文章编号:1000-582X(2008)06-0652-06

一种并行自适应微粒群聚类算法

王华秋[1,2],廖晓峰[1],冯　晋[2]
(1. 重庆大学 计算机学院,重庆 400030;2. 重庆工学院 计算机学院,重庆 400050)

摘　要:大规模的数据挖掘如聚类问题迫切需要大量计算,提出了自适应微粒群优化的并行聚类算法。通过从多种群并行地开始搜索,基于群体搜索技术的微粒群优化算法减少了初始条件的影响,采用任务并行和部分异步通信策略,降低计算时间。结合并行微粒群算法的自适应参数动态优化特性,克服群体逐渐失去迁移性而停止进化的问题,保持群体多样性从而了避免种群退化。仿真实验证明,该算法在并行机群上运行时,加快了聚类算法的计算速度,提高了聚类质量。

关键词:并行聚类;自适应微粒群优化;任务并行;异步通信
中图分类号:TP18　　文献标志码:A

A parallel cluster algorithm for self-adaptive particle swarm optimization

WANG Hua-qiu[1,2], *LAO Xiao-feng*[1], *FENG Jin*[2]
(1. School of Computer Science, Chongqing University, Chongqing 400030, P. R. China;
2. Computer College, Chongqing Institute of Technology, Chongqing 400050, P. R. China)

Abstract: Full-scale data mining, such as in cluster problems, requires large numbers of computations. A parallel cluster algorithm for self-adaptive particle swarm optimization was proposed to deal with this problem. The proposed parallel particle swarm optimization algorithm reduced the impact of the initial conditions via parallel searches of the globally best position amongst a varied population. Task parallelization and partially asynchronous communication of the algorithm were employed to decrease computing time. Furthermore, if combined with the characteristics of self-adaptive and dynamical optimization parameters of the parallel particle swarm algorithm, the problems of particle mobility loss and the end of evolution could be dealt with successfully. When modified thusly, the algorithm maintains individual diversity and restrains degeneration. The simulation experiments indicate the algorithm helps increase computing speed and improve cluster quality.

Key words: parallel cluster; self-adaptive particle swarm optimization; task parallelization; asynchronous communication

微粒群算法是一种基于群体进化计算的算法[1-2],微粒群成为当前一个活跃的研究领域[2-4]。微粒群应用到许多实际应用中,包括求解 NC 难题[5],生物化学过程识别[6],电力能源调度优化[7],神经网络训练和地震分析[2],等。但是,在一些工程应用中,限制求解的高效性的两个主要的障碍是经常遇到的。首先,大规模问题在计算上是经常出现的,在时间和硬件上需要大量的资源去求解,其次,

收稿日期:2008-01-23
基金项目:重庆市自然科学基金资助项目(CSTC2007BB2406)
作者简介:王华秋(1975-),男,重庆工学院副教授,重庆大学博士后,主要从事计算智能、先进控制方面的研究。
廖晓峰(联系人),男,重庆大学教授,博士生导师,(Tel)023-65111874;(E-mail)liaoxf@cqu.edu.cn.

多峰的局部极小和数值噪音常常给工程优化问题带来麻烦，这就需要使用诸如基于群体的全局寻优方法及其改进算法来得到可靠的结果。

针对第 1 个障碍，研究的微粒群优化算法是一种新的全局优化算法，它特别适合于连续变量问题。在工程学领域它常常被用于大规模计算问题，由于是基于群体的方法，容易进行并行化，通过利用机群技术和网络技术，采用 MPI 的通信方式用于开发并行微粒群优化算法，降低了微粒群计算时间消耗。针对第 2 个障碍，其原因就是群体逐渐失去迁移性而停止进化，其结果是使两代之间很相似，然而这种“过分”地相似，有较大程度的局限性，于是提出了一种自适应微粒群算法用于克服局部极小问题。

由于聚类处理对象多为海量数据库和高维数据类型，算法计算的时间和空间复杂性很高。聚类算法一般分为硬聚类算法与模糊聚类算法。基于这样的事实，提出了多种模糊聚类算法，并成功地应用于许多领域。

1）硬聚类并行算法：这类算法在统计和数据库领域得到大量的研究和应用，这些算法往往采用数据并行的设计思想，把局部聚类模式分布在各个计算节点中，以便可以并行地进行聚类模式匹配。文献[8]中指出，面向大规模数据库系统的 BIRCH 算法、处理非数值属性聚类的 CACTUS 算法[9]、处理空间数据的 STING 算法[10]、子空间聚类算法 ENCLUS[11]等都是用并行处理的聚类算法。同时许多相关的工作对并行聚类算法也做了很有意义的探讨和研究。但早期的算法并不完全适合机群系统。由于是基于共享存储系统等低通信延迟系统的，这些并行聚类算法在设计时就没有把通信代价作为重点来考虑。机群系统的通信机制一般是 MPI、PVM 等，基于 TCP/IP 和 UDP 通信协议的消息传递机制，所以通信延迟非常大，一般达到 ms[12]。因此如何在机群环境中有效地减少并行算法通信次数是提高算法效率的重点。文献[13]提出了一种结合数据并行和任务并行的并行聚类算法。

2）模糊聚类并行算法：Lamehamedi 等人提出了并行的 FCM 算法[14]，并行算法采用主/从模式，由一个主进程控制多个从进程共同完成计算。文献[15]提出在分布式互联 PC/工作站环境下的区间数据的并行模糊聚类算法。这些并行模糊聚类算法都采用了数据并行方式计算，将海量的数据分块存储在并行每个节点上，节点只需计算相应的分块数据，并通过并行机群之间的通信相互交换计算结果，以此来缩短计算时间，但是问题是数据如何划分直接影响到聚类效果，如果数据划分不合理，原本属于一类的被划分到了不同的节点上，或者原本不属于一类的被划分到了同一节点上，这样必然降低聚类的质量。

大规模的数据挖掘如聚类问题迫切需要大量计算，甚至在高档处理器上，这样的计算也会导致长时间的求解。为了既快速获得计算的结果，又要提高聚类的质量，研究了一种基于并行自适应微粒群优化的并行聚类算法。

1　自适应微粒群算法

如果粒子被缩减到一维，利用向量符号粒子的轨迹将以简单的图线显示出来。不需要任何信息，两个等式将被转换为一个式子

$$v^{k+1} = wv^k + \phi(p - x^k), \tag{1}$$

其中：p 是两个最好的粒子的加权平均值；ϕ 是分布在区间$[0, (c_1 + c_2)]$的随机数值，它的平均值是$(c_1 + c_2)/2$。

由于数 ϕ 是随机的，粒子准确的位置在每次迭代时改变，为了简化计算，设置 ϕ 为一个常数。在加权值 p 达到最佳点和 ϕ 被设置为常数$(c_1 + c_2)/2$时，粒子的轨迹能被描绘出来，并且被研究。微粒群优化算法的缩减公式如下

$$v^{k+1} = wv^k + \phi y^k, y^{k+1} = -wv^k + (1-\phi)y^k, \tag{2}$$

其中 $y^k = p - x^k$，注意简化模型并不意味着标准微粒群的计算环境发生了改变。一般说来，种群的大小超过 1，并且很少有粒子少于 1 维的。

而且，加权值 p 不是静态的而是动态的，经常表现出复杂的行为来。这个简化模型仅仅用于提供一些每个粒子如何在没有相互作用的情况下的行为的信息。

然而，基于这个模型，当 p 的波动被限制时，每个粒子的稳定性被精确计算出来。既然一个全局最优问题的解限制在一定的搜索空间内（可能的区域），因此 p 的波动也应被限制。因而这个模型能被用于全局优化问题求解过程中的每个粒子稳定性的演变分析。

对于这个降维后的确定性系统可以写成如下形式的矩阵方程

$$\begin{bmatrix} v^{k+1} \\ y^{k+1} \end{bmatrix} = \begin{bmatrix} w & \phi \\ -w & 1-\phi \end{bmatrix} \begin{bmatrix} v^k \\ y^k \end{bmatrix} = \boldsymbol{M} \begin{bmatrix} v^k \\ y^k \end{bmatrix} \begin{bmatrix} v^n \\ y^n \end{bmatrix} = \boldsymbol{M}^n \begin{bmatrix} v^0 \\ y^0 \end{bmatrix}, \tag{3}$$

其中 $\boldsymbol{M}$ 是系统矩阵，

$$\boldsymbol{P}^{-1}\boldsymbol{M}\boldsymbol{P} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}, \boldsymbol{M}^n = \boldsymbol{P} \begin{bmatrix} \lambda_1^n & 0 \\ 0 & \lambda_2^n \end{bmatrix} \boldsymbol{P}^{-1}。$$

粒子在第 k 次迭代时的状态就如下

$$\begin{bmatrix} v^k \\ y^k \end{bmatrix} = \boldsymbol{P} \begin{bmatrix} \lambda_1^k & 0 \\ 0 & \lambda_2^k \end{bmatrix} \boldsymbol{P}^{-1} \begin{bmatrix} v^0 \\ y^0 \end{bmatrix}, \tag{4}$$

其中 $\boldsymbol{P}$ 是对角变换矩阵。λ_1，λ_2 是 $\boldsymbol{M}$ 的特征值

$$\lambda_1, \lambda_2 = \frac{w+1-\phi \pm \sqrt{(w+1-\phi)^2 - 4w}}{2}。$$

这个降维系统的动态特性由 $\boldsymbol{M}$ 决定，这样基于线性系统的稳定性理论，每个粒子的稳定性可以被评价。根据稳定性理论，如果 $|\lambda_1|<1$，$|\lambda_2|<1$，粒子的行为特性是稳定的。除了讨论上述降维系统的稳定性之外，也分析更多的细节分析比如波动周期分析。在 $w+1-2\sqrt{w}<\phi<w+1+2\sqrt{w}$ 情况下，λ 成了一个复数，下面的关系成立：$|\lambda|=\sqrt{w}$. 即是说，降维系统是否稳定取决于参数 w，换言之，降维系统的动态性调节仅相当于调节参数 w。

使用一个参数 $0<k_1<1$，在如下条件下

$w+1-2\sqrt{w}<\phi<w+1+2\sqrt{w}$，

ϕ 表示为

$$\phi = w+1-2\sqrt{w}+4k_1\sqrt{w}。 \tag{5}$$

当 λ 是复数时，它的极坐标为

$$\lambda^k = |\lambda|^k \mathrm{e}^{k\theta},$$

其中 θ 为角频率。这样就可以由上式得出系统的波动周期

$$T = \frac{2\pi}{\theta} = \frac{2\pi}{\tan^{-1}\dfrac{\mathrm{lm}}{\mathrm{Re}}} = \frac{2\pi}{\tan^{-1}\dfrac{\sqrt{4w-(w+1-\phi)^2}}{w+1-\phi}}, \quad (\phi < w+1), \tag{6}$$

其中 $\mathrm{Re}=\dfrac{w+1-\phi}{2}$，$\mathrm{lm}=\dfrac{\sqrt{4w-(w+1-\phi)^2}}{2}$。

通过替换 $w+1-\phi=2\sqrt{w}-4k_1\sqrt{w}$，方程变为

$$T = \begin{cases} \dfrac{2\pi}{\tan^{-1}\dfrac{2\sqrt{k_1-k_1}}{1-2k}}, & (k_1<0.5); \\ \dfrac{2\pi}{\pi+\tan^{-1}\dfrac{2\sqrt{k_1-k_1}}{1-2k}}, & (k_1>0.5)。 \end{cases} \tag{7}$$

通过用另外一个参数 $0<k_2<1$，得

$c_2/c_1 = k_2/(1-k_2)$，且 $c_1 = 2(1-k_2)\phi$，$c_2 = 2k_2\phi$。

以上的结果提供了第 i 个粒子的一个投影位置，定义为

$$\begin{aligned} v_{ij}^{k+1} = {} & w \cdot v_{ij}^k + 2(1-k_2)\phi \cdot \mathrm{rand}() \cdot (p\mathrm{best}_{ij} - x_{ij}^k) + 2k_2\phi \cdot \mathrm{rand}() \cdot (g\mathrm{best}_j - x_{ij}^k) = \\ & w \cdot v_{ij}^k + 2(1-k_2)(w+1-2\sqrt{w}+ \\ & 4k_1\sqrt{w}) \cdot \mathrm{rand}() \cdot (p\mathrm{best}_{ij} - x_{ij}^k) + \\ & 2k_2(w+1-2\sqrt{w}+4k_1\sqrt{w}) \cdot \mathrm{rand}() \cdot \\ & (g\mathrm{best}_j - x_{ij}^k), \end{aligned} \tag{8}$$

其中 $\phi=w+1-2\sqrt{w}+4k_1\sqrt{w}$ $k_2=\dfrac{c_2}{c_1+c_2}$。

通过上述研究，这样微粒群优化算法的降维模型和参数 w，k_1，k_2 之间的搜索轨迹的定性关系可以总结出如下规律。

1）w：如果 $0\leqslant w<1$，降维系统的收敛趋势将随着 w 减小而加强，如果 $w>1$，降维系统的收敛趋势将随着 w 增大而加强。

2）k_1：根据三角函数的性质推导可知，当 k_1 增大时，系统的角频率 θ 就会增加，因此系统的波动周期 T 将增大。当 $0<k_1<1$ 时，降维系统的动态性将随着 k_1 增大而变得波动。

3）k_2：当 $0<k_2<1$ 时，k_2 直接影响系数 c_1，c_2，因此如果 $k_2\approx 0$，稳定点 p 将在局部最优点附近，如果 $k_2\approx 1$，稳定点 p 将在全局最优点附近。

这里给出这几个参数的选择方法，根据各个参数的作用不同设计了 w，k_1，k_2：w 度量系统收敛趋势；k_2 度量靠近局部最优解或全局最优解的程度；k_1 度量粒子的波动性的增加对系统收敛的影响。

在算法运行初期，由于粒子之间的差异较大，局部极值 $p_i\mathrm{best}$ 和个体 x_i 之差一般比较大，惯性因子 w 接近于 1，让系统缓慢收敛并加强局部搜索能力，同理，全部极值 $g\mathrm{best}$ 和局部极值 $p_i\mathrm{best}$ 之差一般比较大，k_2 接近于 0，让微粒群在局部范围内搜索，k_1 的值设计得较大，使得系统波动性在运行初期较强。

粒子在位置更新与速度更新的过程中，如果发现新位置优于个体极值，则将个体极值设置为新位置；同样，如果新位置优于全局极值，则将全局极值设置为新位置 $g\mathrm{best}$。微粒群在运动方程式的作用下向 $g\mathrm{best}$ 靠拢，其间将不断更新个体极值。

对于微粒群中的任意粒子，其最终收敛位置将是整个微粒群找到的全局极值，k_1 随着迭代次数 n

的增加以指数下降，使得系统逐渐趋于稳定。

当算法接近收敛时，局部极值和全局极值的差距逐渐减小并趋向于 0，粒子的波动性减小，惯性因子 w 和波动性因子 k_1 自动调整逐渐接近 0，k_2 逐渐接近 1，使得系统尽可能收敛于全局最优位置。根据以上分析，在上述微粒群运动轨迹的约束条件下，采用如下自适应的方法设计了这几个重要参数

$$\left.\begin{aligned} w &= 1-\exp\left(-\frac{\| p_i\mathrm{best}-x_i \|^2}{2}\right),\\ k_1 &= k_0\exp\left(-\frac{i}{\tau_0}\right),\\ k_2 &= \exp\left(-\frac{\| p_i\mathrm{best}-g\mathrm{best} \|^2}{2}\right)。\end{aligned}\right\} \quad (9)$$

2　并行自适应微粒群聚类算法

由于聚类算法本身复杂度高，收敛速度较慢，因此在机群环境下的并行聚类算法采用任务并行的思想。在聚类算法中，设 M_C 为类的个数，即数据应该聚为多少类。一个粒子代表 M_C 个聚类中心，这样，每个粒子 x_i 就可以如下构造：

$$x_i = (m_{i1}, \cdots, m_{ij}, \cdots, m_iM_C).$$

其中：m_{ij} 表示第 i 个粒子所代表的第 C_{ij} 类的第 j 个聚类中心。因此，一群粒子代表了多个候选中心，微粒群的适应度很容易用量化误差度量出来

$$f_i = \frac{\sum_{j=1}^{M_C}\left[\sum_{\forall Z_p\in C_{ij}} d(z_p,m_j)/\mid C_{ij}\mid\right]}{M_C}, \quad (10)$$

其中：d 是分配给某一类的样本点和该类中心之间的 Euclidean 距离；$|C_{ij}|$ 是属于 C_{ij} 类的样本个数，即是给定类的频数。

利用自适应微粒群优化算法，提出了一种全局优化的并行算法用于聚类，样本数据可以按照如下的方法进行聚类。

1）主机 Master 将样本数据传到机群的从机 Slaver 上；

2）为了保证能对每个从机 Slaver 独立地进行聚类，减少了聚类算法同全局聚类结构的相关性，从机 Slaver 初始化每个粒子，使之随机地形成 M_C 个聚类中心，对每个从机 Slaver 独立地进行聚类，因为要获得全局最佳聚类结果，必须通过通信，从其它从机 Slaver 收集信息。所以让每个从机 Slaver 维护一份自己的的聚类模式。

3）为了减少各个从机 Slaver 之间的通信频率，提高并行实现的性能，采用了部分异步并行实现。设置局部/全局迭代率 s，从机 Slaver 以部分异步并行的方式进行 s 次聚类。每个从机 Slaver 上运行一定数量的工作进程，这些从机 Slaver 独立地执行顺序算法的迭代过程，经过一定迭代数 s 以后，所有的从机 Slaver 需要进行一次全局的同步，即主机 Master将对全局最优轨迹进行全局更新。考虑到如果局部迭代次数过多，每次局部迭代中优化解会被忽略，因此 s 不能设置得过大。采用这种通信方式以后，至少在局部迭代的过程中不必进行同步和通信，因此通信的费用可以在一定程度上减少。这样并行微粒群算法的加速比就有所增加表示为

$$S(m,N) = \frac{O(m^3)}{O(m^3/N)+T_{\mathrm{ovh}}(m,N)/s}。 \quad (11)$$

4）每个从机 Slaver 的粒子群 i 对于每个样本数据 z_p，计算样本到 C_{ij} 的 Euclidean 距离；根据

$$d(z_p,m_j) = \min_{\forall C=1,\cdots,M_C}\{d(z_p,m_{ij})\},$$

分配 z_p 到相应的类 C_{ij} 中去；然后根据公式

$$f_i = \frac{\sum_{j=1}^{M_C}\left[\sum_{\forall Z_p\in C_{ij}} d(z_p,m_j)/\mid C_{ij}\mid\right]}{M_C},$$

计算微粒群的适应度 f_i；通过比较，从机 Slaver 更新微粒群的本地最优位置 x_i；在从机 Slaver 利用式(9)调整自适应参数 w，k_1，k_2。

5）把整个任务分成若干子任务，同时保证聚类算法同数据特性的无关性，即在聚类过程中，不要求输入某些特定的数据；输入任何的数据都可以保证聚类过程正常地进行。从机 Slaver 输出各自的聚类结构，即微粒群的适应度，主机 Master 执行阻塞同步以获得所有从机 Slaver 并行粒子的适应度结果，比较从机 Slaver 传递的本地最佳适应度，得到对应于全局最佳聚类模式的全局最佳适应度，从而引导微粒群移动到最佳位置。

6）通过比较，主机 Master 更新全局最优位置，判断新的全局最优位置的适应度是否符合迭代终止条件，若满足，则终止计算，得到最佳的聚类中心；否则转入下一步。主机 Master 将全局最优位置广播从机 Slaver 上；

7）从机 Slaver 获得当前全局最优粒子的位置，转入第 3 步。

为说明其计算的快速性和准确性，进行了算法实验。

3　实验与对比分析

采用 8 台联想奔月 2000/1G 的计算机作为工作

站,1台DELL服务器作为主结点,用于获得并行自适应粒子群优化聚类算法结果。采用了MPI部分异步并行实现方式,考虑到每次局部迭代中较好的解可能被其它的从机Slaver忽略,设置局部/全局迭代率为$s\leqslant 3$。用于测试的聚类问题如下:一共随机生成6万条[-1,1]数据,约4M,且$x_1,x_2\sim U(-1,1)$,如果$x_1\geqslant 0.7$或者$x_1\leqslant 0.3$且$x_2\geqslant -0.2-x_1$,那么x_1,x_2属于第A类,否则属于B类。

3.1 聚类质量和计算时间比较

比较了用不同节点机群的并行自适应微粒群聚类算法和并行FCM[14]对上述数据集进行聚类的结果,主要目的是比较各自聚类的质量,而质量的评判标准是如下3种。

1)量化误差度量f_i所定义的聚类质量,

$$f_i=\frac{\sum_{j=1}^{M_C}\left[\sum_{\forall Z_p\in C_{ij}} d(z_p,m_j)/\mid C_{ij}\mid\right]}{M_C}。$$

2)簇内距离尽量小,即属于同一簇内的数据向量之间的距离尽可能地最小。

3)簇间距离尽量大,即各个聚类中心之间距离尽可能地最大;满足后面的两个目标可以形成紧凑的聚类。

节点的优化的聚类结果能否体现实际的全局聚类结果,或者说任务并行聚类算法的聚类质量如何呢?下面做了实验来验证这种并行聚类算法。

表1 几种并行聚类算法性能比较

并行聚类算法	平均计算时间/s	聚类误差	簇内距离	簇间距离
8节点并行FCM	117	0.76	1.77	3.67
4节点并行APSO	60	0.58	0.94	4.86
8节点并行APSO	49	0.46	0.73	5.21

从表1可以得出如下结论:

1)随着数据量的增大,节点的聚类结果趋近于全局优化,因为数据量越大,个别数据的影响越小。2)在聚类过程中,通过一定量的同步通信来使各个节点得到优化的聚类模式,从而改善最后的聚类质量。3)随着机群节点的增加,内存也会相应地增加,这对改善聚类质量也有很大的帮助。所以说,采用任务并行的思想是完全可行的。

各节点优化的聚类结果也是能够反映实际的聚类模式的,其聚类质量是可以接受的。并行化对于算法的执行不会带来负面影响,甚至并行计算的结果最终得到更准确的解。这说明在机群环境下,设计良好的数据并行聚类算法不但可以获得较高的加速比,而且可以得到良好的聚类质量。

3.2 并行计算性能分析

3.2.1 相对工作、通信和闲置时间比较

从表2中可以看出,对于较大规模的机群,由于消息等待造成的闲置时间占居优势,随着机群规模的减小,工作时间所占比例显著提高,由于并行实现算法降低了各个进程间的通信频率,从而减少了闲置时间,而通信时间可以忽略不计。机群节点个数对通行比造成一定的影响。随着机群节点个数的增加,不仅通信量在增加,节点进程的闲置时间也在增加。另一方面,由于聚类任务被各个机群节点分担了,计算时间却在减少,从而导致通信比下降。

表2 节点并行计算的时间

节点个数	通信时间/s	工作时间/s	通信比	闲置时间/s	总时间/s
$P=0$	0	98	—	—	98
$P=2$	3	56	18.67	3	62
$P=4$	9	42	4.67	6	57
$P=6$	13	32	2.46	7	52
$P=8$	17	23	1.35	8	48

注:表中$P=0$指只有主节点执行串行计算。

3.2.2 加速比和效率比较

表3中显示了加速比、效率与处理机数目之间的关系。加速比=单机执行的时/n机执行的时间;效率=加速比/机数。

表3 节点并行算法的性能分析

节点个数	加速比	效率
$P=2$	1.56	0.78
$P=4$	1.70	0.43
$P=6$	1.86	0.31
$P=8$	2.02	0.25

从表3可以看出,对于规模较大的测试数据而言,加速比接近于优化值,即处理机数目,效率以非常慢的速度减少。这说明在并行实现算法中,工作时间随着处理机数目的增多而减少说明其利用率降低。因此为了找到最优的处理机数目,需要在加速比增加和效率减小之间找到平衡点。

4 结　论

数据源常常包含海量数据，聚类算法计算时需要大量的 I/O 开销和足够的内存空间，从而成为算法的可扩展性和响应时间的瓶颈。虽然已有一些并行聚类算法，但是一般采用数据并行方式计算，对数据如何合理划分并没有进行考虑，因此聚类质量不高。为了快速获得计算的结果和提高全局搜索的能力，研究了一种基于自适应微粒群优化这种全局寻优方法的并行的聚类算法。通过从多种群并行地开始搜索，基于群体搜索技术的微粒群优化算法减少了初始条件的影响，采用并行自适应微粒群算法进行聚类计算，克服了群体逐渐失去迁移性而停止进化的问题，收敛速度较快，加快了聚类算法的计算速度，提高了聚类质量。实验证明了该算法的有效性。

参考文献：

[1] JUANG C F. A hybrid of genetic algorithm and particle swarm optimization for recurrent network design [J]. IEEE Transactions on Systems, Man and Cybernetics Part B: Cybernetics, 2004, 34(2): 997-1006.

[2] MATEO S. Swarm intelligence [M]. California: Morgan Kaufman, 2001.

[3] CLERC M, KENNEDY J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space [J]. IEEE Transactions on Evolution, Computing, 2002, 6: 58-73.

[4] TASGETIRN M F, LIANG Y C, SEVKLI M, et al. Particle swarm optimization algorithm for makespan and total flowtime minimization in permutation flowshop sequencing problem [EB/OL]. (2004-11-18) [2006-05-21] http://www.fatih.edu.tr/~ftasgetiren/download/EJOR_FTASGETIREN.

[5] TANDON V, EL-MOUNAYRI H, KISHAWY H. NC and milling optimization using evolutionary computation [J]. International Journal of Machine Tools and Manufacture, 2002, 42: 595-605.

[6] COCKSHOTT A R, HARTMAN B E. Improving the fermentation medium for echinocandin B production part II: particle swarm optimization [J]. Process Biochemistry, 2001, 36: 661-669.

[7] ABIDO M A. Optimal power flow using particle swarm optimization [J]. Electrical Power and Energy Systems, 2002, 2: 563-571.

[8] 周兵，沈均毅，彭勤科. 集群环境下的并行聚类算法[J]. 计算机工程，2004, 30(4): 4-7.
ZHOU BING, SHEN JUN-YI, PENG QIN-KE. Parallel clustering algorithm for PCs cluster [J]. Computer Engineering, 2004, 30(4): 4-7.

[9] ALLEN G, BENGER W, DRAMLITSCH T, et al. Cactus tools for grid applications [J]. Journal of Cluster Computing, 2001 (4): 179-188.

[10] MA D Y, ZHANG A D. An adaptive density-based clustering algorithm for spatial database with noise[C]// Proceedings Fourth IEEE International Conference. Brighton, UK: IEEE Computer Society, 2004: 467-470.

[11] WASHIO T, MITSUNAGA Y, MOTODA H. Mining quantitative frequent itemsets using adaptive density-based subspace clustering [C]// Fifth IEEE International Conference on Data Mining. Houston, Texas: IEEE Computer Society, 2005: 27-30.

[12] LIU B. A fast density-based clustering algorithm for large databases [J]. Journal of Machine Learning and Cybernetics, 2006, 8: 996-1000.

[13] BOUTSINAS B, GNARDELLIS T. On distributing the clustering process [J]. Pattern Recognition, 2002, 23: 999-1008.

[14] LAMEHAMEDI H, BENSAID A D, KEBBAL E G, et al. Adaptive programming: application to a semi-supervised point prototype clustering algorithm [C]// International Conference on Parallel and Distributed Processing Techniques. Nevada: CSREA Press, 1999: 2753-2759.

[15] 陆建江，徐宝文. 区间数据的并行模糊聚类算法[J]. 东南大学学报：自然科学版，2003, 33(4): 406-409.
LU JIAN-JIANG, XU BAO-WEN. Parallel fuzzy clustering algorithm for interval data [J]. Journal of Southeast University: Natural Science Edition, 2003, 33(4): 406-409.

（编辑　赵　静）