第 42 卷 第 6A 期
2015 年 6 月

计 算 机 科 学
Computer Science

Vol. 42 No. 6A
June 2015

求解分类问题的文法多蜂算法

刘坤起[1] 周 冲[1] 吴志健[2]

(石家庄经济学院计算机科学系 石家庄 050031)[1] (武汉大学软件工程国家重点实验室 武汉 430072)[2]

摘 要 多蜂算法(Bees Algorithm,BA)和文法演化算法(Grammatical Evolution,GE)是两个著名的演化算法。BA尽管收敛速度较快,但用于求解分类问题时,个体编码不易实现。而基于GE的分类算法的演化算子较简单,仅进行杂交和变异两个操作,但分类精度不高。针对两个算法的优点和不足,将BA和GE相结合,提出了一种新的混合演化算法——文法多蜂算法(Grammatical Bees Algorithm,GBA),并将其用于求解分类问题。在几个标准数据集上的实验验证了GBA的可行性和有效性。与基本基因表达式编程(Gene Expression Programming,GEP)分类算法和改进的GEP分类算法相比,GBA能获得较好的分类精度和更快的收敛速度。

关键词 混合演化算法,演化建模,多蜂算法,文法演化算法,分类问题

中图法分类号 TP311 **文献标识码** A

Grammatical Bees Algorithm for Classification Problem

LIU Kun-qi[1] ZHOU Chong[1] WU Zhi-jian[2]

(Department of Computer Science, Shijiazhuang University of Economics, Shijiazhuang 050031, China)[1]

(State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China)

Abstract Bees Algorithm(BA) and Grammatical Evolution(GE) are two well-known evolutionary algorithms. BA for classification problems has shown faster convergence speed, but the individual coding is complicated. The operators of GE for classification problems are simple, which include crossover and mutation operators, but their classification accuracy is not high. In view of the strengths and weaknesses of the two algorithms, a new algorithm, named Grammatical Bees Algorithm(GBA) combining BA and GE, was proposed to solve the classification problems. Experiments on several benchmark data sets demonstrate the feasibility and effectiveness of GBA. Compared with gene expression programming(GEP) and improved GEP, GBA can achieve better classification accuracy and faster convergence speed.

Keywords Hybrid evolutionary algorithm, Evolutionary modeling, Bees algorithm, Grammatical evolution, Classification

1 引言

分类问题是利用通过训练数据集得到的分类器对新的数据样本进行样本归类的问题。它是数据挖掘领域中的一个基本问题[1]。求解该问题的方法有两类,一类是基于统计学的方法,该方法的特点是分类速度较快,且有较强的理论支持,目前主要有贝叶斯方法[2]、K近邻法[3]、支持向量机[4]等;另一类则是基于规则的方法,其特点是具有易于理解的表现形式,且可以根据需要不断挖掘出更好的规则,该分类方法目前主要有决策树[5]、粗糙集[6]和基于演化的分类[7]等。在基于演化的分类方法中,常见的有遗传算法[8]、遗传程序设计[9]、基因表达式程序设计[10-21]、多基因表达式程序设计[12]等。然而,利用这些演化算法求解分类问题时还存在收敛速度慢或分类精度不高等问题,因此需要发展新的基于演化的分类方法。

多蜂算法[13]和文法演化算法[15-17]均为目前演化计算研究领域中的著名算法,也是目前国内外演化算法研究的热点算法。BA尽管收敛速度较快,但用于求解分类问题时,个体编码不易实现。而基于GE的分类算法的演化算子较简单,但分类精度不高。为了提高求解分类问题的演化算法的收敛速度和求解精度,针对这两个算法的优点和不足,本文将BA和GE结合起来,提出了一种新的混合演化算法——文法多蜂算法,并将其用于求解分类问题。使用几个标准数据集进行仿真实验,结果表明:与基本的GEP算法和改进的GEP算法相比,GBA能获得更好的分类精度,且收敛速度有很大提高,从而说明了GBA的可行性和有效性。

本文第2节为分类问题的描述,第3节讨论多蜂算法,第4节讨论文法演化算法;第5节讨论求解分类问题的文法多蜂算法;第6节为仿真实验及其结果分析;最后为结论。

2 分类问题的描述

分类问题定义为:给定一个数据集,所给定的数据集称为训练数据集。训练数据集由数据库元组(常称作训练样本、实例或对象)和与它们相关联的类标号构成,元组 X 用 n 维属

本文受国家自然科学基金项目(61402481),教育部计算机科学与技术专业综合改革试点(石家庄经济学院)项目,石家庄经济学院博士科研启动基金项目(2011)资助。

刘坤起(1966—),男,博士,教授,主要研究领域为演化计算、程序设计语言、计算机科学教育,E-mail:liu_kq@126.com;周 冲(1989—),男,硕士生,主要研究领域为演化计算;吴志健(1963—),男,博士,教授,博士生导师,主要研究领域为演化计算、图像处理等。

性向量 $X=(x_1,x_2,\cdots,x_n)$ 表示，分别描述元组在 n 个数据库属性 $A_1,A_2,\cdots,A_n$ 上的 n 个度量。每个元组有一个由数据库属性确定的预先定义好的类，称为类标号[1]。用给定的训练数据集建立一个分类函数（常常也称作分类模型或分类器），以便能够使用模型预测类标号未知的对象的类标号。

数学定义描述如下：设训练集为 $\{(t_i,class_j)\}$，寻找训练集分类映射 F，使得

$F(t_i)=class_j$，且 $F(t_{\text{未分类数据}})=class_j$

3 多蜂算法

多蜂算法（BA）是一种模拟蜂群采蜜行为的演化算法。它是由英国卡迪夫大学的 D. T. Pham 和 A. Ghanbarzadeh 于 2005 年提出的[13]。BA 描述如下：

t←0；//初始化演化代数
初始化种群的 $\{X_i(t)|1\leqslant i\leqslant N\}$；//N 为种群的规模
评估种群，并保存种群中的最好个体，记为 $X_{best}(t)$；
do
　选择出 m 个最好的个体用于邻域搜索；
　让 e 个最好的个体进行 nep 次邻域搜索；
　让 m−e 个次好的个体进行 nsp 次邻域搜索；
　让 n−m 个个体进行 1 次全局搜索；
　重组生成新的种群；
　评估种群并保存种群最好个体，记为 $X_{best}(t)$；
　t←t+1；
until 满足终止条件；
输出 $X_{best}(t)$ 和 $fit(X_{best}(t))$。

在 BA 中，采用邻域搜索策略和全局搜索策略。选择表现好的个体进行邻域搜索，其他的个体进行全局搜索。邻域搜索中个体的邻域搜索半径是按照式(1)：

$$a(t+1)\leftarrow 0.8*a(t) \tag{1}$$

动态改变的，并且每个个体设置一个最大搜索次数 $limit$，如果个体在 $limit$ 内没有找到更好的解，那么该个体就被淘汰。BA 的邻域搜索策略按式(2)产生新个体 i。

$$V_{ij}(t)\leftarrow X_{ij}(t)+\varphi_{ij}(X_{ij}(t)-X_{kj}(t)) \tag{2}$$

其中，$i\in(1,2,\cdots,N)$，$j\in(1,2,\cdots,D)$，D 是个体染色体长度，t 是当前的迭代次数，k 为随机在邻域内选择的一个个体号，$k\in(1,2,\cdots,N)$ 且 $k\neq i$，φ_{ij} 是一个在 $[-1,1]$ 之间均匀分布的随机小数。

BA 的全局搜索策略在解空间内随机产生一个新解。

BA 的淘汰策略是个体在 $limit$ 内没有找到更好的解，就随机产生一个新解替换。

4 文法演化算法

文法演化算法（GE）是由英国利默里克大学的 Conor Ryan 和 J. J. Collins 于 1998 年提出的[15]。它是在传统的遗传算法的基础上，利用产生函数表达式的文法（使用 BNF 描述）的最左推导过程，来完成个体（二进制数串）的基因型到其表现型（函数表达式）的转换。因此，与使用语法树代表程序的遗传程序设计相比，GE 采用一个线性二进制串表示其个体（染色体），使得演化操作简单，易于实现，且很容易设计出多种演化算子，它们相互结合使用，扩大了算法的搜索范围，加快了其收敛速度。

下面介绍 GE 利用产生函数表达式的文法来实现个体的基因型转换到其表现型的思想方法。

例如，某函数表达式的文法为：$G=\{N,T,S,P\}$。其中，终端集合 $T=\{+,-,*,/,\sin,\exp,(,)\}$，非终端集合 $N=\{\text{expr},\text{op},\text{pre-op}\}$，开始符号 $S=\langle\text{expr}\rangle$，产生式集合如下：

$\langle\text{expr}\rangle::=\langle\text{expr}\rangle\langle\text{op}\rangle\langle\text{expr}\rangle|$ (0)
$(\langle\text{expr}\rangle\langle\text{op}\rangle\langle\text{expr}\rangle)|$ (1)
$\langle\text{pre-op}\rangle(\langle\text{expr}\rangle)|$ (2)
x (3)
$\langle\text{op}\rangle::=+|$ (0)
$-|$ (1)
$*|$ (2)
$/$ (3)
$\langle\text{pre-op}\rangle::=\sin|$ (0)
$\exp$ (1)

说明：为了便于在算法中操作和实现，在上述每一条产生式规则后均加上了一个编号，用来代替该条产生式规则。

假设某一个二进制编码的个体的染色体为 $C_{GE}=$ 00000000 00000010 00000001 00000011 00000010 00000011，那么其对应的函数表达式（表现型）是什么呢？

在 GE 中，个体的基因型（染色体）向表现型（函数表达式）转化是采用文法的最左推导方法。具体地，先将文法的开始字符映射成非终结符，并取出以该非终结符为左部的产生式数 Num-Rules；然后从个体基因型中从左到右依次读取 8 个基因位，并把这 8 位二进制数转化成一个十进制整数（密码子）Codon-Value；再使用 Codon-Value MOD Num-Rules 来确定待选择的产生式编号 Rule，这样就完成了一步推导。依次读取剩余的密码子，进行最左推导，产生其对应的产生式编号 Rule，直到表现型（函数表达式）生成为止。值得注意的是，在该过程中，若所有密码子均已使用完，但还没有推导结束，此时个体的基因型应设计成一个循环形式，这样就可以重复使用密码子了。这种设计造成有效的染色体长度比其实际长度更长。还有一种情况，若该染色体使用了两遍，其函数表达式还没有最终生成（即还有非终结符），就强制性地把其最右的非终结符删除，即可得到一个最终的有效函数表达式。

若采用上述方法，个体 C_{GE} 的密码子为：0,2,1,3,2,3，其转化成的函数表达式为：$E=\exp(x)*x$。

GE 描述如下：

t←0；//初始化演化代数
初始化种群 $\{X_i(t)|1\leqslant i\leqslant N\}$；//N 为种群的规模
将基因型转化为表现型（函数表达式）；
计算函数表达式的适应度，并保存种群最好个体，记为 $X_{best}(t)$；
do
　在基因型上进行杂交操作；
　在基因型上进行变异操作；
　将基因型转化为表现型（函数表达式）；
　计算函数表达式的适应度，并保存种群最好个体，记为 $X_{best}(t)$；
　采用锦标赛策略更新种群；
until 满足终止条件；
输出 $X_{best}(t)$ 和 $fit(X_{best}(t))$。

5 求解分类问题的文法多蜂算法

文法多蜂算法（GBA）与 BA 的基础算法框架相同，不同之处在于 GBA 的个体编码方式、邻域搜索策略、全局搜索策

略和淘汰策略。

5.1 个体编码

GBA的个体编码方式采用了改进的GE个体编码方式，即将二进制串改为十进制整数串（每位基因的取值范围为0—255）。这种编码方式避免了GE中“把8位二进制数转化成一个十进制整数（密码子）”的操作，从而提高了算法的速度，而且缩短了个体染色体的长度。

5.2 邻域搜索策略和全局搜索策略

(1)邻域搜索

BA的邻域搜索按照式(2)进行，而GE的搜索策略与基本遗传算法的搜索策略相同。实验表明，这两种策略单独使用时求解的结果并不好，且易陷入局部最优。GBA采用了两种邻域搜索策略，而且两种策略的选择是以概率p_n进行选择的。第一种策略是与邻域中的个体进行杂交操作生成一个新个体，具体地，将邻域中的个体k中p_1到p_2基因段置换个体i中的p_1到p_2基因段，从而得到新的个体i。另一种策略是按照下列操作①、②对个体i进行更新。

①$V_{ij}(t)\leftarrow X_{ij}(t)+\sin(t)*(X_{ij}(t)+X_{kj}(t))$；

②若$V_{ij}(t)<0$或$V_{ij}(t)>255$，则$V_{ij}(t)=\mathrm{rand}(0,255)$。

其中，$i\in(1,2,\cdots,N)$，$j\in(1,2,\cdots,D)$，D是个体染色体长度，t是当前的迭代次数，k为随机在邻域内选择的一个个体号，$k\in(1,2,\cdots,N)$且$k\neq i$。

若新个体$V_i(t)$的适应度比老个体$X_i(t)$好，则执行$X_i(t)\leftarrow V_i(t)$操作，T_i置0，否则计数器T_i加1。其中，T_i为没有提高个体i适应度的邻域搜索次数，其初值为0。GBA正是采用了这两种邻域搜索策略，使得个体变化多样，最优表达式段易于保留。

(2)全局搜索

GBA的全局搜索策略是随机产生一个新个体$V_i(t)$，计算$V_i(t)$的适应度，若$V_i(t)$比老个体$X_i(t)$好，则执行$X_i(t)\leftarrow V_i(t)$操作，否则$X_i(t)$不变。

(3)淘汰策略

GBA的淘汰策略为：若T_i大于设定的$limit$，则淘汰该个体i，并执行下面的操作①、②，以生成新个体i，并取代被淘汰的个体。

①若$r\leqslant p_m$或$j=q$，则$X_{ij}(t)\leftarrow X_{bestj}(t)+F*(X_{aj}(t)+X_{bj}(t))$；

②若$X_{ij}(t)<0$或$X_{ij}(t)>255$，则$X_{ij}(t)=\mathrm{rand}(0,255)$。

其中，$r\in[0,1]$是一个随机小数，变异概率$p_m\in[0,1]$，q是染色体中的一点，$F\in[0,2]$，a、b是随机产生的两个个体号且$a\neq b$，$X_{best}(t)$是目前最好的个体。

5.3 适应度计算

由于计算适应度函数有很多，不同的函数对算法的搜索效果不同，GBA的计算适应度函数采用绝对误差适应度函数，即式(3)。

$$f=\min\{\sum_{j=1}^{N}|E_j-O_j|\} \tag{3}$$

其中，E_j是样本j的目标值，O_j为函数表达式利用样本j得到的值，N为样本的个数。

5.4 分类类别的确定和分类精度的定义

在GBA的类别确定中，由于GBA运行结果是函数表达式，计算结果是实数，而类别是整数，因此需要将实数转化为整数。本文按照如下操作①确定分类类别。

①$|e_i-class_i|<\alpha$，则$c_i=class_i$。

其中，e_i为各属性代入分类函数表达式后计算所得的值，$class_i$为第i类，分类阈值α一般设为0.5。

分类精度的定义目前有多种。在GBA的中按照式(4)计算分类精度。

$$f_{cla}=\frac{tr_t+te_t}{tr_a+te_a},\ f_{cla}\in(0,1) \tag{4}$$

其中，tr_a为训练集个数，te_a为测试集的个数，tr_t为训练集中训练正确的个数，te_t为测试集中测试正确的数目，f_{cla}为分类精度。

6 仿真实验及其结果分析

6.1 实验问题描述

将GBA应用到来自UCI数据库[29]的几个标准数据集上，对每个数据集分别做10次5-折交叉验证。数据集信息如表1所列。

表1 数据集信息

数据集 序号	数据集 名称	样本数	属性数	类别数
1	Iris	150	4	3
2	Ionosphere	351	34	2
3	Glass	214	9	6
4	Pima Indian	768	8	2
5	Wine	178	13	3

在本文中，GBA通常以函数表达式作为分类规则。这种分类规则更擅长数值变量的处理，所以在分类前需要对字段属性做统一化处理，将名词属性数值化。具体方法是将该名词属性集直接转化为一组对应的整数集。

6.2 实验环境

GBA在实验平台Code::Blocks 13.12上用C语言实现，操作系统为Windows 7，CPU为Inter(R) Core(TM) i3-2130，内存为2G。

6.3 实验参数设置和文法

根据有利于提高演化速度和分类精度的参数设置原则，对GBA的参数设置如表2所列。GBA的邻域范围设定为：前e个体的邻域为$m-e$，$m-e$个体的邻域为$N-m$。

表2 核心参数设置

参数名	参数值	参数名	参数值
种群大小 N	200	迭代次数	1000
选择邻域搜索策 p_n	0.5	变异概率 p_m	0.8
分类阈值 α	0.5	F	0.8
最大搜索次数 limit	5	选择最好个体 m	10
NSP	7	最好个体 e	3
NEP	4		

GBA使用的文法定义为：$G=\{N,T,S,P\}$

终端集合：$T=\{+,-,*,/,\sin,\exp,\cos,\ln,\mathrm{sqrt},x_1,\cdots,x_n,\mathrm{a}\}$

非终端集合：$N=\{\mathrm{expr},\mathrm{op},\mathrm{pre\text{-}op},\mathrm{var}\}$

开始符号：S=〈expr〉

产生式集合为：

〈expr〉::=〈expr〉〈op〉〈expr〉|〈pre-op〉(〈expr〉)|〈var〉

〈op〉::=+|-|*|/

〈pre-op〉::=sin|exp|cos|In|sqrt

〈var〉::=x_1|…|x_n|a,a∈[0,10]

6.4 实验结果及其分析

(1)分类精度比较

本文把 GBA、GEP[11]、ClonalQuantum-GEP[21] 分别用于求解分类问题,其分类精度的比较如表 3 所列。从表 3 中可以得出,在平均精度方面,GBA 在 Iris、Ionosphere、Pima Indian 数据集上求解的平均精度均高于 GEP 和 ClonalQuantum-GEP 算法,而在 Wine 和 Glass 数据集上求解的平均精度低于 GEP 算法,但高于 ClonalQuantum-GEP 算法。在最高精度方面,GBA 在 Iris、Ionosphere、Pima Indian 和 Wine 数据集上的求解的最高精度均高 GEP 和 ClonalQuantum-GEP 算法,而在 Glass 数据集上求解的最高精度低于 GEP、ClonalQuantum-GEP 算法。

表 3 分类精度比较(%)

数据集		GEP		ClonalQuantum-GEP		GBA	
序号	名称	平均精度	最高精度	平均精度	最高精度	平均精度	最高精度
1	Iris	95.3%	96.0%	96.7%	100.0%	96.9%	100.0%
2	Ionosphere	90.2%	92.3%	87.4%	92.6%	90.8%	94.3%
3	Glass	63.9%	70.1%	56.4%	70.7%	61.1%	64.0%
4	Pima Indian	73.3%	75.1%	70.5%	72.7%	75.4%	78.0%
5	Wine	92.0%	93.8%	90.6%	95.6%	91.4%	95.8%

(2)函数表达式长度和运行时间比较

表 4 列出了 GBA、GEP[11]、ClonalQuantum-GEP[21] 在求解分类问题时的函数表达式长度和运行时间的比较。从表 4 中可以得出,在测试的 5 个数据集上,GBA 求得的函数表达式平均长度比 GEP 和 ClonalQuantum-GEP 均小,与 ClonalQuantum-GEP 相比,在 5 个数据集上分别减少了 60.0%、18.75%、33.92%、70.83%、35.41%。而在平均运行时间方面,GBA 的运行时间比 GEP、ClonalQuantum-GEP 的均少,在 5 个数据集上分别减少了 17.14%、26.21%、22.72%、62.64%、26.41%。

表 4 函数表达式长度和运行时间的比较

数据集		GEP	ClonalQuantum-GEP		GBA	
序号	名称	平均表达式长度	平均表达式长度	平均运行时间(min)	平均表达式长度	平均运行时间(min)
1	Iris	150	50	0.7	20	0.58
2	Ionosphere	40	32	3.7	26	2.73
3	Glass	180	56	2.2	37	1.7
4	Pima Indian	69	48	8.1	14	3.04
5	Wine	75	48	1.7	31	1.25

(3)分类精度最高的函数表达式

下面①—⑤给出了用 GBA 求 5 个数据集分类精度最高的函数表达式。

①$q(x_3)+x_4/q(i(x_2/a))/x_1 * q(a+x_4)/q(i(x_2/a))$, $a=1.9$;

②$x_5 * x_1 * e(q(q(e(x_{30}))+x_{10}/q(x_{13}))+x_8-e(x_{22})+x_{23}/x_5/s(c(x_{24})+x_6)+x_{10})$;

③$e(c(i(q(x_6)))-x_3)+x_4$;

④$s(q(x_6)/c(x_6/x_2+a))$, $a=1.94$;

⑤$q(i(x_4))+x_9/x_1+c(x_7)+e(a)/x_{13}$, $a=3.8$。

其中,函数表达式中的 e 表示自然指数,q 表示 $\sqrt{\ }$,s 表示 sin() 函数,c 表示 cos() 函数,i 表示 In() 函数。

从上面的实验结果对比分析可知,GBA 在一些数据集上计算结果的平均精度、最高精度及运行效率有一定程度的提高。其原因是 GBA 结合了 BA 和 GE 的优点,改进了邻域搜索策略、全局搜索策略和淘汰策略。GBA 采用了两种邻域搜索策略,一种是采用个体杂交操作生成一个新个体,而新个体中被交换的基因段可能是一小段函数表达式,这样可使得好的基因段保留下来,利于产生更好的函数表达式。另一种是通过更新基因的值,使得染色体变化形式多样,进而产生更多形式的函数表达式。同时,邻域搜索策略还使得 GBA 具有更快的收敛速度。而全局搜索使得种群保持较高的多样性,防止陷入局部最优。GBA 的淘汰策略,有利于进一步提高其收敛速度。

结束语 本文在 BA 算法框架的基础上,采用了 GE 中改进的个体编码方式和改进的 BA 中的局部搜索策略和全局搜索策略,以及新的淘汰策略,提出了一个新的混合演化算法——GBA,并将其用于求解分类问题。使用几个标准数据集进行仿真实验,结果表明:与基本的 GEP 算法和改进的 GEP 算法相比,GBA 能获得更好的分类精度,且收敛速度提高很大,从而说明了 GBA 的可行性和有效性。另外,若修改 GBA 中的文法,该算法还可应用到基于规则的分类问题的求解中。但是,GBA 在不平衡和类别较多的数据集上,其分类精度较低,其原因可能是文法设计得不够完善或搜索策略有待进一步改进。由于 GBA 对文法的依赖很强,因此如何改进文法,甚至如何进一步改进 GBA,以期解决其不足之处,是今后努力的方向。

参 考 文 献

[1] 范明.数据挖掘概念和技术[M].孟小峰,译.北京:机械工业出版社,2001

[2] Domingos, Pedro, Pazzanion M. The optimality of the simple bayesian classifier under Zero-One loss[J]. Machine Learning, 1997,29:103-137

[3] Shakhnarovish, Darrell, Indyk. K nearest neighbor methods in learning and vision [M]. MA:The MIT Press,2005

[4] Meyer D,Leisch F,Hornik K. The support vector machine under test[J]. Neuro computing,2003,55(1/2):169-186

[5] Yuan Y,Shaw M J. Induction of fuzzy decision trees[J]. Fuzzy

Sets and Systems, 1995, 69: 125-139

[6] Bazan J, Nguyen H S, Nguyen S H, et al. Rough set algorithms in classification problem[M] // Rough Set Methods and Applications. Physica-Verlag, 2000: 49-88

[7] Alvarez J L, Mata J, Riquelme J C. OBLIC: classification system using evolutionary algorithm[C] // 6th International Work-Conference on Artificial and Natural Neural Networks. 2001

[8] Bandyopadhyay S, Pal S K. Classification and learning using genetic algorithms[M]. Springer Verlag, 2007

[9] Winkler S M, Affenzeller M, Wagner S. Advances in applying genetic programming to machine learning focussing on classification problems [C] // Parallel and Distributed Processing Symposium. 2006

[10] Zhou Chi. Gene expression programming and rule induction for domain knowledge discovery and management [D]. Chicago: University of Illinois at Chicago, 2003

[11] Zhou C, Xiao W, Tirpak T M, et al. Evolving Accurate and Compact Classification Rules with Gene Expression Programming [J]. IEEE Transactions on Evokutionary Computation, 2003, 7 (6): 519-513

[12] 张建伟, 吴志健, 黄樟灿. 基于多表达式编程的分类算法研究[J]. 小型微型计算机系统, 2010, 131(7)

[13] Pham D T, Ghanbarzadeh A, Koc E, et al. The Bees Algorithm [Z]. Technical Note, Manufacturing Engineering Centre, Cardiff University, UK, 2005

[14] Pham D T, Ghanbarzadeh A, Koc E, et al. The Bees Algorithm, A Novel Tool for Complex Optimisation Problems[C] // Proc 2nd Virtual International Conference on Intelligent Production Machines and Systems. Elsevier(Oxford), 2006: 454-459

[15] Ryan C, Collins J J, O'Neill M. Grammatical evolution: evolving programs for an arbitrary language[C] // First EuropeanWorkshop on Genetic Programming, 1998. Paris, France, April 1998: 83-96

[16] O'Neill M, Ryan C. Grammatical Evolution[J]. IEEE Trans. on Evolutionary Computation, 2001, 5(4): 349-358

[17] O'Neill M, Ryan C. Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language[M]. Kluwer Academic Publishers

[18] 刘坤起, 康立山, 赵致琢. 关于认知演化计算分支领域的研究简报(Ⅰ)[J]. 计算机科学, 2009, 36(7): 26-31

[19] 刘坤起, 康立山, 赵致琢. 关于认知演化计算分支领域的研究简报(Ⅱ)[J]. 计算机科学, 2009, 36(8): 35-39

[20] Witten I H, Frank E. Data mining: practical machine learning tools and techniques[M]. San Francisco, CA: Morgan Kaufmann, 2005

[21] 王卫红, 杜燕烨, 李曲. 基于克隆选择和量子进化的 GEP 分类算法[J]. 计算机科学, 2011, 38(10): 236-239

[22] 柳益君, 朱明放, 习海旭, 等. 基于最大隶属度原则的基因表达式编程分类[J]. 计算机工程与应用, 2012, 48(26): 48-52

[23] Sugiura H, Mizuno T, Kita E. Santa Fe Trail Problem Solution-Using Grammatical Evolution[J]. 2012 International Conference onIndustrial and Intelligent Information, Singapore, 2012, 12: 36-40

[24] Ahmad S A. A Study of Search Neighbourhood in the Bees Algorithm[D]. Cardiff University, 2012

[25] Alfonsecaa M, Gil F J S. Evolving an ecology of mathematical expressions with grammatical evolution[J]. BioSystems, 2013, 111(2): 111-119

[26] 王璞. 基于遗传规划的分类算法研究[D]. 安徽: 中国科技大学, 2013

[27] 陈剑, 马光志. 一种基于文法演化自动拟合非线性数据的蜂群算法[J]. 计算机应用研究, 2013, 30(10): 3257-3260

[28] Ganesh Kumar P. Hybrid Ant Bee Algorithm for Fuzzy Expert System Based Sample Classification[J]. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2014, 11 (2): 347-360

[29] UCI 数据集[OL]. http://archive.ics.uci.edu/ml/

(上接第 9 页)

[5] Mathioudakis M, Koudas N. Twittermonitor: trend detection over the twitter stream[C] // Proceedings of the 2010 ACM SIGMOD International Conference on Management of data. ACM, 2010: 1155-1158

[6] Gupta M, Gao J, Zhai C X, et al. Predicting future popularity trend of events in microblogging platforms[J]. Proceedings of the American Society for Information Science and Technology, 2012, 49(1): 1-10

[7] 郑斐然, 苗夺谦, 张志飞. 一种中文微博新闻话题检测的方法[J]. 计算机科学, 2012, 39(1): 138-141

[8] 郭跑秀, 吕学强, 李卓. 基于突发词聚类的微博突发事件检测方法[J]. 计算机应用, 2014, 34(2): 486-490

[9] Zhang J, Liu B, Tang J, et al. Social influence locality for modeling retweeting behaviors[C] // Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence. AAAI Press, 2013: 2761-2767

[10] Zhao S, Zhong L, Wickramasuriya J, et al. Human as real-time sensors of social and physical events: A case study of twitter and sports games[J]. arXiv preprint arXiv: 1106. 4300, 2011

[11] Lee R, Wakamiya S, Sumiya K. Discovery of unusual regional social activities using geo-tagged microblogs[J]. World Wide Web, 2011, 14(4): 321-349

[12] Weiler A, Scholl M H, Wanner F, et al. Event identification for local areas using social media streaming data[C] // Proceedings of the ACM SIGMOD Workshop on Databases and Social Networks. ACM, 2013: 1-6

[13] Boettcher A, Lee D. EventRadar: A real-time local event detection scheme using twitter stream[C] // 2012 IEEE International Conference on Green Computing and Communications (GreenCom). IEEE, 2012: 358-367

[14] Barabasi A L. The origin of bursts and heavy tails in human dynamics[J]. Nature, 2005, 435(7039): 207-211

[15] Leskovec J, McGlohon M, Faloutsos C, et al. Patterns of Cascading behavior in large blog graphs[C] // SDM. 2007, 7: 551-556