结合双粒子群和K-means的混合文本聚类算法*

王永贵，林 琳，刘宪国
（辽宁工程技术大学 软件学院，辽宁 葫芦岛 125105）

摘 要：传统K-means算法对初始聚类中心选择较敏感，结果有可能收敛于一般次优解，为些提出一种结合双粒子群和K-means的混合文本聚类算法。设计了自调整惯性权值策略，根据最优适应度值的变化率动态调整惯性权值。两子群分别采用基于不同惯性权值策略的粒子群算法进化，子代间及子代与父代信息交流，共享最优粒子，替换最劣粒子，完成进化，该算法命名为双粒子群算法。将能平衡全局与局部搜索能力的双粒子群算法与高效的K-means算法结合，每个粒子是一组聚类中心，类内离散度之和的倒数是适应度函数，用K-means算法优化新生粒子，即为结合双粒子群和K-means的混合文本聚类算法。实验结果表明，该算法相对于K-means、PSO等文本聚类算法具有更强鲁棒性，聚类效果也有明显的改善。

关键词：双粒子群；自调整惯性权值；信息交流；K-means算法；文本聚类
中图分类号：TP183；TP301.6 **文献标志码**：A **文章编号**：1001-3695(2014)02-0364-05
doi:10.3969/j.issn.1001-3695.2014.02.011

Hybrid text clustering algorithm based on dual particle swarm optimization and K-means algorithm

WANG Yong-gui, LIN Lin, LIU Xian-guo
(*College of Software Engineering, Liaoning Technical University, Huludao Liaoning* 125105, *China*)

Abstract: As traditional K-means clustering algorithm is sensitive to the choice of initial cluster centers, the results may converge to the general suboptimal solutions, this paper presented a hybrid text clustering algorithm based on dual particle swarm optimization and K-means algorithm. It designed self-adjusting inertia weight strategy which used rate of change of optimal fitness to adjust the inertia weight automatically. Two populations used PSO based on different inertia weight strategies in the process of evolution. Two populations shared the best individual and eliminated the worst individual by exchanging information between the two groups of offsprings as well as offsprings and parents to complete the evolution. The algorithm was named dual particle swarm optimization. The algorithm combined balancing ability of global and local search of dual particle swarm optimization with efficiency of K-means. Every particle was a group of clustering centers and reciprocal of sum of scatter within class was fitness function, then optimized newborn particle with K-means. This was called hybrid text clustering algorithm based on dual particle swarm optimization and K-means algorithm. The results of experiment show that compared with other text clustering algorithms like K-means and PSO *et al*, this algorithm has strong robustness and better clustering results.

Key words: dual particle swarm optimization; self-adjusting inertia weight(SIW); information exchange; K-means; text clustering

0 引言

在这个信息化的时代，文本已经成为重要的信息载体之一。其数量日益庞大，如何在这些庞大而又复杂的文本信息中有效地找到满足用户需求的文本信息是文本挖掘的重要研究课题之一。利用聚类算法将文本自动分类是解决此问题的重要方法。许多学者对文本聚类进行了研究[1~3]。文本聚类的基本原理是通过计算文本间的相似度，将文本集划分为若干个类，使同一聚类中的文本尽可能地相似，不同聚类中的文本尽可能地相异。文本聚类已经广泛用于数据挖掘、信息检索和主题检测等领域[4]。

K-means算法是数据挖掘领域中一种重要方法，其思想简单、局部搜索能力强、收敛速度快。但是它的初始聚类中心是随机选择的，这有可能使同一类别的样本被当做不同类别的聚类中心，导致聚类结果不理想。为了得到高质量的聚类结果，许多学者提出了各种各样的K-means算法的改进算法。文献[5]采用密度敏感的相似性度量来计算对象的密度，启发式生成样本的初始聚类中心。文献[6]对数据集进行多次采样和K-means预聚类，以产生多组不同的聚类结果，利用不同聚类结果的子簇之间存在的交集构造出关于子簇的加权连通图，并根据连通性合并子簇，提高聚类结果的质量。文献[7]利用数据集样本的空间分布信息定义数据对象的密度，并根据整个数据集的空间信息定义了数据对象的邻域，在此基础上选择位于数据集样本密集区且相距较远的数据对象作为初始聚类中心。

粒子群算法(particle swarm optimization, PSO)是由Kennedy等人[8]于1995年提出的一种重要的群体智能算法，源于对鸟类捕食行为的模拟，现已成为进化算法的一个重要分支。该算法通过初始一群随机粒子，每个粒子代表一个潜在的解，通

收稿日期：2013-06-03；**修回日期**：2013-07-25 **基金项目**：国家自然科学基金资助项目(60903082)；辽宁省教育厅项目(L2012113)

作者简介：王永贵(1967-)，男，内蒙古宁城人，教授，硕士，主要研究方向为智能计算、云计算、数据挖掘(lidypli@126.com)；林琳(1989-)，女，硕士研究生，主要研究方向为智能计算、数据挖掘；刘宪国(1981-)，男，讲师，博士，主要研究方向为计算机图形学与CAD、智能计算.

过迭代的方式，使每个粒子向自身最好位置和群体最好位置靠近。该算法思想直观、实现简单、执行效率高且具有较强的全局搜索能力，利用 PSO 算法的全局搜索能力解决 K-means 算法对初始值敏感问题。因此，本文提出结合双粒子群和 K-means 的混合文本聚类算法，并与 K-means、PSO 算法等文本聚类算法作比较，该算法能够取得更好的聚类效果。

1　文本表示

1.1　向量空间模型

文本聚类问题中常采用向量空间模型（vector space model，VSM）[9] 进行文本表示。该模型将每个文本表示成空间向量，特征词作为文本的表示单位，向量的每一维是对应特征词在该文本中的权值。即把文本集 x 表示成 $(x_1, x_2, \cdots, x_n)$，x_i 的向量空间表示为 $(\omega_1(x_i), \omega_2(x_i), \cdots, \omega_m(x_i))$，其中，$m$ 表示特征项的数目，$\omega_j(x_i)$ 表示第 j 个特征项在文本 x_i 中的权值。

1.2　特征项权值的计算

特征项权值反映了特征项在文本中的重要程度，是特征项所在文本与其他文本区分开来的一个度量。计算特征项权值的方法有很多，一般选用 TF * IDF 算法[10]。其计算公式如下：

$$\omega_i(x_j) = \frac{tf_{ij} \times \log_2(N/N_i + a)}{\sqrt{\sum_{i=1}^{n} [tf_{ij} \times \log_2(N/N_i + a)]^2}} \tag{1}$$

其中：tf_{ij} 表示第 i 个文本特征在文本 x_j 出现的次数；N 为文档总数；N_i 为文本集合中出现第 i 个特征词的文本数；$\sqrt{\sum_{i=1}^{n} [tf_{ij} \times \log_2(N/N_i + a)]^2}$ 为归一化因子。

1.3　文本相似度计算

文本相似度是衡量文本间相似程度大小的一个统计量。文本相似性度量有余弦法、内积法、距离函数法等。本文选用距离函数法中的欧氏距离计算文本相似度，计算公式为

$$D(x_i, x_j) = \sqrt{\sum_{k=1}^{n} (\omega_{ki} - \omega_{kj})^2} \tag{2}$$

其中：ω_{ki} 与 ω_{kj} 分别为第 k 个文本特征在文本 x_i 与 x_j 间的特征权值。

2　PSO 优化算法与 K-means 算法

PSO 优化算法在求解优化问题时，粒子被抽象成解空间中的点，具有位置和速度属性，粒子在解空间中飞行，根据自身飞行经验和群体飞行经验动态更新自己的速度和位置。其更新速度和位置的公式如下：

$$v_{id}^{k+1} = \omega v_{id}^{k} + c_1 r_1 (p_{id}^{k} - z_{id}^{k}) + c_2 r_2 (p_{gd}^{k} - z_{id}^{k}) \tag{3}$$

$$z_{id}^{k+1} = z_{id}^{k} + v_{id}^{k+1} \tag{4}$$

其中：z_{id} 为 i 个粒子的 d 维位置矢量；v_{id} 为粒子的飞行速度；p_{id} 为粒子迄今为止搜索的最优位置；p_{gd} 为整个粒子群迄今为止搜索的最优位置；ω 为惯性权值，表示先前粒子的速度对当前速度的影响程度；r_1 和 r_2 为 [0,1] 之间的随机数；c_1 和 c_2 为学习因子，也称加速因子。

粒子群算法虽然编码简单，容易实现，但它在优化过程初期收敛速度较快，后期所有粒子都向最优粒子学习，失去种群多样性，易陷入局部最优。针对这一问题，文献 [11] 提出了线性递减权值（linear decreasing inertia weight，LDIW）策略，即在迭代过程中线性地递减 ω 值，按式 (5) 调整。

$$\omega = (\omega_{start} - \omega_{end})\left(\frac{T_{max} - T}{T_{max}}\right) + \omega_{end} \tag{5}$$

其中：T_{max} 为最大迭代次数；T 为当前迭代次数；ω_{start} 是初始惯性权值；ω_{end} 为进化到最大的迭代次数时的惯性权值。

K-means 算法是一种典型的基于距离的聚类算法，用距离作为相似性度量的标准，距离越小相似性越大，最终把距离紧凑的对象聚为一类。其主要工作过程[12] 为：随机选择 k 个样本作为初始聚类中心，对于剩余的样本按照最邻近原则进行聚类划分，即计算其到各聚类中心的距离，将其放入离它最近的类中；然后重新计算每个类的聚类中心。重复上述过程，直到聚类中心不再变化。从 K-means 算法的聚类过程可以看出，其操作简单，容易实现，但初始聚类中心的选择对结果有很大的影响，不同的初始聚类中心可能会导致不同的聚类结果。

3　结合双粒子群和 K-means 的混合文本聚类算法设计

3.1　双粒子群优化算法

基本的粒子群算法能快速收敛，但搜索方式单一，随着迭代次数的增加，粒子趋向同一化，易陷入局部最优。针对粒子群算法的局限性，设计了一种自调整惯性权值（SIW）策略，并将基于 SIW 策略的 PSO 算法与基于 LDIW 策略的 PSO 算法结合，提出了双粒子群优化算法。两个粒子群采用不同的方式进化，并引入一种信息交流机制，从而提高算法的求解精度和寻优效率。

3.1.1　双种群进化策略

两子群基于不同参数的选取进化，一子群采用基于 SIW 策略的 PSO 算法，另一子群采用基于 LDIW 策略的 PSO 算法，两子群分别按照不同的飞行轨迹进行搜索。该算法可以通过鸟群的群体行为进行说明，基本的粒子群算法以单个鸟群为对象，通过个体经验和群体经验寻找最优位置。双种群优化算法以两个鸟群为对象，两个子群采用不同的进化方式，保证了种群的多样性，提高了全局搜索能力。子群内部根据个体经验和群体经验进行搜索，子群间通过引入信息交流机制，整个过程两子群相互引导，协同进化。

3.1.2　自调整惯性权值策略

在 PSO 算法中，惯性权值 ω 是一个非常重要的参数。较大的惯性权值有利于全局搜索，较小的惯性权值有利于局部搜索，合理地调整惯性权值能够有效地权衡算法的全局与局部搜索能力。因此，本文提出一种自调整惯性权值（self-adjusting inertia weight，SIW）策略，它能改变 ω 为定值的单一模式，较好地权衡全局与局部搜索能力。自调整惯性权值策略如式 (6) (7) 所示：

$$r = \left| \frac{\text{fitness}(t) - \text{fitness}(t-n)}{\text{fitness}(t-n)} \right| \tag{6}$$

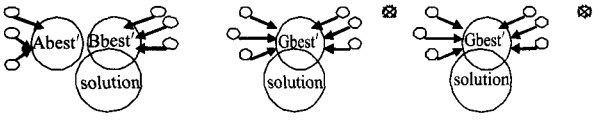
$$\omega = \begin{cases} 0.2 + \theta/3 & r \leqslant 0.3 \\ 0.4 + \theta/3 & 0.3 < r \leqslant 0.6 \\ 0.6 + \theta/3 & r > 0.6 \end{cases} \tag{7}$$

其中：r 为 n 代内最优适应度值的变化率；fitness(t) 为第 t 代最优适应度值；fitness($t-n$) 为第 $t-n$ 代最优适应度值；θ 为均匀分布于 [0,1] 之间的随机数。当变化率较大时，说明算法处于对新空间开发阶段，增大惯性权值，有利于增强其全局搜索能力；当变化率较小时，算法处于局部搜索阶段，减小惯性权值，有利于获得精确的解；当变化率适中时，算法处于全局搜索与

局部搜索之间，选取适当的惯性权值，平衡算法的全局与局部搜索能力。这种自调整惯性权值策略根据最优适应值的变化率灵活地调解 ω 的值，进而提高算法的性能。

3.1.3 信息交流机制原理

两个子群在每次进化前都要进行信息交流，两个子群共享最优个体，淘汰最劣的个体，整个过程两子群相互引导，协同进化。假设两子群分别为 A 和 B 子群。下面以某一次进化中，B 子群进化得到的最优个体更接近目标值为例，说明在该算法中两个子群信息交流的过程。为了方便说明，给出了该算法中个体的运动趋势，如图 1 所示。



(a)Bbest′ 适应度值更接近Solution (b)两子群共享Gbest′ (c)两子群共享Gbest并淘汰最劣个体

图1 算法中某一代个体的运动趋势

图 1 中，Abest′与 Bbest′分别为 A 和 B 子群进化得到的子代最优个体，solution 为目标函数的最优解，Gbest′为 Abest′与 Bbest′两者中适应度值最接近 Solution 的个体，Gbest 为两个子群中父代适应度值最优的个体。在某一次进化中，Bbest′的适应度值更接近 Solution，如图 1(a)所示。经过 Abest′和 Bbest′竞争后得到 Gbest′，当 Gbest′适应度值优于 Gbest 时(设为 condition)，A 子群将不再仅根据自身经验进化，它还要吸收 Bbest′个体的信息。随着 Bbest′的引入，A 子群就会朝着更接近 solution 的方向进化，如图 1(b)所示。当 condition 不成立时，则用两子群父代的最优个体 Gbest 分别替换掉 A 与 B 子群的最劣个体，既保证了整个种群的优良性，同时也加快了进化速度，如图 1(c)所示。A 子群进化得到的最优个体更接近目标值，原理相同。

3.1.4 双粒子群优化算法描述

双粒子群优化算法基本流程如下：

a)随机产生初始种群，大小为 $2N$。

b)设置种群参数(最大迭代次数 Genmax，另初始代数为 Gen = 0，惯性权值 ω，变化率代数 n，初始惯性权值 ω_{start}，进化到最大的迭代次数时的惯性权值 ω_{end})且计算种群的适应度 fitness(为方便下文聚类算法的讨论，设此处是解最大化的目标函数)。

c)随机将种群分为两组，每组 N 个个体，分别为 GroupA、GroupB。

d)选择 GroupA 中 fitness 最大的个体 Abest，按式(5)计算 ω，对 GroupA 中每个粒子按式(3)(4)更新自己的速度和位置，每个粒子更新个体最优值，整个子群更新全局最优值，得到种群 GroupA′。

e)选择 GroupB 中 fitness 最大的个体 Bbest，当 Gen≥n 时，用式(6)计算近 n 代的最优适应度变化率 r，按式(7)自调整惯性权值 ω，对 GroupB 中每个粒子按式(3)(4)更新自己的速度和位置，每个粒子更新个体最优值，整个子群更新全局最优值，得到种群 GroupB′。

f)选择 GroupA′与 GroupB′中 fitness 最大的个体分别为 Abest′与 Bbest′。

g)Abest′与 Bbest′竞争，选择出 fitness 最大的个体设为 Gbest′。如果 Gbest′的 fitness 均大于 Abest 与 Bbest 的 fitness，则分别将 GroupA′与 GroupB′中 fitness 最大的个体替换为 Gbest′，否则用 Abest 与 Bbest 两者中 fitness 最大的个体分别替换 GroupA′与 GroupB′中 fitness 最小的个体。

h)Gen = Gen + 1，若达到 Genmax，则输出当前 fitness 最大个体，算法终止；否则令 GroupA = GroupA′、GroupB = GroupB′跳转到 d)。

该算法与基本 PSO 算法相比有四个特征：a)父代总包含种群最优个体；b)引入自调整惯性权值策略；c)父代产生子代后，用最优个体替代最劣个体；d)采用不同的飞行策略。特征 a)增强了种群的搜索能力；b)增强了算法的全局和局部搜索能力的灵活性；c)加快了收敛速度；d)增强了种群多样性，避免了早熟收敛。

3.2 结合双粒子群和 K-means 的混合文本聚类算法

3.2.1 个体编码

本文中粒子的位置采用基于聚类中心的浮点数编码方式，每个粒子代表一组聚类中心。文本聚类是将经过文本预处理后，维数为 d 的若干个文本组成的文本集合聚成 k 个类的过程。因此，每个粒子的位置是 $k \times d$ 维的向量。由于粒子的速度与位置具有同样的数据结构，所以粒子的速度也是 $k \times d$ 维的向量。其结构如图 2 所示。其中，l_{ij} 表示第 i 个文本的第 j 维的位置，v_{ij} 表示第 i 个文本第 j 维的速度。

$l_{11}, l_{12}, \cdots, l_{1d}$	$v_{11}, v_{12}, \cdots, v_{1d}$
$l_{21}, l_{22}, \cdots, l_{2d}$	$v_{21}, v_{22}, \cdots, v_{2d}$
⋮ ⋮ ⋮	⋮ ⋮ ⋮
$l_{k1}, l_{k2}, \cdots, l_{kd}$	$v_{k1}, v_{k2}, \cdots, v_{kd}$

图2 粒子的编码结构

3.2.2 适应度函数设计

文本聚类的目标是使各类内文本距离之和的总值最小。本文使用欧氏距离进行文本间相似性度量，因此将适应度函数定义如下：

$$\text{fit(ind)} = \frac{1}{1 + \sum_{j=1}^{K} \sum_{X_i \in C_j} D(X_i, Z_j)} = \frac{1}{1 + \sum_{j=1}^{K} \sum_{X_i \in C_j} \| X_i - Z_j \|} \quad (8)$$

其中：K 为聚类数目；X_i 为类 C_j 中的文本；Z_j 为聚类中心，其实际意义是各类文本到其聚类中心距离的总和(即离散度之和)加 1 后求倒数，即为粒子的适应度值。这样粒子的适应度与离散度之和成负相关，离散度之和越小，粒子的适应度值越大。

3.2.3 算法描述

结合双粒子群和 K-means 的混合文本聚类算法的主要思想是：利用双粒子群算法进化多个粒子，每个粒子是一组聚类中心集合，再用 K-means 算法优化每个粒子，最后两子群信息交流得到最优粒子，如此循环，直至达到结束条件。一方面双粒子群算法降低了对初始聚类中心的敏感度，提高了算法的精确性和稳定性；另一方面 K-means 算法加快了算法的收敛速度。

算法的具体流程如下：

a)初始化种群。设 M 个文本聚为 K 类($M \geq K$)，种群个数为 $2N$。随机选取 K 个不同文本作为聚类中心，即粒子的位置编码，并初始化粒子的速度。重复 N 次，生成粒子群 GroupA。以同样的方式生成另一个种群 GroupB。

b)一个粒子相当于 K 个聚类中心点的集合。根据最邻近规则，将 M 个文档用一个粒子进行聚类划分，以同样的方式操作所有的粒子，并按式(8)计算粒子的适应度值。

c)分别选出 GroupA、GroupB 中适应度最大的个体为 Abest、Bbest。两者竞争,得到适应度最大个体 Gbest。

d) GroupA、GroupB 并行进化生成子代种群 GroupA′、GroupB′。

(a)比较每个粒子的适应度值和历史最优位置 p_{id} 所对应的适应度值,如果更优,更新 p_{id}。

(b)比较每个粒子的适应度值和全局最优位置 p_{gd} 所对应的适应度值,如果更优,更新 p_{gd}。

(c) GroupA 按式(5)更新惯性权值,GroupB 按式(6)(7)更新惯性权值。

(d)根据式(3)(4)更新粒子的速度和位置。

(e)对于新生粒子用 K-means 算法优化——按照最近邻规则,用每个粒子对文档进行聚类划分,重新计算新的聚类中心,取代原来的个体编码,按式(8)重新计算新粒子的适应度。

e)分别选出 GroupA′、GroupB′适应度最大个体为 Abest′、Bbest′。两者竞争,得到适应度最大个体 Gbest′。

f)如果 Gbest′的适应度值均大于 Abest 与 Bbest 的适应度值,则分别将 GroupA′与 GroupB′中适应度值最大的个体(即 Abest′、Bbest′)替换为 Gbest′,否则用 Gbest 分别替换 GroupA′与 GroupB′中适应度值最小的个体。

g)判断进化是否达到停止标准,如果达到,则停止进化,跳到 h);否则,令 GroupA = GroupA′、GroupB = GroupB′跳到 c)。

h)选择 GroupA′、GroupB′中适应度值最大的个体作为初始聚类中心,其对应的 K-means 聚类结果为最终聚类结果。

4　仿真实验及结果分析

4.1　实验参数

在采用 SIW 策略的子群中,通常情况下,ω 的取值在 0.4 ~ 0.9 之间。在搜索前期,为使搜索范围较大,ω 应取较大值,ω 初始值取 0.9。变化率代数一般在 5 ~ 10 之间,本文取 $n = 5$。Shi 等人[11]发现,当 $\omega \in [0.4, 0.95]$ 时,算法的性能会大大提高。文献[13]提出的基于 Sigmoid 的惯性权值也恰好位于此区间,算法各性能都有显著的提高。因此,在采用 LDIW 策略的子群中,取 $\omega_{start} = 0.95$, $\omega_{end} = 0.4$, $c_1 = c_2 = 2.0$。为计算 LDIW 策略中 ω 的值,设置 $T_{max} = 40$,但结合双粒子群和 K-means 的混合文本聚类算法的终止条件是整个种群平均适应度值连续多代无明显变化。

4.2　双粒子群算法与粒子群算法的对比

为了对比双粒子群算法与 PSO 算法的性能,用两种算法对以下三个经典测试函数进行了最小值测试:

a) Sphere 函数

$$f_1(x) = \sum_{i=1}^{n} x_i^2$$

$$x \in [-10, 10], f_{min} = 0$$

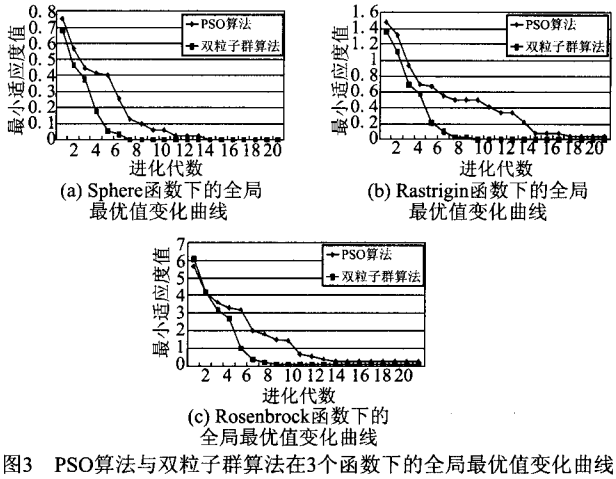
b) Rastrigin 函数

$$f_2(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)] + 10n \quad x \in [-4, 4], f_{min} = 0$$

c) Rosenbrock 函数

$$f_3(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2] \quad x \in [-30, 30], f_{min} = 0$$

Sphere 函数是较为简单的单峰值函数,用于测试算法的寻优精度;Rastrigin 是多峰值函数,局部极小点较多,是难度较大的复杂优化问题;Rosenbrock 函数是一个连续的单峰值函数,全局极值落在一个狭长的山谷中,也是难度较大的复杂优化问题。此次实验绘制了两种算法的全局最优值变化曲线,函数曲线值是 50 次实验中算法每代搜索到的最小值的平均值。实验结果如图 3 所示。



(a) Sphere函数下的全局最优值变化曲线　(b) Rastrigin函数下的全局最优值变化曲线

(c) Rosenbrock函数下的全局最优值变化曲线

图3　PSO算法与双粒子群算法在3个函数下的全局最优值变化曲线

从图 3 可知,双粒子群算法在寻优精度和寻优速度上明显优于 PSO 算法。对于 Sphere 和 Rastrigin 函数,双粒子群算法相对于 PSO 算法都能在较少的迭代次数内获得最优解。如图 3(b)中,在收敛速度上,双粒子群算法明显快于 PSO 算法,其在第 7 代左右就逼近全局最优值,而 PSO 算法在第 14 代才接近全局最优值;在求解精度上,双粒子群算法在第 9 代找到了全局极值 0,而 PSO 算法在第 17 代陷入了局部最优值。由于显示粒度的原因,不能够完全在图中表示出来。对于 Rosenbrock 函数,双粒子群算法较 PSO 算法取得了更好的效果。PSO 算法容易陷入局部极值且收敛速度相对较慢,而双粒子群算法则表现出较高的搜索精度和较快的收敛速度。因此,双粒子群算法相对于 PSO 算法在求解精度和收敛速度上都有着显著提高。

4.3　结合双粒子群和 K-means 的混合文本聚类算法的仿真与分析

实验 1　算法性能测试

实验中,采用 K-means 算法、PSO 聚类算法、PSO + K-means 算法、使用聚类特性的 CFK-means、并行遗传算法 PGA 和本文算法从中国百科术语数据库中抽取 120 篇测试文档(共四类,每类 30 篇),每种算法运行 50 次,测试结果如表 1 所示。

表 1　算法性能测试结果

算法	平均进化代数	进化代数方差	得到最优解的次数	收敛概率/%
K-means	32	0.61	39	78
PSO	29	0.34	45	90
CFK-means	26	0.35	44	88
PGA	29	0.30	45	90
PSO + K-means	22	0.28	46	92
本文	16	0.24	50	100

从实验结果可以看出,K-means 算法的稳定性较差,这正是由于 K-means 对初始聚类中心选择敏感所致;PSO 和 CFK-means 算法在稳定性上有很大的提高,且平均进化代数也相对减小,是相对较好的聚类算法;PGA 算法将遗传算法的并行性考虑其中,稳定性有了明显的提高,但仍有改进的空间;PSO + K-means 算法是一种高效稳定的聚类算法,但是它利用单一的粒子群进行搜索,因此性能仍不如本文算法;本文算法在平均进化代数上大大减小,稳定性有着明显的优势,在收敛概率上,

本文算法达到了 100%，即 50 次实验中，每次都能得到最优解。因此，本文算法在各方面都具有显著优势。

实验 2 聚类结果的准确性测试

本实验的数据集从中国百科术语数据库中抽取 160 篇测试文档。其中，体育类文档 38 篇，教育类文档 50 篇，法律类文档 42 篇，经济类文档 30 篇。基于实验 1 的结果，本次实验选择各性能相对较优的 PSO + K-means 算法、PGA 算法与本文算法作比较，通过计算正确率评价聚类的效果。图 4 和表 2 是本文算法与上述两种算法聚类结果的正确率比较。从图 4 可以看出，本文算法在各类别和总体的正确率均优于 PSO + K-means 和 PGA 算法，PGA 算法正确率相对较低。其中，法律类方面，PSO + K-means 和 PGA 聚类算法的正确率相对较低，都低于 85%，PGA 算法甚至低于 80%，而本文算法的法律类正确率相对于其他两种算法明显较高，在 90% 以上；教育类方面，三种算法的正确率差距较小，这可能是实验数据中教育类文档的选取造成的，但本文算法仍高于其他两种算法。表 2 中，本文算法与上述两种算法的正确率分别是 92.4%、87.8%、85.5%，相对于上述两种算法在正确率上分别提高了 4.6% 与 6.9%。因此，本文算法在聚类效果上有着显著的优势。PGA 算法考虑了遗传算法的并行性，但并没有充分利用此特性。PSO + K-means 算法利用的是单一的粒子群搜索，因此仍不如本文算法。本文算法的误差主要是特征值的抽取与边界文档的聚类划分。

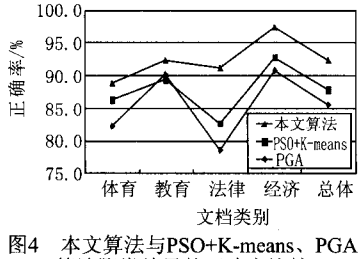


图4 本文算法与PSO+K-means、PGA 算法聚类结果的正确率比较

表 2 本文算法与 PSO + K-means、PGA 算法聚类结果的正确率

算法	体育/%	教育/%	法律/%	经济/%	总数/%
本文	88.9	92.3	91.2	97.3	92.4
PSO + K-means	86.3	89.4	82.7	92.8	87.8
PGA	82.3	90.2	78.6	90.8	85.5

5 结束语

本文提出了双粒子群算法，并将该算法应用到 K-means 文本聚类中，实现了结合双粒子群和 K-means 的混合文本聚类算法。该算法将双粒子群算法和 K-means 算法有机结合，充分发挥两者的优势。实验结果表明，该算法具有较强的稳定性和较高的准确性，产生了较好的聚类效果。本文算法在聚类过程中每个文档只能被唯一地划分到某一类，而实际情况中一些相近文档有可能属于多个类。因此下一步的工作是通过改进算法，解决这些模糊文档的聚类问题，增强算法的精确性。

参考文献：

[1] MAHDAVI M, ABOLHASSANI H. Harmony K-means algorithm for document clustering[J]. Data Mining and Knowledge Discovery, 2009,18(3):370-391.

[2] 徐森，卢志茂，顾国昌. 解决文本聚类集成问题的两个谱算法[J]. 自动化学报，2009,35(7):997-1002.

[3] ZHENG Hai-tao, KANG B Y, KIM H G. Exploiting noun phrases and semantic relationships for text document clustering[J]. Information Sciences, 2009,179(13):2249-2262.

[4] 赵卫中，马慧芳，李志清，等. 一种结合主动学习的半监督文档聚类算法[J]. 软件学报，2012,23(6):1486-1499.

[5] 汪中，刘贵全，陈恩红. 一种优化初始中心点的 K-means 算法[J]. 模式识别与人工智能，2009,22(2):299-304.

[6] 雷小锋，谢昆青，林帆，等. 一种基于 K-means 局部最优性的高效聚类算法[J]. 软件学报，2008,19(7):1683-1692.

[7] 谢娟英，郭文娟，谢维信，等. 基于样本空间分布密度的初始聚类中心优化 K-均值算法[J]. 计算机应用研究，2012,29(3):888-892.

[8] KENNEDY J, EBERHART R C. Particle swarm optimization[C]// Proc of IEEE International Conference on Neural Networks. Piscataway: IEEE Press, 1995: 1942-1948.

[9] SALTON G, WONG A, YANG C S. A vector space model for automatic indexing[J]. Communication of the ACM, 1975,18(11): 613-620.

[10] SALTON G, BUCKLEY B. Term-weighting approaches in automatic text retrieval[J]. Information Processing and Management, 1988,24(5):513-523.

[11] SHI Yu-hui, EBERHART R C. Empirical study of particle swarm optimization[C]//Proc of IEEE Congress on Evolutionary Computational. Piscataway: IEEE Press, 1999: 1945-1950.

[12] 付宁，乔立岩，彭喜元. 基于改进 K-means 聚类和霍夫变换的稀疏源混合矩阵盲估计算法[J]. 电子学报，2009,37(Z1):92-96.

[13] 黄利，杜伟伟，丁立新. 基于 Sigmoid 惯性权重自适应调整的粒子群优化算法[J]. 计算机应用研究，2012,29(1):32-34.

(上接第 350 页)

[3] 林旺群，卢凤顺，丁兆云，等. 基于带权图的层次化社区并行计算方法[J]. 软件学报，2012,23(6): 1517-1530.

[4] KERNIGHAN B W, LIN S. An efficient heuristic procedure for portioning graphs[J]. Bell System Technical Journal, 1970,49(2): 291-310.

[5] NEWMAN M E J. Detecting community structure in networks[J]. European Physical Journal B, 2004,38(2):321-330.

[6] FIEDLER M. Algebraic connectivity of graphs[J]. Czechoslovak Mathematical Journal, 1973,23(2):298-305.

[7] CLAUSET A. Finding local community structure in networks [J]. Physical Review E, 2005,72(2): 026132.

[8] NEWMAN M E J. Fast algorithm for detecting community structure in networks[J]. Physical Review E, 2004,69(6):066133.

[9] NEWMAN M E J, GRIVAN M. Finding and evaluating community structure in networks[J]. Physical Review E, 2004, 69(2): 026113.

[10] PALLA G, DERENYI I, FARKAS I, *et al*. Uncovering the overlapping community structure of complex networks in nature and society[J]. Nature, 2005,435(7043):814-818.

[11] ZHANG Shi-hua, WANG Rui-sheng, ZHANG Xiang-sun. Identification of overlapping community structure in complex networks using Fuzzy C-means clustering[J]. Physical A: Statistical Mechanics and Its Applications, 2007,374(1):483-490.

[12] GREGORY S. An algorithm to find overlapping community structure in networks[C]//Proc of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases. Berlin: Springer-Verlag, 2007:91-102.