自适应种群的高斯动态粒子群聚类算法①

沈 亮 常新功 景丽荣 (山西财经大学 信息管理学院 山西 太原 030006)

摘 要: 聚类问题究其根本在于样本之间相似性的定义和聚类效果优劣的评价。粒子群聚类算法以其较好的聚类效果而受到广大研究者的关注。提出了一种新的衡量聚类效果的函数，并对其进行一定的分析。另外，从分析粒子群算法的拓扑结构出发，在高斯动态粒子群算法的基础上，提出了一种自适应种群的高斯动态粒子群聚类算法。实验表明，该衡量函数能够有效地评价聚类效果的优劣，其算法具有良好的聚类效果，在高维数据上表现优良。

关键词: 聚类；粒子群算法；衡量函数；拓扑结构；自适应种群

Adaptive Population of Gaussian Dynamic PSO Clustering Algorithm

SHEN Liang, CHANG Xin-Gong, JING Li-Rong

(Shanxi University of Finance and Economics，Taiyuan 030006, China)

Abstract: The key issue in Clustering is the definition of similarity between samples and the evaluation of pros and cons of clustering effects. PSO algorithm has drawn more attention from the majority of researchers for its preferable impact. This paper gives a new function that measures the effectiveness of the clustering algorithm and analyzes it thoroughly. In addition, from the topology of the PSO, an adaptive population of Gaussian dynamic PSO clustering algorithm is proposed based on the Gaussian dynamic algorithm. The experiment shows the measure function could effectively evaluate the pros and cons of clustering effects, and its corresponding algorithm has good clustering efficiency, better performance in the high-dimensional data.

Keywords: clustering; PSO; measure function; topology; adaptive population

聚类分析作为数据挖掘的一个重要的分支，已经广泛地应用于很多领域，如：市场研究、图像分割、数据分析、模式识别、机器学习等。聚类分析是指从研究对象的数据中挖掘出具有相似性质的数据集合。在聚类中产生的每一组数据称之为一个簇。聚类的目的是使同一簇中对象尽可能相似，不同簇之间对象的差异性尽可能地大。由于对相似性的表示不同，有不同的聚类方法。比较著名的聚类算法有：K-means 聚类算法、核方法、粒子群聚类算法、基于遗传算法的聚类等。

粒子群算法(particle swarm optimization PSO)是由 Eberhart、Kennedy[1]在 1995 年提出的一种源于对鸟群捕食的行为研究而产生的群体智能优化算法。该算法易于实现，没有很多的参数需要调整，而且它不需要梯度信息。因此，PSO 是非线性连续优化问题、组合优化问题和混合优化问题的有效优化工具。目前它主要应用于工程设计、优化领域、机器人控制、交通运输、数据挖掘以及其他相关领域。

目前，有很多结合粒子群算法的聚类方法。2006 年，刘向东、沙秋夫[2]提出了基于粒子群与 K-means 的混合聚类算法，并从实验上论证了基于子群优化算法的聚类方法在收敛速度方面明显优于基于遗传算法的聚类方法。2008 年，杨久俊、邓辉文[3]等人提出了一种新的基于混合粒子群优化的模糊 C-均值聚类算法，

① 基金项目:国家自然科学基金(60873100);山西省高校科技研究开发项目(20081023)
收稿时间:2009-12-15;收到修改稿时间:2010-01-10

在粒子群算法的基础结合变异算子、混沌优化、逃逸因子等，有效地避免了聚类算法早熟的现象。陈希友，冯少荣等人[4]提出带混沌搜索的粒子群聚类算法，并指出无需将粒子群算法的运算结果作为 K-means 或是其他聚类算法的初始点再进行聚类。

Kennedy[5]等人采用概率生成新种群的方法提出一种高斯动态粒子群算法，本文中以高斯动态粒子群算法为基础上对聚类进行研究与分析，提出了一种自适应种群的高斯动态粒子群聚类算法。实验表明，该算法稳定性好、并且有较好的聚类效果。

本文第二节简要介绍基本粒子群算法与动态高斯粒子群算法以及与聚类的联系，第三节给出了自适应种群的高斯动态粒子群聚类算法的主要思想及理论分析，第四节给出了实验结果，第五节总结全文。

1 研究背景

1.1 基本粒子群算法

1995 年，Eberhart、Kennedy[1]提出了基本粒子群算法，其飞行的速度和方向由三部分动态改变：①粒子本身的探索欲望。粒子在搜索过程不断探索新的区域。②粒子本身的经验认知。粒子根据自身的经验向自身的最优靠拢。③社会群体的经验认知，表示粒子间的协作，粒子根据群体的认知向群体的全局最优靠拢。

其速度和位置的更新公式如下：

$$V_i^{k+1} = wV_i^k + c_1 r_1 (P_{gbest_i} - Z_i^k) + c_2 r_2 (P_{pbest} - Z_i^k) \quad (1)$$

$$Z_i^{k+1} = V_i^{k+1} + Z_i^k \quad (2)$$

其中，Z_i^k 为第 i 个粒子($i=1,2,\ldots,m$)在第 k 次迭代时的 D 维位置矢量。根据设定的适应值函数计算 Z_i^k 的适应值，即可以衡量粒子的优劣。$v_i=(v_{i1},v_{i2},\ldots,v_{iD})$为粒子 i 的飞行速度。即粒子的移动距离。P_{gbest_i} 为粒子 i 迄今为止搜索到的最好位置；P_{pbest}为整个粒子群找的最好位置。w为速度权重。r_1和 r_2为[0,1]之间的随机数，c_1 和 c_2为学习因子，从(1)、(2)两式中可知粒子具有自我总结和向群体中优秀个体学习的能力，从而向自己的历史最优点以及群体内历史最优点靠近。

1.2 高斯动态粒子群算法

2005年Kennedy[5]提出了一种基于概率生成新种群方法的高斯动态粒子群算法，其粒子位置更新公式如下：

$$X(t+1) = X(t) + W_1 * (X(t) - X(t-1)) + W_2 * (avgp - X(t)) + G(0,1) * (frange/2.0) \quad (3)$$

$$avgp = \sum_{k=1}^{K} P_{kd} / K \quad (4)$$

$$frange = \sum_{k=1}^{K} |P_{id} - P_{kd}| / K \quad (5)$$

这里,X(t+1)为第$t+1$次迭代后粒子的位置，W_1和 W_2分别是权值,Kennedy[4]指出W_1的取值一般为0.729，W_2的取值一般为2.187，$G(0,1)$是一个服从正态分布的随机数发生函数,式(4)和式(5)中的下标K表示第i粒子的邻域粒子的个数,P_{kd}表示粒子k的最优位置，P_{id}表示整个粒子群的最优位置。

在基本的粒子群算法中，粒子的位置仅受自身最好位置和整个粒子群最好位置的影响。相比之下，高斯动态粒子群算法中，粒子的位置不仅受其自身和最优粒子的影响，而且还受其邻域粒子的影响。这不仅加强了粒子之间的信息交流，而且使得粒子群算法模型更加符合现实中模型。文献[4,6]中实验表明，高斯动态粒子群算法明显优于基本的粒子群算法。

1.3 聚类分析

聚类问题的描述：对于给定样本点的集合$X=\{x_1, x_2, \ldots, x_n\}$和簇的数目$m$，确定$m$个簇：$C_1\ C_2\ \ldots C_m$，满足：

$$\begin{cases} \bigcup_{i=1}^{m} C_i = X \\ C_i \cap C_j = \Phi (i, j = 1,2,\ldots,m (i \neq j)) \\ C_i \neq \Phi (i = 1,2,\ldots,m) \end{cases}$$

聚类的关键问题在于定义对象之间的相似性函数Sim()，以及对聚类效果的衡量函数E_c()。值得注意的是，在基于粒子群的聚类算法中聚类效果的衡量函数通常与其适应值函数相关，然后再通过粒子群的迭代以搜索到最佳的聚类。文献[2-4]中均提出了不同的适用值函数。文中为了刻画聚类的簇内极大相似，簇间极大不相似，采用了如下衡量函数:

$$E_1 = \sum_{i=1}^{m} \sum_{x_j \in C_i} Sim(x_j, y_{|C_i|}) / |C_i| \quad (6)$$

$$E_2 = \sum_{i=1}^{m} \sum_{j>i}^{m} Sim(y_{|C_i|}, y_{|C_j|}) \quad (7)$$

$$E_c = E_1 / E_2 \quad (8)$$

其中，$Sim(x_j, y_{|C_i|}) = 1/d(x_j, y_{|C_i|})$，d()为距离函数，$|C_i|$ 表示第i个簇内的样本点数。(6)式表示簇内的平

均相似度之和，度量簇内的相似程度；(7)式表示各个簇中心之间的相似度和，度量各个簇之间的相似度。文中采用(8)式二者之间的比值来衡量聚类效果的优劣。该衡量函数不但将簇数引入到衡量函数，而且减少了样本数对聚类效果的影响。

2 本文方法

本节中首先简单介绍了粒子群适用值函数、粒子的构造和一些基本推导，然后大体描述了自适应种群的高斯动态粒子群聚类算法。

2.1 适应值函数和粒子构造

在本文上一节中提到了关于适用值函数与聚类效果之间的衡量函数之间关联，这里采用的适用值函数为：

$$f_c = 1/E = E_2 \Big/ E_1 \tag{9}$$

其中：

$$\sum_{x_j \in C_i} Sim(x_j, y_{|C_i|}) \Big/ |C_i| \geq Sim_{\min}(x_j, y_{|C_i|}) \geq Sim(y_{|C_i|}, y_{|C_j|})(i \neq j)$$

$$\Rightarrow (m-1)\sum_{x_j \in C_i} Sim(x_j, y_{|C_i|}) \Big/ |C_i| \geq Sim(y_{|C_i|}, y_{|C_1|}) + Sim(y_{|C_i|}, y_{|C_2|}) + ... + Sim(y_{|C_i|}, y_{|C_{i-1}|}) + Sim(y_{|C_i|}, y_{|C_{i+1}|}) + ... + Sim(y_{|C_i|}, y_{|C_m|}) = \sum_{i=1}^{m} Sim(y_{|C_i|}, y_{|C_j|})(i \neq j)$$

$$\Rightarrow (m-1)\sum_{i=1}^{m}\sum_{x_j \in C_i} Sim(x_j, y_{|C_i|}) \Big/ |C_i| \geq \sum_{i=1}^{m}\sum_{i \neq j}^{m} Sim(y_{|C_i|}, y_{|C_j|}) = 2\sum_{i=1}^{m}\sum_{j>i}^{m} Sim(y_{|C_i|}, y_{|C_j|}) = 2E_2$$

$$\Rightarrow (m-1)E_1 \geq 2E_2$$

$$\Rightarrow f_c \leq m - 1/2(m > 1) \tag{10}$$

值得引起注意的是，当其中一个聚类中心$|y_{|C_i|}|$是某一样本x_j时，$Sim(x_j, y_{|C_i|}) \to \infty$，$f_c \to 0$，此时，极大地影响了对聚类效果的评价。故文中设定阈值ϵ，若两者之间的相似度大于等于ϵ时，取ϵ。

这时：

$$E_1 = \sum_{i=1}^{m}\sum_{x_j \in C_i} Sim(x_j, y_{|C_i|}) \Big/ |C_i| \leq m\varepsilon$$

$$E_2 = \sum_{i=1}^{m}\sum_{j>i}^{m} Sim(y_{|C_i|}, y_{|C_j|}) \geq mSim_{\min}() = \frac{m}{d_{\max}} > \frac{m}{d_{\max}'}$$

$$f_c = \frac{E_2}{E_1} > \frac{m/d_{\max}'}{m\varepsilon} = \frac{1}{\varepsilon * d_{\max}'} \tag{11}$$

(11)式中$d_{\max}$为两两中心之间的最大距离，$d_{\max}'$为样本点之间的最大距离。

另外，由式(10),(11)可知：

$$\frac{(m-1)}{2} > \frac{1}{\varepsilon * d_{\max}'}$$

$$\varepsilon > \frac{2}{(m-1)d_{\max}'} \tag{12}$$

故ϵ的取值与样本数据及聚类簇数相关。对于一个固定的数据集，ϵ值随着聚类簇数而动态变化。

本文粒子的构造为聚类中每个簇的中心：$Y=(y_1,y_2,\ldots y_m)^T$。这里$y_i=(y_{i1},y_{i2},\ldots,y_{iD})^T$，表示第$i$个簇的中心，$D$为数据维数，$y_{ij}$表示粒子在某一维上的取值，m为簇的数目。

2.2 可变多种群的高斯动态例子群聚类算法

2.2.1 算法分析

文献[6,7]指出，粒子的邻域粒子能够提供以下信息：第一种是当邻域粒子找到质优解时能够指导粒子向质优解靠拢。另一种是邻域粒子之间的“距离”说明粒子之间已达成的某种“共识”，通过这种“共识”影响着邻域粒子的搜索步骤。

在粒子群算法中Gbest模型和Lbest模型为两种基本的模型[6,8]，分别为全连接型拓扑结构和环形拓扑结构。其中，Gbest模型是一种全连接型结构，粒子的邻域粒子包括其他的所有粒子。其信息传播较快，当其中一个粒子找到质优解时，其他粒子能够在很少步骤内收缩到该质优解。因此整个粒子群的收敛速度较快，具有较强的局部搜索能力。Lbest模型相对于Gbest模型其收敛速度较慢，粒子群之间的信息交流较慢，因此具有较强的全局搜索能力。

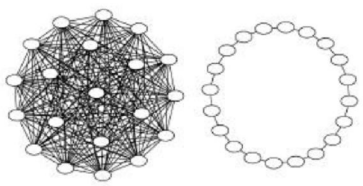

图 1　Gbest 模型和 Lbest 模型

在结合Gbest模型与Lbest模型各自优点的基础上，自适应种群的高斯动态粒子群聚类算法在簇内实现全连接拓扑结构，簇之间实现环形拓扑结构，如图

2所示。

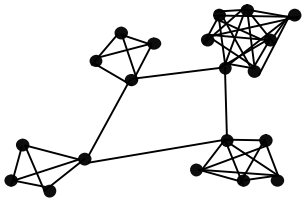

图 2 文中描述的模型

在最优值查找过程中，粒子群算法的快速收敛可能易陷入局部最优而无法找到全局最优点。文中为了扩大粒子群的搜索范围，在搜索前期使用环拓扑结构，加大粒子群对搜索空间的全局搜索。随着粒子群迭代次数的增加，粒子群簇数逐渐减少。在粒子群搜索中期，为了既兼顾粒子群的全局搜索能力，亦考虑到其局部搜索最优解的能力，采用类似于图2所示其拓扑结构。当粒子群的簇数减少到一定程度，其拓扑结构趋向于Gbest模型。此时，粒子群有着较强的局部搜索能力，能够准确找到最优解。

在一个种群规模为25的粒子群算法中，采用自适应种群的高斯动态粒子群算法对Sphere函数进行优化，迭代过程中粒子的位置变化过程。其中，*k*为迭代次数。如图3所示。

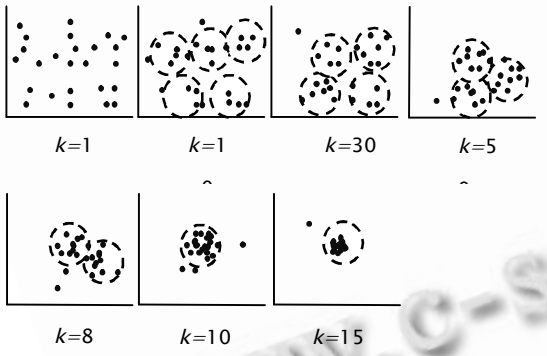

图 3 粒子群簇数随迭代过程的变化

2.2.2 算法步骤

自适应种群的高斯动态粒子群聚类算法步骤如下：

①对数据集进行预处理，设置算法参数，包括最少聚类规模的粒子数*minPts*最少区分距离*d*=kϵ、W_1、W_2等。随机初始化粒子群位置和速度。

②根据数据集和聚类数确定阈值ϵ的取值，计算每个粒子间的相似度并对粒子群进行分簇。

③更新粒子位置和速度矩阵。

④计算适应值，若找到比原来更优的解，更新P_{id}和P_{kd}。

⑤若解满足要求或是到达最大迭代次数，输出。否则，重复步骤②。

3 实验

3.1 实验数据

本文测试数据集全部来自于http://archive.ics.uci.edu/ml/datasets，包括Ionosphere、Iris、Wine、Wine Quality四个数据集。

Iris是3种鸢尾花数据集，该数据集包含3类，即{setosa，versicolor，virginica}，每类中包含50个学习的样本。其属性为{sepal length，sepal width，petal length，petal width}。

Wine是3种植物酿造的酒的数据集，总共包含178个样本数据集。其属性为{Alcohol，Malic acid，Ash，Alcalinity of ash，Magnesium，Total phenols，Flavanoids，Nonflavanoid phenols，Proanthocyanins，Color intensity，Hue，OD280/OD315 of diluted wines，Proline}。

Ionosphere包含351个数据样本，每个样本点有34个属性，分为2类{good bad}。由于数据维数较高，以往算法的聚类效果不佳。

Wine Quality是酒的质量等级数据集，酒的等级分为9个类，分别用1-9标示，共有4898个样本集。其属性为 {fixed acidity ，volatile acidity ，citric acid，residual sugar，chlorides，free sulfur dioxide，total sulfur dioxide ，density ，pH ，sulphates，alcohol}。

3.2 实验设置

本文中选用了 K-means 聚类算法(K-means)、基于粒子群的 K-means 聚类算法(PSOK-means)、以及基于改进粒子群算法的聚类算法(C-PSO)进行对比测试。除 K-means 聚类算法外所有实验的粒子群规模为 25，迭代次数为 1000，对每个数据集上的测试独立运行 50 次。K-means 算法中设置初始 K=每个数据集的类数。在自适应种群的高斯动态粒子群聚类算法中，*minPts*=4、W_1=0.729、W_2=2.187、$\varepsilon = 10 * \frac{2}{(m-1)d_{\max}}$、最少区分距离 *d*=10ϵ。每次运算

给出其聚类的正确率，考察 50 次试验结果的最优值、均值、最差值。

3.3 实验结果

表1给出了在数据集上聚类问题的实验结果。由于实验条件和环境不同，故结果与相关文献数据有一定差别。加黑字体表示最高精度。

表 1 本文方法与其他方法的性能比较

Algorithm	dataset	Worst (%)	Mean (%)	Best (%)
K-means	Iris	70.2	81.5	89.2
	Wine	65	69.2	71
	ionosphere	50.6	51.6	53.8
	Wine Quality	35.2	37.4	40.2
PSO K-means	Iris	75.2	82.4	**91**
	Wine	66.6	70.2	71.6
	ionosphere	53.5	54.8	56.3
	Wine Quality	32.3	36.2	38.6
C-PSO	Iris	86.2	**89.5**	**91**
	Wine	69.2	70.8	**72.4**
	ionosphere	54.4	57.6	61.3
	Wine Quality	36.3	38.2	40.5
本文方法	Iris	85.4	89.4	91
	Wine	**70.2**	**71**	71.5
	ionosphere	**55.7**	**65.4**	**70.3**
	Wine Quality	**43.5**	**45.4**	**49.6**

从表1中可以看出，本文方法在数据维度较高的Ionosphere 和Wine Quality两个数据集上取得明显优于其他的聚类方法，在相比较的其他算法的最好结果上正确率分别提高了15%和23%左右，其效果十分明显。在另外两个较简单的数据集Iris和Wine上与现有的算法性能不相上下。另外，注意到在结果的稳定性方面本文方法较其他方法有一定的改进。

新的评价函数和粒子群算法的引入从总体上提高了聚类的效果。而自适应种群的方式则大大提高粒子群的寻优能力，保证了种群的多样性。以上这些方法都为算法的效率和聚类的结果提供了有益的帮助。

4 结语

本文在 Kennedy[1,5-7,9]等人的基础上，针对数据聚类问题，提出了一种自适应种群的高斯动态粒子群聚类算法。实验表明，该方法在数据维数较高时有着较好的表现，并且稳定性方面比现有算法有一定的改进，比现有方法具有更好的性能。然而，在大数据集上文中方法有待提高。另外，文中方法均是基于数值型数据的聚类，如何改进算法来适应于不同的数据类型是下一步的研究重点。

参考文献

1 Kennedy J, Eberhart RC, Particle swarm optimization. Proc. IEEE Int. Conf. Neural Networks, Perth, Australia. 1995.1942－1948.

2 刘向东,沙秋夫,刘勇查.基于粒子群优化算法的聚类分析.计算机工程, 2006,32(6):201－203.

3 杨久俊,邓辉文,滕姿.基于混合粒子群优化算法的聚类分析.计算机工程与设计,2008,29(22):5820－5822.

4 陈希友,冯少荣.带混沌搜索的粒子群聚类算法.计算机技术与发展, 2008,18(10):93－95.

5 Kenndy J.Dynamic-Probabilistic-particle-swarms. Proc. of the Conf. on Genetic and Evolutionary Computation.Washington:ACMPress, 2005.201－207

6 Kenndy J, Mendes R. Neighborhood topologies in fully informed and best-of-neighborhood particle swarms. Systems, Man, and Cybernetics, Part C: Applications-and-Reviews,IEEE-Transactions, 2006,36(4): 515－519.

7 Kennedy J. Small worlds and mega-minds: Effects of neighborhood topology on particle swarm performance Proc. of the 1999 Conference on Evolutionary Computation. IEEE Computer Society, 1999,1931－1938.

8 倪庆剑,张志政,王蓁蓁,邢汉承.一种基于可变多簇结构的动态概率粒子群优化算法.软件学报, 2009, 20(2):339－349.

9 Mendes R, Kenndy J. Neves J. The fully informed particle swarm: Simpler, maybe better. IEEE Transactions on Evolutionary Computation, 2004,1(1):204－210.