

## 带混沌搜索的粒子群聚类算法

陈希友, 冯少荣

(厦门大学 计算机科学系, 福建 厦门 360015)

**摘要:** 聚类可以看成是寻找  $K$  个最佳聚类中心的过程。文中把一组聚类中心视为一个粒子( $P$ ), 把各个数据到各自聚类中心的欧式距离之和看成优化函数( $f(P)$ ), 使用带混沌搜索的粒子群聚类算法(C-PSO)算法寻找最优函数值, 从而找到最佳聚类中心。该算法改进了粒子速度的初始化, 把混沌搜索嵌入到粒子群的搜索过程中, 提高了粒子群的搜索能力。实验结果表明, 该算法的聚类效果明显好于  $K$ -means 和 PSO 聚类。

**关键词:** 聚类; PSO; 混沌搜索; C-PSO

**中图分类号:** TP301.6

**文献标识码:** A

**文章编号:** 1673-629X(2008)10-0093-03

## Particle Swarm Optimization Clustering Algorithm with Chaos Search

CHEN Xi-you, FENG Shao-rong

(Department of Computer Science, Xiamen University, Xiamen 360015, China)

**Abstract:** Clustering can be regarded as the process of finding  $K$  optimal centers. Considered that a group of centers can be seen as a particle ( $P$ ), and the sum of Euclidean distance between data and its clustering center as optimal function ( $f(P)$ ), and then using particle swarm optimization clustering algorithm with chaos search to find the optimal function value, so as to find the optimal centers. This algorithm improved on initialization of particle velocity, embedding the chaos search into particle search, so improved the capability of global search of particle swarm. The experiment showed that the clustering result of this algorithm was better than  $K$ -means and PSO clustering.

**Key words:** clustering; PSO; chaos search; C-PSO

## 0 引言

聚类分析是数据挖掘领域的一个重要分支, 聚类技术在机器学习、模式识别等领域有广泛的应用。聚类就是把一组对象分成若干子集, 使得同一子集中的对象相似度最大, 不同子集间的对象相似度最小。在过去的十几年中, 学者们提出了很多聚类算法, 主要包括基于划分、基于密度、基于网格和基于模型等聚类算法。

$K$ -means 是当前最流行的聚类算法, 它实现简单, 是一个线性时间复杂度的高效算法, 但是  $K$ -means 算法对初始中心敏感, 容易陷入局部最优解。最近人们把眼光投向了基于群体智能的聚类算法, 例如基于遗传算法的聚类、基于蚁群的聚类和基于粒子群的聚类<sup>[1]</sup>等, 这类算法的优点在于有较好的聚类效果, 但是计算复杂度较高。Ching-Yi Chen 提出了粒

子群聚类算法<sup>[2]</sup>; Yi-Tung Kao 提出了基于带单纯形搜索的粒子群聚类算法(NM-PSO)<sup>[3]</sup>。文中在前人的基础上提出了带混沌搜索的粒子群聚类算法, 并把实验结果和  $K$ -means、PSO 以及文献[3]中的 NM-PSO 做了比较。

## 1 粒子群算法

粒子群优化算法(PSO)是由 James Kennedy 博士和 R.C. Eberhart 博士于 1995 年提出<sup>[4]</sup>。该算法源于对鸟群、鱼群觅食行为的模拟。在 PSO 中, 首先初始化一群随机粒子, 然后通过迭代寻找最优解。在每一次迭代中, 粒子通过跟踪两个极值来更新自己的速度和位置。第一个极值是粒子本身所找到的最优解, 这个解叫做个体极值(pbest); 另一个极值是整个种群目前找到的最优解, 这个极值是全局极值(gbest)。PSO 算法简单易实现, 不需要调整很多参数。

假设用  $X_i = (x_{i1}, x_{i2}, x_{i3}, \dots, x_{id})$  表示第  $i$  个粒子, 其中  $d$  是粒子的维数, 它经历过的最好位置表示为  $P_i = (p_{i1}, p_{i2}, p_{i3}, \dots, p_{id})$ , 而整个群体经历过的最好

收稿日期: 2008-01-21

作者简介: 陈希友(1983-), 男, 福建寿宁人, 硕士研究生, 研究方向为数据库、数据挖掘; 冯少荣, 副教授, 研究方向为 XML、数据库技术、数据库、数据挖掘。

位置表示为  $G = (g_1, g_2, g_3, \dots, g_d)$ 。粒子  $i$  的速度用  $V_i = (v_{i1}, v_{i2}, v_{i3}, \dots, v_{id})$  表示。对于每一代个体,在找到两个最优值时,粒子根据如下公式来更新自己的速度和位置:

$$v_{ik}(t+1) = w \times v_{ik}(t) + c_1 \times \text{rand}() \times (p_{ik}(t) - x_{ik}(t)) + c_2 \times \text{rand}() \times (g_k(t) - x_{ik}(t)) \quad (1)$$

$$x_{ik}(t+1) = x_{ik}(t) + v_{ik}(t+1) \quad (2)$$

其中,  $w$  为惯性权重因子<sup>[5]</sup>,用惯性权重来控制前面的速度对当前速度的影响,较大的  $w$  可以加强 PSO 的全局搜索能力,而较小的  $w$  能加强局部搜索能力。 $v_{ik}(t)$  是粒子当前的速度,  $v_{ik}(t+1)$  是粒子的新速度,  $x_{ik}(t)$  是粒子当前的位置,  $x_{ik}(t+1)$  是粒子产生的新位置;  $c_1, c_2$  为加速系数,是正常数,也称为学习因子,  $c_1, c_2$  分别调节向全局最好粒子和个体最好粒子方向飞行的最大步长,若太小,则粒子可能远离目标区域,若太大则会导致突然向目标区域飞去,容易错过最优解。合适的  $c_1, c_2$  可以加快收敛且不易陷入局部最优;  $\text{rand}()$  为  $[0, 1]$  之间的随机数。

## 2 混沌搜索

混沌现象广泛存在与非线性系统中,混沌运动具有遍历性、随机性和规律性等特点,能在一定范围内按其自身规律不重复地遍历所有状态。因此,利用混沌序列进行优化搜索,其结果优于随机搜索<sup>[6]</sup>。

混沌搜索过程如下:

第一步,产生一个  $d$  维的随机初始向量  $m_0 = (m_{0,1}, m_{0,2}, m_{0,3}, \dots, m_{0,d})$ , 其中  $m_{0k} \in [0, 1]$ , 并且各个值之间的差异较小。

第二步,在初始向量  $m_0$  的基础上,使用 Logistic 方程(3)迭代产生混沌序列  $m_0, m_1, \dots, m_n$ 。

$$m_{i+1} = \mu m_i (1 - m_i) \quad (3)$$

其中  $\mu$  是控制变量,当  $\mu = 4$  时,系统完全处于混沌状态<sup>[6]</sup>。

第三步,利用混沌序列和公式(4)遍历粒子  $X_i$  的邻域,得到更好的位置  $X'_i$ 。

$$X'_i = X_i + R \times \text{rand}() \times m_t \quad (4)$$

其中  $\text{rand}() \in [-1, 1]$ ,  $t \in [0, n]$ ,  $R$  是粒子  $X_i$  的邻域半径。

## 3 带混沌搜索的粒子群聚类算法(C-PSO)

### 3.1 带混沌搜索的粒子群算法

分析标准的粒子群算法可知,当粒子在飞行过程中,全局最优粒子飞到局部最优解时,全部粒子都会向它靠拢,以至于有可能无法通过粒子群自身的飞行逃

离局部最优解的束缚,从而得到一个局部最优解。笔者认为,当全局最优粒子的适应值连续  $N$  次( $N$  是根据具体情况而定的经验值)变化小于阈值( $T1$ ),就断定粒子群已陷入了局部最优解。此时,对全局最优位置和全部粒子的个体最优位置进行混沌搜索,从而使粒子群跳出局部最优解的束缚<sup>[7]</sup>。

C-PSO 算法(见图 1)说明:

初始化粒子群:初始化粒子的位置和速度,个体最好位置 pbest 设置为当前位置,计算出全局最好位置 gbest,设置算法中使用的参数  $c_1, c_2, T1, T2, w$ , 以及粒子群的大小  $N$ 、粒子迭代的次数和混沌序列的长度  $n$ 。

评价粒子:计算粒子的适应值,如果粒子的适应值好于 pbest,则更新 pbest 为当前位置;如果所有粒子的个体极值中最好的好于当前的全局极值,则将 gbest 设置为该粒子的位置。count 是 gbest 连续变化小于阈值( $T1$ )的计数器。

混沌搜索是对所有粒子的个体极值和全局极值的邻域进行混沌遍历,从而找到适应值更好的位置。

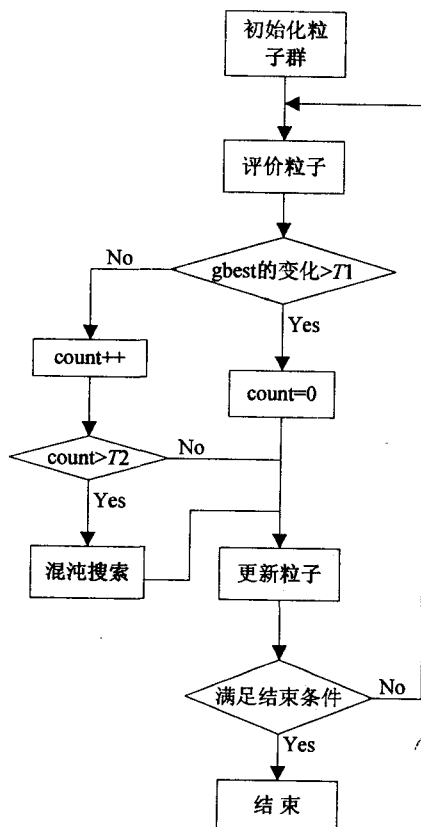


图 1 C-PSO 流程图

### 3.2 带混沌搜索的粒子群聚类算法(C-PSO)

(1)粒子的表示:一个粒子代表一组聚类中心,是  $K$  行  $D$  列的矩阵,  $K$  是聚类中心的个数,  $D$  是数据的维数。粒子群用 3 维矩阵  $P[N][K][D]$  表示,其中  $N$  是

粒子的个数。

(2) 优化函数  $f$ :

$$f(P[i]) = \sum_{k=1}^K \sum_{x \in c_k} \sqrt{(x - P[i][k])^2} \quad (5)$$

表示各个数据到各自聚类中心的欧式距离之和,其中  $c_k$  代表属于第  $k$  个聚类中心的数据集合。 $f(P[i])$  越小,代表聚类越紧凑。

(3) 粒子位置初始化:随机选择原始数据  $x$ , 令  $P[i][k] = x$ , 其中  $i \in [0, N], k \in [0, K]$ 。

(4) 粒子速度初始化:扫描原始数据,分别得到各个维的最大和最小值,第  $d$  维的值记为  $\max(d)$  和  $\min(d)$ , 令  $e(d) = \max(d) - \min(d)$ , 则

$$V[i][k][d] = \sigma \times e(d) \times \text{rand}() \quad (6)$$

其中  $\sigma$  是权重因子,  $\text{rand}()$  是属于  $[0, 1]$  的随机数。粒子的初始速度是粒子搜索能力的重要因素,如果粒子速度太大,则不容易进行局部搜索,从而错过最优解;如果太小,粒子容易陷入局部极值。公式(6)结合了数据的维空间  $e(d)$ , 并用  $\sigma$  进行调节,使得粒子各维的初始速度与  $e(d)$  成正比,从而提高粒子的搜索能力。

## 4 实验分析

### 4.1 实验数据介绍

本实验的数据来源于 <ftp://ftp.ics.uci.edu/pub/machine-learning-databases/>, 数据的情况见表 1。其中, Iris 是 3 种花的萼片长度、萼片宽度、花瓣长度和花瓣宽度的测试数据; Cmc 是 1987 年印度尼西亚妇女避孕方法的调查数据; Wine 是分别用 3 种不同植物酿造的酒的数据。

表 1 实验数据介绍

数据集名称	数据集大小	数据维数	聚类个数	各聚类包含的数据个数
Iris	150	4	3	50, 50, 50
Cmc	1473	9	3	629, 334, 510
Wine	178	13	3	59, 71, 48

### 4.2 实验结果

比较了 K-means、PSO、NM-PSO 和 C-PSO 的聚类结果, 其中 NM-PSO 的数据来源于文献[3]。实验结果包括各个数据集聚类准确率的最大值、最小值和平均值(见表 2), 以及优化函数的最大值、最小值和平均值(见表 3)。

在许多文献中, 都把他们算法的最终聚类中心作为 K-means 的初始中心, 再做 K-means 聚类, 从而得到更好的解。例如在文献[3]中, 作者把 NM-PSO 和 K-means 结合, 从而得到 K-NM-PSO, 从实验结果看, 聚类效果略有改善。笔者认为, 从创新的角度出发, 这样的工作没有太大意义。

从表 2 和表 3 可以看出, PSO 比 K-means 更稳定, PSO 每次执行的结果都较接近于平均解, 而 K-means 容易受初始中心的影响, 导致聚类结果差异较大。由于在 PSO 中嵌入混沌搜索, C-PSO 比 PSO 具有更好的搜索能力; 从实验结果也可以看出, C-PSO 的聚类结果比 PSO 的更好和更稳定, C-PSO 的每次执行结果都较接近最优解。C-PSO 在算法的稳定性上比 NM-PSO 略好。

表 2 各算法的聚类准确率

数据集	类型	K-means	PSO	NM-PSO <sup>[3]</sup>	C-PSO
Iris	best	89.4%	90%	92%	90%
	average	82.7%	87.5%	88.7%	90%
	worst	67.7%	83.3%	—	90%
Cmc	best	45.6%	45.7%	45.62%	45.81%
	average	45.3%	45.6%	45.53%	45.47%
	worst	44.8%	45.32%	—	45.34%
Wine	best	70.1%	71.3%	71.9%	72.0%
	average	70.2%	70.1%	71.5%	71.4%
	worst	69.1%	70.8%	—	70.8%

表 3 各算法的聚类优化函数值

数据集	类型	K-means	PSO	NM-PSO <sup>[3]</sup>	C-PSO
Iris	best	97.32	96.66	96.66	96.65
	average	104.3	102.4	100.72	96.65
	worst	123.84	110.5	—	96.65
Cmc	best	5542.1	5538.3	5537.3	5534.4
	average	5560.8	5734.2	5563.4	5536.5
	worst	5961.2	5876.5	—	5539.2
Wine	best	16555.2	16326.4	16292	16294.3
	average	16701.5	16371	16303	16299.7
	worst	18022.2	16486.9	—	16316.2

## 5 结束语

提出了带混沌搜索的粒子群聚类算法(C-PSO), 改进了粒子速度的初始化, 把聚类看成优化问题, 聚类中心看成优化变量, 各数据到各自聚类中心的欧式距离之和视为优化函数, 在多维数据空间中求解优化问题。实验结果表明 C-PSO 算法的聚类效果比 K-means 和 PSO 更好和更稳定。PSO 算法虽然被广泛地应用, 但是其参数选择仍然没有很好的理论基础, 如何根据被聚类数据自动调整算法参数是一个有意义的问题, 值得进一步研究。

### 参考文献:

- [1] 刘靖明, 韩丽川, 侯立文. 一种新的聚类算法——粒子群聚类算法[J]. 计算机工程与应用, 2005(20): 183-185.
- [2] Chen Ching-Yi, Ye Fun. Particle Swarm Optimization Algorithm and Its Application to Clustering Analysis[C]//International Conference on Networking, Sensing & Control, Proceedings of the 2004 IEEE. Taiwan: [s. n.], 2004: 789-794.

应的独立集报告类型。然而没有其它有利的非诚实揭示。这样,直接证明如果代理 1 总是报告策略上最优的类型,则这个机制的结果总是与最优诚实机制一致。当然,为了代理 1 总是报告策略上最优的类型,当他的类型是  $\theta_Y$  时,则他需要构建一个独立集。因为这个问题是 NPC 问题,猜想代理 1 将不总能构建这样的集合,是合理的。如果代理 1 确实不能构建此情况下的独立集,则结果将是  $x \in Y$ 。因此,在计算能力是无限的假设下,严格大于代理 1 群体利益。

因此,它也比最优诚实机制的社会福利更大。

### 3 结束语

显示原理在机制设计中是一个基础工具。然而,文中表明了,在计算上增加一个合理的约束就可能使显示原理无效。首先,研究了最优诚实机制的情况,说明中心处理这个机制的算法是 NPC 的。通过变换到非诚实机制,它能够从中心到代理转移解决 NPC 问题的负担;其次,提出了一种新的数据库模型,它处理效用

值难以计算的情况,即使在所有相关信息都可以获得的时候下,用这种模型也能处理。通过转移到非诚实机制,它能够转移指数数量询问的负担。在这两种情况里,非诚实机制与无限计算中最优诚实机制一样好。

#### 参考文献:

- [1] 谢识予. 经济博弈论[M]. 上海: 复旦大学出版社, 2002.
- [2] 迈尔森 R B. 博弈论[M]. 于 寅, 费剑平译. 北京: 中国经济出版社, 2001.
- [3] Rothkopf M H, Pekec A, Harstad R M. Computationally manageable combinatorial auctions[J]. Management Science, 1998, 44(8): 1131 - 1147.
- [4] Sandholm T. Algorithm for optimal winner determination in combinatorial auctions[J]. Artificial Intelligence, 2002, 135: 1 - 54.
- [5] Wurman P R, Wellman M P. AkBA: A progressive, anonymous - price combinatorial auction[C]//In Proceedings of the ACM Conference on Electronic Commerce (ACM - EC). Minneapolis, MN: [s. n.], 2000: 21 - 29.

(上接第 95 页)

- [3] Kao Yi - Tung, Zahara E, Kao I - Wei. A hybridized approach to data clustering[J]. Expert Systems with Applications, 2007 (34): 1754 - 1762.
- [4] Kennedy J, Eberhart R. Particle Swarm Optimization[C]//Proc. of IEEE International Conference on Neural Networks (ICNN). Perth, Australia: [s. n.], 1995: 1942 - 1948.
- [5] Shi Y, Eberhart R C. A modified particle swarm optimizer

[C]//Proceeding of IEEE International Conference on Evolutionary Computation. New York, NY, USA: IEEE, 1998: 69 - 73.

- [6] 李 兵, 蒋慰孙. 混沌优化方法及其应用[J]. 控制理论及其应用, 1997, 14(4): 613 - 615.
- [7] 刘华莹, 林玉娥, 张君施. 基于混沌搜索解决早熟收敛的混合粒子群算法[J]. 计算机工程与应用, 2006(13): 77 - 79.

(上接第 98 页)

### 5 结束语

由于网格调度算法对网格系统性能的巨大影响,促使人们对调度算法进行深入研究。文中针对传统的 Min - Min 算法的高效特性和 Max - Min 算法的负载均衡特性,介绍了平衡二者的优缺点的 A - MM 算法,说明了它的优势。在网格调度算法研究中,需要对算法进行深入的性能评估,然而由于受到真实网格环境的高度复杂性的限制,促使人们采用模拟工具来分析调度算法的性能。文中对 GridSim 进行了简单分析,并介绍了如何将 A - MM 算法融入到 GridSim 中去,最后通过模拟实验搭建网格仿真平台对 A - MM 算法进行性能评估。笔者认为,GridSim 仍然是网格仿真技术中最具有创造力的工具。

#### 参考文献:

- [1] Buyya R, Murshed M. GridSim: A Toolkit for the Modeling

and Simulation of Distributed Resource Management and Scheduling for Grid Computing[J]. Journal of Concurrency and Computation: Practice and Experience(CCPE), 2002, 14 (13 - 15): 1175 - 1200.

- [2] Sulistio A, Buyya R. A Grid Simulation Infrastructure Supporting Advance Reservation[C]// Proceedings of the 16th International Conference on Parallel and Distributed Computing and Systems ( PDCS 2004 ). Cambridge, USA: MIT, 2004: 1 - 7.
- [3] 侯 勇, 于 炯. 基于非贡献网格的自适应任务调度算法研究[J]. 微电子学与计算机, 2007(10): 190 - 192.
- [4] Howell F, McNab R. Simjava: a discrete event simulation package for Java with applications in computer systems modelling[C]//In proc. First International Conference on Web - based Modeling and Simulation. San Diego, CA: Society for Computer Simulation, 1998.
- [5] 刘祥瑞, 朱建勇, 樊孝忠. 基于 GridSim 的网格调度模拟[J]. 计算机工程, 2006, 32(2): 42 - 44.