基于免疫遗传的软件衰退检测算法*)

徐 建 游 静 刘凤玉

(南京理工大学计算机科学与技术系 南京 210094)

摘 要 本文吸取了免疫学的灵感,提出了一种新的方法来验证软件衰退的出现,也就是检测软件运行中的性能异常。这种方法结合了阴性选择算法和遗传算法,使用模糊逻辑产生模糊集来区分正常和异常的性能状态,使用了阴性选择算法充当过滤器来消除不合法的检测子、降低搜索空间。最后使用 Mackey-Glass 时间序列产生的数据集和知名的 UCI 数据库的一组数据进行了仿真实验,来验证本方法的可行性和有效性。

关键词 软件衰退,异常检测,阴性选择,遗传算法

An Immuno-Genetic-Based Approach for Software Aging Detection

XU Jian YOU Jing LIU Feng-Yu

(Department of Computer Science and Technology, Nanjing University of Science and Technology, Nanjing 210094)

Abstract This paper presents a new approach inspired by immunology for validation of software aging and detection of system performance anomaly. It combines the negative selection algorithm and genetic algorithm, and generates fuzzy sets with fuzzy logic to code chromosome. The negative selection algorithm serves as a filter to eliminate invalid detectors and reduce search space. Experiments with synthetic and real data sets are performed to show the applicability of the proposed approach.

Keywords Software aging, Anomaly detection, Negative selection, Genetic algorithm

软件衰退的现象指的是一个长时间持续运行的软件系统会发生状态退化和性能降低,最终导致系统崩溃。发生软件衰退现象的主要原因是系统资源的耗尽[1]。常见的软件性能缓慢衰退的例子有内存泄漏和溢出、未释放的文件锁、存储空间的碎片、不足够的交换空间、网络带宽不足、错误的累积和数据的破坏等。许多研究人员[2,3]提出了使用基于度量的可靠性估计方法来检测软件的衰退和评价系统性能状况,他们的方法主要是利用发生故障或错误时所采集的数据进行分析。对于软件衰退的检测和评估,仅利用故障数据是不够的,因此本文的基本思想就是不断监控运行中软件的性能参数,收集性能数据并对其进行分析,检测软件衰退的出现。

异常检测问题已经在很多领域被研究,如入侵检测、故障检测等。相应地,研究人员也提出了不同的方法来解决这个问题,有基于 Chi-square 的统计模型方法[4]、基于机器学习的方法[5]等。本文吸收了自然免疫系统(Natural Immune System, NIS)的免疫机理,从另一个角度提出了解决问题的新方法。NIS 能够有效地把外来的细胞、分子和机体的细胞区分开来,这就是知名的自我非我免疫识别(self-nonself discrimination)。Forrest 等人[6]基于此原理提出了阴性选择算法,成功地应用到了图像识别[7]和网络入侵检测[8]。然而,有两个问题妨碍了它的更为广泛的应用:第一个是严重的扩缩性问题,在面对真实世界的大数据量时,算法效果不佳,J. Kim 在文[9]中证实了这个观点;第二个问题是尖锐的自我和非我的划分,这往往不能满足真实环境的需要。本文为了避免阴性选择方式的扩缩性问题,把阴性选择充当过滤器来删除不合法的检测子,降低搜索空间;为了避免自我非我之间尖锐的划分,使用模糊逻辑来划分是自然的方式。

本文吸取了免疫学的免疫识别机理,提出了一种新的方法来验证软件衰退的出现,也就是检测软件运行中的性能异常。这种方法结合了阴性选择算法和遗传算法,使用模糊逻辑产生规则集来区分正常和异常的性能状态,使用阴性选择算法充当过滤器来消除不合法的检测子、降低搜索空间。并且,进行了仿真实验来验证本方法的可行性和有效性。

1 基于免疫遗传的方法

软件衰退检测属于二元分类的范畴,目的在于把系统的状态划分到两个分类中,即正常状态(自我)和衰退状态(非我)。本文的思想是基于模糊逻辑来生成检测子,对于给定的包含自我和非我的样本集,生成非我空间的检测子来确定一个新的状态属于哪一类。样本空间中的任一样本使用向量的方式$(x_1, x_2, \cdots, x_n)$来表示,其中 $x_i \in [0,1] i=1,\cdots,n$,$n$ 为样本的属性个数。检测子具有如下形式:

IF *condition* THEN *non−sdlf*(非我)

其中 *condition* 可表示如下:

$\langle condition\rangle \rightarrow \langle condition\rangle\langle oper1\rangle\langle atom\rangle \mid \langle atom\rangle$

$\langle atom\rangle \rightarrow \langle variable\rangle\langle oper2\rangle\langle set\rangle$

$\langle variable\rangle \rightarrow x_1 \mid x_2 \mid \cdots \mid x_n$

$\langle set\rangle \rightarrow \langle set\rangle\langle oper3\rangle\langle linguistic\rangle \mid \langle linguistic\rangle$

$\langle linguistic\rangle \rightarrow s_1 \mid s_2 \mid \cdots \mid s_m$

$\langle oper1\rangle \rightarrow \wedge$

$\langle oper2\rangle \rightarrow \in$

$\langle oper3\rangle \rightarrow \vee$

*)基金项目:国家自然科学基金(No. 60273035)。徐 建 博士研究生,主要研究领域为软件自愈与抗衰、信息安全;游 静 博士研究生,主要研究领域为软件自愈与抗衰、信息安全;刘凤玉 教授,博士生导师,主要研究领域为人工智能和网络安全。

$S=\{s_i, i=1,\cdots,m\}$是由模糊隶属函数 F 在实数区间[0.0,1.0]产生的模糊集。

本文提出的算法中使用了遗传算法来进化检测子覆盖非我空间，算法的输入是包含自我和非我的样本集合，以及进化的代数，具体表示如下。

2.1 染色体编码

从以上检测子的形式可以看出，检测子的后件部分都是相同的，因此染色体编码的是规则的前件部分。样本的每一个属性就对应着染色体的一个基因，而每一个基因用 m 长的位串 $s_1^i s_2^i \cdots s_m^i$ 来表示，其中 i 代表染色体的第 i 个基因，m 为模糊隶属函数所产生的模糊集的个数，如果 $\mathrm{Max}(F(x_k)) \in s_t, t=1,\cdots,m$，则 $S_t^i=1$，其余的 $s_w^i=0, w\neq t$。图 1 表示 $m\times n$ 位长的染色体的结构。

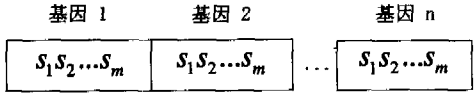

图 1 染色体的编码

2.2 距离度量

为了度量两个个体之间的距离，本文提出了新的部分匹配的距离度量方法。当两个个体相同的基因数大于等于预先定义的阈值 M 时，发生免疫识别，可以用如下公式表示：

$$\mu(R,x)=\begin{cases}1, \text{if } matdh(R,x)\geqslant M\\ 0, \text{otherwise}\end{cases} \tag{1}$$

其中

$$match(R,x)=\sum_{i=1}^{m} geneMatch(R_i,x_i) \tag{2}$$

并且

$$geneMatch(x,y)=\begin{cases}1, \text{if}\ \exists j, 1\leqslant j\leqslant m \text{ and } x^j\equiv y^j\\ 0, \text{otherwise}\end{cases} \tag{3}$$

2.3 适应度评价

每一个检测子的适应度值计算考虑了以下两个因素：一个是检测子所能覆盖的自我样本空间中的自我样本个数，可以通过如下公式表示，

$$num_{\text{self}}(R)=\sum_{x\in \text{self}}\mu(R,x)/|\text{self}| \tag{4}$$

另一个因素是使用模糊逻辑产生的检测子能表示的空间的大小，可以通过如下公式计算：

$$volume(R)=\prod_{i=1}^{n} measure(T_i) \tag{5}$$

其中当且仅当 $s_j^i=1$ 时，$measure(T_i)=\sum_{j=1}^{m}(high_j-low_j)$。

综合考虑以上两个因素，适应度值按照如下公式计算：

$$fitness(R)=volume(R)-Cnum_{\text{self}}(R) \tag{6}$$

C 为惩罚系数。如果某一个检测子覆盖了自我空间的样本，就要受到惩罚。C 值越大，惩罚越大，该检测子适应度值也就越小。

2.4 目标函数

有三个因素决定了本文提出的性能异常检测的性能：一是 TP 值，表示系统检测出的异常样本数目与实际的异常数目的比值；二是 FP 值，表示系统把正常误当作异常检测出来的样本数与实际正常样本数目的比值；三是 SP 值，表示简单性，即用最少数目的检测子覆盖异常样本空间。

$$TP=\sum_{x\in non-self}\mu(R,x)/|non-self| \tag{7}$$

$$FP=\sum_{x\in self}\mu(R,x)/|self| \tag{8}$$

$$SP=|detectors|/|non-self| \tag{9}$$

因此目标函数定义为：

$$cf(TP,FP,SP)=W_{tp}TP-W_{fp}FP-W_{sp}SP \tag{10}$$

W_{tp}, W_{fp}, W_{ap} 分别代表三者的权重。目标是使 cf 取最大值，即 TP 越大越好，FP 越小越好，而检测子数目则是越少越好。

2.5 算法表述

本文提出的算法组合了阴性选择和遗传算法，把阴性选择作为一个过滤算子加入到遗传算法中去，来消除不合法的检测子，降低 FP 的值。算法的输入为正常状态的样本集 PE 和异常状态的样本集 NE，以及进化代数 numGenerations。参数 M 是一个阈值，实验中可以调整。详细的算法描述分成两部分：第一步是把 NE 集合作为候选的初始检测子集合 PRE_DETECTORS，对于其中的每一个样本，如果不与 PE 集合中的任何样本发生免疫识别，则保留下来，否则从 PRE_DETECTORS 中删除此检测子。这里使用的阴性选择算子使用了上述的距离度量公式。其中阈值参数 M 在 1 到染色体的基因数目之间变动。可用如下伪代码表示：

```
设置初始的检测子集合 PRE_DETECTORS 为 NE
For EXi∈NE,i=1,…,|NE|
  For EXj∈PE,j=1,…,|PE|
    对 EXi EXj 应用阴性选择算子；如果相同的基因数不小于 M,则
    从 PRE_DETECTORS 中删除 EXi,否则 BREAK;
  END_For
END_For
```

第二步是使用遗传算法进行进化。与传统的遗传算法不同之处在于引入了阴性选择算子充当过滤器，对进化的子代进行过滤。本文使用的是简单遗传算法，可以很容易地扩展到其他改进过的遗传算法。由于在经过交叉、变异等概率随机事件后，生成的子代个体会与 PE 集中的个体发生免疫识别，因此使用阴性选择算子来删除这些不合法的子代个体。流程图如图 2 所示。

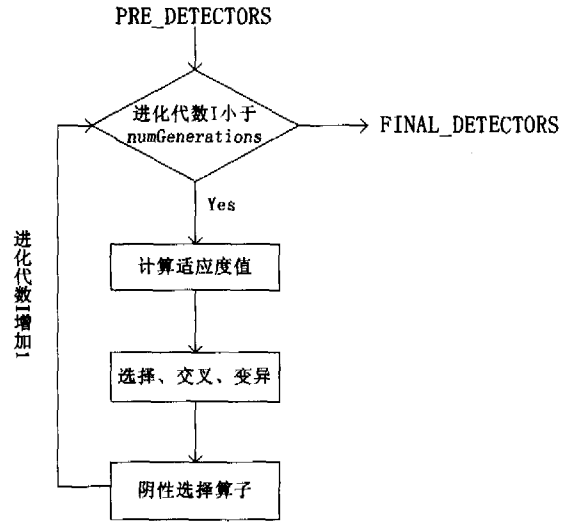

图 2 引入阴性选择算子后的算法流程

3 仿真和结果分析

为了验证算法的有效性，在两组数据上进行了仿真实验：一组是 Mackey-Glass 时间序列数据，另一组是来自 University of California, Irvine (UCI)[11] 的著名机器学习数据库中的乳腺癌数据集。所有的数据首先都被正规化到实数区间[0.0,1.0]。由于模糊隶属函数的合理选择本身就是一个研究课题，故本文只考虑简单而常用的模糊隶属函数，如三角和梯形隶属函数。产生了 5 个基本的模糊集，分别是 S、MS、M、

ML、L，模糊集的个数是可以扩展的，如图 3 所示。

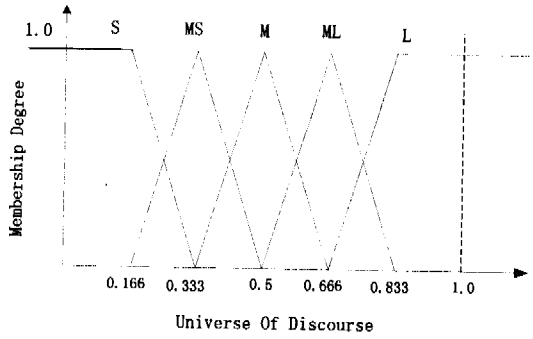

图 3　5 个基本的模糊集

本文实验中简单遗传算法相关参数值选取通常[12]的设置，变异概率 P_m 为 0.016，交叉概率 P_c 为 0.5，进化代数 T 取 200。

3.1　Mackey-Glass 数据集

本实验使用非线性微分方程 Mackey-Glass 来产生数据集，方程如式(11)所示：

$$\frac{dx}{dt}=-\frac{ax(t-\tau)}{1+x^c(t-\tau)}-bx(t) \tag{11}$$

在实验中参数的设置参照了这一领域[10]通常的设置，即取 $a=0.2$，$b=0.1$ 和 $c=10$。而参数 τ 用来控制序列的复杂度，生成不同的样本。实验需要生成正常样本集、异常样本集和测试集三个数据集。本文使用 $\tau=30$ 来产生 1000 个数据的正常样本集。为了克服初始值的影响，舍去方程产生的前 1000 个样本点，结果的时间序列如图 4 所示。在 $\tau=17$ 的情况下，以同样的方式产生 500 个异常样本数据，其结果如图 5 所示。为了产生具有正常和异常样本的测试集，在 300～400 这一时间段改变 τ 的值，取 $\tau=17$，其余时刻取 $\tau=30$，其结果如图 6 所示。阴性选择算子中阈值参数取 $M=10$。

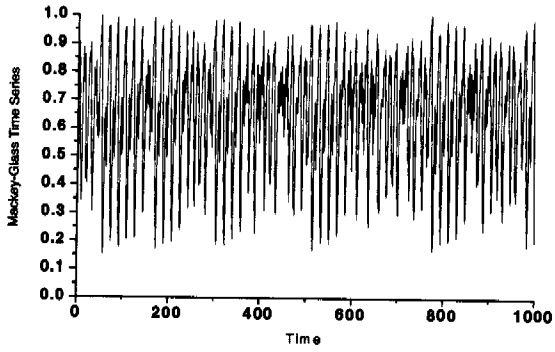

图 4　$\tau=30$ 时的正常样本

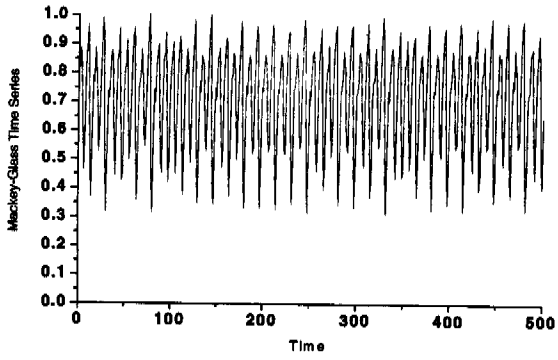

图 5　$\tau=17$ 时的异常样本

图 7 显示了本文提出的算法检测上述测试集的结果，该结果经过了窗口大小为 10 的滑动平均过程的处理。由于实验中设置的部分匹配阈值为 10，大于它的样本值被检测为异常样本，从图上可以清楚地发现在 300～400 这一时间段内检测出了异常的样本。

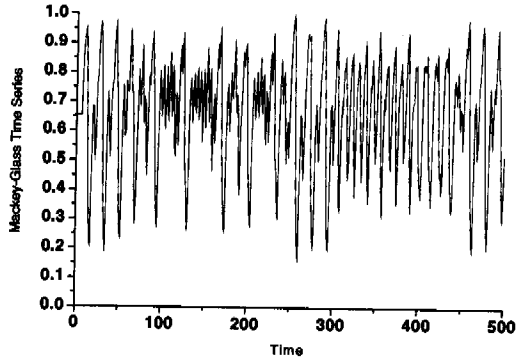

图 6　测试样本在 300～400 时刻，$\tau=17$

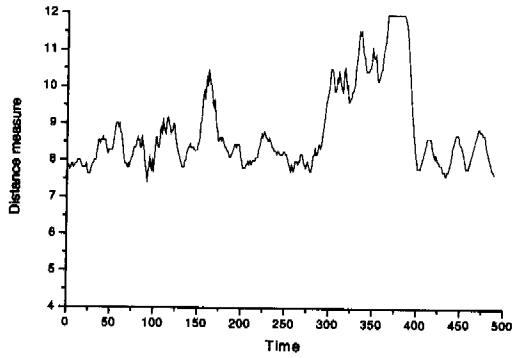

图 7　窗口大小为 10 的平滑后的结果

3.2　乳腺癌数据集

这个数据集共有 569 个样本，其中正例有 357 个、反例有 212 个。每个样本有 32 个属性值，第一个为样本的标识号，第二个为类标识，其余为特征属性。阴性选择算子的阈值参数 M 对于检测的精度影响较大，因此在不同的 M 值下分别完成了实验。

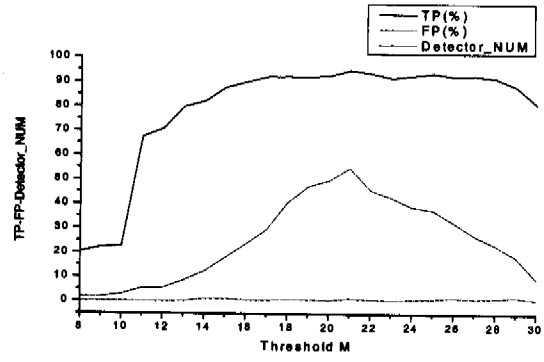

图 8　不同阈值下的 TP 值、FP 值以及检测子数目

图 8 显示了应用上述算法在不同的阈值条件下 TP 值、FP 值、检测子数目。从图上可以看出，从整体来说，FP 的值是比较低的，最大的 FP 值也不大于 2%；阈值 M 在区间[14,28]上变动时，对应的 TP 值都不小于 90.0%，特别是当 M 取 21 时，TP 值取最大值 94.9%；除此之外，检测子的数目也影响了 TP、FP 的值。总的说来，TP 值随着检测子数目的

(下转第 180 页)

3 Wei Qin, Rajagopalan S, Malik S. A Formal Concurrency Model Based Architecture Description Language for Synthesis of Software Development Tools. In: LCTES'04, 2004
4 Rigo S, Araujo G, Bartholomeu M, et al. ArchC: A SystemC-Based Architecture Description Language. In: Proceedings of the 16th Symposium on Computer Architecture and High Performance Computing (SBAC-PAD'04), 2004
5 Freericks M. The nML machine description formalism. Fachbereich Informatik, TU Berlin, 1991
6 Hadjiyiaanis G, Hanono S, Devadas S. ISDL: An instruction set description language for retargetability. In: Proc. of 34th DAC, 1997
7 Gyllenhaal J C. A machine description language for compilation: [Master's thesis]. Dept. of ECE, UIUC, 1994
8 http://www.RCS.virginia.edu/zephyr, The Zephyr compiler infrastructure
9 Halambi A, Grun P, Ganesh V, et al. EXPRESSION: A language for architecture exploration through compiler/simulator retargetability. In: Proc. of DATE, 1999
10 Pees S, Hoffmann A, zivojnovic V, et al. LISA: Machine description language for cycle-accurate models of programmable DSP architectures. In: Proc. of 35th DAC, 1999
11 Marwedel R, Goossens G, et al. Code Generation for Embedded Processors, Kluwer Academic Publishers, 1995
12 Leupers R. Retargetable generator of code selector from HDL processor models. In: European Design and Test Conference (ED&TC), 1997
13 Hanono S Z. Aviv: A Retargetable Code Generator for Embedded Processors: [Ph. D thesis]. Department of Electrical Engineering and Computer Science, MIT, 1999
14 Chang P P, Mahlke S A, Chen W Y, et al. IMPACT: An Architectural Framework for Multiple-Instruction-Issue Processors. In: Proceedings of the 18th Annual Int'l Symposium on Computer Architecture, Toronto, 1991. 266~275
15 Sudarsanam A. Code Optimization Libraries for Retargetable Compilation for Embedded Digital Signal Processors: [PhD Thesis]. Princeton University Department of EE, 1998
16 Mishra P, Mamidipaka M, Dutt N. Processor-Memory Coexploration Using an Architecture Description Language. ACM Transactions on Embedded Computing Systems, 2004, 3(1): 140~162
17 Kejariwal A, Mishra P, Astrom J, et al. HDLGen: Architecture Description Language driven HDL Generation for Pipelined Processors: [Technical Report #03-04]. Center for Embedded Computer Systems, University of California, Irvine, CA, USA, February, 2003
18 Mishra P, Kejariwal A, Dutt N. Rapid Exploration of Pipelined Processors through Automatic Generation of Synthesizable RTL Models. Rapid System Prototyping, San Diego, 2003
19 http://www.cs.virginia.edu/nci/, The National Compiler Infrastructure Project
20 http://www.eecs.harvard.edu/machsuif/index.html, Machine SUIF
21 Morimoto T, Yamazaki K, Nakamura H, et al. Superscalar processor design with hardware description language AIDL. In: Proc. of APCHDL, 1994
22 http://gcc.gnu.org/index.html

(上接第 169 页)

增加而增加。图 9 使用 ROC 曲线来反映检测的精度，从图上可以看出在不同的 *TP* 值下，*FP* 始终处于一个很低的值，这就证明了使用阴性选择算子充当过滤器能够很好地抑制 *FP* 值的增加。

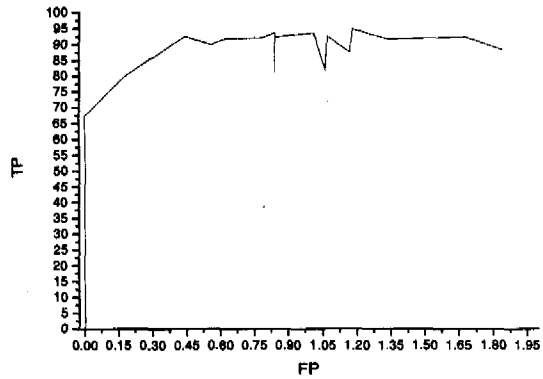

图 9 反映检测性能的 ROC 曲线

结论和展望 本文吸收了生物免疫学中免疫识别的灵感，组合了阴性选择算子和遗传算法来实现性能异常检测。在算法中多次使用阴性选择算子充当过滤器，尤其是把它添加到遗传算法中，有效地抑制了 FP 值的增加；采用了新的部分匹配规则来度量两个个体之间的距离；使用模糊逻辑产生模糊集，更好地反映正常和异常的界线，最后使用两组数据进行了仿真实验。实验的结果表明，算法具有较高的检测精度。本文的工作只是实现了性能异常的检测，并不能对于异常的程度给予量化并且确定软件衰退的原因。未来的工作将研究这些问题，为制定性能恢复策略提供依据。

参 考 文 献

1 Garg S, Moorsel A V. A Methodology for Detection and Estimation of Software Aging[A]. In: Proc. of 9th Intnl Symposium on Software Reliability Engineering[C], Paderborn, Germany, Nov. 1998. 282 ~292
2 Hansen J P, Siewiorek D P. Models for time coalescence in event logs[A]. In: Proc. of 22nd IEEE Intl Symposium on Fault-Tolerant Computing[C], 1992. 221~227
3 Iyer R K, Young L T. Automatic recognition of intermittent failures: An experimental study of field data[J]. IEEE Transactions on Computers, 1990, 39(4): 525~537
4 Ye Nong, Chen Qiang. An Anomaly Detection Technique Based on A Chi-square Statistic for Detecting Intrusions into Information Systems[J]. *Quality and Reliability Engineering International*, 2001, 17(2): 105 ~ 112
5 Lane T, Brodley C E. An Application of Machine Learning to Anomaly Detection[A]. In: Proceedings of the 20th National Information Systems Security Conference[C], Baltimore, MD. Oct. 1997. 366~377
6 Forrest S. Self-nonself discrimination in a computer[A]. In: Proceedings of IEEE Symp on Research in Security and Privacy[C], 1994. 202~212
7 McCoy D F, Devarajan V. Artificial immune systems and aerial image segmentation[A]. In: Proceedings of IEEE International Conference on Systems, Man, and Cybernetics[C]. Orlando, Florida, 1997. 867~872
8 Kim J, Bentley P. Towards an Artificial Immune System for Network Intrusion Detection: An Investigation of Clonal Selection with a Negative Selection Operator[A]. In: Proceedings of on Evolutionary Computation (CEC-2001)[C], Seoul, Korea, 2001. 1244~1252
9 Kim J, Bentley P. An evaluation of negative selection in an artificial immune system for network intrusion detection[A]. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)[C], San Francisco, California, USA, 2001. 1330 ~1337
10 Caudell T, Newman D. An adaptive resonance architecture to define normality and detect novelties in time series and databases [A]. In: Proceedings of(Portland, Oregon)[C], 1993. 166 ~ 176
11 Murphy P M, Aha D W. UCI Repository of machine learning databases, 1992
12 孙瑞祥. 进化计算与智能诊断[D]. 西安交通大学, 2000