DOI:10.19344/j. cnki. issn1671-5276.2023.04.037

基于改进的力导向模型的图布局算法

屈会雪[1],董玉龙[2]

(1. 南京机电职业技术学院 信息学院,江苏 南京 211135;
2. 南瑞集团有限公司,江苏 南京 211106)

摘　要:力导向模型布局算法是一种常用的图可视化算法,在网络设备拓扑图可视化、社交网络关系图可视化、分布式链路追踪可视化等领域都有广泛应用。针对如上领域,以网络拓扑图为应用基础,在传统的力导向模型下,提出一种渐进式的力导向布局算法。在算法输入前对数据进行分类处理,每一个类看作一个节点,进行位置迭代调整,以减少原始算法在每次迭代过程中计算各个节点与相邻节点之间作用力的次数,在类的内部依然使用力导向布局算法。通过实验验证:改进的算法在性能上有明显提高。

关键词:力导向;网络拓扑;分类;布局;模型

中图分类号:TP391　　**文献标志码**:B　　**文章编号**:1671-5276(2023)04-0148-04

Graph Layout Algorithm Based on Improved Force-directed Model

QU Huixue[1], DONG Yulong[2]

(1. School of Computer, Nanjing Vocational Institute of Mechatronic Technoloyg, Nanjing 211135, China;
2. NARI Group Corporation, Nanjing 211106, China)

Abstract: The force-directed model layout algorithm, a commonly used graph visualization algorithm,is widely used in the fields of network device topology visualization, social network relationship graph visualization,anddistributed tracking analysis visualization. By the application of network topology, a progressive force-directed layout algorithm is proposed based on the traditional force-oriented model. Proir to the input of the algorithm, data is classified and analyzed, and each class is regarded as a node and the position is adjusted iteratively so as to reduce the number of times by the original algorithm which calculates the force between each node and adjacent nodes in each iteration, while the force-directed layout algorithm is still used inside the class.The experiments verify the obvious improvement in the performance of the proposed algorithm.

Keywords: force-directed; network topology; classify; layout; model

0　引言

随着 Web 技术的飞速发展,特别是进入大数据时代,越来越多的系统、服务需要通过可视化的手段展现数据,从中挖掘用户更关注的有价值的信息,其中一种可视化应用为图数据可视化,比如社交网络中人物的关系图、网络空间中设备拓扑图等[1]。图数据可视化方法可以分为两类:手动布局算法和自动布局算法[2]。手动干预的布局算法思想是首先将图中的节点随机分布到画布中,然后依据图的复杂程度移动节点位置使图展示得更加可观、清晰,该算法的缺点是在节点数量大的时候不具有可行性。自动布局算法则无需人工干预,在布局的初期就通过计算得到每一个节点的对应位置,使节点在画布的位置分布均匀美观,不影响用户对信息的直观阅读。

网络拓扑可视化在网络管理、网络流量分析、网络模型研究和网络安全风险评估中占有举足轻重的作用,以节点代表网络中的设备,边代表网络连接,将整个网络特点呈现给用户。

目前,自动化布局算法使用最多的是力导向布局算法,该算法的优点是思路简单、图中的边重叠较少,能够直观地看到每一个节点与其他节点的连接情况。但是在节点较多的情况下,画布中的线条排布凌乱,并且该算法的可扩展性和动态性较差,在每次节点变化后要重新计算其他所有节点的位置,性能代价较高[3]。

本文提出一种改进的力导向布局算法,它首先将具有相同特征的点进行分类,然后再使用渐进式的力导向算法,可以减少在节点变化时产生的性能开销,并且可视化效果也更加直观。

1　相关工作

1.1　常用的网络拓扑图布局算法

目前主要的布局算法有:1) 树形布局算法;2) 网格化布局算法;3) 力导向布局算法;4) 启发式布局算法。算法 1) 适用于节点数量较少,没有环状的网络模型;算法 2) 思想简单,缺点是不能真正描述节点之间的相关

第一作者简介:屈会雪(1989—),女,山东枣庄人,讲师,硕士,研究方向为数据处理,528898206@ qq.com。

性;算法3)可以满足大多数拓扑图的可视化要求,空间的利用率高,在合适的参数选择下能够达到满意的效果[4]。早在1963年,TUTTE W T[5]最早提出了力导向模型的布局算法,该算法的理论依据是重心表示理论。之后COHEN R F等[6]于1984年、FRUCHTERMAN T M J等[7]于1991年分别基于spring layout理论改进了该算法,这也是当前最流行的FR布局模型算法。该算法利用物理原理,通过赋予节点物理性质(如质量、电荷量),在布局过程中,具有物理性质的定点会通过相连接的边彼此相互作用,通过多次迭代使整个系统达到一个平衡状态,从而完成整个拓扑图的绘制。力导向布局算法利用物理学中的能量守恒原理,为拓扑图可视化提供了一种新的思路,但是也存在明显的不足之处:性能问题。

1.2　分类算法

分类算法就是确定对象属于哪个预定义的目标类。分类问题是一个普遍存在的问题,有许多不同的应用。例如:根据电子邮件的标题和内容检查出垃圾邮件,根据核磁共振扫描的结果区分肿瘤是恶性还是良性,根据星系的形状对它们进行分类。

常见的分类算法有决策树算法、基于规则的分类算法、神经网络、支持向量机和朴素贝叶斯分类法[8]。K-邻近算法[9]是一种用于分类和回归的非参数统计方法,主要思想是:如果一个样本特征空间中K个最相邻的样本中大多数属于某一类别,则该样本也属于这一类别,并具有这个类别上样本特征。该方法在确定分类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别。K-邻近方法在类别决策时,只与极少量的相邻样本有关。由于K-邻近方法主要靠周围有限的邻近样本,而不是靠判别类域的方法来确定所属类别,因此对于类域的交叉或重叠较多的待分样本集来说,K-邻近方法较其他方法更为适合。

1.3　问题描述和相关定义

定义1:每一个无向图可以表示为$G(V,E)$,其中V是n个节点的集合$\{v_1,v_2,v_3,\cdots,v_n\}$,每一个节点$v_i$的坐标表示为$(xi,\ yi)$;$E$是$m$条边的集合$\{e_1,e_2,e_3,\cdots,e_m\}$,其中每一条边$el=e_{ij}$表示节点$i$和节点$j$之间存在一条边。

分类的目标就是将图中的节点的集合V分成k个类$C_1,\ C_2,\cdots,C_k$,满足式(1)。

$$\begin{cases}\bigcup_{i=1}^{k} C_i = V; C_i \neq \varphi; \\ C_i \cap C_j = \varphi; (i,j=1,2,\cdots,k; i \neq j);\end{cases} \tag{1}$$

定义2:令$k_{\max}$表示分类数目k的上限。算法的终止条件为$1< k< k_{\max}$。

定义3:节点的度表示为与该节点相邻节点的个数,用$D(v_i)$表示,满足式(2)。

$$D(v_i) = \sum_{k=1}^{n} e_{ik} \tag{2}$$

由以上给出的定义得到本文关于改进的力导向模型的拓扑图布局算法描述如下:

1)计算图中每个节点的度$D(v_i)$,选取其中度最大的前k个节点,$v_1,\ v_2,\cdots,\ v_k$;

2)将步骤1)中的k个节点均匀分布到可视化区域;

3)使用K-邻近算法将节点分类,在每个类型内部使用力导向算法进行布局。

一个典型的基于力导向模型的可视化例子如图1所示。

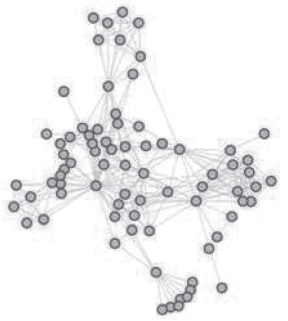

图1　一个典型的力导向模型的布局算法效果图

2　算法的实现

上节给出了改进的力导向模型网络拓扑图布局算法思想和步骤,本节详细介绍每一步中原理和实现细节。

2.1　选取拓扑图中度最大的前k个节点

根据定义1,一个有5个节点、6条边的无向图$G(V,E)$,其中$V=\{v_1,v_2,v_3,v_4,v_5\}$,$E=\{e_{12},e_{14},e_{34},e_{35},e_{25},e_{23}\}$,用邻接矩阵表示如下:

$$\begin{bmatrix}1&1&0&1&0\\1&1&1&0&1\\0&1&1&1&1\\1&0&1&1&0\\0&1&1&0&1\end{bmatrix}。$$

矩阵中的元素只能取1或者0,当节点v_i和v_j之间存在一条边时,矩阵中第i行第j列以及第j行第i列的元素取1;当节点v_i和v_j之间不存在一条边时,矩阵中第i行第j列以及第j行第i列的元素取0。

利用拓扑图的矩阵表示和节点的度(定义3),可以得到算法1,获取度数最大的前k个节点。

算法1:getTopk。

输入:拓扑图的邻接矩阵表示P

输出:节点度数最大的k个节点

初始化一个数组a存放每个节点的度数,长度为节点总数,数组中每一个元素为0

```
FOR i = 0 TO 矩阵行数
    FOR j = i TO 矩阵的列数
        IF 矩阵元素 Pij == 1 THEN
            a[i] ++
        END IF
    END FOR
END FOR
sort(a)
RETURN a[0,…,k]
```

算法1中,通过遍历邻接矩阵中的每一个元素,累加计算每一个节点的度,sort函数对数组a进行降序排序,算法返回数组a中的前k个元素。

2.2　均匀分布 k 个节点

可将可视区看做一个有边界的二维平面,为了简便,设其为矩形平面块,坐标原点为矩形的左下顶点,x 轴与 y 轴分别向右和向上延伸,像素值的大小为其坐标刻度。均匀分布 k 个节点的算法如算法 2 所示。

算法 2:layoutTopk。

输入:k 个待布局的节点

输出:无

初始化一个二维矩形平面,坐标原点位于左下角,水平 x 轴和垂直 y 轴分别向右和向上延伸

$n = k$ 的平方根

将 x 轴可视区平分为 n 等分

将 y 轴可视区平分为 n 等分

FOR $i = 0$ TO k

　　将第 i 个节点的坐标设置为当前第 i 个分割平面的中心坐标

END FOR

算法 2 的思想是将二维平面块划分为 m 个面积相等的矩形平面块,然后将 k 个节点的坐标依次设置为 m 个平面块的中心点。效果如图 2 所示。

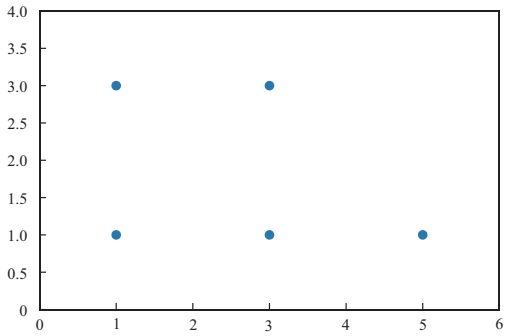

图 2　5 个节点均匀分布到画布上

2.3　使用 K-邻近算法将节点分类

1) 节点之间的距离

K-邻近算法的一个重要步骤是计算各个节点到中心点的距离,一般采用欧式距离计算得到,但是在无向拓扑图的布局算法中,节点之间没有预设的坐标点,只能通过其他特征进行计算每个节点之间的距离。

定义每个节点的特征表示为 $X_i = \{x_1, x_2, \cdots, x_k\}$,其中 x_i 表示该节点和第 i 个分类中心点是否存在一条边,如果存在,则 x_i 为 1,否则为 0。本论文中提出节点距离表达式如下:

$$\mathrm{Dist}(X,Y) = \frac{\sum_{i=1}^{k}(x_i - y_i)^2}{\sum_{i=1}^{k} y_i^2} \tag{3}$$

式中 $\mathrm{Dist}(X,Y)$ 表示节点 X 距离中心点 Y 的距离,可描述为节点 X 与节点 Y 的欧式距离除以节点 Y 的“长度”,既考虑了相邻节点的相关性,又可以均匀地把所有节点分配到预设的 k 个类中。

2) K-邻近算法

算法 3:kNearestNeighbors。

输入:图的邻接矩阵表示 V

输出:分类后的节点结合 S

FOR $i = 1$ TO n DO

　　FOR $j = 1$ TO k DO

　　　　利用式(3)计算节点 i 与第 k 个中心点的距离

　　END FOR

　　选出与节点 i 距离最近的中心点 j,将节点 i 归入第 j 类

END FOR

算法 3 使用 K-邻近算法,通过计算每个节点与中心点的距离,将所有节点归为 k 类,为后续拓扑图排序做准备。

2.4　在每个分类中使用引力-斥力算法对节点进行可视化布局

在基于引力-斥力模型的布局算法中,如何计算节点之间的作用力是重中之重,为了便于计算,本文将节点之间的作用力简化为公式(4)。

$$F = k(x - x_0) \tag{4}$$

式中 x_0 代表两个节点之间距离的阈值,当两个节点之间的距离超过 x_0 时,它们之间的作用力表现为引力,当距离小于 x_0 时,作用力表现为斥力,并且作用力的大小变化呈线性相关。

算法 4 描述了引力-斥力模型布局算法的过程。

算法 4:forceDirectedLayout。

输入:图的邻接矩阵 $\boldsymbol{V}$

输出:无

FOR $i = 1$ TO n DO

　　利用公式(4)计算两个相邻节点之间的作用力 F

　　IF $F>0$ THEN

　　缩小两节点之间的距离

END IF

IF $F<0$ THEN

　　增加两节点之间的距离

END FOR

算法 4 描述了如何使用力导向布局算法对图中的节点进行调整和布局。

3　实验评估

为了验证本文提出的算法有效性和性能,使用 JavaScript 语言在 Eclipse 平台上实现了相应的算法,并且在 Chrome 浏览器中运行该算法,通过分析 Chrome 浏览器开发者工具 Performance 面板中的性能指标数据,对比传统引力-斥力模型布局算法和本文提出改进后的算法,验证本文提出算法的高效性。实验的输入是给定的网络拓扑图 JSON 数据,输出是拓扑图经过布局后的展示结果。

实验环境为 CPU(Intel Core Duo Processor,1.73 GHz)+RAM(4 GB)+Windows10+Eclipse3.4+Chrome98。

3.1　实验设置

本实验采用控制变量法,分别给出在同一数据下传统力导向布局(FR 模型)算法和改进的力导向布局算法的执行时间;同一数据在改进的力导向布局算法中,不同的参数(作用力系数 k、迭代次数 n)算法执行时间;不同数据集在传统力导向布局算法和改进的力导向布局算法中

的执行时间。具体包括:

1)输入 JSON 文件大小分别为 3 MB、4 MB、5 MB、6 MB、7 MB、15 MB,其中包含 70、80、90、100、150、350 个节点,200、240、300、320、400、500 条边,观察两种算法的执行时间(改进的力导向布局算法参数设置为 $k=10$,$n=100$);

2)输入 JSON 文件大小为 3 MB,其中包含 70 个节点,250 条边,通过改变参数观察程序执行结果。

3.2　评价指标

本文所提出的算法主要是改进多节点下 FR 模型算法性能问题,并且判断一个拓扑图布局是否合理的一个重要指标是图中的节点是否分布均匀以及拓扑图中的边尽量不相交。因此提出本实验的评价指标分为两部分,第一个为算法的执行时间以及算法运行时在 Chrome 浏览器开发者工具中 Performance 数据;第二个为指标 R,如式(5)所示。

$$R=N+\rho\times n \tag{5}$$

式中:N 表示拓扑图中边与边交点的个数;ρ 表示单位面积内节点个数的方差,本实验将画布分为 10 等分,每一等分为一个单位面积;n 为节点个数。R 指标越小表明图的可视化效果越好。在 Chrome 开发者工具中,通过浏览器渲染拓扑图动画的 FPS(每秒帧数)和渲染时间占整个网页呈现时间的比重判断可视化效果。

3.3　实验结果与分析

针对 3.1 节 1)中给出的实验设置,使用传统力导向模型(FR 模型)算法,其运行时间如表 1 所示。可视化效果如图 3 所示。

表 1　FR 模型布局算法执行时间

实验编号	输入数据	执行时间/s	R
1	70 节点、200 条边	1.05	35
2	80 节点、240 条边	1.40	35
3	90 节点、300 条边	2.10	56
4	100 节点、320 条边	1.95	57
5	250 节点、400 条边	2.32	90
6	350 节点、500 条边	7.12	110

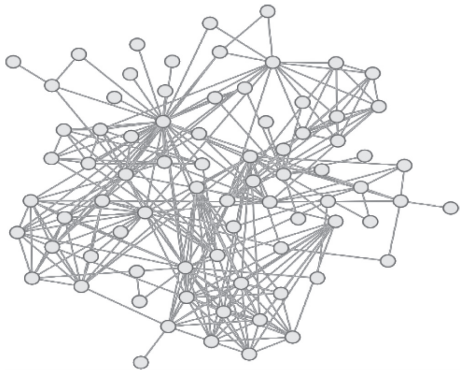

图 3　FR 模型布局算法

本文提出的改进力导向布局算法执行结果如表 2 所示,布局效果如图 4 所示。

表 2　改进的力导向布局算法执行时间

实验编号	输入数据	执行时间/s	R
1	70 节点、200 条边	1.95	12
2	80 节点、240 条边	1.82	15
3	90 节点、300 条边	1.85	30
4	100 节点、320 条边	1.90	32
5	250 节点、400 条边	2.12	46
6	350 节点、500 条边	4.10	56

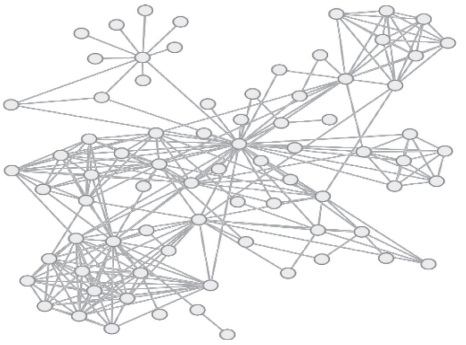

图 4　改进的力导向布局算法

比较表 1 和表 2 数据的结果发现:在节点数据较少的情况下,FR 模型布局算法更优于本文提出的算法,这是由于 FR 模型布局算法没有进行分类预处理;随着节点数量增多,本文提出的改进力导向布局算法在运行时间上优于 FR 模型的布局算法。

Chrome 浏览器开发者工具的性能分析图如图 5 和图 6所示。

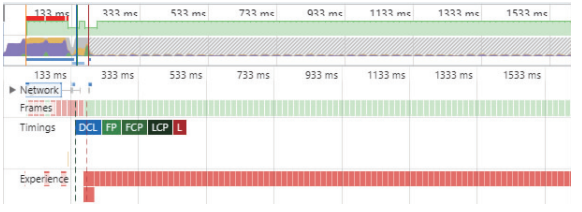

图 5　Chrome 开发工具中 FR 模型布局算法性能指标

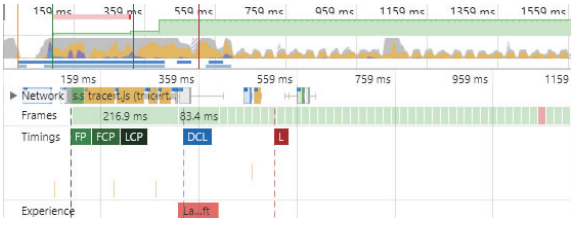

图 6　Chrome 开发工具中改进的力导向布局算法性能指标

图 5 和图 6 中分别展示了两种布局算法在 Chrome 浏览器渲染拓扑图的过程中截取前 1.5 s 的性能指标,其中 Frames 表示渲染动画在当前的 FPS 数,其中绿色代表 60 帧/s 以上,红色表示渲染效果影响用户体验,在 60 帧/s 以下;图 5 和图 6 中顶部彩色折线图代表算法运行过程中计

(下转第 168 页)

综上所述,通过参数变量分析,提高了自动控制器模型的精准度及稳定度,最终构建了高效、高精准度的多自由度机器人抓取末端振动自动控制器,改进了传统方法中存在的不足,具有较好的应用性能。

5　结语

本文提出了多自由度机器人抓取末端振动自动控制研究方法,建立机器人动力学模型,构建机械臂运动规划模型,分析运动过程参量,选取重要参量作为优化目标,构建自动控制器,最终完成多自由度机器人的自动控制。解决了机器人双臂不稳、指令控制度低、控制延迟长的问题。

参考文献:

[1] 孔凡国,谭水生. 两臂二指智能魔方机器人执行控制系统设计[J]. 机床与液压,2020,48(23):40-44.
[2] 庞海. 基于DTN算法的采摘机器人控制系统可行性研究[J]. 农机化研究,2022,44(5):119-123.
[3] 郑文昊,贾英民. 具有状态约束与输入饱和的全向移动机器人自适应跟踪控制[J]. 工程科学学报,2019,41(9):1176-1186.
[4] 蒋沅,公成龙,吕科,等. 基于自适应模糊补偿的不确定性机器人CNF控制[J]. 振动与冲击,2020,39(8):106-111.
[5] 冯浩,殷晨波,曹东辉,等. 挖掘机器人伺服系统神经网络滑模控制[J]. 液压与气动,2021,45(10):104-110.
[6] 张建华,许晓林,刘璇,等. 双臂协调机器人相对动力学建模[J]. 机械工程学报,2019,55(3):34-42.
[7] 李京文,韩行. 基于系统动力学的工业机器人产业发展路径研究[J]. 东北大学学报(社会科学版),2019,21(2):132-138.
[8] 韩江,汪鹏,董方方,等. 基于Udwadia-Kalaba方法的平面冗余并联机器人建模与轨迹跟踪控制[J]. 应用数学和力学,2020,41(11):1183-1196.
[9] 胡章芳,程亮,张杰,等. 多约束条件下基于改进遗传算法的移动机器人路径规划[J]. 重庆邮电大学学报(自然科学版),2021,33(6):999-1006.

收稿日期:2022-07-01

(上接第151页)

算时间Scripting(黄色部分)、布局时间Rendering(紫色部分)、绘制时间Painting(绿色部分)、千统消耗System(灰色部分)占比情况,其中Rendering(紫色部分)和Painting(绿色部分)是与算法相关的,这两部分占比越小代表算法的性能越好(本刊黑白印刷,相关疑问咨询作者)。由于传统力导向布局算法要计算每一个节点的作用力,因此在算法开始时,上述Rendering和Painting部分占比情况对比,图5中的Rendering(紫色部分)和Painting(绿色部分)明显大于图6。

针对3.1节2)中给出的实验设置,实验参数和结果如表3所示。

表3　不同参数下算法的执行时间

实验编号	参数	执行时间/s	R
1	$k=5,n=50$	2.01	15
2	$k=5,n=100$	2.80	17
3	$k=10,n=50$	2.50	20
4	$k=7,n=100$	1.90	22
5	$k=20,n=100$	2.12	40

通过实验表明,不同的参数对本文提出的算法有较大影响,经过多次实验证明,将作用力参数k设置为式(6),算法有较好的性能和布局表现。

$$k=\sqrt{S/|V|} \tag{6}$$

式中:S表示矩形画布的面积;$|V|$表示拓扑图中节点的个数。

4　结语

本文提出了一种高效的大量节点的网络拓扑图布局算法。在传统力导向模型布局的基础上,对图中的节点进行分类预处理,创新地提出了节点之间的距离度量办法,利用K-邻近分类算法将节点归类,在每一个类中并行地使用FR模型布局算法,从而提高算法程序的性能。

然而本文所提出的方法还存在不足地方:若分类中节点数目不均衡导致最后布局节点分布不均匀,可视化效果不好。未来会进一步将工作重心放到处理数量巨大并且分布不均匀的问题上。

参考文献:

[1] 刘召朝,张丹,周琛,等. 基于改进粒子群算法的多分支电缆自动布线技术[J]. 机械制造与自动化,2021,50(1):177-179.
[2] 程远,严伟,李晓明. 基于斥力-张力模型的网络拓扑图布局算法[J]. 计算机工程,2004,30(3):104-105,188.
[3] 水超,陈涛,李慧,等. 基于力导向模型的网络图自动布局算法综述[J]. 计算机工程与科学,2015,37(3):457-465.
[4] 陈昌娜,王雅娟,晏平. 网络拓扑布局技术研究[J]. 信息通信,2018,31(10):73-76.
[5] TUTTE W T. How to draw a graph[J]. Proceedings of the London Mathematical Society,1963,(1):743-767.
[6] COHEN R F,BATTISTA G D I, TAMASSIA R,et al. A heuristic for graph drawing[J]. Congressus numerantium, 1984, 24(3):149-160.
[7] FRUCHTERMAN T M J,REINGOLD E M. Graph drawing by force-directed placement[J]. Software:Practice and Experience, 1991,21(11):1129-1164.
[8] 刘红岩,陈剑,陈国青. 数据挖掘中的数据分类算法综述[J]. 清华大学学报(自然科学版),2002,42(6):727-730.
[9] 平雪良,徐荣礼,孔俊,等. 基于空间划分的海量数据K邻近新算法[J]. 华南理工大学学报(自然科学版),2007,35(5):65-69.

收稿日期:2022-02-10