD(A, B) = \begin{cases} d(A, B), & A \leftrightarrow B \\ d'(A, B), & \text{其他} \end{cases}$$

其中 $d(A, B)$为 A，B之间的欧氏距离，$d'(A, B)$为绕过障碍物的最短距离。

2.2　算法原理

GCOB算法的策略：首先，在无障碍环境下通过最小生成树聚类算法，对空间数据对象进行聚类；其次是在环境中加入障碍物，障碍物将代表同一簇的树又分割为若干子树；然后计算出被分割树的子树之间的最短距离，用该距离和其对应的节点的邻接边的平均长度进行比较，如果小于参数 λ则合并；否则不合并。

第一步，在不考虑障碍物的情况下对空间数据点进行聚类。首先根据空间对象之间的欧氏距离建立图，求出图的最小生成树，最后根据上面的定义找出最小生成树中的不一致边，断开这些边形成一个森林，则森林中的每棵树对应一个聚类。

第二步，引入障碍物再次分割树。本文中采用多边形建模[4]的方法对障碍物进行建模，这样方便在计算机中存储和处理。最小生成树聚类结束后引入障碍物，当障碍物多边形有一条边和每个聚类对应的子树中的边相交时，则断开该边将该子树再分割成两个子树。

第三步，求出分割后形成的每两个子树之间的最小障碍距离，以该距离为边长，如果该边为不一致边，则将其断开，否则合并该边连接的两棵子树。计算最小障碍距离的方法为：假设要计算 T_1、T_2之间的最小障碍距离，那么先求出子树 T_1中每个点到子树 T_2中每个点的障碍距离，这些距离中最小的那个就是 T_1、T_2的最小障碍距离。在障碍环境下求子树 T_1中数据点到子树 T_2中数据点的距离时，先由数据点和障碍物多边形构建为一个图，图的顶点为数据点和障碍物多边形的顶点，图的边为这些点中任意两个可视点之间的连线，如图 3所示，然后通过 Dijkstra算法计算两个数据点之间的最短路径，路径长度为这两个点之间的障碍距离。

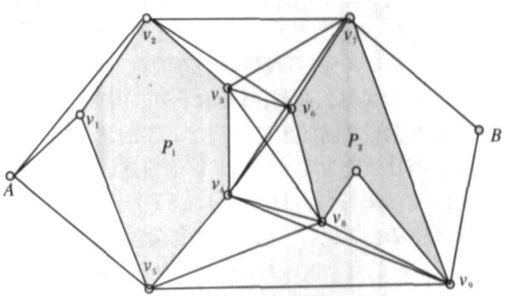

图 3　求解 AB之间最短路径

3　GCOB算法的实现

3.1　算法的描述

该算法分为三个基本步骤，算法描述如下：

输入：数据集 D，障碍物多边形集合 P，参数 λ

输出：数据类标识 clusterID

```
1)  MST(D)                                   //求最小生成树
2)  FOR each edge e in E^MST_D
3)      IF(|e|/m > λ)
4)          Remove e from MST_D              //删除不一致边
5)      End IF
6)  End FOR
    //此时最小生成树就变成一个森林 F={T1, T2, …}，后面就引入障碍物分割聚类 定义一个队列 Queue
7)  FOR each obstacle p in P
8)      FOR each edge ep of p
9)          IF(any edge e in F cross ep)
10)             Remove e                     //删除 e后形成的两棵子树为 Ti, Tj
11)             InQueue(<Ti, Tj, p>)
                //保存入队列以便后面检查是否能合并。
12)         End IF
13)     End FOR
14)     End FOR
15)     While(Length of queue<>0){
```

```
16)        OuQueue(< Tm, Tn, p>);
17)    Min_distance=MaxDouble
                    //设开始时 Tm, Tn的最小障碍距离为无穷大
18)    FOR v1 in Tm
19)    FOR v2 in Tn
20)    Building_graph( v1, v2, p);                    //构建图
21)    通过 Dikstra算法,计算图中 v1和 v2之间的最短路径为 s
22)    IF( s< Min_distance)
23)    Min_distance= s
24)    END IF
25)    END FOR
26)    END FOR
27)    以 Min_distance为边长构建连接 Tm, Tn的边 eb
28)    IF( |eb|/m <= λ)
29)    JOIN( Tm, Tn)                  //用边 eb将 Tm, Tn合并; }
30)    最终形成的每棵子树就形成一个类。
```

3.2 复杂度分析

GCOB是一种基于图论的算法,用到了图论中的知识,算法的时间复杂度可以分为三个模块计算。第一步,在无障碍环境下,基于最小生成树上聚类,也就是逐一考虑最小生成数的边,所以时间复杂度为 $O(n-1)$;而后是对最小生成树中的每条边进行判断裁剪,设每条边的邻边有 $m(m \ll n)$条,那么该步骤的时间复杂度不超过 $O(mn)$;第二步,引入障碍物,设障碍个数为 k,平均有 r条边,则时间复杂度不超过 $O(kr)$;第三步是合并,这一步主要是将障碍物分割开的两个子树合并,站在障碍物的角度考虑,假设障碍物等的每条边都断开一个树那么共有 kr对关系入队列,断开的子树的顶点数为 v,计算子树中每两对顶点之间的距离时构建的图的顶点数为 g,则第三步的时间复杂度为 $O(krvg)$。综上所述,总时间复杂度为 $O(n-1+kr+krvg)$,其中 $v=n/c$(c为聚类的数目),而 $g \ll v$,$kr \ll v$,那么简化后算法的时间复杂度为 $O(n)$,与算法 COD_CLARANS的复杂度[9]($O(n^k)$一般情况障碍物个数都大于 2)相比具有较大优势。

4 仿真实验

实验环境:Intel Pentium(R) 2.8 GHz双核,内存 1 GB,windowsXP SP2,Microsoft Visual studio 2005。仿真实验程序用C#语言编写,对平面二维坐标点进行障碍环境下的聚类。

仿真程序的界面如图 4所示,在右边空白区域通过点击鼠标左键创建数据点,通过点击右键创建多边形障碍物。数据集,是通过上述操所创建的人工数据集 ds该数据集有 5个障碍物,500个数据点。如图 5所示,数据点之间的连线为最小生成树的边。

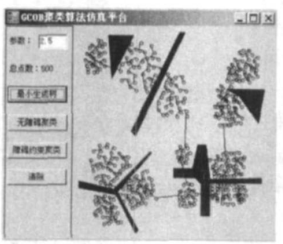

图 4 实验程序界面及数据集 ds

在不考虑障碍物的情况下,对 ds进行的聚类,结果如图 5所示,数据点旁的数字为该点所属的类标号,这种情况下共得到 6个聚类。

引入障碍物后进行聚类的结果如图 6所示,原来的类 1被分称三个子类 1、2、5,原来的类 2也被分割为类 6、9、11,原来的类 5被分割为簇 3、4,原来的类 3没有被障碍物完全分开所以还是同一个类标号 7,同样原来的类 4和 6也没分割,对应新的类标号分别为 8和 10。

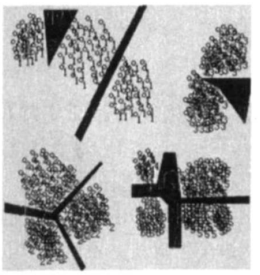

图 5 不考虑障碍的聚类结果

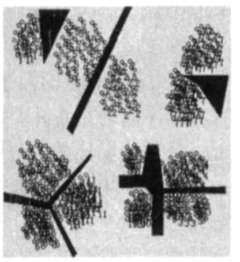

图 6 考虑障碍约束的聚类结果

5 结语

在对空间数据对象进行聚类时常常受到各种障碍物的约束,例如超市或者银行选择地址,要方便用户,但是因为河流,公园或者一些大单位围墙的阻挡,使得原本很近的用户要绕道,怎样找到最佳地址就是一个障碍约束的空间聚类问题。本文提出了一种解决该问题的基于图论聚类算法 GCOD,理论分析和实验验证都证明了算法的可行性。算法可以获取任意形状任意密度的类,要求输入的参数只有一个。因为最小生成树是在计算相似度矩阵的同时计算的,而实际聚类过程中的时间复杂度很低。

参考文献:

[1] HAN JIAWEI, KAMBER M. 数据挖掘:概念与技术[M]. 2版. 范明,孟小峰,译. 北京:机械工业出版社,2007.

[2] MOHAMED A, EL-ZAWAWY, MOHAMED E. Clustering with obstacles in spatial databases[C]// Proceedings of the 1st IEEE International Symposium on Bioinformatics and Biomedical Engineering. Washington, DC: IEEE Computer Society, 2002: 6.

[3] 周丽华,王丽珍,陈克平. 带障碍的空间分级聚类算法[J]. 计算机科学, 2006, 33(5): 182-187.

[4] HOU J. Clustering with obstacle entities[D]. Canada: University of Waterloo, 1999.

[5] TUNG A K H, HOU J, HAN JIAWEI. Spatial clustering in the presense of obstacles[C]// Proceeding of International Conference on Data Engineering. New York: IEEE, 2001: 359-367.

[6] ESTIVILL-CASTRO V, LEE I. AUTOCLUST+: Automatic clustering of point data sets in the presence of obstacles[C]// Temporal, Spatial, and Spatio-Temporal Data Mining. Berlin Heidelberg: Springer, 2001: 14.

[7] ZAIANE O R, LEE C H. DBCluC: Clustering spatial data when facing physical constraints[EB/OL]. [2008-06-16]. http://www.cs.ualberta.ca/~zaiane/postscript/icdm02-3.pdf.

[8] WANG X N, ROSTOKER C, HAMILTON H J. Density based spatial clustering in the presence of obstacles and facilitators[C]// Proceeding of the 8th European International Conference on Principles and Practice of Knowledge Discovery in Databases. New York: Springer-Verlag, 2004: 446-458.

(下转第 411页)

检测引擎模块。本文对数据挖掘算法中的关联规则算法 Aprior算法进行适当改进,使其挖掘出正常行为模式。异常检测引擎模块在 Snort程序中使用 C语言来实现。

4)利用关联规则算法 Aprior算法对分离出来的新入侵数据进行关联分析,生成关联规则并转换成适合 Snort检测的新规则。这部分在 ACID中使用 PHP语言来实现。

4　模拟实验及结果

为了对该模型的检测效果进行评估,本文进行了一些相关的模拟实验。本实验分为两个部分,其一是对于改进效率的测试,其二是对于规则扩充能力的测试。

实验方法:利用 Profile工具来进行实验。VC++6.0提供的 Profile工具可以帮助程序员发现程序运行的瓶颈,找到耗时所在。

实验数据:由于在局域网环境下,每一时间段的网络流量不同,来自局域网的实际网络数据不能准确评价系统改进前后的效率,因此,本文采用美国麻省理工林肯实验室(MIT Lincoln Lab)提供的 1999 DARPA Intrusion Detection Evaluation Data Sets[7]来进行对比实验,以说明改进效果。

用网络分析软件 Ethereal打开数据集 outside. tcpdump共有 233 428条记录,本文就该数据集的前 1 000、5 000和 10 000条记录对改进前后的系统进行实验,给出改进前后系统的总耗时和 _DETECT函数调用次数的对比,结果如表 1所示。

表 1　改进前后检测效率对照表

数据记录/条	改进后总耗时/ms	改进前总耗时/ms	改进后调用 _DETECT函数次数	改进前调用 _DETECT函数次数
1 000	24 085.952	24 265.795	788	986
5 000	94 590.281	95 201.430	3 442	5 138
10 000	326 387.279	329 169.852	5 016	10 409

从上面的实验结果可以看出,改进后的系统总耗时明显减少,而且调用 Snort原有检测函数 _DETECT的次数也明显减少。这就说明改进后的系统减轻了 Snort原有检测引擎的负担,提高了 Snort系统的检测效率。

对新规则生成模块进行实验的过程中发现,网络数据的特定形式不适应 Aprior算法直接进行挖掘。因为,源 IP和目的 IP属于两个不同的项,但值可能相同,源端口和目的端口也存在这样的问题,直接对这些数据进行挖掘,将产生大量冗余的频繁项集和强关联规则。因此,必须对数据进行预先处理,本文的做法是对每条记录的源 IP和源端口加入标识符 s,目的 IP和目的端口加上标识符 d,然后,利用关联分析算法 Aprior算法对处理后的数据进行挖掘,设定最小支持度和最小置信度分别为 10%和 98%,改进系统在对 50条新入侵数据进行挖掘之后,自动生成了 3条规则并存入 Snort规则库 rules下的 new. rules中,如下所示:

alert TCP 172.16.117.52 1051 -> 207.46.130.139 80
alert TCP 172.16.117.52 1052 -> 207.46.130.139 80
alert TCP 172.16.117.52 1049 -> 207.46.130.139 80

最后,重新运行系统,发现 Snort在初始化时已经将这三条规则加入到规则链中,并且对测试数据中匹配到该规则的数据进行报警,相应的报警信息可以在报警文件 alert. ids中找到,而且在 ACID分析控制台也可以查看到这些报警信息,图 4中给出了其中的部分报警信息:

图 4　新规则生效后的报警信息

上面的测试表明:该系统可以生成新的入侵规则并扩充 Snort的规则库,因而改进后的 Snort系统对新类型的入侵行为具有了一定的防御能力。

5　结语

本文基于数据挖掘理论,提出了一种 Snort系统的改进模型,并通过 Snort的插件机制实现了该模型。经过采用美国麻省理工林肯实验室(MIT Lincoln Lab)提供的测试数据集进行改进前后的对比实验,结果表明该模型不仅提高了 Snort系统的检测效率,并且可以不断扩充 Snort的规则库,使其能够检测到新的入侵行为。本文对数据挖掘技术在入侵检测系统中的应用进行了初步尝试,更进一步的研究工作如挖掘项对于检测效果的影响、最小支持度和最小置信度对模式库精确度的影响等还有待深入。

参考文献:

[1] LEE W, STOLFO S J. A framework for constructing features and models for intrusion detection systems[J]. ACM Transactions on Information and System Security, 2000, 3(4): 227-261.

[2] HAN JIAWEI, KAMBER M. 数据挖掘概念与技术[M]. 范明, 孟小峰, 译. 北京:机械工业出版社, 2001.

[3] 陈耿, 朱玉全, 孙志辉, 等. 一种基于异常检测的关联模式挖掘模型[J]. 计算机工程与应用, 2004, 40(12): 158-198.

[4] 宋世杰, 胡华平, 胡笑蕾, 等. 数据挖掘技术在网络型异常入侵检测系统中的应用[J]. 计算机应用, 2003, 23(12): 20-23.

[5] 贾世国, 张昌城等. 基于数据挖掘的网络入侵检测系统的设计与实现[J]. 计算机工程与应用, 2008, 44(14): 134-137.

[6] 宋世杰, 胡华平, 胡笑蕾, 等. 数据挖掘技术在网络型误用入侵检测系统中的应用[J]. 计算机工程, 2004, 30(16): 126-127.

[7] MIT Lincoln Lab. Intrusion detection evaluation data sets[EB/OL]. [2008-06-10]. http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1999data.html.

(上接第 408页)

[9] 杨扬, 孙志伟, 赵政. 一种处理障碍约束的基于密度的空间聚类算法[J]. 计算机应用, 2007, 27(7): 1688-1692

[10] 薛丽霞, 汪林林, 王佐成, 等. 基于 Voronoi图的有障碍物空间聚类[J]. 计算机科学, 2007, 34(2): 189-192

[11] 严馨, 周丽华, 陈克平. 一种改进的带障碍的基于密度和网格的聚类算法[J]. 计算机应用, 2005, 25(8): 1818-1822

[12] ZHANG XUEPING, WANG JIAYAO, FAN ZHONG-SHAN, et al. Spatial clustering with obstacles constraints using ant colony and particle swarm optimization[C]// Proceeding of PAKDD 2007 Workshops. Berlin: Springer, 2007: 344-356.

[13] GRYGORASH O, ZHOU YAN, JORGENSEN Z. Minimum spanning tree based clustering algorithms[C]// Proceedings of 18th IEEE International Conference on ICTAI. Washington: IEEE Computer Society, 2006: 73-81.