文章编号: 1001-0920 (2016) 10-1879-05 DOI: 10.13195/j.kzyjc.2015.1040

基于多种群协同微粒群优化的流数据聚类算法

张 勇, 夏长红, 巩敦卫, 荣 淼

(中国矿业大学 信息与电气工程学院，江苏 徐州 221116)

摘 要: 针对流数据的实时、有序和维数高等特点, 提出一种基于多种群协同微粒群优化的流数据聚类算法. 该算法利用变量分而治之的思想, 多个种群协同优化多个类中心, 进而求出问题完整的类中心集合. 给出一种类中心变化趋势的预估策略, 以快速追踪环境变化. 为防止多个子微粒群同时优化一个类中心, 提出一种相似子微粒群的合并策略. 最后将所提出的算法用于多个数据集, 实验结果验证了算法的有效性.

关键词: 流数据；协同微粒群；聚类；预估

中图分类号: TP273 **文献标志码:** A

Streaming data clustering using cooperative particle swarm optimization

ZHANG Yong, *XIA Chang-hong*, *GONG Dun-wei*, *RONG Miao*

(School of Information and Electrical Engineering，China University of Mining and Technology，Xuzhou 221116, China．Correspondent：XIA Chang-hong，E-mail：xch@cumt.edu.cn)

Abstract: Focusing on the stream data real time performance, orderliness, and high dimension, a streaming data clustering algorithm based on cooperative particle swarm optimization is proposed, which divides sequential stream data into several data subsets according to the time stamp. For any data subset, the high-dimensional clustering problem is firstly transformed into the low dimensional sub-problem with only one class center. Then, one sub-swarm optimizes one clustering sub-problem independently, and all the sub-swarms cooperate with each other to find the whole solution of the streaming data. Moreover, in order to enhance the speed of tracking the environment changes, a forecast strategy is designed to predict the change trend of class centers. In order to avoid multiple sub-swarms repeatedly searching for the same class center, a merging strategy of similar sub-swarms is proposed. Finally, the proposed algorithm is applied to multiple data sets, and experimental results show the effectiveness.

Keywords: stream data；cooperative particle swarm optimization；clustering；forecast

0 引 言

流数据是一种实时连续的数据序列, 聚类是数据处理的重要技术之一. 针对传统的静态聚类问题, 学者们提出了众多聚类方法[1]. 然而, 流数据聚类不同于传统的数据聚类, 其数据是动态的, 具有规模大、聚类结果随时间变化等特点, 处理难度大.

微粒群优化 (PSO) 是一种受鸟群或鱼群觅食行为的启发而产生的全局搜索方法, 由于具有易于实现、收敛速度快等优点, 近年已广泛应用于静态数据聚类问题[2-3]. 最近, 学者们开始尝试将其用于流数据聚类问题. Yingmei 等[4]分析了群体智能算法在流数据聚类中的应用, 指出该算法能有效抑制流数据聚类问题中的噪声. Ke 等[5]提出了基于网格和微粒群优化的流数据聚类算法. Elsayed 等[6]提出了基于网格和密度混合的微粒群流数据聚类方法. 这些都展现出微粒群在处理流数据聚类问题时的优势. 然而, 由于问题的复杂度和数据维数呈指数关系, 聚类问题存在维数灾难现象. 协同进化是受自然界物种之间相互协作启发而产生的[7], 其思想是, 通过对变量空间的分割, 将一个复杂的多变量优化问题分解为多个简单的少变量的子问题, 再利用若干子种群同时优化这些子问题. 由于可以显著缩小子种群的搜索空间, 提高算法的搜索效率, 近年来学者们开始尝试将其用于复杂优化问题[8-9]. Jiang 等[10]尝试将其用于高维聚类问题, 并通

收稿日期: 2015-08-18；**修回日期:** 2016-03-16.

基金项目: 国家自然科学基金项目(61473299)；中国博士后科学基金项目(2014T70557, 2012M521142)；江苏省博士后科学基金项目(1301009B).

作者简介: 张勇 (1979–), 男, 副教授, 博士, 从事数据挖掘、群体智能等研究；夏长红 (1990–), 女, 硕士生, 从事智能优化的研究.

过实验验证了该类方法的有效性. 然而, 所提出的方法只适用于静态数据, 无法应用于具有实时快速变化的流数据.

本文针对高维流数据聚类问题, 提出一种基于多种群协同微粒群优化的流数据聚类算法. 利用协同进化思想, 将高维流数据聚类问题划分为多个简单的子优化问题, 以降低问题的复杂度. 当环境变化时, 采用预测技术估计类中心的变化趋势, 引导微粒群的进化, 以加快算法对环境的响应速度. 引入相似种群合并、冗余微粒随机初始化策略, 以提高算法的性能. 最后将所提出的算法用于多个数据集, 实验结果验证了算法的有效性.

1 问题描述

流数据是动态变化的, 一般可定义为带有时间窗 $\{t_1, t_2, \cdots, t_m, \cdots\}$ 的数据块 $\{Z_1, Z_2, \cdots, Z_m, \cdots\} \in R^d$, 其中 d 为数据的维度, 用 $A_1, A_2, \cdots, A_d$ 描述属性, 每块都可以在内存中进行处理. 以数据块 Z_m 为例, 具体表示为

$$Z_m = (z_1, z_2, \cdots, z_N).$$

其中: $z_i = (z_{i1}, z_{i2}, \cdots, z_{id})$, $i = 1, 2, \cdots, N$, N 为样本数目.

对按序到达的数据块进行聚类, 采用基于欧氏距离的相似度指标评价类内样本的相似性. 具体地, 样本 z_i 与 z_j 之间的欧氏距离可以表示为

$$D(z_i, z_j) = \|z_i, z_j\| = \sqrt{\sum_{g=1}^{d}(z_{ig} - z_{jg})^2}. \tag{1}$$

基于式 (1), 关于数据块 Z_m 的聚类问题描述为

$$\min \sum_{k=1}^{K}\sum_{i=1}^{N} D(z_i, c_k). \tag{2}$$

其中: c_k 为第 k 类的中心, $D(z_i, c_k)$ 为第 k 类中第 i 个样本到其类中心的欧氏距离. 可以看出, 基于类中心的数目, 式 (2) 可以分解成如下 K 个子问题:

$$\Big\{\min\Big(\sum_{i=1}^{n_1} D(z_i, c_1) + \sum_{i=1}^{n_2} D(z_i, c_2) + \cdots + \sum_{i=1}^{n_K} D(z_i, c_K)\Big)\Big\}, \tag{3}$$

其中 $n_1 + n_2 + \cdots + n_K = N$.

使用协同微粒群优化处理上述问题, 必须考虑如下内容: 1) 问题被划分成 K 个子问题后, 如何编码实现对问题搜索空间的分割; 2) 优化变量分散于 K 个子问题中, K 个子微粒群必须相互协作, 如何交换各个子问题之间的信息; 3) 如何保证不同微粒群优化不同的类中心; 4) 环境变化时, 如何加快算法对环境变化的响应速度.

2 基于多种群协同微粒群的流数据聚类算法

针对按序到达的数据块, 利用变量分割思想将一个高维聚类问题转化为多个仅包含一个类中心的低维子聚类问题, 一个子微粒群优化一个子聚类问题, 多个子微粒群协同进化, 进而求出问题完整的类中心集合.

2.1 子微粒群的编码

数据聚类的目的是找到问题的 K 个类中心. 基于式 (3), 上述高维流数据聚类问题可以分解为 K 个仅包含一个类中心的低维子聚类问题, 一个子微粒群优化一个子聚类问题. 针对每个子微粒群, 微粒的编码方式如下:

$$X_i = (x_1, x_2, \cdots, x_d),$$

其中 X_i 为第 i 个子微粒群优化的类中心. 相对于传统的采用一个编码串表示所有 K 个类中心的方法, 本文方法显著缩短了微粒的搜索空间, 搜索空间由 $K \times d$ 缩减到 d.

2.2 子微粒群的进化

采用具有较少控制参数的骨干微粒群优化算法[11], 更新微粒的位置, 具体公式如下:

$$\begin{aligned} &X_i(t+1) = N(\mu_i(t), \sigma_i(t)),\\ &\mu_i(t) = (r \times P_i(t) + (1-r) \times P_g(t))/2,\\ &\sigma_i(t) = \|P_i(t) - P_g(t)\|. \end{aligned} \tag{4}$$

其中: $X_i(t+1)$ 为当前子微粒群中第 i 个微粒的位置, $P_i(t)$ 为该微粒的个体极值点, $P_g(t)$ 为微粒群的全局极值点, t 为微粒群进化算法的迭代次数, r 为 [0,1] 之间随机数, $N(a, b)$ 为均值是 a 方差是 b 的高斯分布函数.

每次迭代时, 针对子微粒群中的任意新生微粒, 如果其适应值好于历史个体极值点, 则将该微粒位置作为新的个体极值点, 否则, 保持微粒的历史个体极值点不变. 在子微粒群中, 比较所有微粒的个体极值点, 将胜出者作为当前微粒的全局极值点.

2.3 微粒适应值的计算

由于每个子微粒群仅优化一个类中心, 评价一个子微粒群中微粒的适应值时, 需要事先构造完整的类中心集. 本文采用其他子微粒群的全局极值点, 与当前子微粒群中待评价微粒组成一个完整解. 以子微粒群 P_1 中第 i 个微粒 X_{1i} 为例, 假设其他子微粒群的全局极值点为 $G_{\mathrm{best}_2}, G_{\mathrm{best}_3}, \cdots, G_{\mathrm{best}_K}$, 微粒 X_{1i} 对应的完整解为 $(X_{1i}, G_{\mathrm{best}_2}, G_{\mathrm{best}_3}, \cdots, G_{\mathrm{best}_K})$, 将该微粒对应的完整解 (类中心集) 代入式 (3), 得到其适应值.

2.4 类中心的预测及其利用

为了更好地保持历史数据特征, 同时利用已有的统计信息预测类中心的变化趋势, 给出一种类中心的线性预测策略. 该策略利用前两个时刻得出的类中心 $\{c\}^{(t-1)}$、$\{c\}^{t}$ 预估 $t+1$ 时刻类中心 $\{c\}^{(t+1)}$ 的可能位置, 具体预测公式为

$$c^{(t+1)} = c^{t} + (c^{t} - c^{(t-1)}) + \varepsilon. \tag{5}$$

其中

$$\varepsilon = N(0,\sigma),\ \sigma = D(\{c\}^{t}, \{c\}^{(t-1)})^{2},$$
$$D(\{c\}^{t}, \{c\}^{(t-1)}) = \|\{c\}^{t}, \{c\}^{(t-1)}\|.$$

当一个子微粒群收到新的数据块时, 利用式 (5) 预测出新数据块可能的类中心, 在估计出的类中心附近重新初始化一定比例微粒的位置, 剩余微粒在整个空间中随机产生. 可以看出, 在预测类中心附近重新初始化部分微粒位置, 可以增强算法对可能最优解空间的搜索力度, 加快算法对环境变化的响应速度; 在整个空间中随机产生部分微粒, 可以保证微粒群的多样性, 防止算法局部收敛.

2.5 相似子微粒群的处理

由于不同子微粒群优化的类中心可能重叠, 传统协同进化方法存在多个子微粒群优化同一类中心的不足. 为引导不同子微粒群在搜索空间朝着不同的方向搜索, 给出一种相似子微粒群的合并策略. 对于任意两个子微粒群, 计算它们所得最优解之间的距离, 如果该距离小于两个类的半径之和, 则认为两个子微粒群相似. 此时, 随机保留一个子微粒群, 并初始化另一个子微粒群. 保留下来的种群继续开发现有类中心, 随机初始化的种群可以全力探索新的类中心.

2.6 算法的执行步骤

所提出流数据聚类算法步骤如下.

Step 1: 初始化, 随机产生 S 个维数为 $K \times d$ 的个体, 通过变量分割方式, 将这些个体平均划分到 K 个子微粒群, 每个子微粒群包含 S 个维数为 d 的微粒. 对每个子微粒群, 初始化微粒的个体极值点为其自身, 其全局极值点为在相应维数上适应值最小的个体位置.

Step 2: 对每个子微粒群作如下操作:

Step 2.1: 采用骨干微粒群优化算法更新子微粒群中微粒的位置;

Step 2.2: 由第 2.3 节方法计算微粒的适应值, 并更新问题的最优完整解;

Step 2.3: 由第 2.2 节方法更新微粒的个体极值点以及全局极值点;

Step 2.4: 由第 2.5 节方法, 判断与其他子微粒群的相似性, 并合并相似子微粒群;

Step 2.5: 检测算法是否符合结束条件, 如果满足, 则找到所有子微粒群中适应值最小的全局极值点, 其对应的完整解即为当前数据块的类中心集合, 并等待新数据集, 否则, 转至 Step 2.1;

Step 2.6: 判断数据流是否结束, 如果收到下一个时刻数据块, 则预测该时刻类中心的变化位置, 并由第 2.4 节方法初始化该子微粒群, 转至 Step 2.1, 如果一直未收到新的数据块, 则终止整个算法.

3 实验结果

选择 7 组人工流数据集和两组实际流数据集作为测试对象, 与 2 种已有聚类算法 (K-means 聚类算法[12]和流数据聚类算法 PSOCK[13]) 和 1 种微粒群优化算法 (骨干微粒群优化算法[11]) 进行比较. 为了将 K-means 算法应用于流数据, 每当新的数据到达时, 重新执行一次该算法. 与本文所提方法的区别在于: 采用一个微粒群优化所有类中心, 且没有类中心预测机制, 每一算法均采用相同的种群规模和迭代次数.

3.1 人工流数据集

以 UCI 提供的静态数据集为基础, 人工产生 7 组流数据. 表 1 展示了所选静态数据集, 这些数据集已在聚类中得到了广泛的应用[9,13].

表 1　所选数据集

数据集	聚类中心数 K	样本个数 n	维数	数据描述
Cancer	2	683	9	Wisconsin breast canner
Ionosphere	2	351	34	Radar observations
Iris	3	150	4	Fisher’s iris data
CMC	3	1 473	9	Contraceptive method choice
Glass	6	214	9	Glass identification data
Control	6	600	60	Synthetic control chart
Image	7	2 100	19	Image segmentation

将数据集扩展为动态流数据, 以数据集 Cancer 为例, 首先求出所有样本的平均位置 $C_m = (c_{m,1}, c_{m,2}, \cdots, c_{m,d})$, 以该平均值为基础, 产生一个随机量为

$$\Delta = 0.01 \times (10 + \mathrm{rand}) \times C_m.$$

每变化一个时间窗 $\{t_1, t_2, \cdots, t_m, \cdots\}$, Cancer 中样本位置即增加一个 Δ.

本文采用聚类正确率 CA、类内距离和 SD 作为聚类结果的评价指标, 聚类正确率为所有数据样本中正确划分的样本个数占总样本个数的比例. CA 取值在 [0,1] 之间, 值越高代表分类效越好; SD 越小代表聚类性能越好.

3.2 类中心预测方法分析

通过分析第 2.4 节种群初始化比例对算法性能的

表 2 Control 和 Glass 数据集所得聚类结果

数据集 预测比例	Control				Glass			
	CA		SD		CA		SD	
	mean	std	mean	std	mean	std	mean	std
0	67.666 7	0.654 4	2.23e+04	57.458 2	50.633 3	1.349 7	2.30e+02	2.428 2
0.1	67.219 3	0.306 3	2.24e+04	41.785 3	50.972 2	0.667 3	2.30e+02	1.279 6
0.3	66.282 5	0.479 2	2.26e+04	62.201 8	51.855 6	1.032 5	2.29e+02	2.397 5
0.5	69.099 4	0.39 7	2.22e+04	66.213 9	51.644 4	0.608 7	2.28e+02	1.489 9
0.7	**69.894 7**	0.272	**2.19e+04**	57.008 2	**51.994 4**	0.938 8	**2.22e+02**	1.684 9
0.9	68.070 2	1.121 5	2.20e+04	61.286 3	50.688 9	1.263	2.29e+02	2.749 4
1.0	68.471 9	1.310 8	2.20e+04	68.322 4	50.161 1	1.181 3	2.22e+02	5.918 8

表 3 不同算法所得聚类正确率

算法	PSOCK		BPSO		K-means		本文算法	
	mean	std	mean	std	mean	std	mean	std
Cancer	93.462 6	2.206 7	94.981	0.292	96.046 9	0	**96.561 4**	0.176 7
Ionosphere	64.486 6	3.263 1	63.814 8	2.344 1	**71.225 1**	0	67.647 7	0.933 3
Iris	89.376 2	4.369 6	89.533 3	1.568 2	79.066 7	16.579 6	**90.673 3**	1.262 6
CMC	39.665 7	0.823 6	39.820 1	0.121 7	39.341 5	0.107 3	**43.621 6**	0.239 8
Glass	47.156 1	2.329 7	48.509 3	0.750 1	49.813 1	3.349 5	**51.994 4**	0.938 8
Control	41.226 7	7.203 5	40.636 7	2.104 8	58.083 3	9.791 1	**69.894 7**	0.272
Image	14.964 3	1.890 6	16.323 8	1.087 5	50.852 4	4.725 1	**54.070 2**	0.496 7

表 4 不同算法所得聚类的类内距离和

算法	PSOCK		BPSO		K-means		本文算法	
	mean	std	mean	std	mean	std	mean	std
Cancer	4.36e+03	2.088 2	3.63e+03	46.254 1	1.93e+04	0	**2.96e+03**	1.802
Ionosphere	1.00e+03	23.334 2	1.02e+03	19.217 5	2.42e+03	0	**8.08e+02**	12.498 1
Iris	1.14e+02	6.642 8	1.02e+02	1.672 7	9.84e+01	31.269 7	**9.67e+01**	0.322 8
CMC	6.06e+03	10.367 1	5.79e+03	18.761 6	2.37e+04	6.451 6	**5.19e+03**	11.222 8
Glass	3.51e+02	10.549 3	3.25e+02	6.792	3.85e+02	69.795 8	**2.22e+02**	1.684 9
Control	4.15e+04	125.250 2	4.58e+04	539.333 1	1.03e+06	6.96e+04	**2.19e+04**	57.008 2
Image	2.92e+05	1.23e+03	2.76e+05	1.62e+03	1.41e+07	2.54e+06	**1.27e+05**	1.03e+03

影响, 验证预测方法的有效性. 表 2 为不同初始比例情况下, 所提出方法所得聚类正确率和类内距离等性能指标.

由表 2 可见, 对于数据集 Control 和 Glass, 当预测类中心初始化种群的比例为 70% 时, 本文算法所得聚类正确率 CA 的平均值最大, 类内距离和 SD 的平均值最小. 当没有预测机制, 即初始化比例为 0 时, 本文算法的性能最差. 鉴于此, 建议初始化比例设置在 0.7 左右.

3.3 算法比较

将本文算法与其他聚类算法进行比较, 验证其有效性. 表 3 为不同算法所得的聚类正确率, 表 4 为类内距离和.

由表 3 可见, 对于数据集 Cancer 和 Iris, 本文算法得到了最好的聚类正确率, K-means 算法得到了最差的聚类正确率. 对于数据集 Ionosphere, K-means 得到了最优的聚类正确率, 其次是本文算法, 最差的是 BPSO 算法. 对于数据集 CMC 和 Glass, 本文算法依然得到了最好的聚类正确率, K-means、PSOCK 和 BPSO 表现相差不太大. 对于数据集 Control 和 Image, 本文算法在聚类正确率上均表现最优, 而K-means、PSOCK 和 BPSO 的结果均较差.

由表 4 可见, 对于所有数据集, 所提出算法均得到了最优的类内聚类和, 且对于数据集 Iris、CMC、Glass、Control 和 Image, 本文算法所得方差值也小于其他算法; 对于数据集 Cancer 和 Ionosphere, K-means 算法的方差最优.

3.4 真实数据集

从 20 newsgroups 语料库[14]中选取 A2、B2、A4 和 B4 等 4 组不同结构的数据集, 文档样本总数为 4 216, 最大特征维数为 490. 网络入侵检测数据集采用 KDD-CUP99[15], 从 42 维属性中选取 34 维连续属性. KDD-CUP99 实验数据的样本数为 494 021, 维数为 34, 类中心个数为 40.

对于文本数据集和 KDD-CUP99 数据集, 图 1 和图 2 为不同算法所得的聚类正确率. 可以看出, 本文

算法取得了最优的聚类结果, 其次是BPSO算法和PSOCK, 最差的是 K-means 算法.

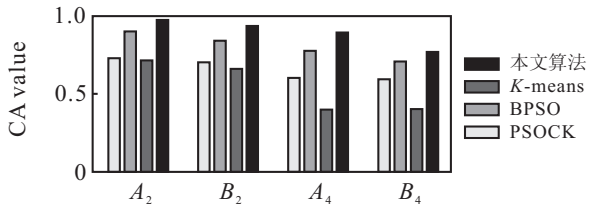

图 1　文本数据集实验结果

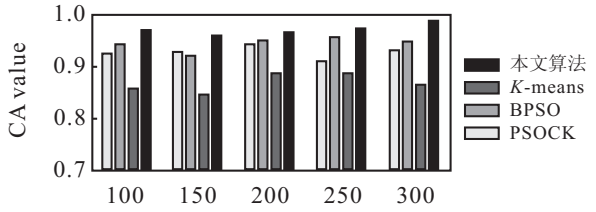

图 2　KDD-CUP99 数据集实验结果

4　结　论

本文提出了一种基于多种群协同微粒群优化的流数据聚类算法. 该算法将高维聚类问题转化为多个仅包含一个类中心的低维子聚类问题, 显著缩小了微粒的搜索空间. 与 K-means 聚类算法和PSOCK 聚类算法相比, 实验结果表明了所提出算法的有效性. 但是, 本文算法所考虑类中心的变化趋势较为平缓, 如何处理类中心变化不规则的情况, 是今后的研究重点.

参考文献(References)

[1] 孙吉贵, 刘杰, 赵连宇. 聚类算法研究[J]. 软件学报, 2008, 19(1): 48-61.
(Sun J G, Liu J, Zhao L Y. Clustering alorithms research[J]. J of Software, 2008, 19(1): 48-61.)

[2] 付柳强, 张洪伟, 徐开阔. 基于 K-means 的量子微粒群动态聚类[J]. 四川理工学院学报: 自然科学版, 2013, 26(6): 28-32.
(Fu L Q, Zhang H W, Xu K K. Quantum-behaved particle swarm dynamic clustering based on K-means[J]. J of Sichuan University of Science & Engineering: Natural Science Edition, 2013, 26(6): 28-32.)

[3] Aljarah I, Ludwig S A. Towards a scalable intrusion detection system based on parallel pso clustering using mapreduce[C]. Proc of the 15th Annual Conf Companion on Genetic and Evolutionary Computation. Cancun: ACM, 2013: 169-170.

[4] Yingmei L, Weining X, Yuyan H, et al. Research on stream data clustering based on swarm intelligence[C]. Int Conf on Computer Science and Network Technology. Harbin: IEEE, 2011, 1: 573-576.

[5] Ke L, Lin W. Data streams clustering algorithm based on grid and particle swarm optimization[C]. Int Forum on Computer Science-Technology and Applications. Chongqing: IEEE, 2009, 1: 93-96.

[6] Elsayed S M, Sarker R A, Essam D L. Multi-operator based evolutionary algorithms for solving constrained optimization problems[J]. Computers & Operations Research, 2011, 38(12): 1877-1896.

[7] Potter M A, De Jong K A. A cooperative coevolutionary approach to function optimization[M]. Parallel Problem Solving from Nature-PPSN III. Berlin: Springer-Heidelberg, 1994: 249-257.

[8] Li X, Yao X. Cooperatively coevolving particle swarms for large scale optimization[J]. IEEE Trans on Evolutionary Computation, 2012, 16(2): 210-224.

[9] 陶新民, 王妍, 徐晶, 等. 求解最小属性约简的病毒协同进化微粒群算法[J]. 控制与决策, 2012, 27(2): 259-265.
(Tao X M, Wang Y, Xu J, et al. Minimum rough set attribute reduction algorithm based on virus-coordinative discrete particle swarm optimization[J]. Control and Decision, 2012, 27(2): 259-265.)

[10] Jiang B, Wang N. Cooperative bare-bone particle swarm optimization for data clustering[J]. Soft Computing, 2014, 18(6): 1079-1091.

[11] Kennedy J. Bare bones particle swarms[C]. Proc of the IEEE Swarm Intelligence Symposium. Piscataway NJ: IEEE, 2003: 80-87.

[12] Benmounah Z, Meshoul S, Batouche M. Cooperative parallel multi swarm model for clustering in gene expression profiling[C]. Proc of the 5th Int Conf on Swarm Intelligence. Hefei, 2014: 450-459.

[13] MacQueen J. Some methods for classification and analysis of multivariate observations[J]. Proc of the 5th Berkeley Symposium on Mathematical Statistics and Probability, 1967, 1(14): 281-297.

[14] Cristian F, Feter C. Probabilistic internal clock synchronization[C]. Proc of the 13th Symposium on Reliable Distributed Systems. 1994, 10: 22-31.

[15] Stolfo S, Wenke Lee, Chan P K, et al. Datamining-based intrusion detectors: An overview of the Columbia IDSproject[J]. ACM SIGMOD Record, 2001, 30(4): 5-14.

(责任编辑：郑晓蕾)