基于蚁群聚类算法的优化与改进①

林金灼, 叶东毅

(福州大学 数学与计算机科学学院, 福州 350108)

摘 要: 传统的蚁群聚类算法将聚类数据的每一维属性都等同看待, 而在实际的应用中各维属性对聚类的贡献率不一, 具有主次之分, 若将所有属性赋予相同的权重, 将对聚类的效果造成影响. 为了克服这个缺陷, 本文将主成份分析(PCA)方法引入到蚁群聚类当中, 利用 PCA 计算属性的贡献率并以此构建属性的权重. 在此基础上, 结合一个新的初始化策略, 提出了一种属性带权的改进蚁群聚类算法. 通过对多个 UCI 数据集的测试, 验证了本算法的有效性. 实验结果表明, 合理的权重分配能够有效的提高蚁群聚类的质量.

关键词: 蚁群聚类算法; PCA; 贡献率; 属性带权

Optimization and Improvement Based on Ant Colony Clustering Algorithm

LIN Jin-Zhuo, YE Dong-Yi

(Department of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China)

Abstract: The traditional ant colony clustering algorithm treats all features of data set equally. But in practice, the contribution rate of attributes is different from each other. Therefore, giving all features the same weight will eventually affect the quality of clustering. To overcome the defect, the method of principal components analysis is introduced into the ant colony clustering algorithm to calculate the contribution rates of attributes and to construct the weights of attributes. On this basis, combined with a new initialization strategy, an improved ant colony algorithm with weighted attributes is proposed in this paper. The experiments on several UCI data sets validated the effectiveness of the proposed algorithm. The results show that reasonable weight distribution can effectively improve the quality of clustering.

Key words: ant colony clustering algorithm; PCA; contribution rate; weighted attribute

1 引言

聚类是机器学习和数据挖掘领域的重要研究内容之一, 由于其不需要任何先验知识, 又被称为无监督学习. 聚类的基本原则是根据数据间的不同特征对数据集进行分簇, 发现数据中隐含的模式, 使得同在一个簇的数据相似度尽量大, 不在同一簇的数据相似度尽量小, 最终将有限的无标号数据集划分为几个确定的簇. 经过长时间的发展, 主要的聚类算法大致分为以下几种: 基于划分的方法、基于层次的方法、基于密度的方法、基于网格的方法以及基于模型的方法[1,2].

Deneubourg 等人最早把基于群体智能思想的蚁群算法引入到聚类和分类问题中[3], 并提出了一种基本模型(basic model, 简称 BM)用来解释蚂蚁堆积尸体形成墓穴的行为. Lumer 和 Faieta 扩展了 BM 模型, 给出了数据对象的相似性度量表达式, 设计了用于数据聚类的 LF 算法[4-6]. 但是 BM 模型和 LF 算法在处理聚类问题时, 存在着一些比较难以解决的问题: 一方面, 要形成高质量的聚类需要耗费很长的时间, 这在一些注重实时处理的应用中比较不可取; 另一方面, BM 和 LF 算法中对参数设置具有敏感性, 特别是一些关键参数的设定, 非常依赖于使用者的经验, 使得聚类缺少鲁棒性, 聚类的效果受到影响. 徐晓华等人于 2007 年提出一种新的人工蚂蚁聚类模型, 即蚂蚁睡眠模型(ants sleeping model, 简称 ASM)和在此模型基础上的一个自适应的蚂蚁聚类算法(ant clustering based on cellular automata, 简称 A^4C)[7,8]. 这种算法不同于 LF

① 基金项目:国家自然科学基金(71231003);福建省自然科学基金(2012J01262)
收稿时间:2013-05-05;收到修改稿时间:2013-05-29

算法中的 BM 模型, 相比 BM 和 LF 算法取得了更好的聚类效果, 同时聚类的效率也大大改善.

然而, A^4C 算法采用的是欧几里德距离来度量数据间的差异, 对数据间的每一维属性都等同看待. 在实际的应用中, 各个属性的作用不同, 有的属性甚至不起作用, 形成冗余, 这势必会对聚类的最终结果造成影响. 针对此缺点, 本文采用了一种属性带权的蚁群聚类算法, 其中利用主成分分析(PCA)计算属性的贡献度并以此作为加权的依据. 同时, 鉴于 A^4C 算法运行初期聚类簇的形成缓慢, 本文采取一种新的处理策略来取代算法早期数据的随机投点策略, 有效加快了聚类的速度.

本文的安排如下: 第一节简单介绍传统的自适应蚁群聚类算法(A^4C)的一些基本概念和算法思想; 第二节主要介绍本文提出的一种改进的蚁群聚类算法 IA^4C; 第三节进行了算法的实验比较分析; 最后一节给出研究结论.

2 基本A^4C算法[7,8]

由于本文提出的改进的自适应蚁群聚类算法是基于徐晓华等人提出的 A^4C 算法, 为了更方便描述本文的工作, 先简略介绍一下 A^4C 算法, 详细内容参见[7].

A^4C 算法基于细胞自动机理论, 模拟蚂蚁因为安全需求而产生聚类的行为, 其中用一只人工蚂蚁来代表一条数据. 人工蚂蚁遵循给定的激活概率函数和聚类规则不停的寻找合适的位置, 从而使蚂蚁群体动态自组织地形成聚类, 其主要模型和计算过程如下:

2.1 蚂蚁睡眠模型

蚂蚁睡眠模型启发于自然界中蚂蚁的分巢居住行为, 即各方面习性都比较相似的蚂蚁会自发聚集在一起, 同时会排斥那些习性相差比较大的蚂蚁, 久而久之, 在蚂蚁居住环境中就会形成一个个的巢. 巢穴内的蚂蚁习性比较接近, 巢穴间的蚂蚁习性相差比较大. 在蚂蚁睡眠模型中, 用一个适应值函数来判别蚂蚁之间的相似与否. 蚂蚁有两种状态: 活跃和睡眠, 两种状态之间以一定的概率互相转化. 如果蚂蚁处于睡眠状态但适应值比较低的情况下, 它就会觉得不够安全, 从而以一个比较高的概率转成活跃状态, 然后去寻找一个适应值比较高的位置, 一旦该蚂蚁找到这样的一个位置就会再度以一定的概率转换成睡眠状态.

2.2 蚂蚁活动空间

蚂蚁的活动空间 $G=[0\cdots w(n)-1]*[0\cdots h(n)-1]$是一个二维网格, 网格里面的每一个小格最多容纳一只蚂蚁. 网格空间的大小取决于待聚类的数据的多少. 此网格空间不是通常意义下的长方形, 其上边界连接着下边界, 左边界连接着右边界, 因而是拓扑等价于球面的网格. 空间大小的设置如下:

$$w(n)=2(\lfloor\sqrt{n}\rfloor+1),h(n)=2(\lfloor\sqrt{n}\rfloor+1) \tag{1}$$

其中 $w(n)$代表网格空间的宽度, $h(n)$代表网格空间的高度, n 代表待聚类的数据量大小.

2.3 蚂蚁的邻域 N

蚂蚁的邻域定义如下:

$$N(agent_i)=N(x_i,y_i)=\{(x \bmod w(n), y \bmod h(n)) \mid |x-x_i|\le s_x, |y-y_i|\le s_y\} \tag{2}$$

其中 $agent_i$ 表示第 i 只蚂蚁, 位于(x_i, y_i), S_x 和 S_y 分别代表该蚂蚁水平方向和竖直方向的视野.

常见的邻域有八邻域, 如图 1 所示, 此时 S_x 和 S_y 都为 1.

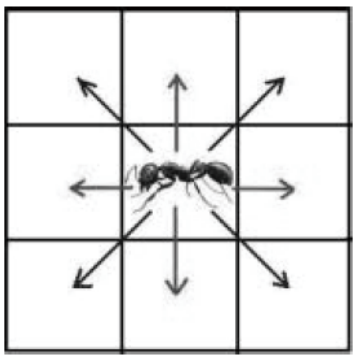


图 1 八邻域示例

2.4 数据间差异度的定义

数据间的差异度量采用欧几里德距离, 即

$d_{ij}=d(agent_i, agent_j)=d(data_i, data_j)=\|data_i-data_j\|_2$

2.5 人工蚂蚁的适应值定义

$$f(agent_i)=\max\left\{0,\frac{1}{(2s_x+1)\times(2s_y+1)}\sum_{agent_j\in N(agent_i)}\left(1-\frac{d(agent_i,agent_j)}{a}\right)\right\} \tag{3}$$

$$a=\frac{1}{n*(n-1)}\sum_{i=1}^{n}\sum_{j=1}^{n}d(agent_i,agent_j) \tag{4}$$

$f(agent_i)$代表第 i 只人工蚂蚁的适应值, α 为群体

相似系数, 其值对聚类中心的个数和聚类的质量具有重要的影响. α 的值随着算法迭代过程自适应的更新.

2.6 蚂蚁的激活概率函数

$$p_a(agent_i)=\frac{\beta^{\lambda}}{\beta^{\lambda}+f(agent_i)^{\lambda}} \quad (5)$$

这里的 β 是蚂蚁活跃适应值的阈值. λ 是一个参数. 在 A^{4}C 算法中 β 取值 0.1. λ 的值随着迭代自适应的更新.

2.7 蚂蚁移动策略

(1) 随机法: 在蚂蚁的视野范围内随机选择一个空闲的位置.

(2) 贪婪法: 以一定的概率选择蚂蚁视野范围内适应值最高的一个空闲位置.

2.8 聚类规则

(1) 如果蚂蚁处于睡眠状态, 则该蚂蚁的类号跟其邻域内类的个数最多的那个类一样.

(2) 如果蚂蚁处于活跃状态, 则该蚂蚁的类保持不变.

3 改进的蚁群聚类算法(IA^{4}C)

3.1 特征带权

传统的蚁群算法进行聚类分析时总是假设构成模式矢量的特征是独立而且无冗余的, 并且将模式矢量的各维特征看成是同等重要的. 然而在实际运用中, 各维属性对分类的贡献率往往并不相同, 有的甚至是冗余的, 不起作用[9]. 针对此缺陷, 本文提出了一种特征加权的蚁群聚类算法. 在该算法中, 考虑了特征对分类的贡献率, 有效的改善了聚类的质量.

考虑到待聚类数据的特征可能很多, 而且有些特征可能对聚类不起作用, 所以如果能够过滤掉数据中的冗余属性, 将数据降维到一个比较低维的空间来处理可能会改善聚类的质量. 鉴于此, 本文将主成分分析方法引入到蚁群聚类算法当中.

主成分分析(PCA)是将多个变量通过线性变换以选出较少个数重要变量的一种多元统计分析方法, 又称主分量分析. PCA 是最简单的以特征量分析多元统计分布的方法, 其结果可以理解为对原数据中的方差做出解释: 哪一个方向上的数据值对方差的影响最大? 换而言之, PCA 提供了一种降低数据维度的有效办法, 如果分析者在原数据中除掉最小的特征值所对应的成分, 那么所得的低维度数据必定是最优化的, 即这样降低维度必定是失去讯息最少的方法. 通过对数据进行主成分分析之后, 可以有效的去除数据之间的相关性和冗余属性[10].

设 $X=(x_1, x_2, \cdots, x_p)$是一个 p 维的向量, PCA 就是对其做如下的线性变换:

$$y_1=a_{11}x_1+a_{12}x_2+\cdots+a_{1p}x_p$$

$$y_2=a_{21}x_1+a_{22}x_2+\ldots+a_{2p}x_p$$

$$\ldots$$

$$y_p=a_{p1}x_1+a_{p2}x_2+\ldots+a_{pp}x_p$$

其中 $y_1, y_2, \cdots y_p$ 分别称为原变量的第一主成分, 第二主成分, …第 p 主成分.

协方差 $Cov(y_i, y_j)=u_i^T\Sigma u_j$, 设协方差的特征根为 $\lambda_1\geqslant\lambda_2\geqslant\cdots\geqslant\lambda_p\geqslant 0$.

定义 $\lambda_k\Big/\sum_{i=1}^{p}\lambda_i$ 为主分量 y_k 的贡献率, $\sum_{i=1}^{m}\lambda_i\Big/\sum_{i=1}^{p}\lambda_i$ 为主分量 $y_1, y_2, \cdots, y_m$ 的累积贡献率. 为了达到降维的目的, 一般只要取前几个主分量即可, 在本文改进的 IA^{4}C 算法当中, 取累积贡献率达到 90%的前几个主分量.

标准的 A^{4}C 算法采用了基于欧式距离的相似度度量方式, 其假设每一个属性在聚类过程中具有同等重要性. 实际上, 各个属性在聚类中所取的作用不一. 因此, 本文改进了数据相似度的计算方式:

$d_{ij}^{w}=\sqrt{\sum_{k=1}^{m}w_k(x_{ik}-x_{jk})^2}$, $w=(w_1,w_2,\ldots w_m)$是与属性相对应的一个权重矢量. w_i 为该分量的贡献率, 刻画了第 i 维属性在聚类过程中的重要程度. 例如, 采用主成分分析得到了新的 m 个属性来刻画原本的数据, 然后取 $\lambda_k\Big/\sum_{i=1}^{p}\lambda_i$ 作为第 k 个新属性的权重.

在 IA^{4}C 算法中, 采用了 PCA 计算属性的贡献率并以此构建属性的权重, 有效的避免了属性权重学习所需的计算时间, 同时也满足了贡献率越大的属性在聚类过程中取到更大作用的特性, 有效的改善了聚类的质量.

3.2 算法初期的投点问题

A^{4}C 算法初期是将数据点采用随机投点方式投到定义的二维网格当中, 使数据点均匀分布在蚂蚁的活动空间中, 如图 2 所示.

本文采用的是一种新的初始化策略: 将 PCA 处理过的数据的前两维(贡献率最高的两维)对应二维网格进行投影, 如图 3 所示. 该初始化方法使得算法初始

阶段，各数据与其周围环境中的其他数据已经具有比较高的相似性，有助于提高算法收敛的速度和聚类的精度[11].

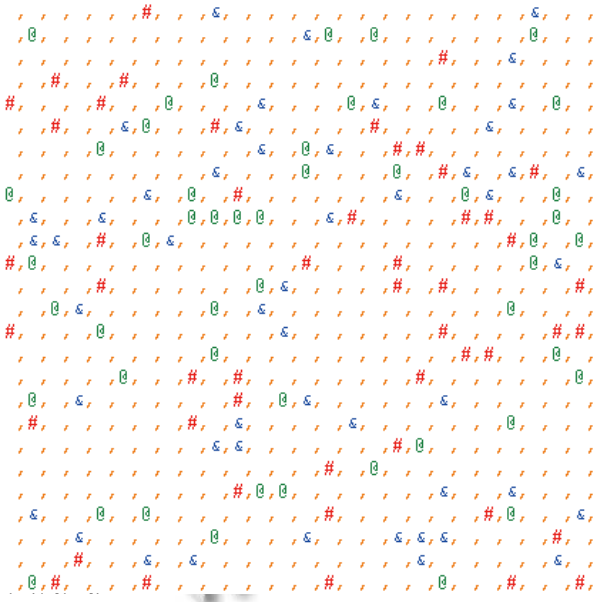


图 2　随机投点方式

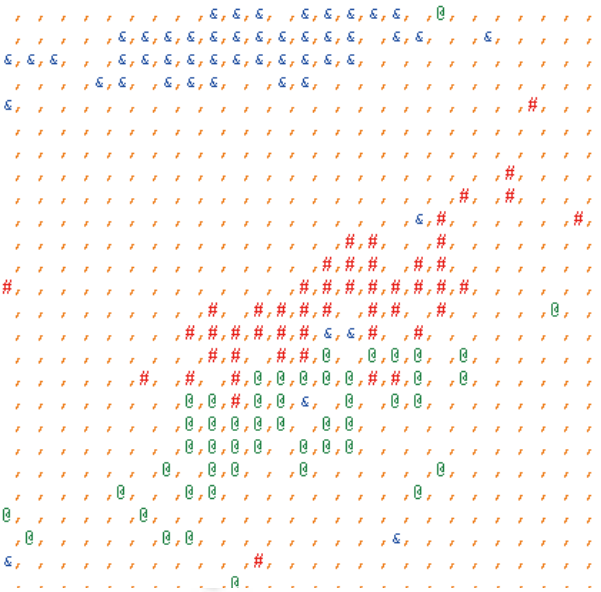


图 3　新的投点方式

3.3 蚂蚁的视野

标准的 A^{4}C 算法蚂蚁的视野取决于 $N(agent_i)$里面的 S_x 和 S_y 的值，一般情况下选择 S_x=S_y=1(8 邻域). 从 $f(agent_i)$的定义可以看到它的取值不仅取决于邻域内数据的相似性还跟邻域内数据的密度有关. 邻域如果取值较大，即使其内的数据非常相似但数目较少的话，$f(agent_i)$的值也会相对比较小，不利于生成小型簇；邻域如果取值较小会导致蚂蚁的视野受限，不利于算法后期一些相似的小型簇进行融合. 为了克服此缺陷，本算法将蚂蚁的视野与适应值函数里面蚂蚁的邻域概念区别开来，适应值函数 $f(agent_i)$邻域从始至终都按 8 邻域进行处理，而蚂蚁的视野则随着算法迭代过程逐渐增加.

经过实验发现：算法初期采用比较小的视野能有效加快聚类簇的形成，而到了算法后期一个相对比较大的视野则能够促进一些相似的小型簇的融合，提高聚类的质量.

3.4 适应值函数

A^{4}C 算法中蚂蚁的适应值函数定义如下:

$$f(agent_i)=\max\left\{0,\frac{1}{(2s_x+1)\times(2s_y+1)}\sum_{agent_j\in N(agent_i)}\left(1-\frac{d(agent_i,agent_j)}{a}\right)\right\}$$

适应值函数中的群体相似系数 α 对聚类的质量和簇的个数具有比较重要的影响，α 取值过大可能会造成粘簇，α 取值过小会阻碍簇的形成. 因而在改进的 IA^{4}C 算法中，α 初始值定义为待聚类的数据的距离平均值，见公式(4). 而 α 的更新方式如下:

$$a=\begin{cases}a*0.99\\ a*1.01 \quad if\ \bar{f}<\beta\end{cases} \tag{3}$$

$\bar{f}$ 为蚂蚁的平均适应值，β 为蚂蚁活跃适应值的阈值，α 每隔 Δt 时间更新一次.

3.5 改进的蚁群聚类算法(IA^{4}C)步骤

(1) 初始化参数，用 PCA 方法对数据进行预处理;

(2) 将经 PCA 处理过的数据的前两维对应二维网格进行投影，每只人工蚂蚁代表着一条数据,并置每只蚂蚁的类号为其标号;

(3) While (not termination)

(4) 对每只蚂蚁计算其适应值和活跃概率 $P_a(agent)$，然后选择一个在[0，1]区间内满足均匀分布的随机数 r，如果 r<P_a，该蚂蚁将变成活跃状态，并按贪心法在其视野范围内选择一个空闲位置并移动到该位置；反之，蚂蚁将继续呆在原位置，并陷入睡眠状态，按照聚类规则更新蚂蚁的类号;

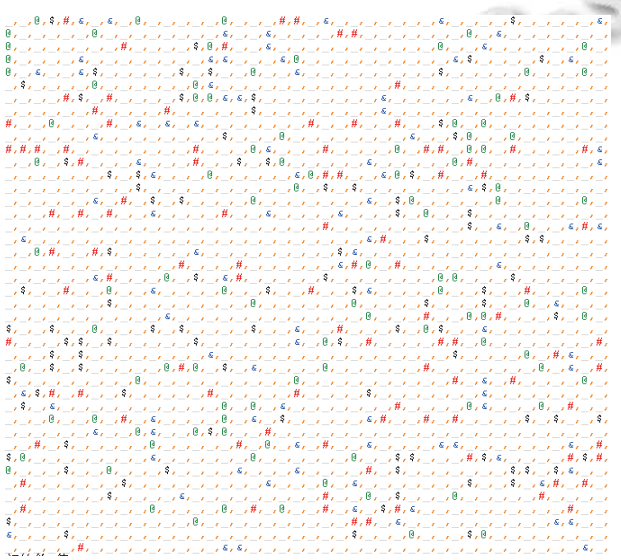
(5) 更新参数;

(6) 如果满足停止条件，则结束算法，并输出蚂蚁

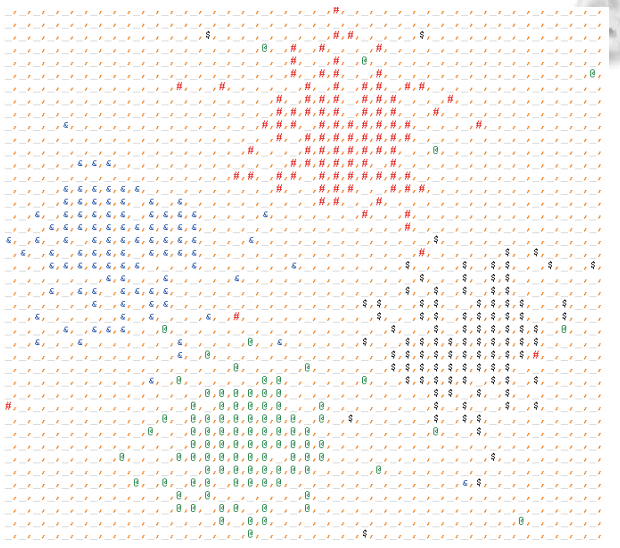
聚类的情况; 否则, 转向(4).

4 算法实验与分析

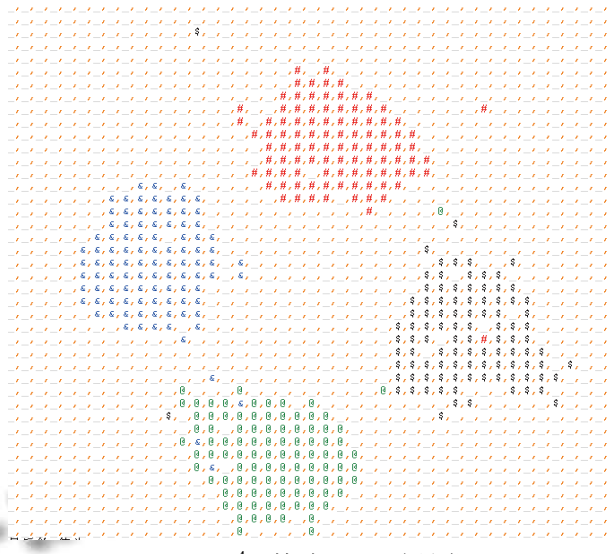
实验环境是一台 PC 机(英特尔奔腾双核 E2200, 1G 内存, Windows XP 环境, VC++6.0 开发平台). 首先本文使用一个满足正态分布的数据集来展示 IA^{4}C 算法的聚类结果. 这个数据集是由 400 条分属为四类的二维数据(x, y)组成, 每类各有 100 条数据, x 和 y 均满足正态分布 $N(\mu,\sigma^2)$, 如 $(N(0,1.5^2), N(0,1.5^2))$, $(N(0,1.5^2), N(8,1.5^2))$, $(N(8,1.5^2), N(0,1.5^2))$, $(N(8,1.5^2), N(8,1.5^2))$. 具体的算法展示见图 4, 图中分别用符号@, #, &, $来代表四类不同的数据, 并用不同的颜色加以标注.



(a) A^{4}C 算法随机投点



(b) IA^{4}C 算法投点



(c) IA^{4}C 算法 1000 次迭代

图 4 IA^{4}C 算法聚类过程

图 4(a) 为 A^{4}C 算法的随机投点方式图, (b)为 IA^{4}C 算法的投点方式图, (c)为 IA^{4}C 算法 1000 次迭代聚类结果图.

从图 4(b)中可以看到采用新的投点方式后依稀可见簇的轮廓, 这大大提高了算法的运行效率.

除了满足正态分布的数据集外, 本文还选用了 UCI 机器学习库[12]里面的 iris, wisconsin 和 seed 三个数据集来对比 A^{4}C 算法和 IA^{4}C 算法. iris 数据集包含 150 条数据, 其有四个属性且分为三类, 其中一类跟另外两类线性可分, 另外两类有一些交集. wisconsin 是一个包含 699 条数据, 有 10 个属性并分属为两类的数据集. seed 数据集包含 210 条数据, 具备 7 个属性, 分属为三类.

图 5 是 iris 数据集中 150 个数据采用 IA^{4}C 算法投点方式的投影图. 图 6 是 IA^{4}C 算法 1000 次迭代聚类的结果. 图中分别用&, #和@号来代表 iris 数据集中 setosa, versicolor, virginica 的三个类别的数据, 并用不同颜色加以标注.

由于原文中作者没有描述其对数据集预处理的方法, 在这里本文假定对各个数据集采取 z-score 标准化处理, 对 A^{4}C 算法和本文改进的 IA^{4}C 算法分别就选取的三个数据集各进行 100 次实验, 结果如表 1, 表 2, 表 3 所示.

从上面几个表中可以看到在 iris 和 seed 数据集中, IA^{4}C 算法在效率和精度上都稍高于原文的 A^{4}C 算法.

至于 wisconsin 数据集, 虽然 IA^{4}C 算法在聚类的错误率上稍高于 A^{4}C 算法, 但是 A^{4}C 算法在此数据集的收敛性不好, 具体可以从图 7 和图 8 两张图的对比中看到, 图中分别用#和$来表示 wisconsin 数据集的两个类别, 并用不同的颜色加以分别.

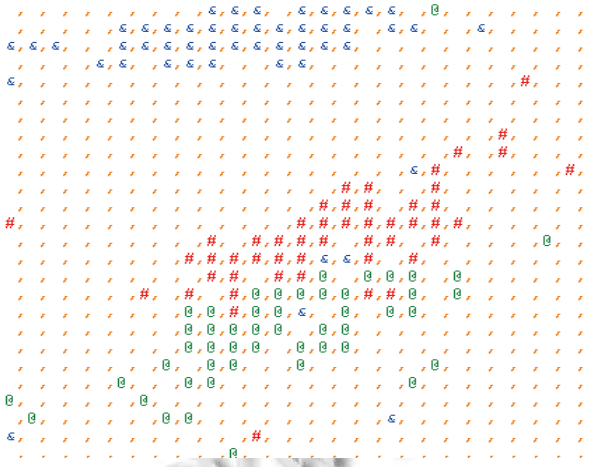


图 5　IA^{4}C 投点方式

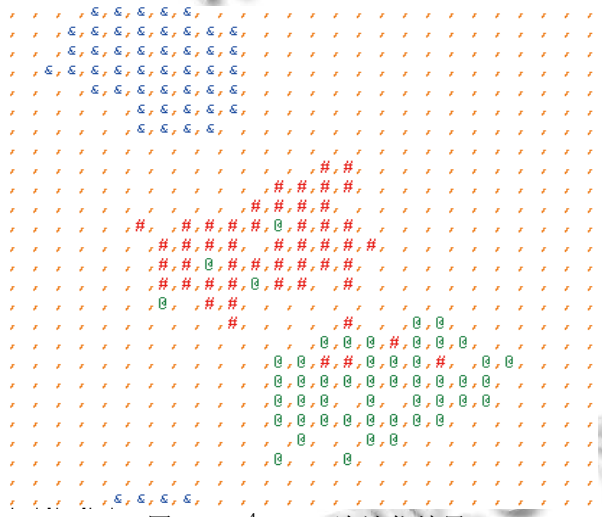


图 6　IA^{4}C1000 次迭代结果

表 1 iris 数据集比较结果

	iris 数据集	
	A^{4}C	IA^{4}C
迭代次数	5000	1000
最小错误个数	11	6
最大错误个数	48	23
平均错误个数	24	13.96
错误率	16%	9.31%
运行时间(s)	2.08	0.36

表 2 seed 数据集比较结果

	seed 数据集	
	A^{4}C	IA^{4}C
迭代次数	5000	1000
最小错误个数	12	19
最大错误个数	67	41
平均错误个数	33.15	29.2
错误率	15.78%	13.90%
运行时间(s)	3.91	0.53

表 3 wisconsin 数据集比较结果

	wisconsin 数据集	
	A^{4}C	IA^{4}C
迭代次数	5000	1000
最小错误个数	14	15
最大错误个数	27	31
平均错误个数	18.13	23
错误率	2.59%	3.29%
运行时间(s)	32.18	3.39

图 7　A^{4}C5000 次迭代结果

图 8　IA^{4}C1000 次迭代结果

从图 7 和图 8 的对比中可以看到 A^{4}C 算法 5000

次迭代后空间中还是存在很多孤立点, 而 IA^{4}C 算法在聚类效果上就好很多, 孤立点也相对较少. 虽然 IA^{4}C 在此数据集中正确率上有所不足, 但是聚类的正确率也仅仅是衡量聚类质量的一个标准. 簇的个数过多有时候会使得聚类的正确率有所提高, 极端情况下一个簇一条数据正确率将会达到 100%, 但这显然不是聚类所要的结果, 所以在考虑聚类正确率的同时也要兼顾聚类簇的个数.

5 结语

本文在 A^{4}C 算法框架的基础上, 提出了一种特征带权的蚁群聚类算法 IA^{4}C. IA^{4}C 将主成分分析方法引入到蚁群聚类当中, 借以消除数据间的冗余, 并针对不同的特征赋予不同的权重, 有效的提高聚类的质量. 此外本文还通过 PCA 计算出贡献率最高的二维属性对应二维网格进行投影用以取代原算法当中的随机投点方式, 有效提高算法运行的效率. 实验结果表明, 本文改进的 IA^{4}C 算法能够有效的改善聚类的质量和效率.

参考文献

1 Han JW, Kamber M.范明,孟小峰译.数据挖掘:概念与技术.北京:机械工业出版社.2007:196–216.

2 孙吉贵,刘杰,赵连宇.聚类算法研究.软件学报,2008,19(1):48–61.

3 Deneubourg JL, Goss S, Franks N, Sendova-Franks A, Detrain C, Chrétien L. The dynamics of collective sorting robot-like Conference on Simulation of Adaptive Behavior on: from ants and ant-like robots. Proc. of the First International Animals to Animats. Cambridge. MIT Press. 1990. 356–363.

4 Lumer ED, Faieta B. Diversity and adaptation in populations of clustering ants. Proc. of the Third International Conference on Simulation of Adaptive Behavior: from Animals to Animats 3. Cambridge. MIT Press. 1994. 501–508.

5 Handl J, Meyer B. Improved ant-based clustering and sorting in a document retrieval interface. In Goos G, Hartmanis J, Leeuwen J, eds. Lecture Notes in Computer Science. Proc. of the 7th International Conference on Parallel Problem Solving from Nature PPSN VII. Berlin. Springer. 2002, 2439. 913–923.

6 Handl J, Knowles J, Dorigo M. Ant-based clustering and topographic mapping. Artificial Life, 2006, 12(1): 35–61.

7 Xu X, Chen L, He P. A novel ant clustering algorithm based on cellular automata. Web Intelligence and Agent Systems, 2007, 5(1): 1–14.

8 Xu XH, Chen L. An adaptive ant clustering algorithm.Journal of Software,2006,9(17):1884–1889.

9 王丽娟,关守义,王晓龙,王熙照.基于属性权重的 Fuzzy C Mean 算法.计算机学报,2006,29(10):1797–1802.

10 陆虎.基于 PCA 与属性权重模糊聚类的入侵检测方法.江苏科技大学学报(自然科学版),2008,(2).

11 张蕾,曹其新,李杰.一种基于群体智能聚类的设备性能横向比较算法.上海交通大学学报,2006,40(3):439–443.

12 http://archive.ics.uci.edu/ml/.

(上接第 103 页)

9 Fu JY, Liang SG, Li QS. Prediction of wind-induced pressures on a large gymnasium roof using artificial neural networks. Computers & Structures, 2007, 85 (3-4): 179– 192.

10 康洪铭.装甲车辆电源系统测试性设计与故障诊断方法研究.北京:装甲兵工程学院, 2012.

11 郑连清.基于神经网络的三相全控桥整流电路故障诊断.重庆大学学报,2004,27(9):72–74.

12 张德丰.Matlab/Simulink 建模与仿真实例讲解.北京:机械工业出版社.2010:275–288.