ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.3, March 2008, pp.650−662
DOI: 10.3724/SP.J.1001.2008.00650

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于 MHC 调控的免疫公式发现算法*

胡 珉[1+], 吴耿锋[2], 杨 晶[2]

[1](上海大学 悉尼工商学院,上海 200072)

[2](上海大学 计算机工程与科学学院,上海 200072)

MHC Regulation Based Immune Formula Discovering Algorithm

HU Min[1+], WU Geng-Feng[2], YANG Jing[2]

[1](Sydney Institute of Language and Commerce, Shanghai University, Shanghai 200072, China)

[2](School of Computer Engineering and Science, Shanghai University, Shanghai 200072, China)

+ Corresponding author: Phn: +86-21-69980028 ext 8509, Fax: +86-21-69980017, E-mail: minahu@shu.edu.cn

Hu M, Wu GF, Yang J. MHC regulation based immune formula discovering algorithm. *Journal of Software*, 2008,19(3):650−662. http://www.jos.org.cn/1000-9825/19/650.htm

Abstract: Aiming at the difficulty in good segments of the formula to be inherited in formula discovering using gene expression programming (GEP), this paper proposes an innovative immune formula discovering algorithm (IFDA), which is actually inspired by MHC (major histocompatibility complex) regulation principle of immune theory. In IFDA, the formula are mapped as a tree structure and transformed into both constant and variation section of antibody with a depth-first mechanism while its fragment is encoded into the MHC. By the feature of MHC regulation, IFDA mines the dataset to discover the proper formula very quickly. Many data are benchmarked for verifying the performance of IFDA in which all results from experiments show that the IFDA can really provide better performance than GEP in both convergence speed and formula complexity.

Key words: major histocompatibility complex; immune theory; formula discovering; gene expression programming; immune formula discovering algorithm

摘 要: 在分析了基于遗传原理的公式发现方法的优势与不足的基础上,根据免疫原理和 MHC(major histocompatibility complex)在免疫系统中的调控作用,提出了一种应用于公式发现领域的算法 IFDA(immune formula discovering algorithm)来解决公式进化中优良结构不易保护的问题.该算法将公式翻译成树状图,并按深度优先的编码方法形成抗体的恒定区和可变区代码,把公式片段编码成为 MHC 代码,借鉴 MHC 调控原理指导抗体进化,寻找出数据集合中蕴涵的规律,并用公式的形式表示.通过对多组基准数据的实验说明,此方法在公式复杂度和收敛速度方面比基因表达式算法有更好的性能.

关键词: 主要组织相容复合体;免疫原理;公式发现;基因表达式编程;免疫公式发现算法

中图法分类号: TP18 **文献标识码**: A

* Supported by the National Natural Science Foundation of China under Grant No.60275220 (国家自然科学基金); the Science Development Foundation of Shanghai of China under Grant No.012112027 (上海市科技发展基金)

Received 2006-08-10; Accepted 2006-12-27

数据是人们进行科学计算和工程设计的主要依据,用一个抽象的公式直观地归纳数据之间的关系是科学发展史上必不可少的过程.因此,如何迅速而准确地发现数据中蕴含的规律并用公式来表达,成为研究热点.

对数据归纳分析并发现公式的过程通常是由该领域研究人员完成的.随着人工智能技术的发展,公式发现技术发展迅速.目前常用的分析方法有遗传规划(genetic programming,简称 GP)[1]、遗传算法(genetic algorithm,简称 GA)[2]和基因表达式编程(gene expression programming,简称 GEP)[3].但如何选择合适的模型结构是回归分析中最为困难的问题.GP,GA 和 GEP 都是在遗传算法基础上发展起来的搜索算法. GP 采用的是动态的树状结构编码方式,树的结点由终止符、原始函数与运算符组成.这种树状结构的层与结点是可变化的,非常适合表达复杂的公式结构.但是由于组成群体的个体对应于不同的公式,因而长短不一,降低了寻找公式的速度.GA 采用串状基因编码方式,每个基因的长短一致,但是单纯的串结构很难描述层次化的问题,缺少动态可变性,不太适应复杂公式的发现.GEP 结合了 GA 和 GP 各自的优点,它的编码方式是先将个体编码为固定长度的线性串,待进行优化求解时再对操作对象重新翻译成树.这样结合,既能解决复杂问题又提高了求解速度,受到研究者的广泛关注[4].

然而,如何提取和保留公式中有用的片断,使公式中良好的特性得以继承是公式发现领域的一个难点.以 GEP 算法为例,其表现型(树状图)到基因型(字符串)转换时采用从左至右、从上到下的层次遍历得到.图 1 和图 2 就是式(1)在 GEP 中的不同表现形式.从图 2 中可以看出,基因型与表现型定义的映射关系使公式中原来在一起的片断,如 *a***b*,被分开放置,这不利于对公式子结构的保护.

$$\frac{a\cdot b}{c}+\sqrt{d-e} \tag{1}$$

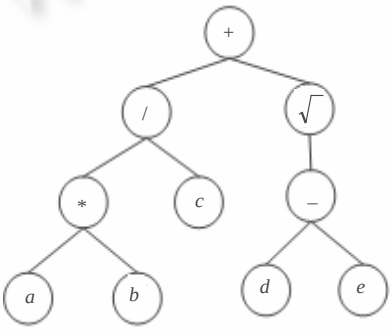


Fig.1 Representation type graph

图 1 表现型图

Fig.2 Genetic type graph

图 2 基因型图

公式发现从本质上讲是一个优化问题,而免疫算法在许多优化问题[6−8]中都得到了有效的运用.因此,本文试图将免疫原理引入公式发现领域,提出了基于免疫原理的公式发现算法 IFDA(immune formula discovering algorithm).

生物免疫系统是一个多层结构的防御系统,具有强大的识别学习和记忆的能力以及分布式、自组织和多样性等特性.其中,特异免疫响应是指抗体(antibody,简写 Ab)与外来入侵者抗原(antigen,简写 Ag)发生的生物化学反应.抗体是具有活性的免疫球蛋白,它由 4 条肽链组成,在肽链上有氨基酸数量和排列比较稳定的恒定区,称为 *C* 区;还有氨基酸的排列因抗体不同而不同的可变区,称为 *V* 区.尽管抗体数量比抗原少很多个数量级,但是,通过 *V* 区的高频变异和 *C* 区的变化,确保抗体几乎能够与所有的抗原结合.除了 Ab 和 Ag 以外,免疫响应过程中的另一个物质——主要组织相容性复合体(major histocompatability complex,简称 MHC)起到了关键的调控作用,指导抗体尽快地朝着与抗原结合的方向变化.

IFDA 算法模拟了 Ab,Ag 和 MHC 在免疫响应中的相互关系,将抗体编码映射为表达式(公式),将待分析数据集比作抗原,表达式的一部分比作 MHC,用抗体与抗原的亲和度体现为公式对数据集的拟合/归纳能力,而抗体的寻优过程就是体现数据集各变量相互关系的公式逐步得以浮现的过程.

1 免疫学中 Ag,Ab 和 MHC 定义[9]

在免疫系统中,除了抗体与抗原的相互作用以外,主要组织相容性复合体在免疫响应中也扮演着一个极为重要的角色.

1.1 抗原Ag

在生物免疫学中,抗原被定义为能够在机体中引起特异性免疫应答的物质,如细菌、病毒或其代谢产物等.

1.2 抗体Ab

抗体是在抗原刺激下机体免疫应答的产物,它与抗原产生选择性的反应.抗体由重链和轻链构成 *V* 型核心区域,分为恒定区和可变区两部分.在给定的物种中,不同抗体分子的恒定区都具有相同的或几乎相同的氨基酸序列;而可变区内有一小部分氨基酸残基变化特别强烈,这些氨基酸的残基组成和排列顺序更易发生变异.

1.3 主要组织相容性复合体MHC

主要组织相容性复合体是由一群紧密连锁的基因群组成,定位于抗体的特定区域,呈高度多态性.MHC 不仅与移植排斥反应有关,还广泛参与抗原递呈、制约细胞间相互识别等诱导免疫应答的诱导和调节,具有重要的免疫学意义.Messaoudi 等人[10]指出,根据实验和大量数据统计分析,MHC 的多态特性使 MHC 分子能够训练 *T* 细胞,确保 *T* 细胞在感染一开始就能尽快地杀死病原体,避免病原体所造成的疾病和死亡.这说明 MHC 在抗体选择和进化过程中起到了关键的调控作用,它不仅带来了抗体的多样性,并且调控抗体在免疫应答过程中尽快地与抗原结合.

2 IFDA 基本定义

2.1 抗原Ag

在 IFDA 中,准备用于公式发现的数据集合被定义为抗原.这些数据集通常为真实的实验数据或工程量测数据.

2.2 抗体Ab

在IFDA中,抗体对应于表达式(公式),其结构可以分为两部分,一部分是恒定区(低变异区),另一部分是可变区(高变异区).因此,在 IFDA 算法中也将抗体分为两个部分,*G* 区和 *C* 区.*G* 区存放公式的结构,类似于可变区;而 *C* 区存放公式中的一些系数,即常数,类似于恒定区.这两个区域被连接成一个线性串,如图 3 所示.

对于表达公式结构的 *G* 区,为了与代数表达式建立对应关系,借鉴 GEP 的方法,通过树状图进行映射,但是,GEP 采用的是从左到右“层序存放”形成树状图,而 IFDA 采用深度优先的“根序存放”形成树状图.以式(1)为例,采用深度优先的方法,得到的 *G* 区表达形式如图 4 所示.

Fig.3 The structure of antibody

图 3 抗体的结构

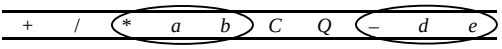


Fig.4 String type of antibody's *G* area in IFDA

图 4 IFDA 抗体串形表示(*G* 区)

图4中,*Q* 表示平方根.可以看出,在公式中比较接近的内容片断如 *a*b*,*d–e*,在抗体串型表达式中依旧连在一起,这样就比较容易进行片断的提取和继承了.

对于树状结点图,如果一个树结点最大允许的叶结点为 *M* 个,则很容易证明,如果该树有 *K* 个结点,那么总结点数不大于 *M*×*K*+1.因此,当运算符的最大目数(即运算符进行运算所必须的参数个数)已知且运算符个数确定的前提下,IFDA 的 *G* 区的长度就能被确定.

例如,已知运算符集{+,–,*,/,*Q*,*E*}中的最大目数是两目,其中,*Q* 表示平方根,*E* 是以自然数为底的指数表达式,要求公式中最多含有 5 个运算符,*G* 区的长度为 11.为了确保生成的公式有效,要求串的第 1 位为运算符.图 5

显示了部分随机产生抗体串的 *G* 区部分.对应的表达式树结构图如图 6 所示.

+	a	–	c	Q	a	*	a	/	b	c

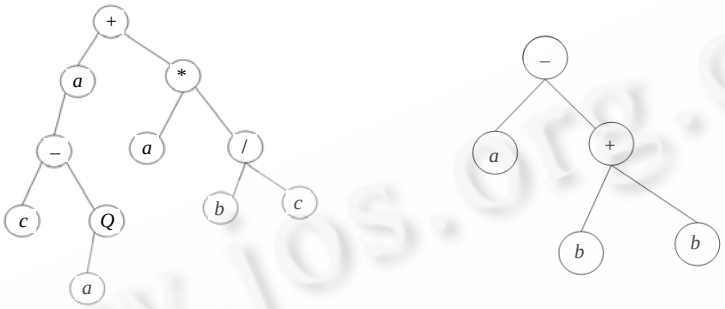
(a)

–	a	+	b	b	a	+	*	a	a	/

(b)

Fig.5 String type example of antibody's *G* area

图 5 抗体 *G* 区串型表达举例



(a) (b)

Fig.6 Tree type example of antibody's *G* area

图 6 抗体 *G* 区树型表达举例

从图 5 和图 6 可以看出,采用深度优先原则,自顶向下将串结构翻译成树结构,当树结构已经表达完整时,后面未被翻译 *G* 区的部分内容将被忽略.例如,图 5(a)的内容被完全映射成了表达式树,而图 5(b)只是串结构前面的有效部分被映射成了树结构.

抗体的 *C* 区用于说明公式中的常数项的具体内容.如果 *G* 区长度为 *L*,其中,运算符个数为 *k*,它所对应的 *C* 区长度为 *L*–*k*,这样就保证了公式中的每一位系数都能对应到 *C* 区中的一位.

在 IFDA 算法中设计了一个常数项集.常数项集通常随机产生,常数项中的常数可以是整数,也可以是浮点数.*C* 区的每一位用一个数字表示所对应的系数是常数项集中的第几个数据(即索引号).例如,设常数项集中由 8 个数据{1.5,2.34,5.6,0.8,1.2,3.14,9.36,10}组成.当 *C* 区的长度为 6 时,*C* 区内容与对应的数据关系如图 7 所示.

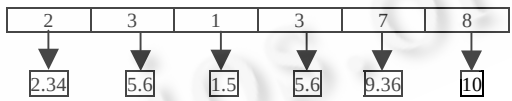


Fig.7 Example of antiboy's *C* area

图 7 抗体 *C* 区举例

因此,一个数学表达式可以通过抗体的串结构和表达树两种方法来显示,这三者存在着相互的对应关系.图 8 为抗体编码的例子.图 8(a)是抗体的串结构表示形式,左侧有浅色底纹的部分是表示公式结构的 *G* 区,右侧无底纹的部分为 *C* 区.*G* 区由 3 类符号组成:一类是运算符,如 *Q* 表示平方根;一类是公式中的变量,如 *x*;还有一类表示公式中的系数,均用"?"表示.每个"?"所代表的具体数值依照其先后次序,从 *C* 区中依次取出常数集中的索引号,再找到对应值.*C* 区的长度是根据 *G* 区而定的.以图 8(a)为例,假设确定的公式复杂度为包含 5 个运算符(最大目数为 2),那么根据前面的分析,*G* 区的长度为 5×2+1=11,而含有 5 个运算符的公式最多含有 6 个常数,因此,*C* 区的长度为 11–5=6.由于随机产生的 *G* 区还包含了 3 个公式变量 *x*,因此,*G* 区中只有 3 个常数,所以,实际使用的是 *C* 区的前 3 位.图 8(b)的树结构是按照自上向下、深度优先的方法,结合常数项集的内容得到的.而式(2)就是这两种结构对应的数学表达式.

$$y=\sqrt{x\cdot(2.34-x)\cdot 5.6/1.5} \tag{2}$$

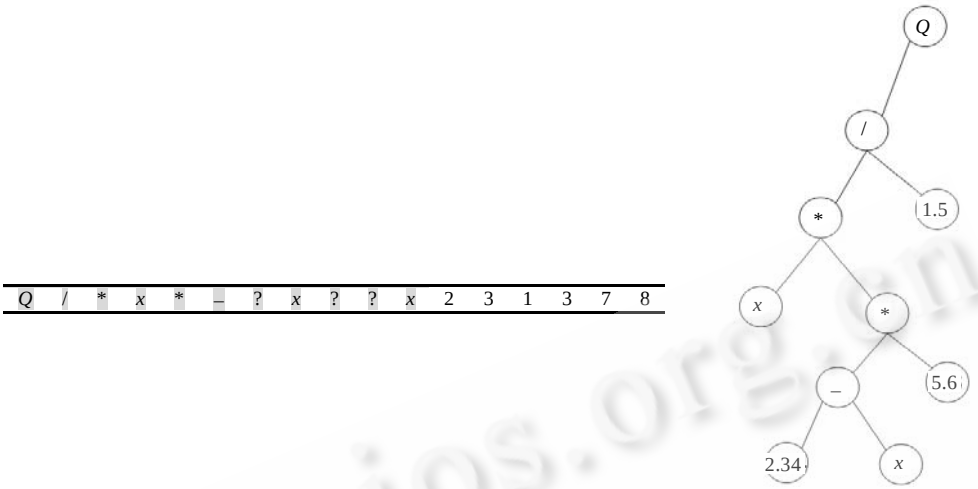


Fig.8 Example of antibody encode

图 8 抗体编码举例

2.3 适应度计算

适应度函数用来计算各抗体所对应个体的适应度,指导种群的进化.因此,适应度函数的选择直接影响到整个遗传操作的方向.在 IFDA 算法中,适应度函数的设计见式(3).

对于第 i 个抗体,

$$f(i)=\frac{1}{1+\sqrt{\sum_{j=1}^{S}(C_{ij}-T_j)^2}} \tag{3}$$

其中,

C_{ij}:第 i 个抗体对应的公式表达式利用第 j 个样本中的变量数据求得的值;

T_j:第 j 个样本中包含的实际测得的该目标函数值的真实值;

S:表示样本的数量.

因此,理想状态下,抗体的最大适应度值为 1.

2.4 MHC集合

2.4.1 MHC 集合生成

根据 MHC 限制原理,在后天免疫应答中,抗体必须识别 MHC 才能与抗原反应.因此,把抗体 G 区中的片段作为 MHC 分子,以确保生成的抗体的合法性.MHC 分子的长度小于抗体 G 区的长度,其编码方式与抗体相同.模拟免疫学中 MHC 的进化方式[11],IFDA 中 MHC 集合初始来源分为两个部分:一部分来源于与该公式有关的领域知识,如挖掘一个与能量有关的公式,按照经验可能与速度的平方有关,可以把"*VV"作为 MHC 分子(如果没有任何领域知识可以借鉴,可以将这部分的比例设置为 0);另一部分 MHC 则随机产生.MHC 初始集合中的 MHC 分子将以一定的比例替换随机产生的初始抗体群的 G 区,对初始抗体群进行修正.例如,初始的抗体群的数量是 100,MHC 的替换比例是 30%(替换的具体方法见第 3.3 节),那么,有 30%初始抗体的 G 区将进行 MHC 替换操作.

2.4.2 MHC 集合调整

MHC 的集合不是一成不变的,它随着抗体的进化过程进行动态更新.当抗体进化停滞代数大于 MHC 集合

更新代数的阈值时,MHC 集合根据强度进行调整.在 MHC 集合准备进行调整之前,除了原来 MHC 集合中的分子作为候选 MHC 以外,新增的候选 MHC 是从适应度大的优秀抗体集合片断 G 区中进行截取的.假设有一优秀抗体同图 8(a)所示,按照长度为 3、第一位必须是运算符的原则,可以截取的片段有 $Q/*,/*x,*x*,*-?,-?x$.

MHC 的强度计算公式为

$$I_j=\frac{\sum_{i=1}^{n}Find(MHC_j,Ab_i)\cdot f(Ab_i)\cdot Good(Ab_i)}{\sum_{i=1}^{n}Find(MHC_j,Ab_i)\cdot f(Ab_i)} \tag{4}$$

其中,

MHC_j:第 j 个 MHC;

Ab_i:第 i 个抗体;

$f(Ab_i)$的适应度;

I_j:第 j 个 MHC 的强度;

n:抗体集中抗体的数量;

$Find(MHC_j,Ab_i)$:判断 MHC_j 串是否在 Ab_i 串中,如果在 Ab_i 串中,则返回 1;否则,返回 0;

$Good(Ab_i)$:判断 Ab_i 是否属于优秀抗体集合,如果属于,则返回 1;如果不属于,则返回 0.

根据 MHC 分子的不同强度和 MHC 集合中的数量 k,选取强度最大的前 k 个 MHC,组成新的 MHC 集合,完成 MHC 集合的调整.

3 算子描述

进化算子是所有进化算法的核心,不同的算子设计对进化将产生极大的影响.

3.1 选择和复制

在 IFDA 算法中,选择与复制的依据是适应度值,采用轮盘赌加精英保留法策略来完成.由于抗体之间的差异非常大,所以,计算得到的适应度值的差距也非常大,如果直接用抗体的适应度作为比例计算的依据,可能会导致只有 1 个抗体被选择、复制,从而使进化过程过早地收敛.因此,在计算复制比例时,没有直接选用适应度值,而是根据适应度值的排位计算,见式(5).

$$R_i=\frac{(n-Rank(Ab_i))^2}{\sum_{i=1}^{n}i^2} \tag{5}$$

其中,

Ab_i:第 i 个抗体;

R_i:第 i 个抗体复制比例;

n:抗体集中抗体的数量;

$Rank(Ab_i)$:抗体在抗体集合中的排位,与抗原结合越好,适应度就越高,排位数也就越小.适应度最高的抗体,最小排位数为 0.

3.2 变 异

变异可以发生在抗体的任何部位.然而,抗体的结构组织必须保持完整无缺.在 G 区,头部的符号只能进行运算符变异,抗体其他位置的符号都可以变成其他符号(运算符、变量或常量),但是必须保证变异结束后整个抗体的运算符的个数不变;在 C 区,只能进行常数项位置的变异.这样,抗体的结构组织被保留下来,并且所有经变异生成的新个体在结构上都是正确的编排.

例如,考虑每个抗体进行双点变异的情况.如表 1 所示,设抗体为:

Table 1 Antibody before mutation

表 1 变异前抗体

Position	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7
Antibody	*Q*	/	*	*x*	*	–	?	*x*	?	?	*x*	2	3	1	3	7	8

假设变异位置选择为位置 3 和 8,统计被选择的点位变异前的运算符总数为 1,为了保证变异结束后整个抗体的运算符个数不变,这两个变异点位变异后的运算符总数也必须为 1.如果位置 3 的“*”变成了 *x*;位置 8 的 *x* 只能变异为运算符,假如变异成了“/”,得到新的抗体,见表 2.

Table 2 Antibody after mutation

表 2 变异后的抗体

Position	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7
Antibody	*Q*	/	*x*	*x*	*	–	?	/	?	?	*x*	2	3	1	3	7	8

3.3 MHC串替换

按照 MHC 的强度和 MHC 串替换率进行抗体的更新.MHC 串替换的起始位置选择与 MHC 长度和抗体 *G* 区的长度有关.如果 *G* 区的长度为 *L*,而 MHC 串长度是 *t*(通常 MHC 长度取为最大运算符目数加 1).当 MHC 的起始字符是运算符时,可以选择的位置范围在 1~*L*–*t*+1 之间;当 MHC 的起始字符不是运算符时,可以选择的位置范围缩小为 2~*L*–*t*+1 之间.

以表 1 表示的抗体为例,假设选择的位置是 1,MHC 分子的内容是 *E*–*x*,那么,MHC 替换后的抗体见表 3.

Table 3 Antibody after MHC replacement

表 3 MHC 替换后的抗体

Position	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7
Antibody	*E*	–	*x*	*x*	*	–	?	*x*	?	?	*x*	2	3	1	3	7	8

替换后的抗体进行运算符个数的校验,如果运算符的个数与设定要求不一致,则通过强制变异进行调整,以满足限制条件.

3.4 MHC串插入

按照 MHC 的强度和 MHC 串插入率进行抗体的更新.MHC 串插入的起始位置选择的范围与 MHC 串替换相同.以表 1 所示的抗体为例,假设选择的位置是 5,MHC 分子的内容是 *E*–*x*,那么,MHC 插入后的抗体如表 4 中上方的表格所示.

Table 4 Antibody after MHC insertion

表 4 MHC 插入后的抗体

Position	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7
Antibody	*Q*	/	*	*x*	*E*	–	*x*	*	–	?	*x*	2	3	1	3	7	8

Validation and adjustment (mutation)

Position	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7
Antibody	*Q*	/	*	*x*	*E*	–	*x*	?	*x*	?	*x*	2	3	1	3	7	8

插入串后的抗体需要进行运算符个数的校验,如果运算符的个数与设定要求不一致,则通过强制变异进行调整.以表 4 为例,MHC 插入后运算符总数减少了一个,根据不满足抗体的运算符数量不变的原则,必须通过变异,确保运算符的数量不变.可以采取的变异措施是:如果运算符个数大于规定运算符,则选择离插入串尾部最近的运算符(不包括插入串部分)变异为操作数;如果运算符个数小于规定运算符,则选择离插入串头部最近的操作数(不包括插入串部分)变异为运算符.表 4 所示,经过 MHC 串插入操作以后,运算符总数由 5 变成了 7,因此,对第 8 位和第 9 位进行强制变异,第 8 位的内容“*”变异为“?”,第 8 位的内容“–”变异为 *x*.

3.5 移 位

算法中,可移位的元素是一些抗体的片段,这些片断可以被激活并跳转到抗体的其他部位.移位可以发生在 *G* 段,也可以发生在 *C* 段,但不能在这两段之间发生互相之间的移位.以表 1 所示的抗体为例,假设移位点位是 *G* 段的 3 到 5,移入的位置是 2,那么,移位后的抗体见表 5.

Table 5 Antibody after shift

表 5 移位后抗体

Position	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7
Antibody	*Q*	*	*x*	*	/	–	?	*x*	?	?	*x*	2	3	1	3	7	8

3.6 交 换

算法中可以进行两个元素的交换.与移位相同,交换可以发生在 *G* 段,也可以发生在 *C* 段,但不能在这两段之间发生互相之间的移位.以表 1 所示的抗体为例,假设交换点位是 *G* 段的 3 和 7 的位置,那么,交换后的抗体见表 6.

Table 6 Antibody after exchange

表 6 交换后抗体

Position	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7
Antibody	*Q*	/	?	*x*	*	–	*	*x*	?	?	*x*	2	3	1	3	7	8

3.7 常数集更新

公式的优劣除了与公式的结构形式有关以外,与公式中常数的选取也有密切的联系.因此,在算法的操作算子中设置了常数项的变异算子,进行常数项集更新.但是,这个特殊的算子并不是每一代都运行,而是只在停滞代数大于常数集更新代数阈值时才运行.项集中常数变异的概率与其每个常数在优秀个体中使用的比例有关,使用比例越高的常数,其变异率越低.

4 算法实现

IFDA 算法.

Step 1:算法参数初始化

基本参数设定,包括单条抗体的运算符数、抗体群数目、MHC 数目、最大运行代数、最大允许相同运算代数、MHC 集合更新代数阈值、变异概率、MHC 串替换概率、MHC 串插入概率、移位概率、交换概率和常数集更新代数阈值.

Step 2:群体初始化

MHC 初始化,抗体初始化.

Step 3:计算抗体群亲和度

分别计算抗体群中 *n* 个抗体的亲和度,选出优秀抗体.

Step 4:结束判断

更新运行代数和停滞代数:如果停滞代数大于最大允许相同运算的代数或者运行代数大于最大运行的代数程序,则转向 Step 8;否则,转至 Step 5.

Step 5:MHC 集合调整

如果停滞代数大于 MHC 集合更新代数阈值,则根据优秀抗体产生候选 MHC 集合,计算 MHC 强度,进行 MHC 集合调整.

Step 6:常数项集合调整

如果停滞代数大于常数项集合更新代数阈值,则进行常数项集合调整.

Step 7:抗体进化

包括抗体的选择和复制、变异、MHC 串替换、MHC 串插入、移位和交换操作.返回 Step 3.

Step 8:结束

从算法流程可以看出,除了抗体正常的进化过程以外,算法还通过 MHC 集合的调整以及常数项集合的调整,进行公式结构调整的指导以及公式常数项调整的指导.

5 实验与结果

5.1 实验数据

由于公式发现问题比较复杂,没有得到普遍承认的基准测试数据,因此,CS[12]从各类科学文献中收集了 220 组数据(http://www.ics.uci.edu/~mlearn/databases/function-finding/function-finding.data)作为评判公式发现问题的基准测试数据,这些数据包含了各类误差和各种不确定的因素,比使用人为公式设计的测试数据更能反映公式发现方法在性能上的优劣.

在本实验中,从这 220 组数据中随机选取了 20 组数据进行性能测试,被选中的测试数据集编号分别为 11,22,23,41a,41b,47,58,68,73,84a,84b,90,127a,127b,135d,143,148,160,179,212a,215.

5.2 实验设置

为了客观地分析 IFDA 的性能,选择了在公式发现上表现出色的基因表达式算法进行对比.整个实验按终止条件分为两个类别进行:第 1 类的程序运行终止条件为:如果进化出现 K(分别取 K 为 30 和 100)代以上的停滞,则程序终止运行;第 2 类的运行终止条件是进化运行 1 000 代后,程序终止运行.对于 IFDA 算法,将其分为两种方式进行:一种是基本方式 IFDA_1,这种方式的初始 MHC 集合中不包含人为设定的 MHC;另一种是改进方式 IFDA_2,在基本方式基础上,初始 MHC 集合中包含了两个人为设定的 MHC("/??"和"^??"),其目的是为了扩大公式中常数项值的区间;同时,IFDA_2 常数项集处理也与 IFDA_1 有所不同,采用了整数优先策略.也就是说,初始生成的常数均为整数(目前采用的取值空间为–10~10),在整数常数进化受阻后,变异为浮点数.

5.2.1 GEP 参数设置

在所有的实验中,设置每条抗体的基因数为 3,基因头部长度为 6,抗体数为 300,抗体变异率为 0.044,常数变异率为 0.044,IS 移位率为 0.1,IS 串长度为 0.1,IS 常数移位率为 0.1,IS 常数串长度为 3,基因移位率为 0.1,单点交叉率为 0.3,两点交叉率为 0.3,基因交叉率为 0.1,常数项集规模为 10,包括一个固定的常数 2.781 8 和 3.141 57,其他由–1~1 之间的随机数组成.

5.2.2 IFDA_1 参数设置

在所有的实验中,设置单条抗体的运算符数为 10,抗体群数目为 300,MHC 数目为 10,MHC 集合更新代数阈值为 20,变异概率 0.3,MHC 串替换概率为 0.15,MHC 串插入概率为 0.15,移位概率为 0.2,交换概率为 0.2,常数集更新代数阈值为 20.常数项集规模为 10,包括一个固定的常数 2.781 8 和 3.141 57,其他由–10~10 之间的随机浮点数组成.

5.2.3 IFDA_2 参数设置

在所有的实验中,设置单条抗体的运算符数为 10,抗体群数目为 300,MHC 数目为 10,初始设定 MHC 为"/??"和"^??",MHC 集合更新代数阈值为 30,变异概率为 0.3,MHC 串替换概率为 0.15,MHC 串插入概率为 0.15,移位概率为 0.2,交换概率为 0.2,常数集更新代数阈值为 20.常数项集规模为 10,初始包括一个固定的常数 10 和 1,其他由–10~10 之间的随机整数组成.

每种方式均运行 10 次,通过其适应度平均值比较性能.

5.3 实验结果

IFDA 与 GEP 对比的数据拟合度和收敛特性比较见表 7.

Table 7 Algorithm performances comparison

表 7 算法性能比较

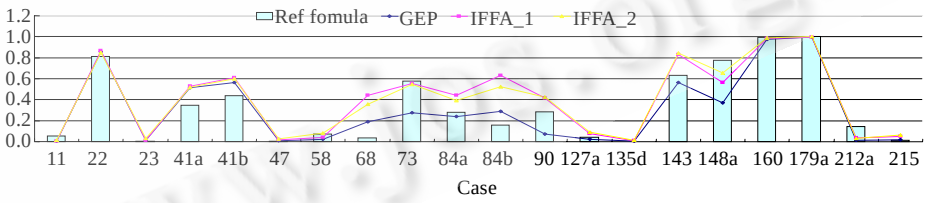
Case	Average fitness (stagnation for 30 generations)			Average fitness (running 100 generations)			Convergence rate (stagnation for 100 generations)		
	GEP	IFDA_1	IFDA_2	GEP	IFDA_1	IFDA_2	GEP (%)	IFDA_1 (%)	IFDA_2 (%)
11	0.003 9	0.006 7	0.011 3	0.005 1	0.015 5	0.058 5	20	40	60
22	0.855 8	0.865 3	0.850 2	0.897 6	0.884 3	0.893 1	30	60	90
23	0.013 6	0.015	0.030 3	0.005 5	0.029 8	0.032 1	20	80	100
41a	0.513	0.527 4	0.520 4	0.513 2	0.548 2	0.538 9	40	90	90
41b	0.565 9	0.607 7	0.601 3	0.512 6	0.612 4	0.615	10	80	90
47	0.007 7	0.019 8	0.025 6	0.008 4	0.0289	0.043 3	10	80	90
58	0.022 2	0.041 3	0.082	0.022 6	0.083 2	0.086	10	30	30
68	0.190 1	0.443 7	0.356 7	0.203 8	0.443 7	0.387 9	30	90	80
73	0.274 7	0.556 2	0.543 7	0.340 1	0.569 8	0.623 5	20	80	90
84a	0.237 5	0.441 1	0.391 3	0.279 8	0.599 9	0.493 2	40	100	80
84b	0.286 7	0.630 4	0.523 4	0.320 9	0.726 6	0.753 5	40	50	70
90	0.07	0.414 1	0.423 4	0.279	0.644 2	0.653 2	10	40	40
127a	0.024 4	0.075	0.089 5	0.055 2	0.103 6	0.126 8	20	30	30
127b	0.004 6	0.010 8	0.012 1	0.019	0.037 6	0.057	20	50	90
135d	0.562 9	0.830 9	0.845 2	0.634 5	0.865 4	0.893 1	50	90	90
143	0.371 4	0.563 8	0.653 2	0.525 7	0.674	0.693 3	20	80	90
148a	0.972 3	0.981 9	0.989	0.976 5	0.985 2	0.991 7	30	70	80
160	0.999	0.998 1	0.999 9	0.999 5	0.998 4	0.999 9	30	40	50
179a	0.015 6	0.038 2	0.032 8	0.014 4	0.080 4	0.040 9	30	60	90
212a	0.019 7	0.05	0.064 2	0.023 9	0.082 8	0.100 7	40	70	100
215	0.003 9	0.006 7	0.011 3	0.005 1	0.015 5	0.058 5	20	100	100
Avg	0.29	0.39	0.38	0.32	0.43	0.44	26	66	78

Remark: Data 拟合度计算公式为式(2),Avg. is the average of these cases

除了将 IFDA 与 GEP 相对比以外,还与提供原始实验数据的研究人员在文献中归纳的公式进行对比,绘制了数据拟合度的对比曲线图,如图 9 所示.

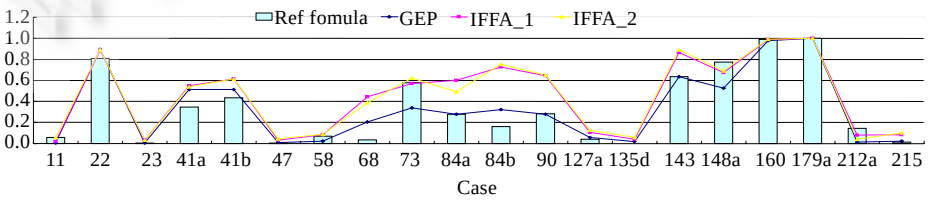
为了比较各种方法得到的公式的复杂程度,以公式中运算符为依据,绘制了公式复杂度比较图(如图 10 所示).图中的纵坐标表示公式中含有的运算符的个数.

为了比较 IFDA 和 GEP 方法的运行速度,统计了各个算例运行 1 000 代所需要的平均时间,如图 11 所示.同时,以每代的最小拟合误差为基准,绘制误差变化曲线图(如图 12 所示),从而探寻这两类算法搜索特性的异同点.



(a) Stop when stagnation for 30 generations

(a) 当 30 代停滞时程序结束



(b) Stop when stagnation for 100 generations

(b) 当 100 代停滞时程序结束

Fig.9 Formula fitness comparison

图 9 公式拟合度比较

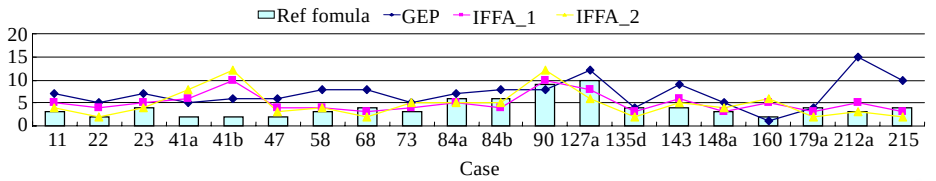


Fig.10 Formula complexity comparison

图 10 公式复杂度比较

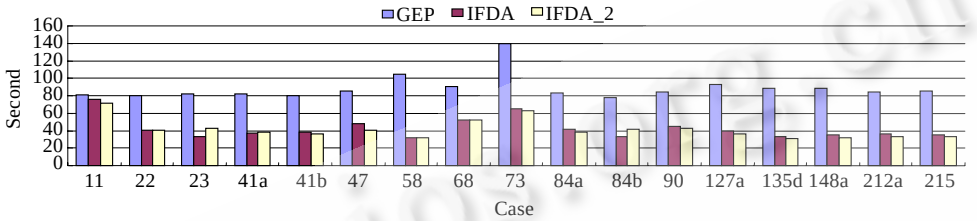


Fig.11 Time per generation comparison (run for 1 000 generation)

图 11 运行时间比较(运行 1 000 代)

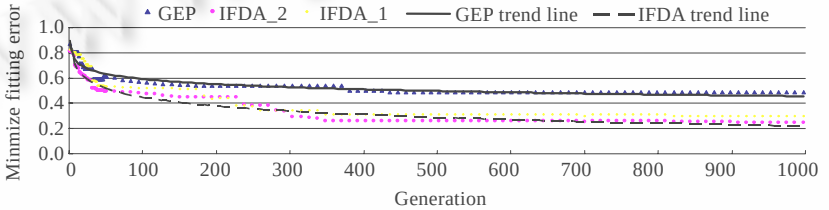


Fig.12 Evolutionary curve comparison (run for 1 000 generation)

图 12 进化曲线比较(运行 1 000 代)

5.4 结果分析

从表 7 和图 9 可以看出,在 30 代停滞进化结束的实验中,IFDA 在 case41a,case41b,case68,case84a,ccase84b,case90,case127a,case143 和 case215 这 9 个算例中适应度明显高于原始研究给出的拟合公式;但 case11,case148a 和 case212a 中低于原始研究给出的拟合公式在其他算例中拟合效果类似.IFDA 与 GEP 算法相比,IFDA 适应度几乎全部优于 GEP 得到的公式的适应度,特别是 case68,case73,case84a,case84b,case90,case127a,case143 和 case148a 这 8 个算例的效果更为明显.IFDA_1 和 IFDA_2 两种方法相比,效果没有明显的差异.

在进化 1 000 代的实验中,IFDA 在 case41a,case41b,case68,case84a,ccase84b,case90,case127a,case143 和 case215 这 9 个算例中适应度明显高于原始研究给出的拟合公式;但在 case11,case148a 和 case212a 中低于原始研究给出的拟合公式,在其他算例中拟合效果类似.这一点与 30 代停滞进化结束的实验结果完全相同,但是进化 1 000 代的实验的适应度值高于 30 代停滞进化结束的实验.

IFDA 与 GEP 算法相比,IFDA 适应度全部优于 GEP 得到的公式的适应度,特别是 case41a,case41b,case58,case68,case73,case84a,case84b,case90,case127a,case143,case148a,case212 和 case215 这 13 个算例的效果更为明显.可见,在同等运算次数下,IFDA 比 GEP 算法效果更为明显.另外,IFDA_1 和 IFDA_2 两种方法相比,IFDA_2 效果略好于 IFDA_1.

从图 10 可以看出,原来研究提供的参考公式都比较简洁.GEP 算法除了在 case90,case135 和 case160 中得到了较为简洁的公式形式以外,其他时候的公式都非常繁琐.而 IFDA 有 14 个算例的公式与参考公式同样简洁或更为简洁,其中有 5 个算例得到的公式形式与参考公式类似,而其余 9 个算例得到的公式形式不相同.观察与参考公式形式同样简洁的 IFDA 的算例:case47,case68,case135,case179,发现 IFDA 得到的公式更能反映出实际

测量值的情况,即适应度更高.而 IFDA_1 和 IFDA_2 性能类似,IFDA_2 略优于 IFDA_1.

从图 11 可以看出,就每一代的平均运算速度而言,在绝大部分算例中,IFDA 要比 GEP 快两倍以上.而 IFDA_1 和 IFDA_2 的速度基本相同,在 case23 和 case84b 中,IFDA_2 比 IFDA_1 略快,其他算例两者基本相同.从图 12 中可以看到,在达到相同拟合程度的情况下,IFDA 比 GEP 需要的搜索代数要少.这说明 IFDA 与 GEP 比较,进行公式发现的时间明显少于 GEP 算法.表 7 统计了各算例在运行出现 100 代以上停滞时,程序结束的条件下能够达到满意拟合度(以大于已知最优解的 75%为条件)的收敛比例.GEP,IFDA_1 和 IFDA_2 的平均值分别为 26%,66%,76%.从总体上看,在此种运行条件下,IFDA 算法的收敛性能好于 GEP 算法.

综上所述,IFDA 能够比 GEP 更好地拟合测量数据集合,得到的公式的形式也易为人们所接受.IFDA 算法在搜索公式机制方面比较灵活,在研究人员没有领域知识时,算法能够较快地找到合适的公式;当研究人员具有一定的领域知识时,算法能够有效地利用经验知识,缩小搜索范围,更快地找到能够反映数据的公式.

6 结 论

公式发现方法的研究在实际中具有重要的意义.由于公式的表现形式非常复杂,因此,关于公式发现的研究进展不是很大,而进化算法的出现给公式发现研究带来了新的生机.本文借鉴各种进化算法在公式发现问题上的求解思路,提出了利用免疫系统的抗体进化来模拟公式,从而提高拟合能力的 IFDA 算法.IFDA 主要有 3 个特点:(1) 采用了深度优先的“根序存放”进行树与串之间的相互转换,保证了树结构紧密关联的内容在串结构的物理位置上也紧密关联;(2) 利用树形结构的特点,分析了公式中运算符和系数之间的关系,制定了等长串的生成规则,既保证了公式的复杂度,也保证了操作的简便性;(3) 充分应用了免疫系统 MHC 调控原理,提高了公式发现的能力,加快了公式发现的速度.

IFDA 的研究为免疫算法开辟了新的应用场合,但这只是一个开始.如何进行分段公式发现等问题,还有待进一步的探索.

致谢 我们衷心感谢论文评审专家给予本文非常有价值的建议和指导.

References:

[1] Koza JR. Genetic Programming II. Cambridge: MIT Press, 1994.

[2] Goldberg DE. Genetic Algorithms in Search Optimization and Machine Learning. Addison-Wesley, 1989.

[3] Ferreira C. Gene expression programming: A new adaptive algorithm for solving problems. Complex Systems, 2001,13(2):87−129.

[4] Tang CJ, Zang TQ, Zuo J, Wang R, Jia XB. Knowledge discovery based on gene expression programming-history, achievements and future directions. Computer Application, 2004,24(10):7−10 (in Chinese with English abstract).

[5] Schaffer C. A proven domain-independent scientific function-finding algorithm. In: Proc. of the 8th National Conf. on Artificial Intelligence. Boston: MIT Press, 1990. 828−833. http://citeseer.ist.psu.edu/7808.html

[6] De Castro LN, Timmis J. An artificial immune network for multimodal function optimization. In: Proc. of the IEEE Congress on Evolutionary Computation (CEC 2002). Honolulu: IEEE Press, 2002. 699−704.

[7] Endo S, Toma N, Yamada K. Immune algorithm for *n*-TSP. In: Pattipati K, *et al*., eds. Proc. of the IEEE Int'l Conf. on Systems, Man and Cybernetics. San Diego: IEEE Press, 1998. 3844−3849.

[8] Toma N, Endo S, Yamada K, Miyagi H. An immune optimization inspired by biological immune cell-cooperation for division-and-labor problem. In: Proc. of the 4th Int'l Conf. on Computational Intelligence and Multimedia Applications. Yokusika: IEEE Press, 2001. 153−157. http://ieeexplore.ieee.org/search/srchabstract.jsp?arnumber=970460&isnumber=20917&punumber=7656&k2dockey=970460@ieeecnfs&query=an+immune+optimization+inspired+by+biological+immune+cell-cooperation+for+division-and-labor+problem&pos=0&access=no

[9] Lydyard PM, Whelan A, Fanger MW. Immunology. Beijing: BIOS Scientific Publishers Limited, 2000. 113−130.

[10] Messaoudi I, Guevara Patino JA, Dyall R, LeMaoult J, Nikolich-Zugich J. Direct link between MHC polymorphism, T cell avidity, and diversity in immune defense. Science, 2002,298(29):1797−1780.

[11] Borghans JAM, Beltman JB, De Boer RJ. MHC polymorphism under host-pathogen coevolution. Immunogenetics, 2004,55: 732–739.

[12] Schaffer C. Bacon, data analysis and artificial intelligence. In: Maria A, ed. Proc. of the 6th Int'l Workshop on Machine Learning. Ithaca: Morgan Kaufmann Publishers, 1989. 174–178. http://citeseer.ist.psu.edu/144070.html

附中文参考文献:

[4] 唐常杰,张天庆,左劼,汪锐,贾晓斌.基于基因表达式编程的知识发现——沿革,成果和发展方向.计算机应用,2004,24(10):7–10.

胡珉(1970－),女,浙江上虞人,博士,副教授,主要研究领域为智能信息处理,地下工程信息处理.

杨晶(1982－),女,主要研究领域为数据挖掘,人工智能.

吴耿锋(1945－),男,教授,博士生导师,主要研究领域为智能信息处理,人工智能应用.