一种基于构件组装方式的特征干扰检测方法*)

胡金柱　费丽娟　李　敏
(华中师范大学计算机科学系　武汉430079)

摘　要　随着基于构件的软件开发技术(CBSDT)的不断发展,构件组装中的特征干扰问题逐渐受到大家的重视,针对这一问题的研究也成为一个热点。基于软件构件的特点,提出采用时序逻辑语言XYZ/E形式化构件,并根据构件组装的不同方式对特征干扰问题进行检测。
关键词　特征干扰,时序逻辑,XYZ/E语言,构件组装方式

Detecting Feature Interactions Based on Component Composition Mechanisms

HU Jin-Zhu　FEI Li-Juan　LI Min
(Department of Computer Science, Central China Normal University, Wuhan 430079)

Abstract　With the development of Component-Based Software Development Technology (CBSDT), people find it is important to resolve the feature interaction problems in component composition. Based on the characteristic of software component, in this paper, the temporal logic language XYZ/E is used to formalize components, and a method based on component composition mechanisms is proposed to detect feature interactions.
Keywords　Feature interaction problem, Temporal logic, XYZ/E, Component composition mechanisms

随着基于构件的软件开发技术的不断发展,构件可能来自于不同厂商,或者可能属于不同的应用系统,因此其组装后的行为变得难以预测。构件作为一个单独的模块使用时很简单,功能很明确,但当多个构件组装形成复合构件或是构件加入到某个应用系统时,就会引发很多问题,人们将这种现象称为"特征干扰"。其中那些干扰或破坏构件的正常操作、影响系统正常工作的特征干扰被称为"特征干扰问题",这种造成不良影响的特征干扰是必须被检测出来并予以解决的[1]。

"特征干扰"问题最早在电信领域被提出,并展开了一系列的研究。在研究基于构件的软件开发过程中的特征干扰问题时,可以借鉴电信领域中特征干扰问题的检测、解决方法。但是需要注意的是,由于构件组装方式有许多,不同的组装方式会产生不同的特征干扰。所以在本文,我们提出了一种基于构件组装方式的特征干扰检测方法,并给出了使用XYZ/E语言的形式化描述。

1　相关工作

目前,对基于构件的软件领域的特征干扰问题的研究已经有不少进展。例如,Kerth P. Pomakis 和 Joanne M. Atlee 提出一种思路,采用服务和特征的堆栈结构,通过区分先后次序的方法来解决不确定性问题,从而检测在数据库系统、操作系统、控制系统等领域中的特征干扰[2]。M. Heisel 和 J. Souquieres 提出将需求表示为公式(约束),如果约束具有相交的前置条件,同时具有矛盾的后序条件则出现特征干扰,这个检测方法可以用于解决需求阶段的特征干扰[3]。

但是,这些研究或者是基于某一特定领域,或者是针对某一特定类型的特征干扰。由于构件具有不同的抽象层次、不同的粒度以及不同的组装方式,决定了它们之间的特征干扰现象的多样性,不能简单地用一种标准检测是否产生了特征干扰。所以本文提出了一种基于组装方式的特征干扰检测方法。

构件的组装方式是多种多样的,例如并行组装、选择组装、顺序组装等无连接件的组装,过程调用、消息传递等有连接件的组装[4]。构件的组装方式不同,其产生特征干扰的原因和现象也各不相同。

例如,在升降机系统中假设构件A的功能是:升降机停在某层且门的当前状态为开,如果在时间 t 内没接受到该层的任何外部命令,则门自动关闭。用XYZ/E描述如下:$HALT(i) \wedge DOOR\_OPEN \wedge \sim press(i) \Rightarrow \Diamond close$ 构件B的功能是:升降机停在某层且门开,如果升降机超载,则不能关门。$HALT(i) \wedge DOOR\_OPEN \wedge OVERLOARDED \Rightarrow \$Ooren$ 两个构件要并行组装,由于 $HALT(i) \wedge DOOR\_OPEN \wedge \sim press(i) \wedge OVERLOARDED == \$T; close \wedge open == \$F$ 可知,它们的前置条件能同时满足,但由于后序条件矛盾,产生了特征干扰。

又比如,有音乐播放器构件MusicPlayer和VCD播放器构件VcdPlayer,现在要将它们组装成一个媒体播放器,既能播放音乐,又能播放VCD,如果这两个构件采用选择组装方式,则当两个构件有能够共同播放的文件,例如CD,就会产生特征干扰。也即如果两个构件以选择方式组装,则它们的前置条件不能同时满足。

由此可见,采用不同组装方式构件,其产生特征干扰的方式是不同的,也就有必要对不同组装方式下的特征干扰分别进行研究。

2　构件描述

时序逻辑语言XYZ/E基于一阶线性时序逻辑,既是一个时序逻辑系统,又是一种程序语言,能在统一框架下,既描述软件系统静态语义,又描述软件系统动态语义[5,6]。所以我们采用XYZ/E语言作为构件描述语言。构件描述由以下几部分组成:

(1)*数据结构*　具体描述如下:

$\%\mathrm{Type}[Struct\_Name == Record(Field\_List)]$

*)本项目研究得到湖北省科技攻关重大项目(2003AA101C26)支持。胡金柱　教授,主要研究领域为数据库与软件工程。费丽娟　硕士研究生,主要研究领域为基于构件的软件工程和数据库。李　敏　硕士研究生,主要研究领域为数据库与软件体系结构。

其中 *Struct_Name* 表示数据结构的名称，结构定义与高级语言中的 Record 相似。但是，由于 XYZ/E 语言数据类型是基于 PASCAL 的，它所提供的数据类型是面向软件编码和实现的，而我们在对特征干扰问题进行检测时要求有较高的抽象，把握构件的本质，简明地表达数据和其间的复杂关系。所以我们需要对 XYZ/E 语言的数据结构进行扩充，将幂集类型、笛卡儿积类型、序列、包、关系、函数和模式都引入作为数据类型。同时允许用户运用方括号定义基本数据类型，而不关心其内部结构[7]。

(2)过程计算　描述如下：

%Proc Proc_Name(*Paameters*)==
[*Satements*]WHERE(*Constraint*)

其中 Proc_Name 表示过程/进程名(对于函数，还需要定义返回直的类型)，Statements 为实现该过程的语句序列，在此表示 Elements(条件元)序列，Elements 的定义为：

Elements==[*LB*=*definitionallabel* ∧ *condition*⇒ $*O*(变量序列)=(值序列) $*OLB*=*forwardlabel*]　或

Elements==[*LB*=*definitionallabel* ∧ *condition*⇒◇(一阶公式 ∧ *LB*=*forward label*)]

(3)包块

%*Pack Pack_Name*==[*Data Structure*;$Comp_1$ $ \vee \wedge $ $ \vee Comp_k$]WHERE(*Constraint*)

其中 *Pack_Name* 表示包块名，此处"$*V*⋯$*V*"可以写成"*V*⋯*V*"。

(4)选择结构

$LB = y \wedge R \Rightarrow !![Cond_1 \mid > ExeAct_1, \cdots, Cond_k \mid > ExeAct_k]$

或

$LB = y \wedge R \Rightarrow \$O(Cond_1 \wedge ExeAct_1 \$V' \cdots \$V' Cond_k \wedge ExeAct_k)$

其含义是：在前置条件 *R* 满足的情况下，当外界环境中出现事件 *Cond*$_i$ 时，执行动作 *ExeAct*$_i$[6]。

运用 XYZ/E 语言提供的程序构造单元，可以分别建模不同属性的软件构件；过程对被动执行的具有单个完整功能的软件构件进行建模；进程对能够主动执行和并发执行的软件构件进行建模；包块对围绕一数据结构封装一组运算而成的软件构件进行建模；代理机构由一数据包块和一与之相匹配的进程部分组成，实现静态语义和动态语义的联接和组合，对能够主动执行和通信的包块构件进行建模[7]。

一个程序的功能和性质通常可通过程序的初始状态和终止状态之间的关系来表示，程序的初始状态和终止状态所满足的一阶公式分别称为程序的前置条件(pre-condition)和后序条件(post-condition)。当我们用前置条件和后序条件的序对来描述一个程序的基本功能时，这个前置条件和后序条件的序对便被称为程序的规范[5]。我们利用这种规范对构件组装中的行为进行检测，从而发现其中的特征干扰问题。

3　特征干扰的检测

下面我们考虑几种组装情况下的特征干扰及其检测方法。假设构件 *A* 的前置条件为 P_1，后序条件为 Q_1；构件 *B* 的前置条件为 P_2，后序条件为 Q_2。

(1)构件 *A* 和构件 *B* 同时执行　则

$LB=y \wedge P_1 \wedge P_2 \Rightarrow \$OLB=w \wedge Q_1 \wedge Q_2$

此时，如果 *A* 和 *B* 的前置条件相矛盾，或者在前置条件不矛盾的情况下出现了矛盾的后序条件，则构件 *A* 和 *B* 不能同时执行，出现了特征干扰。在 XYZ/E 中表现为如下形式：

$(P_1 \wedge P_2 == \$F) \vee ((P_1 \wedge P_2 == \$T) \wedge (Q_1 \wedge Q_2 == \$F))$　(1)

(2)构件 A 和构件 B 必须且只能执行一个　则

$LB=y \Rightarrow \$OLB=k$;
$LB=k \wedge P_1 \Rightarrow LB=w \wedge Q_1$;
$LB=k \wedge P_2 \Rightarrow LB=w \wedge Q_2$。

此时，如果构件 *A* 和 *B* 能同时满足或者同时不满足初始条件，则构件 *A*、*B* 可能同时执行或都不执行，与用户的期望相矛盾，出现了特征干扰现象。

$(P_1 \wedge P_2 == \$T) \vee (P_1 \vee P_2 == \$F)$　(2)

(3)构件 *A* 先执行，然后构件 *B* 执行

$LB=y \wedge P_1 \Rightarrow \$OLB=k \wedge Q_1$;
$LB=k \wedge P_2 \Rightarrow \$OLB=w \wedge Q_2$;

特征干扰可能出现在以下两种情况：①构件 *B* 的前置状态包含构件 *A* 执行完后的状态，此时构件 *B* 有部分功能不能实现；②构件 *A* 执行完后的状态包含构件 *B* 的前置状态，此时构件 *A* 执行完后有可能不满足 *B* 的状态，而使构件 *B* 不能执行。也即

$((Q_1 \rightarrow P_2) \wedge \sim(P_2 \rightarrow Q_1)) \vee$
$((P_2 \rightarrow Q_1) \wedge \sim(Q_1 \rightarrow Q_2))$　(3)

(4)过程调用　构件 *A* 执行过程中调用构件 *B* 的过程，*B* 的过程执行完毕后返回 *A* 继续执行。构件 *A* 中相关语句如下：

$LB=l_1 \wedge P_1 \Rightarrow \$OLB=l_2$;
$LB=l_2 \Rightarrow B(parameter\_name) \wedge \$OLB=l_3$;
$LB=l_3 \Rightarrow Q_1 \wedge \$OLB=l_4$;

构件 *B* 的规范(静态语义)表示为：

$P_2 \Rightarrow Q_2$;

在以下几种情况下可能会产生特征干扰：①调用发生时，构件 *A* 的状态与构件 *B* 的前置状态矛盾；②构件 *B* 的前置状态包含构件 *A* 调用时的状态，或者构件 *A* 调用时的状态包含构件 *B* 的前置状态，但两者不等价；③构件 *B* 执行完毕后返回的状态与构件 *A* 的状态矛盾。也即

$(P_1 \wedge P_2 == \$F) \vee ((P_1 \rightarrow P_2) \wedge \sim(P_2 \rightarrow P_1))$
$\vee ((P_2 \rightarrow P_1) \wedge \sim(P_1 \rightarrow P_2) \vee (Q_1 \wedge Q_2 == \$F))$　(4)

(5)隐式调用　构件 *A* 执行过程中出现事件 P_1，触发了构件 *B* 的过程或进程。构件 *A* 描述如下：

$LB=l_1 \wedge P_1 \Rightarrow B \wedge \$OLB=l_2$;

构件 *B* 的规范表示为：$P_2 \Rightarrow Q_2$;

如果构件 *B* 执行后又使构件 *A* 回到触发前的状态，即又出现了事件 P_1，则又将再次触发构件 *B*。如果这种触发没有限制条件，将无限制进行下去。也即

$Q_2 \Rightarrow P_1$　(5)

4　实例

例1　在一个路由器的 IP 过滤系统中，管理员设置各种规则以指导路由器对接收到的 IP 包的处理，如：

规则1：目标端口为21的包一律丢弃；

规则2：目标端口小于255的包执行规则35；

规则3：源地址在202.112.58.0到202.112.58.24的包一律通过。

等等[8]。这些规则用形式化的方法描述如下：

基本数据结构：
%*TYPE*[
[*IP Address*]
//定义 *IP Address* 为基本数据类型，表示 *IP* 地址[*Port*]
//定义 *Port* 为基本数据类型，表示目标端口
]

(下转第137页)

结论 作为一类概率算法，消息传递算法(如 BP,SP)有其快速、高效的特点。但 BP 算法在 Hard-SAT 区域常常无法收敛。而改进后的 SP 算法则表现得相当出色。但 SP 算法在 SAT/UNSAT 相变区附近，出现不收敛，越过 SAT/UNSAT 相变区后(如 $\alpha=4.3$)，则绝大情况仍然无法收敛。也许这是由于在此区域，解的个数过少(或者无解?)，以至无法对这些解作有效的统计分析。作者采用的回溯策略，也只能是在一定程度上解决这个问题，而且这纯粹是经验性的。在 satlib 的一些难解结构化实例上(如 g125.18.cnf,g250.15.cnf 等)，SP 也没有好的表现。常常出现 0/0，作者估计这可能是公式(20,21)计算出的问题，即所有警告为 1。也可能在 SP 算法眼里，这些结构化实例是约束过强的。

参 考 文 献

1 张健.逻辑公式的可满足性判定一方法、工具及应用.北京:科学出版社,2000
2 Braunstein A,Mezard M,Weigt M,Zecchina R. Constraint satisfaction by survey propagation. http://fr.arxiv.org/abs/cond-mat/0212451,Dec. 2002
3 Braunstein A,Mezard M,Zecchina R. Survey propagation: an algorithm for satisfiability. submitted to Random structures and algorithms. http://fr.arxiv.org/abs/cs.CC/0212002,Dec. 2002
4 Mezard M,Parisi G,Zecchina R. Analytic and Algorithmic Solution of Random Satisfiability Problems. Science,2002,297,812
5 Parisi G. Some remarks on the survey decimation algorithm for K-satisfiability. http://arxiv.org/abs/cs.CC/0301015,Jan. 2003
6 Kschischang F R,Frey B J,Loeliger H A. Factor Graphs and the Sum-Product Algorithm. IEEE Trans. Infor. Theory 2002,47:498
7 Kirkpatrick S,Gelatt CD,Jr.,Vecchi MP. Optimization by simulated annealing. Science,1983(4598)
8 鄢庆增.自旋玻璃和复杂系统.物理,1992,21(7)
9 Mezard M. Spin glasses: an introduction. In:Proc. of the Cargese summer institute on. From statistical physics to statistical inference and back,Sep. 1992
10 Parisi G. A backtracking survey propagation algorithm for K-satisfiability. http://arxiv.org/abs/cond-mat/0308510 v1, Aug. 2003
11 Zhou Haijun. Vertex cover problem studied by cavity method: Analytics and population dynamics. Eur. Phys. J.,2003,B 32:265～270
12 Weiss Y. Approximate inference using loopy belief propagation. Tutorial given at UAI 2001
13 Mezard M. Constraint satisfaction networks: Belief propagation and beyond. Invited Talks at NIPS 2003 Workshop on Robust Communication Dynamics in Complex Networks
14 Mezard M. Passing Messages Between Disciplines. SCIENCE,2003,301
15 Mertens S,Mezard M,Zecchina R. Threshold values of Random K-SAT from the cavity method. http://fr.arxiv.org/abs/cs.CC/0309020,Sep. 2003
16 Survey Propagation: Constraint Satisfaction via Message Passing, Constantine Caramanis, slides of MIT course 6.454: Graduate Seminar in Area I,Fall 2003

(上接第129页)

规则1:

```
%VAR[
      aimport,port1:Port
//定义 Port 类型的变量,代表目标端口
]
%Proc Rule1(aimport)[
  LB=START-Rule1⇒$OLB=l1;
  LB=l1∧aimport==port1
    ⇒abandon()∧$OLB=EXIT;
  //调用 abandon()函数,丢弃包
  //port1是用户定义的目标端口号
  LB=l1∧~(aimport==port1)
    ⇒accept()∧$OLB=EXIT;
  //调用 accept()函数,接收包
]
```

规则2:

```
%VAR[
      aimport,port1:Port
]
%Proc Rule2(aimport)[
      LB=START-Rule2⇒$OLB=l1;
      LB=l1∧aimport<port1⇒Rule35()
          ∧$OLB=EXIT;
    //调用规则35,规则35的作用及描述省略
      //port1是用户定义的端口号
      LB=l1∧aimport>=port1⇒$OLB=EXIT;
]
```

规则3:

```
%VAR[
  sourceaddress,low,high:IP Address,
  //定义 IPAddress 类型的变量,表示源地址
]
%Proc Rule3(sourceaddress,low,high)[
  LB=START-Rule3⇒$OLB=l1;
  LB=l1∧AddBetween(low,high)
    ⇒accept()∧$OLB=EXIT;
  //AddBetween(low,high)函数的作用是判断 IP 地址
  //是否在 low 和 high 之间,这两个参数由用户设置
  LB=l1~AddBetween(low,high)
    ⇒abandon()∧$OLB=EXIT;
]
```

这三个规则要同时满足，则要用公式(1)进行判断。由于

$(aimport==port1 \wedge AddBetween(low,high)==\$T)\wedge(abandon()\wedge accept()==\$F)$

所以有可能产生特征干扰。例如：当一个包来自于202.112.58.1，目标端口为21时，路由器根据规则1将其丢弃，但这种行为破坏了规则3希望该包通过的意图，即发生了特征干扰问题。

结束语 本文中我们提出了一种基于构件组装方式的特征干扰检测方法。这种方法是根据构件组装的不同方式，采用不同的规则检测特征干扰。需要指出的是，由于对不同的应用系统和不同的用户，其需求各不相同，对构件间交互的期望也各不相同。所以我们在检测过程中采用的方法是尽量找出组装过程中可能出现的特征干扰，让用户决定哪些是他所期望的，哪些则是必须排除的。

在以后的研究中，我们还有许多工作要做。一是我们只给出了几种组装方式下检测特征干扰的规则，这些规则本身还不完善；而其它构件组装方式中的特征干扰问题的方式和检测规则还有待研究。二是我们研究的只是简单构件组装过程中的特征干扰的检测，在这种情况下的特征干扰一般都是比较明显的、容易被发现的；在此基础上，我们还要研究多个构件以多种不同方式组装时特征干扰的检测规则。

参 考 文 献

1 李敏,费丽娟,胡金柱.软件系统中特征干扰问题的研究.计算机科学,2003,30(8,增)
2 Pomakis K P,Atlee J M. Reachability Analysis of Feature Interactions: A Progress Report. In:The international symposium on software testing and analysis. California,United States,1996
3 Heisel M,Souquieres J. A heuristic approach to detect feature interactions in requirements. In:Fifth Intl. Workshop on Feature Interactions in Telecommunications and Software Systems. Lund,Sweden,1998
4 任洪敏,钱乐秋.构件组装及其形式化推导研究.软件学报,2003,14(6):1065～1072
5 唐稚松,等.时序逻辑程序设计与软件工程(上).科学出版社,1999
6 唐稚松,等.时序逻辑程序设计与软件工程(下).科学出版社,2002
7 任洪敏,朱承,钱乐秋.基于时序逻辑软件构架形式化方法研究.计算机科学,2003,30(5):1～3
8 梅宏,黄罡,邢岩,彭枫.特征交互问题导论.电子学报,2002,30(12A):1923～1927