下一代通信软件中的特征冲突检测*

王 栋[1,2+], 梅 宏[1]

[1](北京大学 信息科学技术学院 软件研究所,北京 100871)

[2](贝尔实验室 中国基础科学研究院,北京 100080)

Detecting Feature Interactions in Next Generation Communication Software

WANG Dong[1,2+], MEI Hong[1]

[1](Institute of Software, School of Electronics and Computer Science, Peking University, Beijing 100871, China)

[2](Bell Laboratory Research China, Lucent Technologies, Beijing 100080, China)

+ Corresponding author: Phn: +86-10-68748088 ext 8427, E-mail: wangd@lucent.com, http://www.pku.edu.cn

Received 2005-01-20; Accepted 2005-03-11

Wang D, Mei H. Detecting feature interactions in next generation communication software. *Journal of Software*, 2005,16(7):1232−1241. DOI: 10.1360/jos161232

Abstract: Features denote the extensions of the basic function set of the communication software, while Feature Interactions (FIs) mean the unexpected interference between the features. The paper studies the FIs in the next generation communication software mainly on the aspect of the distributed implementation and deployment of the features. Based on the Communication Finite State Machine (CFSM) model, the authors design an FI detection method using the system verification technique. For the state explosion problem during verification, an optimization scheme is presented to reduce the complexity. Its validity is proved in theory and illustrated by examples.

Key words: next generation communication software; feature interaction; communication finite state machine (CFSM); system verification; state explosion

摘 要: 通信软件在其基本服务基础上进行扩展而得到的附加功能被称为特征,由于特征之间的相互干扰所导致的软件系统的异常行为被称为特征冲突问题.研究了下一代通信软件中由于特征的分布式实现和部署而产生的特征冲突问题.基于通信有限状态机模型,采用系统验证的技术设计了一种分布式特征冲突检测方案;对于生成全局状态自动机所导致的状态爆炸问题,提出了 4 条消除冗余状态和变迁的优化规则.对所提出的方案及优化策略进行了理论上的证明和实际中的验证,结果均表明,该方案对于通信软件中的特征冲突检测是有效的.

关键词: 下一代通信软件;特征冲突;通信有限状态机;系统验证;状态爆炸

中图法分类号: TP311 **文献标识码**: A

* Supported by the National Natural Science Foundation of China under Grant No.60233010 (国家自然科学基金)

作者简介: **王栋**(1976－),男,山西壶关人,博士生,主要研究领域为网络软件;**梅宏**(1963－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为软件工程.

1 研究背景介绍

通信软件是指用于构建和管理各种通信网络的软件系统,而后者所能提供的各种服务也主要由前者来实现,因此通信软件的正确性对于整个通信网络而言意义重大.在影响通信软件质量的各类因素中,特征冲突是最难以解决的问题之一,对该问题进行的研究工作现在已经发展成为一个相对独立的领域[1].

在通信软件中,通过扩展系统的基本功能而得到的各项附加服务通常被称为特征(feature).一般情况下,特征是随着用户需求和应用环境的不断改变而逐渐添加到系统中的,因此特征的设计和开发往往具有相互之间的独立性.当把多个独立设计的特征集成到一个系统中以后,特征之间一些未曾预料到的相互作用会使某些特征、有时甚至是整个系统行为异常.这种现象就被称为特征冲突问题(feature interaction,简称 FI).

对特征冲突问题的研究开始于 20 世纪 80 年代贝尔实验室对传统电信软件系统的开发过程中[2],大部分的早期工作也局限于该领域.随着技术的发展,下一代通信软件系统正在逐渐地取代传统电信系统,开始提供更加丰富和灵活的服务.然而特征冲突问题并没有随着这种技术的革新而有所缓解,相反地,由于下一代通信系统中服务种类的增加和定制灵活性的增强,特征冲突问题依旧是影响新特征正常问世的主要障碍之一,因此,对特征冲突的研究也开始向这个新的领域扩展,相关的工作主要有以下一些:Kolberg 等人[3]首次对部署在基于 IP 分组交换网络中的服务之间出现的不兼容性进行了讨论;并在文献[4]中对一般的 Internet 服务之间的特征冲突问题进行了研究;Yoneda 等人[5]报告了一个在 VoIP 系统中应用已有形式化技术进行特征冲突检测的实验结果;Turner[6]提出了一种新的描述 VoIP 特征的方法;Wu 等人[7]对 VoIP 系统中部署于终端上的特征之间的冲突问题进行了针对性的研究,等等.但总而言之,由于研究开展的时间不长,目前已有的方法还停留在对特征的描述方法和一些较为简单的实验阶段,尚没有得到一些成熟的理论和可以实际使用的有效的冲突检测技术.

与传统的电信网络相比,下一代通信软件中,特征的实现和部署具有很多不同的特点,也引发了不同的表现形式和内在原因的特征冲突问题[8].本文着眼于其中最为重要的一点,即特征的分布式实现和部署方面,研究了下一代通信软件中的特征冲突问题.本文的创新性主要体现在使用通信有限状态机模型对整个分布式系统进行了描述,并在此基础上采用系统验证的技术设计了一种特征冲突检测方案;对于系统验证中的状态爆炸问题,提出了 4 条冗余状态消除的规则,并对其正确性和有效性进行了理论和实验的证明.

本文第 2 节介绍下一代通信软件中的特征冲突问题;第 3 节对所考察的系统进行模型描述;第 4 节在前面模型的基础上,提出进行特征冲突检测的基本方法及优化措施;第 5 节进行实例分析;第 6 节总结全文,并指出将来工作的方向.

2 下一代通信软件中的特征冲突问题

前文已经提到,通信软件是指用于构建和管理各种通信网络的软件系统.根据通信技术不同的发展阶段,我们可以把通信软件分为传统的电信交换系统软件和下一代通信软件(next generation communication software)两个类别.前者是指基于传统的链路交换网络所开发的软件系统.这类软件最大的特点是其复杂的信令系统和相对集中式的服务提供模式;而后者则是指基于现代互联网络(Internet)的新型通信软件系统.由于更高效的分组交换技术的采用和终端系统智能的提高,下一代通信软件的特点表现为简单、灵活且容易扩展的信令系统,以及更加分布化的特征实现和部署方式,目前最主要的实现之一就是基于会话启动协议(session initiation protocol,简称 SIP)[9]的软交换系统.

虽然上述两种软件系统存在一些功能上的重合之处,但随着下一代通信软件逐渐得到部署和应用,人们发现,基于二者之间不同的特点,特征冲突问题本身也发生了很大的变化.我们对此加以举例说明.

在传统电信软件的特征冲突问题研究中,一个经常被提到的例子就是无条件呼叫转移(call forwarding unconditionally,简称 CFU)和遇忙呼叫转移(call forwarding when busy,简称 CFB)两个特征之间的冲突情况,前者是指定制该特征的用户将所有的来电无条件地转移到第三方,而后者是指只有当定制该特征的用户处于“Busy”状态时,才将来电转移到第三方.如图 1 所示,用户 *A* 同时定制了这两项特征,呼转的目标分别是 *C* 和 *B*.在这种情况下,设想如下场景:即当用户 *A* 处于“Busy”状态,且又有来电呼入.这时,根据各自特征的定义,CFU 要

求将这个新的来电转移到 *C*,而 CFB 则希望它被转移到 *B*,如果 *B* 和 *C* 不同,则二者的冲突不可避免.从中也可以得出结论,在传统电信软件中,只有当 CFU 和 CFB 两个特征的呼转目标不同时,二者才会出现相互冲突的场景.而这一点在下一代通信软件中则变得有所不同.

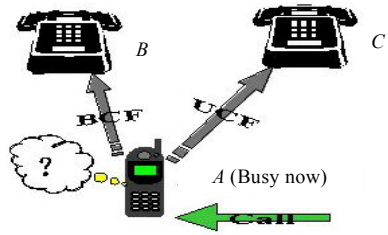

Fig.1 Interactions between features CFU(*A*,*C*) and CFB(*A*,*B*)
图 1 特征 CFU(*A*,*C*)和特征 CFB(*A*,*B*)之间的冲突情况

下一代通信软件中,特征的分布式实现指的是特征功能的完成需要依赖于物理上相互分离的设备所提供的信息.例如,与传统电信软件不同,终端状态信息在下一代通信软件中不再集中存储于交换设备中,而是由端系统本身决定.因此,在特征 CFB 的实现过程中,为了获取被叫方当前的状态,特定的消息交互过程不可缺少.图 2 显示了特征 CFU 和 CFB 在下一代通信软件中实现的基本消息流程.其中特别需要注意的是,在特征 CFB 的实现中,代理服务器则必须首先将 INVITE 消息转发到特征的定制方(用户 *A*),只有当用户 *A* 作出响应,说明自己正处于"Busy"状态时,代理服务器才能将 INVITE 消息转发到呼转目标(用户 *C*).这种实现方式所造成的结果就是即使两项特征的呼转目标相同,但由于二者对代理服务器在接收到同一条消息时的动作有不同的要求,因此也会产生相互之间的冲突问题.

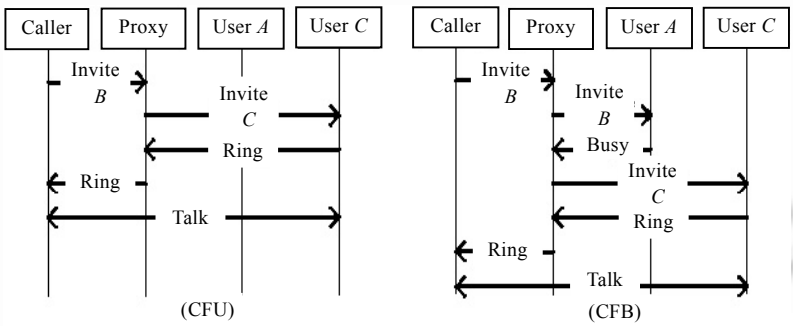

Fig.2 Interactions between features CFU(*A*,*C*) and CFB(*A*,*C*) in next generation communication software
图 2 特征 CFU(*A*,*C*)和特征 CFB(*A*,*C*)在下一代通信软件中的冲突情况

从上述例子可以看出,特征的分布式实现和部署是造成下一代通信软件中出现特征冲突且不同于传统电信软件的重要原因之一,因此从这个方面入手,以基于 SIP 协议栈的通信软件为例,我们对其中的特征冲突检测问题进行了研究.

3 问题的模型描述

对具体问题建立抽象的数学模型是提出具有一定普遍意义的解决方案的前提.下面,我们首先介绍一个典型 SIP 软件系统的具体组成和功能,然后对其进行抽象的模型描述.

3.1 一个典型的SIP软件系统

图 3 是一个基于 SIP 的通信网络示意图,其中包含了一些可以部署特征的 SIP 设备,如注册服务器(registrar),用于维护用户的帐户信息和进行安全认证等;代理服务器(proxy),用于转发 SIP 消息到正确的地址;媒体服务器(media server),用于提供附加的媒体流服务等等.与传统电信网络不同,在 SIP 系统中,用户终端(*Tn*)作为智能设备,可以控制实际的呼叫过程,因此也属于 SIP 系统中可以部署特征的设备.

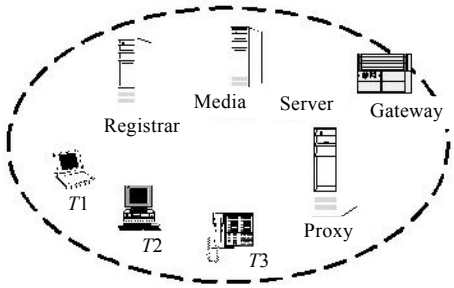

Fig.3 A typical SIP system
图 3 一个典型的 SIP 系统

我们把运行于上述网络中各个设备上的软件的集合称为 SIP 软件系统,它是下一代通信软件的一个典型实例.由于在 SIP 软件系统中,不同特征的实现和部署位于不同的网络设备上,即软件子系统中,构成了一个分布式的软件系统,因此特征冲突的发生不再仅仅局限于单个设备,而需要从整个系统的角度进行全面的考察和分析,这对采用的系统描述模型也提出了相应的要求.

3.2 特征的模型描述

建模的第 1 步是对局部的特征进行描述,我们采用有限状态机(FSM)[10]对上述网络中运行于单个设备上的软件子系统进行描述,并由此得到特征的定义.特征本质上是对系统某些方面行为所进行的修改或增强,在 FSM 模型中,这种修改和增强可以表现为特定状态和变迁的删除、改变和增加.通过对系统在特征集成前后的 FSM 模型进行比较,可以将其中的状态和变迁分为两种,一种是未受到新特征影响的状态和变迁,另一种是改变和增加了状态和变迁,而后者就反映出特征对集成后系统的影响,也是特征逻辑的实际表达,因此我们可以据此定义特征.

定义 1(特征). 在有限状态机模型中,特征就是在系统的集成过程中,增加或是修改了的状态和变迁.

图 4 对上述的定义作出了形象的说明,其中,在集成了系统 S_0 的基本功能和特征 f_i 的有限状态机 S 中,粗线条表示的部分可以认为是对特征的描述.

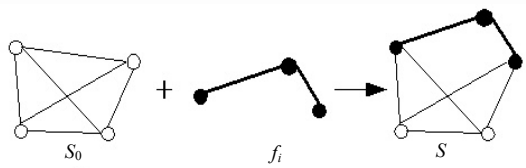

Fig.4 Feature described in FSM model
图 4 特征的模型描述

3.3 特征的分布式集成

由于特征冲突发生在特征的集成过程中,因此在得到特征定义之后,建模的第 2 步是考察如何进行特征的集成操作.由于我们所考察的系统包含了多个物理上相互分离的独立子系统,子系统之间只能通过协议规定的消息进行相互通信,而特征却可以部署在任何子系统中,因此,研究特征如何进行集成操作,实际上是考察特征之间如何通过网络消息交换来影响彼此的状态演化行为.为了能够描述这种相互间的影响,我们需要全面考察整个系统的行为,即子系统本地逻辑和网络中的消息交换两个方面.为达到这个目的,我们采用通信有限状态机(communication FSM,简称 CFSM)[11]来对整个分布式系统进行描述,以便可以在一个统一的模型中考察各个特征的行为及其相互间的影响.有关 CFSM 的定义和操作请参考文献[11],下面,我们给出在这种模型中特征集成操作的定义.

定义 2(特征的集成). 对于 N 个集成了特征的软件子系统 $F_i=\{I_i,O_i,S_i,\delta_i,\lambda_i,o_i\}$,$i=1,\ldots,N$,其中 I_i,O_i,S_i,分别表示各自的输入、输出和状态集合,δ_i,λ_i 是变迁函数和输出函数,o_i 是系统的初始状态,它们的分布式集成结果可以表示为

$$F=\{\{S_i\}_{i=1}^N,\{o_i\}_{i=1}^N,\{M_{ij}\}_{i,j=1}^N,succ\},$$

其中,$\{M_{ij}\}_{i,j=1}^N$是N^2个从子系统i到j的单向消息的集合.M_{ii}恒为空,表示子系统不会向自己发送消息,$succ$是一个映射,可以有两种形式,$S_i\times M_{ij}\to S_i$或$S_i\times M_{ji}\to S_i$,分别表示第i个子系统发出或收到一条消息之后的状态变化.

CFSM模型的确从整体上描述了SIP软件系统所有可能发生的消息交换和状态变迁,但在表现方式上通常不如一般的有限状态机那么直观,而且直接判断CFSM的正确性与否也是较为困难的.为了解决这个问题,我们引入了分布式系统全局状态*这一概念,并在此基础上,定义系统的全局状态自动机(global state automata,简称GSA)来直观地表明系统整体所处的状态和系统局部的变迁对整体产生的影响.我们将系统的全局状态定义为所有子系统状态的一个快照(snapshot),而任何子系统状态的变化都可以使全局状态产生变迁,在这个过程中,既可以存在网络消息的交换,也可以是由某个软件子系统自身的变化所引发.因此,我们可以得到从CFSM转化到GSA的算法genGSA,如下所示.

算法 1. 全局状态自动机的生成 genGSA(curState s_G).

输入:GSA的初始状态 $s_{G0}=\{o_1,o_2,\ldots,o_N\}$.

输出:全局状态自动机 $A=\{s_{G0},S_G,d\}$.

```
1.  for each state s_i in curState s_G={s_1,s_2,…,s_N}
2.     for each state s_j that s_i×m_ij→s_j or s_i×m_ji→s_j
3.        if (s_j≠o_i){
4.           s′_G={s_1,s_2,…,s_{i-1},s_j,s_{i+1},…,s_N};
5.           S_G=S_G∪{s′_G}
6.           δ=δ∪{s_G→s′_G}
7.           genGSA(s′_G);}
8.  return
```

在算法中,我们使用了深度优先的递归搜索算法,其中通信有限自动机本身作为全局变量供各个步骤使用,而系统的当前状态作为递归调用参数在算法各级之间传递,且其初值为整个系统的初始状态.根据通信有限自动机的变迁函数,得到了等价的全局状态自动机.需要指出的是,算法genGSA的正确性需要基于一个重要的前提,就是在特征冲突问题的研究中,我们总是认为集成了单个特征的系统是没有错误的.表现在模型中就是说,相应的有限状态机中的各项变迁将会形成多条正常的路径,并且对于通信软件系统而言,这些路径最终都将返回到系统的初始状态.

全局状态自动机兼顾了分布式系统本地的逻辑变化和相互之间的消息交换,是从整体上对分布式系统所有可能行为的一种全面描述,也反映了分布式环境中不同设备上所部署特征的相互影响关系,因此我们可以使用它作为分布式系统特征集成的一种描述工具,特征可以部署在任意的子系统中,由此产生的对其他子系统和系统整体的影响都反映在新构造的全局状态自动机中,这种方法解决了由于特征部署在不同物理设备上的分布式集成问题.

4 特征冲突检测

4.1 特征冲突的表达和基本的系统验证方法

特征冲突问题是当特征集成到系统中时,对正常的系统行为产生的意外影响.这种意外体现在全局状态自动机中,就是它不能满足特定的性质.对通信软件而言,这些性质包括通用属性和时序属性两类.前者包括了系统中没有死锁(deadlock)、活锁(livelock)和非正常结束(abnormal termination)状态等等,而后者又可以分为两种

*本文对于全局状态的定义有别于文献[6],不同点主要在于,本文在CFSM向GSA转换的过程中,不考虑消息在网络中的传输过程,因为这种因素不会对特征冲突的检测产生影响.

属性:状态属性(state properties)和变迁属性(transition properties).其中,状态属性是由单个原子属性或多个原子属性的逻辑连接所构成,这里,原子属性是指对软件系统各种变量的判断;而变迁属性是指系统变迁路径上状态属性之间的关系.

判断一个软件系统能否满足上述这些属性的过程称为"系统验证(system verification)"[12].系统验证的过程可以分为两个主要的步骤,一是系统建模和性质表达,二是对第 1 步得到的结果进行详尽的检测,以得到其是否满足特定性质的结论.对于基本的系统验证技术,可以参考文献[12],我们将在第 5 节中给出使用基本验证技术得到的结果.

4.2 Shrink-CFSM方案

在上述基本的验证过程中,一个主要的困难在于,由分布式系统生成的全局状态自动机中出现的状态和变迁路径数量爆炸问题,根据生成算法,在最坏情况下,全局状态自动机的状态数目是 $\prod_{i=1}^{N}|S_i|$,即所有子系统状态数的乘积,而变迁的数目则为 $\prod_{i=1}^{N}|S_i|(\prod_{i=1}^{N}|S_i|-1)/2$.虽然通过全局状态自动机的生成算法,已经删除了一些不可达的状态和不必要的变迁,但就检测特征冲突这个目的而言,我们依然可以发现上述全局状态自动机中存在的冗余,因此本节我们讨论如何减少这些冗余的状态和变迁.

在定义特征时,我们已经指出,在分布式系统的各个子系统中,状态可以分为两类,即与所考察特征相关的状态和与特征无关的状态.与此相类似,我们可以将系统的全局状态分成两类,即与某个或多个特征相关的全局状态和与任何特征都无关的全局状态.以子系统数目 N=2 为例,如果我们记子系统中和所考察特征相关的状态为 B 类状态,而与所考察特征无关的状态为 W 类状态,那么,在全局状态自动机中,就存在 4 种不同类别的状态,分别为 BB,BW,WB 和 WW,而我们将前 3 种均归入和所考察特征相关的全局状态,而最后一种为与所考察特征无关的全局状态.

在特征冲突的研究中,一个合理的假设是在所考察的特征部署之前,整个分布式系统中并没有冲突发生,也就是说,当时的状态机可以满足针对各种性质的验证而没有"错误"出现,因此一个自然的思路就是在生成的全局状态自动机中,将某些和特征无关的状态去除而不影响对特征冲突的检测.下面我们就此问题展开讨论,首先针对全局变量自动机提出 4 条状态的优化规则,分别对应于自动机的中间、起始和结束 3 个阶段,然后证明其正确性.

优化规则 1. 对于全局状态自动机内部没有分支的路径 $s_{G,a},s_{G,a+1},...,s_{G,a+b}$ 上,如果 $s_{G,a+1}$ 到 $s_{G,a+b-1}$ 为连续的特征无关状态,而 $s_{G,a}$ 和 $s_{G,a+b}$ 为特征相关状态,则可以将由 $s_{G,a+1}$ 到 $s_{G,a+b-1}$ 的状态去除,如图 5 所示.

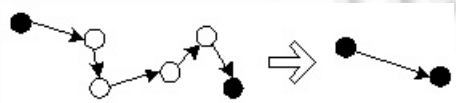

Fig.5 Optimization rule 1

图 5 优化规则 1

优化规则 2. 对于全局状态自动机内部两个相邻的特征无关状态 $s_{G,a}$ 和 $s_{G,a+1}$,如果 $s_{G,a}$ 可以去除,而且 $s_{G,a}$ 是 $s_{G,a+1}$ 唯一的父节点,则 $s_{G,a+1}$ 也可以去除,该规则可以递归使用,如图 6 所示.

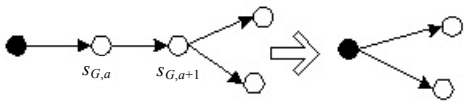

Fig.6 Optimization rule 2

图 6 优化规则 2

优化规则 3. 如果全局状态自动机的初始状态为特征无关状态,那么由初始状态出发的所有路径上,直到第 1 个特征相关状态、或是第 1 个可以由本路径上前缀之外的状态变迁而来的特征无关状态(不包含)为止,所有的特征无关状态可以去除,但初始状态保留,该规则可以递归使用,如图 7 所示.

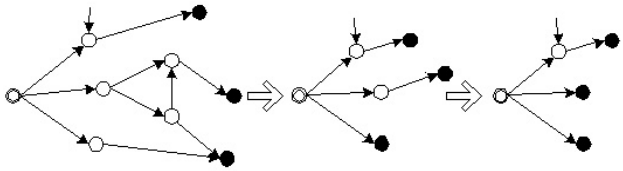

Fig.7 Optimization rule 3
图 7 优化规则 3

优化规则 4. 如果全局状态自动机的结束状态为特征无关状态,那么由结束状态反向回溯的所有路径上,直到第 1 个特征相关状态、或是第 1 个可以由本路径上前缀节点之外的节点变迁而来的特征无关状态(不包含)为止,所有的特征无关状态可以去除,但该结束状态保留,该规则可以递归使用,如图 8 所示.

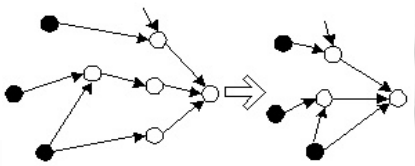

Fig.8 Optimization rule 4
图 8 优化规则 4

对于上述优化规则的正确性,我们提出如下两条定理.

定理 1. 上述规则 1~规则 4 的优化措施对系统是否满足以下性质没有影响,这些性质指的是死锁、活锁、非正常结束以及状态属性等,而且它们均被用来判断系统中是否存在特征冲突.

证明:证明可以分为两个方面,一是原系统中由于特征冲突而不能满足的任意性质在经上述优化之后依然不能满足,即特征冲突的情况没有被减少;二是系统优化之前能满足而之后不能满足的性质不涉及特征冲突的判断,即特征冲突的情况没有增加.我们对定理中提及的 4 种属性:死锁、活锁、非正常结束以及状态属性一一进行讨论.

(1) 死锁

① 各种优化规则不会增加状态,因此不会在系统中增加死锁的状态;

② 各种优化规则不会减少系统中的死锁状态,这一点我们通过反证法加以说明.假设由于某条优化规则而去除的状态为死锁状态,那么由于我们去除的都是特征无关状态,因此该死锁状态也应当属于特征无关状态.全局状态自动机中的特征无关状态是由原来分布式系统未部署特征之前的状态生成,因此其推论就是在原来的分布式系统中存在死锁的情况,这个结果与我们的前提相互矛盾,因此假设不成立,即优化规则不会减少死锁.

(2) 活锁

① 各种优化规则不会增加状态,因此不会在系统中增加活锁的发生;

② 分布式系统中的活锁在全局状态自动机中是一个环路,在上面的状态优化过程中,我们保留了所有可能的环路,因此优化规则不会减少活锁.

(3) 非正常结束

对于非正常结束状态的讨论类似与死锁,参考上面的(1).

(4) 状态属性

对于特征冲突的检测而言,有意义的应当是对特征相关状态的属性判断.在上面的各项优化措施中,规则保留了所有的特征相关状态,因此不会影响判断特征冲突的状态属性. □

定理 2. 规则 3 的优化措施不会减少系统中由于特征冲突而不能满足的变迁属性.

证明:我们依然采用反证法,证明系统中任意一条由于特征冲突而不能满足的变迁属性在优化的系统中依然存在.假设存在一条由于特征冲突而不能满足的变迁属性在系统优化之后变得可以满足,那么可以得出的结论是在优化之前导致该属性得不到满足的状态就是优化过程中去除的状态.根据规则 1 的定义,我们去除的是

系统所有变迁路径上特征相关状态之前的特征无关状态(甚至更前),由此,又可以得出的推论是由系统初始状态开始的特征无关状态的变迁中存在冲突,这与我们的前提是矛盾的,因此假设不成立,即规则 1 的优化不会减少由于特征冲突而使得系统不能满足的变迁属性. □

上述定理说明,我们提出的优化规则其正确性是有保证的,但是对于某些性质(如变迁属性)而言,尚没有找到消除这类冗余的充要条件.这也是下一阶段工作的主要内容之一.

5 实例研究

为了验证上述方法的有效性,我们选取了目前基于 SIP 系统设计的一项新的特征,个性化回铃音(personalized ringback tone,简称 PRT)和一项传统特征无条件呼叫转移在 SIP 中的实现作为例子进行讨论,首先对它们在 SIP 系统中各自的实现进行形式化的建模,然后对它们之间潜在的特征冲突进行检测.

5.1 特征的实现及建模

SIP 系统中特征无条件呼叫转移的实现是较为简单的.下面,我们对在 SIP 系统中如何实现个性化回铃音进行详细的说明.

如图 9 所示,SIP 系统中个性化回铃音特征的实现要基于包括媒体服务器(medai server)和用户 *B* 的呼入代理在内的多个物理上相互分离设备的参与,是一个典型的分布式实现的特征.其基本过程可以描述如下,用户 *A* 通过其代理向用户 *B* 发送呼叫请求(INVITE),当该请求到达用户 *B* 的代理服务器时,代理服务器一方面将请求向用户 *B* 转发,另一方面通过 SIP 系统的第三方呼叫控制(3PCC)机制,将用户 *A* 和媒体服务器相联,通过另一条媒体通道向用户 *A* 播放 *B* 事先指定的回铃音.我们可以对特征 PRT 在代理服务器和媒体服务器上不同的实现逻辑使用有限状态机模型进行描述,如图 10(a)和图 10(b)所示,而图 10(c)描述了用户 *A* 的代理服务器中部署了 CFU 之后的状态转换行为.图中 xxx-idle 为各个设备的初始状态,在变迁过程中,+*msg* 表示收到消息 *msg*,而–*msg* 表示发出消息 *msg*.基于这些模型及它们之间的消息交互关系,我们就可以得到整个系统的通信有限状态机模型,进而生成表示整个系统逻辑变化关系的全局状态自动机(状态过多,略去).

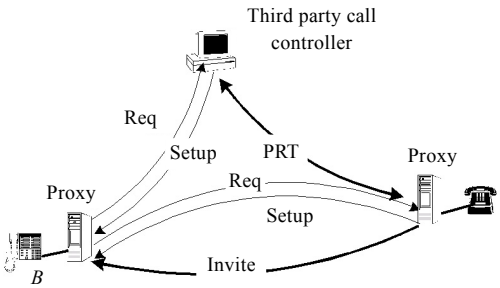

Fig.9 Implementation of the feature PRT

图 9 特征 PRT 的实现

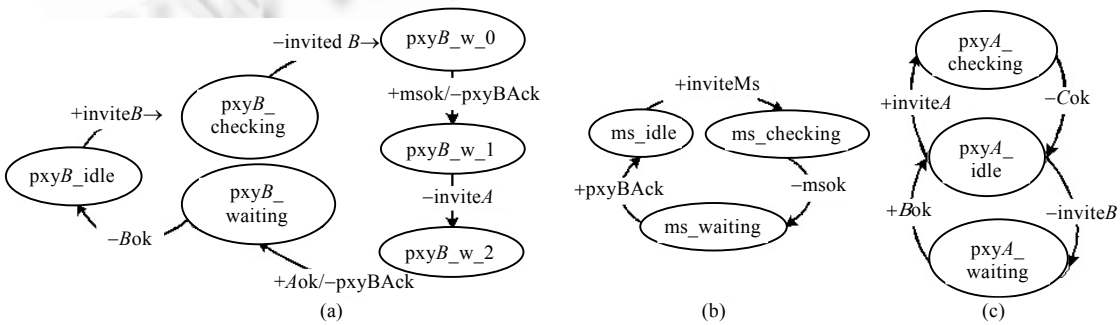

Fig.10 FSM model for each sub-system

图 10 各个子系统的有限状态机模型

5.2 基本验证方法的检测

我们采取了模型验证(model checking)[12]的方法对上节生成的全局状态自动机进行验证,这样做的好处是,在发现错误的时候,可以通过路径的回溯发现出现错误的原因,即举出反例.通过验证,我们首先发现了一条对用户 *A* 的通信能力有额外要求的变迁路径,如图 11 所示.在图 11 所示的最后一个状态中,用户 *A* 的代理正在等待用户 *B* 发回的 OK 消息,而用户 *B* 的代理也正在等待用户 *A* 发回的 OK 消息,而二者之间的信令通道中仅有一条 invite*A* 消息,因此,如果用户 *A* 此时不能处理该消息,则整个系统将陷入死锁状态.通过分析我们可以看出,这个异常状态产生的原因在于特征 PRT 要求主叫方具备同时和多个通信实体建立媒体通道的能力,而不能仅仅具有点到点的通信功能.这项能力对于目前支持 SIP 协议栈的众多软件终端而言一般不成问题,然而对于某些设计时没有考虑这项功能而且逻辑固化的硬件 SIP 终端而言,则是很难支持的.

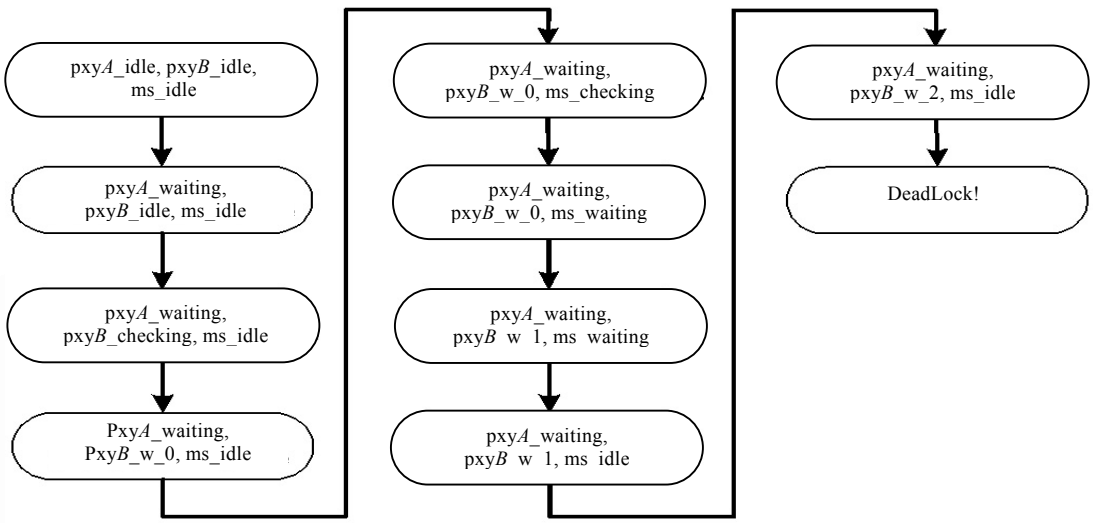

Fig.11 Error scenario I

图 11 出错场景之一

我们假设所考察环境中的终端具备该项能力,从而可以将这条路径上的验证继续下去,然而又出现了新的错误状态,如图 12 所示.

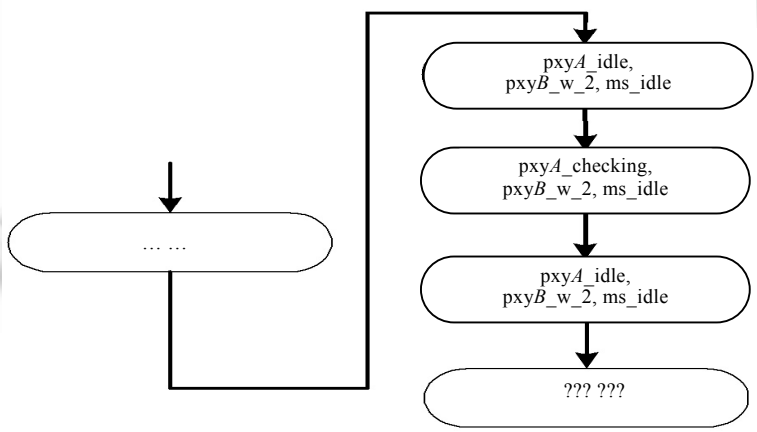

Fig.12 Error scenario II

图 12 出错场景之二

图 12 展示了特征 PRT 和 UCF 相互冲突的一个场景,用户 *B* 的代理在 3PCC 的过程中等待用户 *A* 发来的 *A*ok 消息,然而由于用户 *A* 代理上部署了 UCF 特征,将 *B* 代理的呼叫请求转发到了用户 *C*,因此 *A* 代理所能回应

的只能是 *C*ok 消息,这种不一致就反映出了两种特征的相互冲突之处.

5.3 应用优化策略的结果

在上述系统生成的全局状态自动机模型中,状态数为 54 个,而变迁为 108 条.为了减少需要验证的状态和路径的数量,我们应用上述优化规则对该自动机进行优化处理.在我们的实验所涉及到的各设备的有限状态机模型中,可以将下列状态视为由特征的部署而引入的新状态,它们是用户 *A* 代理服务器中的 pxy*A*_checking,以及用户 *B* 代理服务器中的 pxy*B*_w_0,pxy*B*_w_1 和 pxy*B*_w_2,其余状态为系统原有状态,在此基础上,应用 4 条优化规则的结果见表 1,总共有 17 个和特征冲突发现无关的状态被消除,而且对上述验证的结果没有影响.

Table 1 Result after optimizing the global state automata

表 1 对全局状态自动机进行优化的结果

Optimization rule	Times applied	Number of erased states	In all
1	0	0	17
2	14	14	
3	0	0	
4	3	3	

6 结论及将来的工作

本文研究了下一代通信软件系统中的特征冲突问题,着眼于其中最重要的一个特点,即特征分布式的实现与部署.通过使用通信有限状态机模型对分布式系统的整体进行描述,得到了反映系统整体变化的全局状态自动机,并通过对后者进行系统验证来得出实际的软件系统中是否存在特征冲突的结论.对于全局自动机中冗余状态和变迁的去除,本文提出了一种 Shrink-CFSM 的优化策略,由 4 条优化规则构成,并对其有效性进行了理论证明和实例检验,对其中尚存的不足之处进行了说明.

我们下一步的工作方向主要包括了在更大的范围内应用本文提出的技术方案,并在实际中不断加以改进;同时,对于现有的一些不足,如对于系统变迁属性进行优化的充要条件等方面进行更为深入的研究.

References:

[1] http://www.cs.le.ac.uk/~srm13/fiw05/. 2005.

[2] Bowen TF, Dworack FS, Chow CH, Griffeth N, Herman GE, Lin YJ. The feature interaction problem in telecommunications systems. In: Proc. of the 7th Int'l Conf. on Software Engineering for Telecommunication Switching Systems (SETSS'89). London, 1989. 59−62. http://ieeexplore.ieee.org

[3] Kolberg M, Magill EH. Handling incompatibilities between services deployed on IP-based networks. In: Proc. of the IEEE Intelligent Networks (IN 2001). Boston: IEEE Press, 2001. 360−370. http://ieeexplore.ieee.org

[4] Kolberg M, Magill EH, Marples D, Tsang S. Feature interactions in services for Internet personal appliances. In: Proc. of the Int'l Conf. on Communications (ICC 2002). New York, 2002. 2613−2618. http://ieeexplore.ieee.org

[5] Yoneda T, Kawauchi S, Yoshida J, Ohta T. Formal approaches for detecting feature interactions, their experimental results and application to VoIP. In: Ayot D, Logrippo L, ed. Feature Interaction in Telecommunications and Software Systems VII. Ottawa: IOS Press, 2003. 205−212.

[6] Turner KJ. Representing new voice services and their features. In: Ayot D, Logrippo L, ed. Feature Interaction in Telecommunications and Software Systems VII. Ottawa: IOS Press, 2003. 123−140.

[7] Wu XT, Schulzrinne H. Feature interactions in Internet telephony end systems. Technical Report, New York: Columbia University, 2004.

[8] Lennox J, Schulzrinne H. Feature interaction in Internet telephony. In: Calder M, Magill E, ed. Feature Interaction in Telecommunications and Software Systems VI. Glasgow: IOS Press, 2000. 38−50.

[9] Rosenberg J, Schulzrinne H, Camarillo G, Johnston A, Peterson J, Sparks R, Handley M, Schooler E, eds. SIP: Session initiation protocol. RFC3261, IETF, 2002.

[10] Lee D, Yannakakis M. Principles and methods of testing finite state machines—A survey. Proc. of the IEEE, 1996,84(8): 1090−1123.

[11] Brand D, Zafiropulo P. On communicating finite state machines. Journal of the Association for Computing Machinery, 1983, 30(2):323−342.

[12] Clarke EM, Jr Grumberg O, Peled DA. Model Checking. London: MIT Press, 1999.