计算机科学2005Vol. 32№. 3

多 Agent 系统中信任的动态性处理

王 平 张自力
(西南师范大学计算机学院 重庆400715)

摘 要 信任是多 Agent 系统中进行决策和交互的重要内容。收集必要的信息确定信任关系,动态地管理、维护信任关系,以及监控和重估已有的信任关系是多 Agent 系统中信任管理的关键问题。虽然研究者对上述关键问题提出了一系列解决方案,但依然存在一些问题有待进一步解决。本文针对信任的动态性处理这一问题,在分析现有的典型信任模型基础之上,提出一个具有动态性的 Confidence-Reputation 信任模型。模型中,我们不仅考虑了 Agent 的直接交互历史(Confidence)和信誉(Reputation),同时也考虑了信任的本体性。

关键词 多 Agent 系统,信心,信誉,信任管理

Dynamic Trust Processing in Multi-Agent Systems

WANG Ping ZHANG Zi-Li
(School of Computer Science, Southwest China Normal University, Chongqing 400715)

Abstract Trust is one of the most important concepts guiding decision making and contracting in multi-agent systems. Collecting necessary information to make a trust relationship decision, continuously managing and maintaining rust relationships overtime as well as monitoring and re-evaluating existing trust relationships are the key issues for trust management in multi-agent systems. While previous work provides some solutions to these key issues, there are still some problems remained unsolved especially the trust dynamic processing problem. In this paper we propose a confidence-reputation model with trust network for trust dynamic processing so that agents can autonomously deal with deception and identify trustworthy parties in multi-agent systems. When evaluating the trustworthiness of the target agent, our approach not only considers the confidence information (based on direct prior interactions with the target agent) and the reputation information from trust network, but also considers the trust's ontological property.

Keywords Multi-agent systems, Confidence, Reputation, Trust management

1 引言

智能 Agent 是处在某个环境中的计算机系统,该系统有能力在这个环境中自主行动以实现其设计目标[1]。多 Agent 系统(MAS)即由多个 Agent 构成的系统,可泛指所有由多个自治或半自治模块组成的系统。一般情况下,多 Agent 系统中的 Agent 往往代表具有不同目标和动机的用户进行工作,Agent 需要其他 Agent 的合作以实现其目标。然而在实际交互中,由于能力的缺乏或环境的变化,Agent 并不能保证完全履行其承诺。在这种不确定的环境中,信任能促进合作,减低合作中的风险和开销[2]。

信任是一复杂的概念,它涉及到对事物的诚信,真理,能力,依赖性等诸多方面的信念。本文中信任被定义为信任者和被信任者之间的关系,信任使得被信任者能够使用或操作信任者所拥有的资源或者影响信任者是否使用被信任者提供的服务[3]。

信任管理这一概念最早出现在文[4]中。由于受网络安全研究的影响,Blaze 等针对如何管理公钥授权问题,给出了信任管理的定义,然而该定义忽略了:(1)信任关系的分析;(2)决策过程中,冒险、经验等辅助因素的使用;(3)信任的动态性:信任会随着时间而发生变化。因此本文中信任管理被认为是:收集、编码、分析和表示证据的活动,而这些证据与网络应用中进行信任关系评估决策的能力、诚信、安全和依赖信相关。这些证据可以是证书识别号、质量信任状、冒险假设,或是使用的经验以及推荐信息等。综上所述,信任管理就是收集必要的信息确定信任关系,动态地管理、维护信任关系,以及监控和重估已有的信任关系。

针对多 Agent 系统中的信任管理,研究者们作了大量的研究:Amdreas Birk 采用遗传算法中的学习策略来进化 Agent 间的合作与信任[5];Ana L. C. Bazzan 和 Rafael H. Bordinic[6]在 Agent 交互中引入道德规范来强化 Agent 性能;Giorgos Zacharia 和 Pattie Maes[7]将社会网分析融入信誉模型;Miquel Montaner[8]等以用户的相似度为基础提出观点过滤的方法;S Braynov 等[9]采用鼓励和睦相处的机制,主体在交互之初便公开其诚信信息。虽然研究者们对信任管理中的关键问题提出了一系列解决方案,但依然有些问题有待进一步解决。现有的多 Agent 系统的信任管理模型或机制中,普遍存在的现象是在交互之初就定义了一种静态形式的信任。文[5]中采取随机选择标签的方法来建立信任,而选择的标签在 Agent 交互之初便已经定义。然而信任具有动态性,可随时间的变化而变化。而且 Agent 的交互环境完全开放,信任是在系统错综复杂的交互环境中形成的,不能事先定义。Ana L. C. Bazzan 和 Rafael H. Bordinic[6]在 Agent 交互中引入道德规范来强化 Agent,但是使用道德规范来强化 Agent 受到群体规模的影响。此外,在信任的决策过程中,即使在信任者和被信任者之间已存在一定的关系,然而被信任者有可能不能完全履行先前达成的合同。因此,在信任的决策过程中有进行协商的必要,文[8]并没有考虑信任决策中的协商问题。Giorgos Zacharia 和 Pattie Maes[7]虽然将 Agent 的社会性引入信誉模型中,但并没有考虑恶意 Agent 的欺诈。由于在电子商务交易中,参与交互的每一方都想获得更多的利益,因此鼓励和睦相处的机制并不实用[9]。

王 平 硕士研究生,主要研究方向:人工智能、多 Agent 系统。张自力 教授,硕士生导师,主要研究方向:人工智能、多 Agent 系统。

本文针对信任管理中信任的动态性处理这一问题，提出一个具有动态性的 Confidence-Reputation 信任模型。该模型不仅考虑了 Agent 的直接交互历史(Confidence)，同时考虑信任的社会性(Reputation)以及信任的本体性，更好地解决了信任的动态性问题。

在对 Agent、信任及信任管理的基本概念和重要性以及现有的典型信任模型/机制有了基本了解后，在第2节我们将针对多 Agent 系统中信任的动态性处理提出一个具有动态性的 Confidence-Reputation 信任模型，文章的最后是本文的总结。

2 Confidence-Reputation 信任模型

为了更好地理解多 Agent 系统中的信任特性，首先来了解囚徒两难问题[10]。人们经常使用囚徒两难问题来研究 Agent 社会的合作性问题。标准的囚犯两难问题是两个人被共同起诉一项罪名，被关押在隔离的牢房里，他们彼此之间不能相互通信，也没办法达成一致，这两人被告知：(1)如果其中一人承认有罪而另一人没有承认，承认者将被释放，另一个人将被关押3年；(2)如果两人都承认有罪，则每人被关押2年；两个囚犯都否认有罪，则每人被关押1年。当该场景反复进行时，我们称之为迭代囚徒两难问题 IPD，这时相互欺骗不再是唯一的解决方法，但这时会碰到理性选择的问题。这种思想现已应用于多 Agent 系统中信任管理的研究[11]。一方面，交互双方彼此信任可以促进相互合作从而避免相互的欺诈；此外只有在参与者彼此反复交互的情况下才能建立信任。社会奖惩的影响远大于个人奖惩，信任机制有利于理性合作的维护。

电子商务交易过程中，正确估价交易双方的信任值是交易成功的关键。购买者在某一时间购买某一产品，交易中需要考虑的因素有价格、质量以及交付的日期等。买方选择卖方，如果卖方愿意同买方合作，就向买方发出提议，而后买方考虑是否接受卖方的提议，如果买方接受卖方的提议，就执行交易。然而值得注意的是，实际的交易结果并不一定和卖方最初的提议相同：一个带有诈骗意图的卖者可能提高价格，降低质量或者推迟交付日期；另一方面购买者可能不交付足够的货款甚至不交付货款。因此在交易过程中买方和卖方不得不在知道对方的行为之前估价对方的信任度，以决定其正确的策略。

下面我们就以电子商务交易过程中信任的建立为例来阐述 Confidence-Reputation 信任模型。

2.1 信任的基本特征

信任具有以下特征：

1、信任是基于某特定内容的信任；

2、信任具有非传递性：如果 A 信任 B，B 信任 C，A 并不一定信任 C；然而 A 对 C 的信任值将高度地依赖于 B 对 C 的信任值；

3、信任受前驱交互历史的影响：Agent 能够识别与同一 Agent 的反复交互历史；

4、在信任的决策过程中，Agent 能够彼此交互信誉信息；

5、信任并不是单一抽象的概念，而是一个组合概念。

2.2 信任值的计算

在 Agent 社会中，用于估价目标 Agent 的信任值有两种方法：一是利用与目标 Agent 过去的直接交互信息(Confidence)；二是利用其他 Agent 对目标 Agent 的信任信息(Reputation)[12,13]。大量文献中的信任模型例如 Birk[5]，只采用了直接交互的机制。而基于社会关系网的研究表明人与人之间的联系是信息收集与分发的重要途径(Katz&Lazarsfeld1995)，Reputation 在进行有效的信任决策中具有重要作用。因此，我们提出的 Confidence-Reputation 模型不仅考虑了 Agent 的直接交互历史(Confidence)，同时也考虑了信任的社会性(Reputation)及信任的本体性[14,15]。

2.2.1 Confidence 信心(Confidence)是通过分析与目标 Agent 的前期直接交互历史而建立的信任值。模型中以元组 $o=(a,b,I,X^c,X,t)$ 表示 Agent 间的一次直接交互结果：其中 a,b 为参与交互的两个 Agent，I 为协议议题集，X^c 和 X 分别表示议题集在协议中的取值以及在交易完成时的实际取值，t 为协议达成的时间。在电子商务交易中，我们假设交易双方的议题集为 $I=\{Price, Deliver\text{-}date, Service\}$。

ODB 定义为 Agent 社会的所有交互结果，则 Agent a,b 间的直接交互结果表示为 $ODB^{a,b}\in ODB$。$ODB^{a,b}_{\{i_1,\cdots,i_n\}}\subseteq ODB^{a,b}$ 表示协议中具有议题集$\{i_1,\cdots,i_n\}$的交互结果集，例如，$ODB^{a,b}_{\{price\}}$ 表示 Agent a 与 Agent b 间有关议题 Price 的交互结果。根据 Agent 的交互结果 ODB 计算 Agent 的信心时，模型中引入了信任类型 φ 及 grounding relation(gr)。卖方 Agent 的信任类型及其相应的 grounding relation 如下所示。

φ	$gr(\varphi)$
To-overcharge	{*Price*}
To-deliver-late	{*Delivery-date*}
Service -swindler	{*Service*}

我们使用权值的方法来计算两 Agent 直接交互后的 Confidence，其公式如下：

$$C_{a\xrightarrow{\varphi}b}(\varphi)=\sum_{O_i\in ODB^{a,b}_{gr(\varphi)}}\rho(t,t_i)\cdot imp(o_i,gr(\varphi)) \tag{1}$$

其中，$p(t,t_i)=\dfrac{f(t_i,t)}{\sum O_i\in ODB^{a,b}_{gr(\varphi)}f(t_j,t)}$，$t$ 为实际交互时间，$f(t_i,t)$为时间函数，此类时间函数的一个简单例子如：$f(t_i,t)=t_i/t$。

Im $p(O_i,gr(\varphi))$为交易结果 o_i 的估价函数，与信任类型 φ 及协议议题集相关：

$$\text{Im } p(o,gr(\varphi))=g(v(x')-v(x^c)) \tag{2}$$

估价函数中，$v(x^c)$为向量效用函数[*Jennings*]，x' 的取值如下：

$$x'_i=\begin{cases}x_i & \text{if } i\in gr(\varphi)\\ x^c_i & \text{otherwise}\end{cases} \tag{3}$$

最后，使用函数 $g(x)=\sin(\pi/2 * x)$来建模 Agent 直接交互后的 Confidence。

2.2.2 Reputation 当与目标 Agent 没有交互历史时，Agent 将向其熟人(即与当前 Agent 有直接交互历史的 Agent)询问有关目标 Agent 的情况。如果某一 Agent 收到请求，该 Agent 将首先确认目标 Agent 是否为其有直接交互历史的熟人：如果是其熟人，该 Agent 将返回其对目标 Agent 的 Confidence；如果不是的话，该 Agent 将基于其交互历史返回推荐 Agent。然后，当前 Agent 询问被推荐的 Agent 关于目标 Agent 的情况，依此形成推荐链，建立信任网来计算目标 Agent 的信誉[16,17]。信任网的相关定义如下：

定义1 $r\langle A_i,A_j\rangle$表示 Agent A_i对 Agent A_j 的推荐。

则从 Agent A_r 到目标 Agent A_g 间的一条推荐链表示为 $\langle A_r,\cdots,A_i,A_{i+1},A_g\rangle$，其中 A_{i+1}与 A_i 之间为熟人关系。

定义2 信任网 Trust Net $TN(A_r;A_g;A;R)$ 是一有向图，其中 A 为有限 Agent 集$(A_1,\cdots,A_n)$，R w 为推荐集$\{r_1,r_2,\cdots,r_m\}$。

假设有一推荐集$\{r_1,r_2,\cdots,r_n\}$，估价 Agent A_r 合并满足条件的每一条推荐 $r_i=\langle A_i,A_j\rangle$ 生成信任网。算法1描述了信任网的生成过程。根据 small-world work[18]，模型中 Agent 的熟人数定义为4，每一推荐链的深度不大于6，即 depth $(A_i)<=$ depth Limit $=6$。

Algorthml 信任网的生成

1: Suppose agent Ar is the requesting agent, R is a series of referrals and A is a finite set of agents being visited. Ar first sends a query to some of its neighbors. For any referral $r=\langle A_i,A_j\rangle$, agent Ar adds r to the trust network TN
2: If(depth$(A_j)<=$depth Limit)then
3: if (A_j not in A)AND(A_j returns a confidence to A_g)then
4: add A_j to the set of testimonies
5: record the confidence form A_j
6: else if A_j not in A then
7: Append r to the trust network
8: send a request to A_j
9: else
10: Ignore the referral r
11: end if
12: end if

图1给出了 Reputation 通过信任网聚集的示例。图中 Agent A 想获得 Agent E 的 Reputation。其中$\langle A,B,E\rangle$和$\langle A,C,D,E\rangle$为从估价 Agent A 到目标 Agent 的两条推荐链。

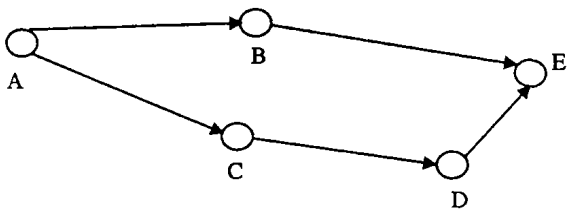

图1 信任网的生成示例

信任网建立后，计算目标 Agent 的 Reputation 需要熟人 Agent 的证词。假设 $\chi=\langle a,\cdots,w_i,w_{i+1},\cdots,w_n,b\rangle$为从 Agent a 到 b 的一条推荐链，在推荐链中相邻的两 Agent 间有直接的交互历史，则从 a 到 b 与 χ 相关的一条证据为：

$$E(a,\chi,b)=c_{a\xrightarrow{0}w_1}(\varphi)\otimes C_{w_1\xrightarrow{0}w_2}(\varphi)\otimes\cdots\cdots\otimes C_{w_n\xrightarrow{0}b}(\varphi) \tag{4}$$

其中，$\otimes$为信任传播符，其定义如下：

定义3 $x\otimes y=if(x\geqslant 0\wedge y\geqslant 0)$ then $x\times y$ else $-|x\times y|$

假设 $\varepsilon=\{E_1,E_2,\cdots,E_L\}$为 a 关于目标 Agent b 的可靠的证据集(即 $E>0$)，则目标 Agent b 的 Reputation 为：

$$R_{a\xrightarrow{0}b}(\varphi)=1/L\sum_{|i-1|}^{i-l}E(a,\chi_i,b) \text{ where } |\varepsilon=\{E_1,E_2,\cdots,E_L\}|=L \tag{5}$$

2.2.3 Trust= Confidence +Reputation Confidence-Reputation 信任模型中，Agent 的信任由两部分组成：Confidence 和 Reputation。假设 a,b 为参与交易的两个 Agent，其中 a 为估价 Agent，b 为目标 Agent，则 Agent a 对 Agent b 的信任为：

$$T(a,b,\varphi)=k\cdot c_{a\xrightarrow{0}b}(\varphi)+(1-k)\cdot R_{a\xrightarrow{0}b}(\varphi) \tag{6}$$

(使用系数 k 和$1-k$表示 Confidence 和 Reputation 在信任值的计算中的重要性，这里 $k>1-k$。)

前述中的 Confidence 和 Reputation 只是基于议题集中某一单方面的信任，我们可以将这些单方面的信任组合得到更复杂的信任。例如，在电子商务买卖交易中，根据协议议题集 I={Price,Deliver-date,Service}，一个好的卖家的信任与价格、分发的时间以及服务的质量有关。因此，将卖家的 To-deliver-late 信任，To-overcharge 信任 和 Service -swindler 信任组合得到对卖家的 good-seller 信任，如图2所示。

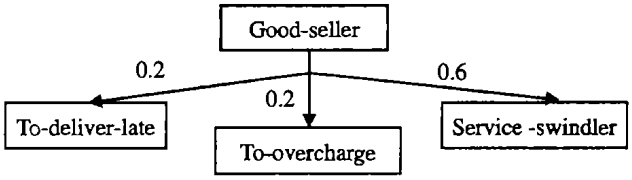

图2 本体结构图

综上所述，我们给出如下组合信任的计算方式。设 $I=\{i_1,i_2,\cdots,i_n\}$ 表示协议中涉及到的 n 个议题，则 Agent a 对 b 的组合信任为：

$T(a,b,I)=g(T(a,b,i_1),\cdots,T(a,b,i_n))$，其中 g 是 suitable aggregation function 适应聚集函数，$g:[0.1]^k\rightarrow[0,1]$。即

$$T(a,b,I)=\sum\tilde{\omega}_j\cdot T(a,b,i_j) \text{ where } i_j\in I$$
$$\text{and } \sum\tilde{\omega}_j=1,0\leqslant\tilde{\omega}_j\leqslant 1 \tag{7}$$

结论 从近年大量的研究项目可以看出信任管理已经逐渐成为多 Agent 系统的一个重要研究方向。本文在总结现有的典型信任模型的基础上，讨论了多 Agent 系统中信任管理的关键问题。虽然针对这些关键问题提出了很多好的解决方法，但依然有些问题尚未解决或有待进一步解决，尤其是信任的动态性处理问题。本文在分析现有的典型信任模型基础之上，提出一个具有动态性的 Confidence-Reputation 信任模型。该模型不但考虑了 Agent 的直接交互历史，同时考虑了 Agent 社会的 Reputation 以及信任的本体性。在未来的工作中，我们将以 Agent builder 为开发平台来模拟实现这一模型。

参 考 文 献

1 Wooldridge M. An Introduction to Multiagent Systems. Wiley, Chichester, England, 2002
2 Ramchurn S D, Jennings N R. A Computational Trust Model for Multi-agent Interaction based on Confidence and Reputation. In: Proc. of 2nd Intl. Joint Conf. on Autonomous Agents and Multi-Agent Systems, 2003
3 Gambetta D. Can We Trust Trust? In, Trust: Making and Breaking Cooperative Relations. In: Gambetta D, ed. Basil Blackwell. Oxford, 1990
4 Blaze M, Feigenbaum J, Lacy J. Decentralized Trust Management. In: IEEE Conf. on Security and Privacy, Oakland California, USA, 1996
5 Birk A. Boosting Cooperation by Evolving Trust. Applied Artificial Intelligence, 2000, 14(8)
6 Bazzan A L C, Bordini R H. Evolving agents with moral sentiments in an iterated prisoner's dilemma exercise. In: Second workshop on game theoretic and decision theoretic agents. Proc. of the workshop on Game Theoretic and Decision Theoretic Agents, Boston, 2000
7 Zacharia G, Maes P. Trust Management Through Reputation Mechanisms. Artificial Intelligence, 2000, 14: 881～907
8 Montaner M, López B, de la Rosa J Ll. Opinion-based Filtering through Trust. Lecture Notes in Computer Science (Artificial Intelligence) 2002, 2446: 164～178
9 Braynov S, Holm T S. Trust revelation in multi-agent interaction. In: Proc. of CHI'02 Workshop on The Philosophy and Design of Socially Adept Technologies, Minneapolis, 2002. 57～60
10 Axelrod R. The Evolution of Cooperation. Basic books, New York, 1984
11 Yao D. How important is your reputation in a Multi-Agent Environment. In: Proc. of the 1999 IEEE Intl. Conf. on Systems, Man, and Cyber metric, 1999. 575～580

12 Sabater J,Sierra C. Reputation and social network analysis in multi-agent systems. AAMAS,2002. 475～482
13 Barber K S,Kim J. Belief Revision Process Based on Trust: Agents Evaluating Reputation of Information Sources. Trust in Cyber-societies LNAI2246,Spring-Verlag Berlin H,2001. 73～82
14 Esfandiari B,Chandrasehharan S. On how Agents Make Friends: Mechanisms for Trust Acquisition. In:Proc. of the 4th workshop on Component-Oriented Programming,2001
15 Sen S. believing others: Pros and cons. Artificial Intelligence, 2002:142:179～203
16 Yu B,Singh M P. Detecting Deception in Reputation management. In:Proc. of 2nd Intl. Joint Conf. on Autonomous Agents and Multi-Agent Systems,2003. 73～80
17 Yu B, Singh M P. Searching Social Networks. In: Proc. of 2nd Intl. Joint Conf. on Autonomous Agents and Multi-Agent Systems,2003. 65～72
18 Watts D J,Strogatz S H. Collective dynamics of small-world networks. Nature,1998,393:440～442

(上接第161页)

5.2 测试驱动程序的构造

利用测试驱动程序框架来构造 *Tetris* 类的驱动程序的步骤如下。

步骤一：TetrisTesterDriver：public TestDriver，表示TesrisTesterDriver 从 TestDriver 继承而来。

步骤二：覆盖 TesterDriver 中的 *RunTest*()。其中 *RunTest*()的设计按照前面所提到的算法来实现(为了说明的方便本文以伪代码的形式给出)。

```
RunTest(){
tetris. block = I;
while casefile not end
    {
      for(casepoint from testcase. first to testcase. end )
       { if( * casepoint="INIT")
                 { tetris. block->Init();
                   oracleblockinit()}
                      M
                   if( * casepoint="Rotate")
                 { tetris. Rotate();
                   oracle. rotate();}
       }
    }
}
```

说明：对于 *RunTest*()函数，对于不同的 CUT，*RunTest*()是不同的。其中的省略号部分表示的是测试用例中每个方法的调用，及其对应方法的测试预测程序(orcacle)的执行。

步骤三：注册该被测类驱动，在 *main*()函数中得到语句：

```
#ifdef TetrisTesterDriver
      TestDriver *r1=new Tetris
      r->Runtest()
#end if
```

步骤四：将被测类，被测类的驱动类和主控程序一起编译连接生成可执行的程序，从而运行测试用例。通过调用 CL 编译器完成。其中我们将编译连接的参数设置和连接参数的设置以 compileset. txt，linkset. tx 的形式存放。

```
cl@compileset. txt
link@linkset. txt
```

即可生成我们所需要的可执行的驱动程序，利用生成的可执行程序可以驱动测试用例运行，从而完成了 *Tetris* 类的测试。运行后，我们将会得到一个存放运行结果的文件 TestResult. txt。

运行结果(即 TestResult. txt 里面的内容)如下：

```
Test case1: INIT MoveLeft MoveRight block=T
    pass
Test case2: INIT MoveLeft MoveDown MoveRight block=Square
    pass
Test case3: INIT MoveLeft MoveDown MoveRight block=S
    pass
Test case4: INIT MoveLeft MoveRight block=Rectangle
    pass
Test case5: INIT MoveLeft MoveDown MoveRight block=Rectangle
    pass
Test case6: INIT MoveLeft MoveDown AddBlock Rotate block=
    Square
    pass
total test cases: 6
PassedTests: 6 passRatio: 100%
failedTests: 0 failedRatio: 0%
```

总结 本文提出了一种面向对象类测试驱动程序的构造框架，介绍了该框架的设计。将类驱动设计成了一个独立的类，使得驱动程序有了很好的复用性；利用条件编译实现被测类的驱动类注册，使得在一个主控程序下，能够同时测试多个类，并且条件编译可以比较灵活地控制驱动程序的执行。最后通过实例说明了这种框架的可行性。利用该驱动程序构造框架，可以方便快速地生成类测试驱动程序。

本文所提出的框架还存在有待于进一步研究和改进的地方，例如对于驱动中的测试结果的预测。因为对于测试结果预测的自动化，将会进一步简化驱动程序的设计。因此希望在以后的工作中在此方面取得进展。

从目前的研究现状来看，对于类测试驱动程序的构造，现在还没有一个完全的解决方案。但对于类测试驱动的研究将对提高类测试的效率，实现类测试的自动化，以及类的回归测试都有着十分重要的意义。

参考文献

1 Doong R K,Frankl P G. The ASTOOT Approach to Testing Object-oriented Programs [J]. ACM Transactions on Software Engineering and Methodology,1994,3(2):101～130
2 Ihssan A,Carver D. A Testing Assistant for Object-oriented Programs [J]. In:1998 IEEE Aerospace Conf.
3 刘玲. 基于面向对象形式规格说明的测试用例生成技术[D]. 上海大学,2004
4 McGrgor J D,Sykes D A 著. 面向对象的软件测试[M]. 北京：机械工业出版社,2002
5 Luo Gang,Probert R L,Dral H. Approach to construction software unit testing tools [J]. Software Engineering Journal,1995. 11
6 Liu Ling,Miao Huaikou,Zhan Xuede. A Framework for Specification-Based Class Testing. In:Eighth IEEE intl. conf. on engineering of complex computer systems,Dec. 2002. 153～162
7 Doong R K,Frankl P G. The ASTOOT approach to testing object-oriented programs [J]. ACM Transactions on Software Engineering and Methodology,1994,3(2):101～130
8 张雪萍，张慧档，庄雷. 面向对象软件的类测试技术[J]. 微机发展，2002,5:74～77
9 郑春一，宋雨，孙文婧. 面向对象类测试方法分析[J]. 微机发展，2003,1:57～59
10 陈站华. 软件单元测试[J]. 软件与测试,2003,5:50～51