文章编号: 1001-0920(2010)01-0053-06

基于并行二进制免疫量子粒子群优化的特征选择方法

朱颢东, 钟 勇
(1. 中国科学院 成都计算机应用研究所, 成都 610041; 2. 中国科学院 研究生院, 北京 100039)

摘 要: 为提高文本挖掘算法的运行速度,降低占用的内存空间,提出一种基于并行二进制免疫量子粒子群优化的特征选择方法. 该方法采用二进制免疫量子粒子群优化搜索特征子集,利用并行算法来提高时间效率,从而较快地获得较具代表性的特征子集. 实验结果表明该算法是有效的.

关键词: 特征空间; 特征选择; 并行二进制免疫量子粒子群优化
中图分类号: TP301 **文献标识码**: A

Feature selection method based on PBIQPSO

ZHU Hao-dong, ZHONG Yong
(1. Chengdu Institute of Computer Application, Chinese Academy of Sciences, Chengdu 610041, China; 2. The Graduate School, Chinese Academy of Sciences, Beijing 100039, China. Correspondent: ZHU Hao-dong, E-mail: zhuhaodong80@163.com)

Abstract: In order to enhance the operating speed and reduce the memory space occupied and filter out irrelevant or lower degree of features, a feature selection method based on the parallel binary immune quantum-behaved particle swarm optimization (PBIQPSO) is presented, which uses the binary immune quantum-behaved particle swarm optimization to select feature subsets and takes advantage of multiple computing nodes to enhance time efficiency, so can acquire quickly the feature subsets which are more representative. Experimental results show the effectiveness of the algorithm.

Key words: Feature space; Feature selection; Parallel binary immune quantum-behaved particle swarm optimization (PBIQPSO)

1 引 言

在文本分类中,文本通常以向量形式表示,其特点具有高维性和稀疏性[1]. 而在中文文本分类中,通常采用词条作为最小的独立语义载体,原始的特征空间可能由出现在文章中的全部词条构成. 由于中文词条总数有 20 多万条,这使得其高维性和稀疏性更加明显,大大限制了分类算法的选择余地,降低了分类算法的效率和精度. 为此,寻找一种高效的特征选择方法,以降低特征空间维数、避免维数灾难、提高文本分类的效率和精度,成为文本自动分类中亟待解决的重要问题[2].

现存诸多特征选择方法都是基于英文的,例如词频(WF),文档频(DF),信息增益(IG),互信息(MI),x^2 统计量(CHI)等[3,4]. 由于中文与英文的文本分类问题具有相当大的差别,体现在原始特征空间的维数更大,文章表示更加稀疏,词性变化更加灵活等多个方面. 在英文文本分类中表现良好的特征抽取方法未必适合中文文本分类,并且这些特征抽取方法都是串行化的,遇到中文海量数据集时效率较低,因此利用并行思想来研究适合于中文的特征选择方法十分必要.

粒子群算法[5]是 20 世纪 90 年代兴起的一种最优化方法,目前已在多个领域得到广泛应用. 文献[6]对 PSO 算法进行修正,提出了基本二进制粒子群算法,该算法可用于求解组合优化问题,拓展了粒子群优化算法的应用范围. 粒子群算法具有结构简单、易于编程、计算量小的优点,但算法中粒子被束缚在局部最优点及整个种群全局最优点附近,从而导致其搜索区域有限,易陷入局部最优. 人们针对这些问题提出许多改进算法,如变异粒子群算法、自适

收稿日期: 2009-02-23; **修回日期**: 2009-05-06.
基金项目: 四川省科技计划项目(2008GZ0003); 四川省科技厅科技攻关项目(07GG006-019).
作者简介: 朱颢东(1980—),男,河南虞城人,博士,从事软件过程技术与方法、文本挖掘的研究; 钟勇(1966—),男,四川南充人,研究员,博士生导师,从事软件过程技术与方法等研究.

应调整粒子群算法[7]、免疫粒子群算法[8]等,这些算法在一定程度上可避免 PSO 陷入局部最优. 受量子力学启发,文献[9,10]提出了量子粒子群(QPSO)算法. 在量子系统中,粒子能够以某一确定的概率出现在可行解空间中的任意位置,因此有更大的搜索范围.

虽然量子有更大的搜索空间,但是量子(粒子)进化过程中,缺乏很好的方向指导. 针对这个缺陷,本文对进化过程中的粒子进行有效疫苗接种,能够指导粒子朝着更好的进化方向发展,提高量子粒子群的收敛速度和寻优能力. 使用粒子群这类智能优化算法进行特征优选具有一个共同的缺点,就是计算适应度的工作量较大. 为了克服这一缺点,增加粒子群算法的实用性,本文结合计算机技术的发展,引入并行技术,以缩短计算耗时,提高效率. 在粗糙集决策表的基础上,提出一个基于并行二进制免疫量子粒子群优化的特征选择方法,该方法利用二进制免疫量子粒子群优化的极强的搜索能力来发现较好的特征子集,在大规模集群上采用并行算法来降低算法运行时间,从而较快地获得较具代表性的特征子集. 实验结果表明,该算法是有效的.

2 并行算法简介

当今并行计算的研究主要集中在并行计算和并行应用两个方向. 近年来,并行算法的研究更讲究实用性,更多地集中在应用领域并行算法研究上[11].

目前主要利用集群系统进行并行计算,其特点是:低成本 + 高性能 + 短周期. 相对于单个的超级计算机而言,集群式系统可以利用普通的 CP 机、工作站来完成工作. 更重要的是集群式系统的性能不比单个的超级计算机的性能差,而且相对于单个的超级计算机而言其研制周期短,只需将普通计算机联网便可完成集群式系统硬件上的架设. 正是由于这些优点,其被越来越多的国家和研究机构作为并行计算的硬件系统[12,13].

目前应用最多的并行编程环境是消息传递模型 MPI,是目前最重要的并行编程工具之一. 该工具具有移植性好、功能强大、效率高等多种优点,且有多种不同的免费高效实用的实现版本,几乎所有的并行计算机厂商都提供对它的支持,这是其他所有的并行编程环境都无法比拟的[14].

3 粗糙集基本理论

本文要用到粗糙集理论的一点基本知识,限于篇幅,只介绍紧密相关的粗糙集知识,具体请参阅文献[15].

定义 1[13] 设信息系统 $S = \langle U, C \cup D, V, f\rangle$, $U/C = \{X_1, X_2, \cdots, X_n\}$, $U/D = \{Y_1, Y_2, \cdots, Y_m\}$,则 D 关于 C 的支持度定义为

$$\gamma_C(D) = \frac{1}{|U|}\sum_{i=1}^{m}|\mathrm{Pos}_C(Y_i)|,\ Y_i \in U/D. \tag{1}$$

定理 1[15] 设信息系统 $S = \langle U, C \cup D, V, f\rangle$, $B \subseteq C$,如果有 $\gamma_B(D) = \gamma_C(D)$,则 B 是 C 的一个约简集.

定义 2[15] 属性 a 加入 $R \subseteq C$,对于分类 U/D 的重要度定义为

$$\mathrm{SGF}(a, R, D) = \gamma_{R+(a)}(D) - \gamma_R(D). \tag{2}$$

$\mathrm{SGF}(a, R, D)$ 的值越大,说明在已知属性集 R 的条件下,属性 a 对决策 D 就越重要.

4 二进制并行免疫量子粒子群优化

人工免疫系统是近年发展起来的模仿自然免疫系统功能的一种新的智能搜索方法. 免疫机理往往用于改进其他智能优化算法,例如免疫 - 遗传算法等. 免疫接种算子根据所求问题具备的或多或少的先验知识,有选择、有目的地提取出一些特征信息或知识,形成“疫苗”并通过“疫苗接种”和“免疫选择”来指导搜索过程,提高优化能力. 将免疫机理、二进制粒子群优化、并行计算相结合,给出了二进制并行免疫量子粒子群优化.

4.1 粒子编码方案

在特征选择问题中,一个特征要么在粒子中要么不在粒子中,因此对粒子采用 0 和 1 的二进制一维编码形式是合适的. 其具体方法如下:假设原始特征集 $C = \{c_1, c_2, \cdots, c_n\}$,其空间可以方便地影射为粒子. 粒子是一个长度为 n 的 0 和 1 字串,每位对应一个特征. 如果在某个位置为 1,则表明选择该特征;否则不选. 这样每个粒子就对应一个特征子集.

4.2 适应度函数设计

根据本文实际情况,定义适应度函数如下:

$$F(x) = \begin{cases} \dfrac{\mathrm{Card}(C - B(x))}{\mathrm{Card}(C)}\gamma_{B(x)}(D), \cdots, \\ \gamma_C(D) - \gamma_{B(x)}(D) \geq \varepsilon 1; \\ \dfrac{2 \times \mathrm{Card}(C - B(x))}{\mathrm{Card}(C)}\gamma_{B(x)}(D), \cdots, \\ \gamma_C(D) - \gamma_{B(x)}(D) < \varepsilon 1. \end{cases} \tag{3}$$

其中,$\varepsilon 1$ 表示精度误差,本文取 $\varepsilon 1 = 0.01$;$B(x)$ 表示个体 x 中对应位为 1 的属性组成的集合;$\gamma_{B(x)}(D)$ 表示类集 D 关于属性集合 $B(x)$ 的支持度. 很明显,属性集 $B(x)$ 元素的数越少,对决策属性 D 的支持度越大,适应度函数值也就越大,而适应度函数中的 $\dfrac{\mathrm{Card}(C - B(x))}{\mathrm{Card}(C)}\gamma_{B(x)}(D)$ 正好反映了这一思想. 如

果 $\gamma_C(D) - \gamma_{B(x)}(D) < \varepsilon 1$,则说明此时属性集 $B(x)$ 接近最优,应适当提高 X 的适应度,由此选择的适应函数可获得知识约简的最佳的搜索效果.

在串行算法中,循环计算每个粒子的适应度是该算法中最耗时的部分,可以利用并行计算来解决. 在并行计算中,把循环进行分解,也就是把要计算适应度的粒子通过消息发送到集群中的多个节点中,并行计算适应度,然后把计算好的适应度发送回来,也就是所谓的主从式方法[16-18]. 方法流程如下:

Step1: 主节点获得参数处理器 CPU 个数 N 和粒子规模 M.

Step2: 计算每个节点要计算的个体数目 P. 由于 CPU 数目不一定能整除种群中的粒子,所以能整除时每个 CPU 要计算的粒子数为 $P = M/N$; N 不能整除时,将剩余的粒子给编号较大的那些 CPU 多发送一个. 每个 CPU 接收到的粒子数目在不能整除时只差一个,这便是一个简单的负载均衡程序.

Step3: 主节点利用 MPI_Isend 函数发送粒子编码到从节点.

Step4: 各个从节点并行计算适应度,并利用 MPI_Isend 函数发送回主节点.

Step5: 主节点利用 MPI_Recv 接收从节点返回的适应度值.

4.3　疫苗抽取设计

Step1: 按照公式 $\mathrm{SGD}(\{c\}, D) = \gamma_{\{c\}}(D)$ 计算各个特征的重要度,并按照从小到大排序,这一步使用 4.2 节所述的主从式方法并行计算.

Step2: 将特征重要度影射为

$$H(c) = \frac{\mathrm{SGD}(\{c\}, D) - \mathrm{SGD}_{\min}}{\mathrm{SGD}_{\max} - \mathrm{SGD}_{\min}}.$$

其中: $c \in C$, $\mathrm{SGD}_{\min}$ 为最小重要度, $\mathrm{SGD}_{\max}$ 为最大重要度.

Step3: 确定特征集的疫苗位模式 $HC = \{H(c_i) \mid i = 1, 2, \cdots, n\}$, n 为总的特征个数,这一步使用 4.2 节所述的主从式方法并行计算.

4.4　初始种群设计

随机初始化 N 个粒子,本文取 $N = 50$. 特征 c_i 的权重 $H(c_i)$ 越大,它对应位 $x_i = 1$ 的概率越大;权重 $H(c_i)$ 越小, $x_i = 0$ 的概率越大. 本文根据下式产生粒子:

$$x_i = \begin{cases} 1, & \mathrm{rand} \geq H(a_i); \\ 0, & \mathrm{rand} < H(a_i). \end{cases} \tag{4}$$

4.5　位置更新策略

在量子空间中,粒子的位置和速度不能同时确定,因此可通过波函数(波函数的平方是粒子在空间中某一点出现的概率密度) 来描述粒子的状态,并通过求解薛定谔方程得到粒子在空间某一点出现的概率密度函数,随后通过蒙特卡罗随机模拟的方式得到量子空间中粒子的位置方程,如下式所示[19]:

$$p = a \times p_b(i) + (1 - a) \times g_b, \tag{5}$$

$$m_b = \frac{1}{N}\sum_{i=1}^{N} p_b(i), \tag{6}$$

$$b = 1 - i_t / i_{t\max} \times 0.5, \tag{7}$$

$$p_{\mathrm{pos}} = p \pm b \times | m_b - p_{\mathrm{pos}} | \times \ln\frac{1}{u}. \tag{8}$$

其中: p 为 p_b 与 g_b 之间的随机位置; m_b 为所有粒子个体最佳位置 p_b 的平均值; N 为粒子个数; b 为收缩扩张系数,在 QPSO 算法收敛的过程中线性减小; i_t 为当前迭代次数; $i_{t\max}$ 为设定的最大迭代次数; p_{pos} 为粒子的当前位置. a 和 u 都是 0～1 之间的随机数,当 $u \geq 0.5$ 时,式(11) 取“ —”号;当 $0 < u < 0.5$ 时,式(8) 取“ + ”号.

对收缩扩张系数 b 的选择和控制是非常重要的,它关系到整个算法的收敛性能[17]. 在文献[17]中已经证明,当 $b < 1.7$ 时,粒子收敛,靠近粒子群的当前最佳位置;当 $b > 1.8$ 时,粒子发散,远离粒子群的当前最佳位置. 从式(7) 可以看出,收缩扩张系数 b 在粒子进化过程中随着进化代数的增加而线性减小,这种固定的变化并不能自适应避免早熟趋势. 对此,本文作出如下改进:

$$b = 2 \times h, \quad h < 0.5; \tag{9}$$

$$b = 1 + h, \quad h \geq 0.5. \tag{10}$$

其中: $h = f_{\mathrm{bfit2}} / f_{\mathrm{fitness}}(i)$, f_{bfit2} 为上一代群体获得最佳位置 g_b 的适应度, $f_{\mathrm{fitness}}(i)$ 为第 i 个粒子的当前适应度; h 为两者的比值, h 越小,说明粒子越远离粒子群的当前最佳位置, h 越大,说明粒子越靠近粒子群的当前最佳位置. 本文以 h 的值是否小于 0.5 为分界,如果 $h < 0.5$,说明粒子远离群体最佳位置 g_b,收缩扩张系数 b 应该小于 1.7,使它收敛,因此将 b 值设为 $2 \times h$,使它不超过 1;若 $h \geq 0.5$,说明粒子靠近群体的当前最佳位置 g_b,因此将 b 值设为 $1 + h$,增加其大于 1.8 的概率,使它尽量发散,扩大搜索范围.

4.6　精英策略

$x^i(t)$ 为 t 时刻的粒子, $x^i(t+1)$ t 为该粒子下一刻移动后的粒子,如果 $\mathrm{Fitness}(x^i(t+1)) \leq \mathrm{Fitness}(x^i(t))$,则 $x^i(t+1) = x^i(t)$, $i = 1, 2, \cdots, m$.

4.7　变异概率

假设粒子 X 的第 k 位为 X_k, X_k 对应的属性为 c_k 的疫苗位为 $H(c_k)$,则定义 X_k 发生变异的概率

$$\beta_k = K_1 (H(c_k) - X_k)^2 - K_2. \tag{11}$$

$H(c_k)$ 与 X_k 的差异越大, β_k 就越大,发生变异的概率也就越大; $H(c_k)$ 与 X_k 的差异越小, β_k 就越

小,发生变异的概率也就越小. 其中 K_1 和 K_2 为参数,本文分别取值 0.2 和 0.05.

4.8 疫苗接种和免疫选择

对每个粒子 X,按位进行疫苗接种产生新粒子 X^*. 具体过程如下:

对 X 中的任一位 X_k,按照公式计算该位的变异率 β_k;然后判断

if rand < β_k, then $X_k^* = 1 - X_k$,

Else $X_k^* = X^*$,

其中 Rand 为[0,1]区间的随机数.

免疫选择

if Fitness(X^*) ≥Fitness(X),

Then $X = X^*$.

这一步可使用 4.2 节所述的主从式方法并行计算来提高时间效率.

4.9 整个算法的简单过程

Step1: 设定各个参数(参数取值均为本文设定):粒子规模 $N = 50$,总的迭代次数 $T = 600$,粒子最大限制速度 maxV = 20,粒子保护周期 $K = 10$,精度误差 $\varepsilon 1 = 0.01$.

Step2: 根据 4.3 节所述产生 N 个初始粒子群 $P(0)$,初始速度向量 $V(0)$ 为 0 向量,局部最优 Lbest$(0,t) = 0$,全局最优 Gbest = 0,初始化已经迭代次数 $t = 1$.

Step3: 按照 4.4 节所述抽取疫苗.

Step4: 按照式(3)并行计算 t 时刻各个粒子 X 的适应度,更新

Lbest(t) = max(Lbest(t), Fitness(X)),

Gbest = max(Lbest(t), Gbest)).

Step5: 更新粒子群:

Step5.1: 对粒子群 $P(t)$ 的各个粒子按照适应度从小到大排序,取前 K 个适应度最高的粒子组成一个粒子群,记为 $KM = \{p_1, p_2, \cdots, p_k\}$;

Step5.2: 按照 4.5 节所述的策略更新粒子群 $P(t)$ 中粒子的位置;

Step5.3: 对 Step5.1 选出的 KM 按照 4.6 节所述的精英策略进行更新;

Step5.4: 按照 4.7 节和 4.8 节所述进行疫苗接种和粒子更新.

Setp6: 若满足 $t \geq T$,则算法结束,输出最优粒子对应的特征子集;否则 $t = t + 1$,转 Step4.

5 实验例证

5.1 实验语料库

对文本分类进行实验,语料库的选择非常重要,需采用国内外使用广泛、权威标准和规范的原则. 这样使得实验和国内外同行的试验结果具有可比性,同时也便于分析实验数据、分析算法的优劣.

在中文文本分类方面,经过分析和比较,本文选用的分类语料库是复旦大学中文文本分类语料库. 该语料库由复旦大学计算机信息与技术系国际数据库中心自然语言处理小组构建,语料文档全部采自互联网,它可以从网上免费下载,网址为 http://www.nlp.org.cn/categories/default.php?cat_id=16.

复旦大学中文文本分类语料库中包含 20 个类别,分为训练文档集和测试文档集两个部分. 每个部分都包括 20 个子目录,相同类别的文档存放在一个对应的子目录下;每个存储文件只包含一篇文档,所有文档均以文件名作为唯一编号. 共有 19637 篇文档,其中训练文档 9804 篇,测试文档 9833 篇;训练文档和测试文档基本按照 1:1 的比例来划分. 去除部分重复文档和损坏文档后,共保留文档 14378 篇,其中训练文档 8214 篇,测试文档 6164 篇,跨类别的重复文档没有考虑,即一篇文档只属于一个类别. 该语料库中文档的类别分布情况不均匀,其中训练文档最多的类 Economy 有 1369 篇;训练文档最少的类 Communication 有 25 篇;训练文档数少于 100 篇的稀有类别共有 11 个. 训练文档集和测试文档集之间互不重叠. 本文只取前 10 个类的部分文档,其类别文档统计数如表 1 所示.

表 1 中文文本分类语料库各类别文档数统计

类别	训练文档数目	测试文档数目
经济	480	419
体育	584	489
计算机	628	591
政治	573	482
农业	547	435
环境	405	371
艺术	510	286
太空	506	248
历史	466	468
军事	74	75

5.2 实验环境及参数设置

实验所用计算机配置如下:操作系统为 Microsoft Windows XP Professional (SP2),CPU 规格为 Intel(R) Celeron(R) CPU 2.40 GHz,内存 512 M,硬盘 80 G.

进行中文分词处理时,采用的是中科院计算所开源项目汉语词法分析 ICTCLAS 系统.

实验使用的软件工具是 Weka,这是纽西兰的 Waikato 大学开发的数据挖掘相关的一系列机器学习算法. 实现语言是 Java,可直接调用,也可在代码

中调用. Weka 包括数据预处理、分类、回归分析、聚类、关联规则、可视化等工具,对机器学习和数据挖掘的研究工作很有帮助. 它是开源项目,网址为 http://www.cs.waikato.ac.nz/ml/weka/. 实验使用的计算工具为 Matlab7.0.

本文算法中各参数需要反复试验才能得到,实验算法中各参数最后设置如下:粒子规模 $N = 40$,总的迭代次数 $T = 600$,粒子最大限制速度 maxV = 20,粒子保护周期 $K = 10$.

5.3　实验所用分类器及其评价标准

本实验旨在比较本文方法与 IG,CHI,MI 三种特征选择方法对后续文本分类精度的影响,因此本实验在各种特征选择方法后采用相同的分类器对文本进行分类. 本实验中使用 KNN 分类器比较这几种特征选择方法(K 设置为 10).

为了评价分类效果,实验中选择分类正确率和召回率作为评价标准:准确(Precision) $= a/(a+b)$,它是所判断的文本与人工分类文本吻合的文本所占的比率. 召回率(Recall) $= a/(a+c)$,它是人工分类结果应有的文本与分类系统吻合的文本所占的比率. 在实际中,查准率比查全率重要. 其中 a, b, c 代表相应的文档数,它们的含义如表 2 所示.

表 2　二值联表

	真正属于此类	真正不属于此类
判断属于此类	a	b
判断不属于此类	c	d

为了评价各个特征选择方法的时间效果,串行算法采用特征选择过程所消耗的时间,而并行算法常采用加速比与效率分析. 加速比 $S_p = T_s/T_p$,其中 T_s 是求解一个问题最快的串行算法在最坏情况下的运行时间,而 T_p 是求解同一个问题的并行算法在最坏情况下的运行时间. 可见加速比是评价算法的并行性对运行时间改进的程度. 效率 $E_p = S_p/p$,其中 p 为处理器的个数,效率反映了并行系统中处理器的利用情况.

5.4　实验结果及其分析

表 3 表明了 4 种方法的准确率和召回率. 可以看出,它们从大到小的顺序依次为本文方法,IG 方法,CHI 方法,MI 方法. 本文方法在选择特征时,不但考查了特征的权重,而且还考查了它们之间潜在的隐含关系,对要选择的特征进行了较全面的考查,所以效果最佳;IG 方法受样本分布影响,在样本分布不均匀的情况下,它的效果大大降低,但从整体上看本文所选样本分布相对均匀,只有极个别相差较大,所以总体效果次之;MI 方法仅考虑了特征发生的概率,而 CHI 方法同时考虑了特征存在与不存在时的情况,所以 CHI 方法比 MI 方法效果要好.

表 4 和表 5 表明,在选择特征子集过程所消耗的时间上,本文方法在一个 CPU 上处理的时间要劣于其他 3 种方法,但采用并行策略后所需时间要远远少于其他 3 种方法.

表 5 给出了本文方法在 CPU 个数变化时的运行时间、加速比和效率. 从该表可看出,随着结点数的增加,速度有明显提高;在子 CPU 个数是种群规模的约数时,效率比较高,达到 88 %左右;当子 CPU 个数在 16 时,效率下降到 84 %左右;当子 CPU 个数在 32 时,效率下降到 79 %左右;整个效率趋势是下降的. 产生这种情况的原因在于:当子 CPU 个数是种群规模约数时,各个子 CPU 分配的种群个数相等,此时系统负载较平衡,各个子 CPU 可以较好地并行工作,主 CPU 不需要单独等待某个子 CPU 便可工作;当子 CPU 个数不是种群规模约数时,此时负载不平衡,子 CPU 分配的工作量不同,因而完成的时间也不同,主 CPU 必须等待各个子 CPU 都

表 3　准确率和召回率　　%

类别	本文方法		IG		CHI		MI	
	准确率	召回率	准确率	召回率	准确率	召回率	准确率	召回率
经济	96.28	94.56	82.52	80.83	79.31	87.67	75.63	76.99
体育	94.17	93.67	83.88	82.93	81.71	85.60	79.54	80.78
计算机	95.07	93.56	87.64	88.43	82.41	83.51	80.71	77.91
政治	94.67	90.33	78.78	84.29	83.29	78.80	79.99	80.72
农业	93.33	92.18	83.27	89.67	79.56	77.23	72.48	79.45
环境	96.48	90.67	81.67	86.42	81.93	86.56	76.42	80.13
艺术	95.27	94.78	80.55	85.81	82.51	82.78	80.51	81.81
太空	94.67	95.88	82.46	87.47	80.84	79.23	78.57	78.47
历史	95.73	91.91	80.33	87.39	78.34	80.42	77.45	81.92
军事	92.81	94.95	75.53	79.73	60.94	87.67	63.67	74.71
平均率	94.95	93.25	81.66	85.30	79.08	82.95	76.50	79.29

表 4 所用串行算法消耗的时间 s

IG	CHI	MI
1427	1538	1496

表 5 不同处理器个数下时间、加速比、并行效率列表

CPU 个数	时间/ s	加速比	效率
1	1738	1.00	1.0000
2	882	1.97	0.9853
4	451	3.85	0.9634
8	237	7.33	0.9167
10	192	9.05	0.9052
16	129	13.47	0.8421
20	98	17.73	0.8867
32	68	25.59	0.7987

结束工作后才能工作;由于随着 CPU 的个数不断增加,每个 CPU 的计算量在不断减小,这样数据传送时间与整个时间的比值就越大,导致效率逐渐降低.从而可以说明,在并行算法中,不要为了追求时间效率而无限增加 CPU 个数,那会造成资源的极大浪费,应该在加速比和效率之间做出一个权衡.

6 结 论

本文提出了一种用于特征选择的并行免疫克隆算法,利用免疫克隆算法极强的搜索能力来发现较好的特征子集,在大规模集群上采用并行算法来降低算法运行时间,从而较快地获得较具代表性的特征子集.实验证明,本文特征选择方法同 IG,CHI,MI 三种特征选择方法相比,有较高的准确率和召回率,而且花费的时间远远低于这 3 种方法所需的时间.本文方法在文本分类中有一定的使用价值,同时为中文文本特征选择提供一种思路,也为后续的知识发现算法减少了时间与空间复杂性.

参考文献(References)

[1] Delgado M, Martin-Bautista M J, Sanchez D, et al. Mining text data: Special features and patterns [C]. Proc of ESF Exploratory Workshop. London, 2002: 32-38.

[2] 朱颢东, 钟勇. 一种新的基于多启发式的特征选择算法[J]. 计算机应用, 2009, 29(3): 849-851.
(Zhu H D, Zhong Y. New feature selection algorithm based on multiple heuristics [J]. J of Computer Applications, 2009, 29(3): 849-851.)

[3] Yang Y, Pedersen J O. A comparative study on feature selection in text categorization [C]. Proc of the 14th Int'l Conf on Machine Learning (ICML '97). Nashville: Morgan Kaufmann Publishers, 1997: 412-420.

[4] 张海龙, 王莲芝. 自动文本分类特征选择方法研究[J]. 计算机工程与设计, 2006, 27(20): 3838-3841.
(Zhang H L, Wang L Z. Automatic text categorization feature selection methods research [J]. Computer Engineering and Design, 2006, 27(20): 3838-3841.)

[5] Kennedy J, Eberhart R C. Particle swarm optimization [C]. Proc IEEE Int Conf on Neural Networks. New York, 1995, 4: 1942-1948.

[6] Kennedy J, Eberhart R C. A discrete binary version of the particle swarm algorithm[C]. Proc of the 1997 Conf on Systems, Man and Cybernetics. Piscataway: IEEE Press, 1997: 4104-4109.

[7] Yang X M, Yuan J S, Yuan J Y, et al. A modified particle swarm optimizer with dynamic adaptation [J]. Applied Mathematics and Computation, 2007, 189(9): 1205-1213.

[8] 廖建坤, 叶东毅. 基于免疫粒子群优化的最小属性约简算法[J]. 计算机应用, 2007, 27(3): 550-555.
(Liao J K, Ye D Y. Minimal attribute reduction algorithm based on particle swarm optimization with immunity[J]. J of Computer Applications, 2007, 27(3): 550-555.)

[9] Sun J, Feng B, Xu W. Particle swarm optimization with particles having quantum behavior[C]. Proc of Congress on Evolutionary Computation. Portland, 2004: 238-246.

[10] Liu J, Xu W, Sun J. Quantum-behaved particle swarm optimization with mutation operator [C]. Proc of the 17th TEEE Int Conf on Tools with Artificial Intelligence (ICTAI'05). Hong Kong: IEEE Press, 2005: 871-878.

[11] 谷建军. 粗糙集理论在数据约简中的应用研究[D]. 济南: 山东师范大学, 2007.
(Gu J J. Application of rough set theory in data reduction[D]. Ji'nan: Shandong Normal University, 2007.)

[12] 旷海兰. 基于粗糙集理论的数据挖掘算法研究[D]. 长沙: 长沙理工大学, 2006.
(Kuang H L. The research on data mining algorithms based on rough set theory[D]. Changsha: Changsha Polytechnic University, 2006.)

[13] 陈鑫影. 基于粗糙集理论约简算法研究[D]. 长春: 吉林大学, 2005.
(Chen X Y. Research of reduction algorithm based on rough set theory [D]. Changchun: Jilin University, 2005.)

[14] 曾维宏. 基于粗糙集理论的数据挖掘算法研究[D]. 郑州: 郑州大学, 2005.
(Zeng W H. Research of reduction algorithm based on rough set theory [D]. Zhengzhou: Zhengzhou University, 2005.)

(下转第 63 页)

$-\gamma^{-1}\Gamma(t)$, 因此对于任意的 $t > t_0$, 有

$$V(t) - V(t_0) \leq -\gamma^{-1}\int_{t_0}^{t}\Gamma(\tau)\,\mathrm{d}\tau. \qquad (26)$$

再次利用 $V(t) \geq 0$, 有

$$\int_{t_0}^{t} z^{\mathrm{T}}(\tau)\,z(\tau)\,\mathrm{d}\tau \leq \gamma V(t_0) + \gamma^2\int_{t_0}^{t} w^{\mathrm{T}}(\tau)\,w(\tau)\,\mathrm{d}\tau. \qquad (27)$$

这表明得到了 H_∞ 扰动抑制水平. □

注 1　在现有的参考文献中, μ_{i1} 和 μ_{i2} 的值是以时间的方式调节的, 即 μ_{i1} 和 μ_{i2} 值的改变依赖于时间. 这样做主要是为了保证系统状态进入到某一特定区域. 由于系统中存在未知扰动输入 $w(t)$, 使得依赖于时间的调节方法不能被采用. 为了克服这个困难, 提出了一种依赖状态调节 μ_{i1} 和 μ_{i2} 值的方法. 与其他参考文献指出的一样, 对于系统建模不完全的情形, 这种依赖状态的方法比依赖时间的方法通常具有更好的鲁棒性.

注 2　尽管 H_∞ 扰动抑制水平 γ 在本文中是固定值, 但对于任意正数 $\gamma > \gamma_{\mathrm{opt}}$, 分散状态反馈量化控制器都是成立的, 其中 γ_{opt} 是系统 (1) 和 (2) 的最优 H_∞ 范数指标.

注 3　定理 1 中的条件 (15) 是灵活的, 在某种意义上可以选择矩阵 P_i, R_i, M_i (或 K_i) 使得这些条件都满足. 这些矩阵不是互不相关的, 它们要满足矩阵不等式 (13). 而在求解式 (13) 时, 可通过加入优化条件来获得更大的设计自由度.

4　结　　论

本文研究了同时具有状态和控制输入两量化器的关联状态反馈网络系统的稳定性和 H_∞ 扰动抑制水平问题. 对于调节量化器参数, 提出一种局部状态依赖的控制方法, 使得闭环系统全局渐近稳定, 并得到了与未加量化器一样的 H_∞ 扰动抑制水平.

参考文献(References)

[1] Bushnell L G. Special section on networks and control [J]. IEEE Control Systems Magazine, 2001, 21(1): 22-99.

[2] Ishii H, Francis B. Limited data rate in control systems with networks[M]. Berlin: Springer, 2002.

[3] Delchamps D F. Stabilizing a linear system with quantized state feedback[J]. IEEE Trans on Automatic Control, 1990, 35(8): 916-924.

[4] Brockett R W, Liberzon D. Quantized feedback stabilization of linear systems [J]. IEEE Trans on Automatic Control, 2000, 45(7): 1279-1289.

[5] Liberzon D. Hybrid feedback stabilization of systems with quantized signals[J]. Automatica, 2003, 39(9): 1543-1554.

[6] Matsumoto Y, Zhai G, Mi Y. Stabilization of discrete-time LTI systems by hybrid quantized output feedback [C]. Preprints of the 46th Japan Joint Automatic Control Conf. Okayama, 2003: 799-802.

[7] Zhai G, Matsumoto Y, Chen X, et al. Hybrid stabilization of linear time-invariant systems with two quantizers[C]. Proc of the 2004 IEEE Int Symposium on Intelligent Control. Taipei, 2004: 305-309.

[8] Zhai G, Mi Y, Imae J, et al. Design of H_∞ feedback control systems with quantized signals[C]. Preprints of the 16th IFAC World Congress. Prague, 2005: Fr-M17-TO/1.

[9] Zhai G, Chen N, Gui W. Quantizer design for interconnected feedback control systems[C]. Preprints of the 17th IFAC World Congress. Seoul, 2008: 8707-8712.

[10] Petersen I R. A stabilization algorithm for a class of uncertain linear systems [J]. Systems & Control Letters, 1987, 8(1): 351-357.

(上接第 58 页)

[15] 曾黄麟. 智能计算[M]. 重庆: 重庆大学出版社, 2004.
(Zeng H L. Intelligent computation[M]. Chongqing: Chongqing University Press, 2004.)

[16] 吴昊, 程锦松. 用并行遗传算法解列车控制问题[J]. 微机发展, 2002, 12(1): 50-52.
(Wu H, Chen J S. Parallel genetic algorithm for solving the train control problem[J]. Microcomputer Development, 2002, 12(1): 50-52.)

[17] 陈睿, 谷艳昌. 基于并行自适应变异粒子群算法的渗透系数反分析[J]. 水力发电, 2008, 34(2): 17-19.
(Chen R, Gu Y C. Back analyzing seepage coefficients with parallel AMPSO[J]. J of Hydroelectric Power, 2008, 34(2): 17-19.)

[18] 于洽, 陈波. 入侵数据特征并行选择算法[J]. 电子科技大学学报, 2008, 37(2): 266-269.
(Yu L, Chen B. Parallel algorithm of feature reduction in intrusion data [J]. J of University of Electronic Science and Technology of China, 2008, 37(2): 266-269.)

[19] Sun J, Feng B, Xu W B. Particle swarm optimization with particles having quantum behavior[C]. Proc of 2004 Congress on Evolutionary Computation. Piscataway: IEEE Press, 2004: 325-330.