基于双格的软件产品线模型检测

石玉峰 魏 欧 周 宇
(南京航空航天大学计算机科学与技术学院 南京 210016)

摘 要 软件产品线在保留每个产品的可变性前提下通过最大化产品间的共性实现资源的再利用,从而提高生产效率和节约生产成本。近年来,基于特征的状态迁移系统应用于软件产品线的建模和验证中。然而现有的方法不能很好地支持软件产品线中存在的信息不确定和不一致的情况。为此,首先提出一种基于双格的特征迁移系统,用于软件产品线的行为建模,采用投影的方法定义产品的行为模型;然后采用动作计算树逻辑描述系统的时序属性,并且给出它在新系统上的语义,用于支持基于双格的模型检测;最后,采用多值模型检测工具 χchek 对方法的有效性进行实验分析。

关键词 模型检测,软件产品线,多值逻辑
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.2.036

Model Checking of Software Product Line Based on Bilattices

SHI Yu-feng WEI Ou ZHOU Yu
(College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

Abstract Software product line (SPL) maximizes the commonality between similar software products to reduce production costs and improve productivity. Recently, state transition systems based on features have been widely used in behavioral modeling and verification of SPLs. However, they can't nicely support uncertain and inconsistent information. Therefore, firstly a formalism, bilattice-based featured transition systems (BFTS), was proposed for model checking of SPLs with uncertain and inconsistent information, where product was defined via projection. Furthermore, action computation tree logic (ACTL) was used to describe temporal properties; its semantics on BFTS was defined for model checking. Finally, based on multi-valued model checker χ chek, a case study was conducted to illustrate the effectiveness of our approach.

Keywords Model checking, Software product line, Multi-valued logic

1 引言

软件产品线是一组在公共核心资源的基础上,按照规定的方式开发的软件密集系统的集合。这些系统共享一组公共的、可管理的、能够满足特定的市场或者任务需求的功能集合[1]。近年来,软件产品线已广泛应用于航天、汽车以及航空电子等关键系统,这使得软件产品线的模型检测的作用日益凸显。

模型检测[2]是一种自动形式化验证技术,用于判断计算机系统的正确行为属性,其基本方法是用一个状态迁移图 M 来表示待检测系统的模型,用时序逻辑公式 φ 来描述系统的正确行为属性,然后通过对模型状态空间的穷举搜索来判断 φ 是否在 M 上被满足。如果 φ 在 M 上被满足,则系统的正确性得以证实;否则,系统中存在错误,系统的正确性被证伪。与测试和习惯化推理等其他验证技术相比,模型检测的优点在于它能够检测系统模型所包含的所有行为,并能返回反例帮助找到错误产生的原因。

软件产品线的模型检测问题比单个系统的模型检测问题更为复杂。从模型检测的基本方法来看,实现软件产品线的模型检测需要判断软件产品线上每一个产品的正确行为属性,而每一个产品可视为一个单个系统,这就意味着完成软件产品线的模型检测相当于完成多个单个系统的模型检测。在实际中,软件产品线可用特征(feature)来描述,一条产品线可看作是一个有层次关系的特征的集合。所谓特征,是指软件系统或系统中用户可见的、显著或与众不同的方面、品质或特点[3]。软件产品线上的产品数量是随特征数量呈指数增长的。即若一条产品线有 n 个特征,那么它可以生产的产品种类最多可达到 2^n。这使得通过传统的逐一检测方法难以实施。

为此,目前已有相关工作对软件产品线的模型检测问题进行了研究,实现了在软件产品线层面上对产品属性的检测。

到稿日期:2014-04-07 返修日期:2014-06-12 本文受国家自然科学基金项目(61170043,61202002),国家重点基础研究发展计划(973)项目(2014CB744904)资助。

石玉峰(1990-),女,硕士生,CCF 会员,主要研究领域为模型检测、软件产品线,E-mail:shiyufeng1990@foxmail.com;**魏 欧**(1974-),男,博士,副教授,主要研究领域为形式化方法、软件自动验证,E-mail:owei@nuaa.edu.cn;**周 宇**(1981-),男,博士,副教授,主要研究领域为软件工程、形式化方法,E-mail:zhouyu@nuaa.edu.cn。

大部分研究是基于模态迁移模型[4](Modal Transition Systems,MTS)。在这些研究中,软件产品线采用迁移(transition)描述系统的所有行为(behavior)。迁移的种类分为必选型和可选型,其中必选型迁移表示必须存在的迁移,用来描述软件产品线的共性(commonality);可选型迁移表示允许存在的迁移,用来描述软件产品线中的可变性(variability)。

这些方法实现了对软件产品线可变性和共性的描述,但忽略了软件产品线的特征对系统行为的影响,导致它们的模型检测结果缺乏系统行为与特征之间的联系。文献[5]考虑到特征与行为之间的联系,提出特征迁移系统(Featured Transition System,FTS)。在 FTS 中,每条迁移采用特征标记(label)。然而,FTS 定义每条迁移只能标记一个特征,一方面,这将导致 FTS 不能很好地支持软件产品线中特征之间的不一致性。如在手机系统中,"飞行模式"和"浏览器"这两种功能分别设为特征 f 和特征 e,系统行为"信号接通"设为迁移(s,a,t)。若"飞行模式"禁止"信号接通"而"浏览器"需要"信号接通",则仅标记一个特征无法描述特征 f 和特征 e 对迁移(s,a,t)的不一致观点;另一方面,FTS 不能很好地支持软件产品线中特征与行为之间关系的不确定(unknown)情况。如在之前的手机系统中,现新增加一个功能"照相机",设为特征 c。其中,特征 c 与"信号接通"是否相关暂时不确定。根据 FTS 的定义,迁移(s,a,t)用特征 e 标记,则特征 f 和特征 c 视为与迁移(s,a,t)无关,这显然与已知条件中的情况不相符。

为此,本文提出一种行为模型-基于双格的特征迁移系统(Bilattice-based Featured Transition System,BFTS)。与传统的 FTS 不同,BFTS 定义了多值的迁移描述产品线中不确定和不一致情况。在 BFTS 中,有序对$\langle U,V\rangle$表示逻辑值,其中 U、V 分别代表依赖和排斥该迁移的特征集合,则既不在 U 中也不在 V 中的特征视为暂时不确定与该迁移是否有关。上述手机系统的 BFTS 中,迁移(s,a,t)的逻辑值则为$\langle\{e\},\{f\}\rangle$。由此,特征 f 和特征 e 之间的不一致性以及特征 c 与行为"信号接通"关系的不确定性得以描述。此外,本文采用投影(project)的方法定义产品的行为模型。

进一步,本文采用动作计算树逻辑(Action Computation Tree Logic,ACTL)[6]描述时序属性,在 BFTS 上定义 ACTL 的语义,实现了基于双格的模型检测。传统的 FTS 的模型检测结果无法表示特征对属性的影响,而 BFTS 的模型检测结果能表示影响该属性的特征。最后,为证明该方法的有效性,本文运用模型检测工具 χchek[7] 对例子中的软件产品线进行模型检测。

本文第 2 节概述相关的预备知识,包括特征图、多值逻辑的思想以及 ACTL 逻辑;第 3 节展示基于双格的特征迁移系统;第 4 节解释 ACTL 逻辑在 BFTS 上的语义;第 5 节呈现实验结果;最后讨论相关工作。

2 预备知识

本节将介绍与本文研究相关的预备知识,其中主要包括特征图、基于世界的双格逻辑(world-based bilattice)和 ACTL 逻辑。

2.1 特征图

软件产品线上的产品之间既存在相同或相似的部分,同时也存在着差异。可变性则用于描述软件产品线上产品之间的差异,这是软件产品线工程区别于其他软件开发过程的重要特征[8]。

可变性建模方法主要包括两类,即特征建模(Feature Modeling)和行为建模(Behavioral Modeling)。特征建模主要侧重于定义软件产品线中特征之间的层次关系。基于特征的可变性建模是最早用于可变性建模的方法[8]。

定义 1[9] 特征图(Feature Diagram,FD)是一种以图的形式表示的基于特征集合 F 的树状特征模型。其中,树中每条边代表一个特征约束。特征约束分为或(or),可选(optional),多选一(alternative)、必选(mandatory) 4 种,即:

$root(f)\equiv f$

$mandatory(p,f)\equiv f\Leftrightarrow p$

$alternative(p,\{f_1,\cdots,f_n\})\equiv((f_1\vee\cdots\vee f_n)\Leftrightarrow p)\wedge\bigwedge_{i<j}\neg(f_i\wedge f_j)$

$or\ (p,\{f_1,\cdots,f_n\})\equiv(f_1\vee\cdots\vee f_n)\Leftrightarrow p$

其中,p,f,f_i 为特征集合 F 中的元素。

定义 2 设产品线的特征集合 F,若存在集合 P 满足 $P\subseteq F$ 且 P 中的特征满足 F 中的特征约束关系,则称 P 是 F 的产品。

例 1 饮料自动售货机(简称饮料机)例子是软件产品线研究领域中代表性例子之一,最早在文献[10]中用以介绍 MTS 对软件产品线的行为建模。本文根据需要对原有的饮料机例子进行修改。其基本工作流程如下:饮料机初始状态为待机状态→顾客投进硬币→顾客选择饮料是否含糖→顾客选择饮料的种类→饮料机根据顾客的选择提供相应饮料→结束后显示"完成"→饮料拿走后回到初始状态。

以上是产品线的共性,其可变性如下:

- 可至少提供一种饮料:咖啡、茶或卡布其诺;
- 可有响铃功能:显示"完成"后响铃或不响铃;
- 可选择"快速服务":只允许提供茶或咖啡且显示"完成"后不响铃以减少等待的时间。

饮料机产品线的特征图如图 1 所示。图中每一个特征均用一个小写字母代替,如 c 代表特征"咖啡($coffee$)"。根据特征图的语法,饮料机的产品线的特征集合为$\{v,f,b,r,c,t,o\}$,产品线上所有的产品集合为:

$\{\{c,b,v\},\{b,t,v\},\{b,o,v\},\{c,r,b,v\},\{c,b,f,v\},\{t,r,b,v\},\{b,t,f,v\},\{c,b,t,v\},\{r,b,o,v\},\{b,o,f,v\},\{c,b,o,v\},\{b,t,o,v\},\{t,r,b,f,v\},\{c,r,b,f,v\},\{c,r,b,t,v\},\{c,b,t,f,v\},\{c,r,b,f,v,t\},\{o,r,b,f,v\},\{c,r,b,o,v\},\{c,b,o,f,v\},\{c,r,b,f,v,o\},\{t,r,b,o,v\},\{o,b,t,f,v\},\{r,b,f,v,o,t\},\{c,b,t,o,v\},\{c,r,b,v,o,t\},\{c,b,t,f,v,o\},\{c,r,b,f,v,o,t\}\}$

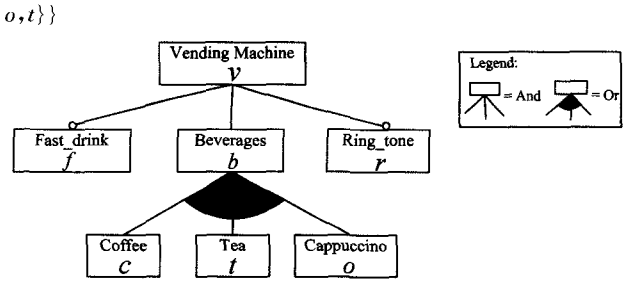

图 1 饮料机产品线的特征图

2.2 基于世界的双格

我们提出模型的基本思想是基于世界的双格所定义的多值逻辑。在此逻辑中，逻辑值之间不仅存在真值上的序关系还有信息上的序关系，逻辑值本身不仅存在真值方向上的含义也存在信息方向上的含义。因此，用它作为模型检测的结果在反映检测结果的同时也能体现结果的信息量。世界双格所定义的多值逻辑有以下定义。

定义 3[11] 设 W 为世界的集合，$P(W)$ 为 W 的幂集。基于世界的双格(world-based bilattice)为 $B_W=\langle P(W)\times P(W),\leqslant_t,\leqslant_i,\rightarrow\rangle$，对于任意的 $\langle U,V\rangle,\langle S,T\rangle\in P(W)\times P(W)$，有以下条件成立：

1) $\langle U,V\rangle\leqslant_t\langle S,T\rangle\stackrel{\triangle}{=}U\subseteq S\wedge T\subseteq V$

2) $\langle U,V\rangle\leqslant_i\langle S,T\rangle\stackrel{\triangle}{=}U\subseteq S\wedge V\subseteq T$

3) $\rightarrow\langle U,V\rangle\stackrel{\triangle}{=}\langle V,U\rangle$

其中，$\leqslant_t$ 和 $\leqslant_i$ 分别表示真值序关系与信息序关系。

有序对 $\langle U,V\rangle$ 表示逻辑值，如果某个命题 p 的逻辑值为 $\langle U,V\rangle$，则表示 p 在 U 这个集合的世界里都为真，在 V 这个集合的世界里都为假。

下面分别给出真值序关系和信息序关系上的交、并操作，其中 $\wedge$、$\vee$ 与 $\otimes$、$\oplus$ 分别对应于真值序关系与信息序关系。

定理 1[11] 设 $B_W=\langle P(W)\times P(W),\leqslant_t,\leqslant_i,\rightarrow\rangle$，对任意的 $\langle U,V\rangle,\langle S,T\rangle\in P(W)\times P(W)$，满足以下条件：

$\langle U,V\rangle\wedge\langle S,T\rangle=\langle U\cap S,V\cup T\rangle$

$\langle U,V\rangle\vee\langle S,T\rangle=\langle U\cup S,V\cap T\rangle$

$\langle U,V\rangle\otimes\langle S,T\rangle=\langle U\cap S,V\cap T\rangle$

$\langle U,V\rangle\oplus\langle S,T\rangle=\langle U\cup S,V\cup T\rangle$

2.3 ACTL 逻辑

实现在产品线上时序逻辑属性的模型检测，需要选择合适的时序逻辑描述系统属性。常见的时序逻辑有计算树逻辑(Computation Tree Logic，CTL)[12]和线性时序逻辑(Linear-time Temporal Logic，LTL)[13]。CTL 和 LTL 由于都是针对采用 Kripke 结构作为系统的计算模型，并不适用于我们提出的模型，因此本文采用一种基于动作的 CTL 逻辑-ACTL 描述系统的时序逻辑属性。

定义 4 ACTL 公式由下列规则进行归纳：

$\varphi::=\text{true}\mid\text{false}\mid\neg\varphi\mid\varphi\wedge\varphi'\mid\varphi\vee\varphi'\mid(\varphi\rightarrow\varphi')\mid AF\varphi\mid EF\varphi\mid EG\varphi\mid AG\varphi\mid A[\varphi U\varphi']\mid E[\varphi U\varphi']\mid\mid([a]\varphi)\mid\langle a\rangle\varphi$

其中，

- true，false：逻辑常量，分别表示“真”、“假”；
- $\wedge$，$\vee$，$\neg$，$\rightarrow$：基本逻辑连接词，分别表示“且”、“或”、“非”、“蕴含”；
- A：表示“所有路径”；
- E：表示“存在一条路径”；
- F：表示“将来某个状态”；
- G：表示“将来所有状态”；
- U：表示“直到”；
- a：动作(action)；
- $\langle a\rangle$：表示“存在某条迁移做 a 动作到达下一状态”；
- $[a]$：表示“所有做 a 这个动作的迁移到达下一状态”。

除了 $\langle\rangle$ 和 $[]$，每个 ACTL 时态连接词都是一对符号。符号对中的第一个是 A 或 E，符号对中的第二个符号是 F、G 或 U。

以例 1 中的饮料机产品线为例，ACTL 公式 $AG[sugar]AF\langle pour\_sugar\rangle true$ 描述的属性是“任何时候若顾客选择饮料加糖，则饮料机将来会在饮料里加糖”。

3 软件产品线的行为建模

不同于第 2 节中介绍的软件产品线的特征建模，软件产品线的行为建模主要侧重于描述产品线上所有产品在每一个系统所处状态所能发生的行为。常用的行为模型主要分为两类：基于状态的模型(state-based model)(如：Kripke 结构模型)和基于动作的模型(如：MTS)。我们提出的 BFTS 是一种基于动作的模型。本节首先给出 BFTS 的定义，然后介绍得到产品的 BFTS 的投影方法。

3.1 基于双格的特征迁移系统

传统的 FTS 中，每条迁移采用特征标记，如 $s\xrightarrow{a/f}t$ 表示从状态 s 出发，做 a 这个动作，到达状态 t，其中该迁移是特征 f 依赖的。但传统的 FTS 由于标记迁移的特征个数只有一个，这使其无法表示特征之间的不一致性和特征与行为的不确定性。为此，在 BFTS 中，我们采用有序对 $\langle U,V\rangle$ 表示迁移的逻辑值。

定义 5 一个基于双格的特征迁移系统是一个六元组：$M=(S,Act,s_0,\rightarrow,B_W,R)$，其中，

1) S 是一个有穷状态集；

2) Act 是一个有穷行为集；

3) $s_0\in S$ 是唯一的一个初始状态；

4) $\rightarrow\subseteq S\times Act\times S$ 是一个有穷迁移集；

5) $B_W=\langle P(W)\times P(W),\leqslant_t,\leqslant_i,\rightarrow\rangle$ 是一个基于世界的双格，其中 W 为一个有穷特征集；

6) R 是转换函数 $S\times Act\times S\rightarrow B_W$ 上的映像；

对于 M，若 $(s,a,s')\in\rightarrow$，则表示当前状态为 s，动作为 a 时，系统将到达下一个状态 s'，我们把 s' 称作 s 的后继状态；若 $(s,a,s')\notin\rightarrow$，则表示系统不存在一条迁移从状态 s 发出 a 这个动作后到达下一个状态 s'，因此该迁移被所有特征排斥，即迁移逻辑值为 $\langle\emptyset,W\rangle$。若 $R(s,a,s')=\langle U,V\rangle$，则表示迁移 (s,a,s') 的值为 $\langle U,V\rangle$，即该迁移被 U 中的特征需要，被 V 中的特征反对，与 $W\backslash(U\cup V)$ 中的特征关系尚未确定。此外，我们记 $\pi_t(\langle U,V\rangle)=U,\pi_f(\langle U,V\rangle)=V$。

例 2 以例 1 中的饮料机产品线为例，我们对饮料机产品线进行行为建模，如图 2 所示。在 BFTS 中迁移 (s_0,pay,s_1) 饮料机的“投币”行为，其中 s_0 为系统的初始状态。该行为被特征 v 依赖并且没有被产品线所包含的任何特征排斥。因此，$R(s_0,pay,s_1)=\langle\{v\},\emptyset\rangle$。同理，特征 c 要求饮料机能提供咖啡($coffee$)、倒咖啡($pour\_coffee$)以及加糖($pour\_sugar$)，3 种行为在 BFTS 中的逻辑值均为 $\langle\{c\},\emptyset\rangle$。

通过有序对来表示迁移的逻辑值，BFTS 能更好地反映特征之间的不一致性。如图 2 中 $R(s_{11},display\_done,s_{12})=\langle\{r\},\{f\}\rangle$，则表示迁移 $(s_{11},display\_done,s_{12})$ 被特征 r 依赖但被特征 f 排斥。于是，特征 r 与特征 f 的互斥关系在迁移 $(s_{11},display\_done,s_{12})$ 上得以体现。同理，特征 o 与特征 f 的互斥关系通过 $R(s_3,cappuccino,s_7)=\langle\{o\},\{f\}\rangle$ 得以体现。

而这些互斥关系在例 1 的特征图中并未描述，这最终会导致通过特征图语义得到的产品出现不一致的情况，如同时

包含特征 r 和特征 f 的产品$\{v,b,c,r,f\}$。因此，BFTS 在某种意义上也起到了完善特征建模的作用。

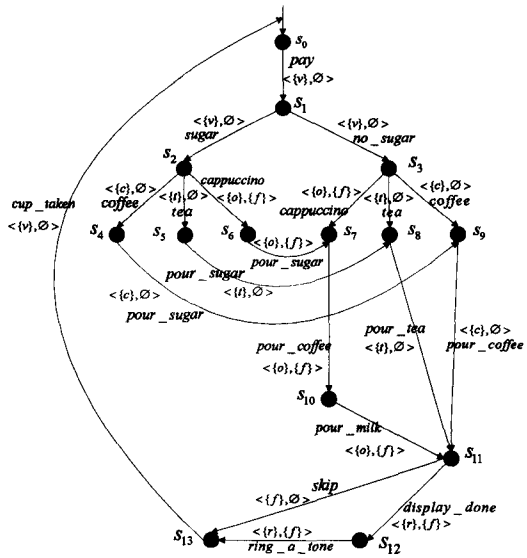

图 2　饮料机产品线的 BFTS 模型

3.2　产品的行为模型

本节主要介绍如何通过提出的投影方法得到产品的行为模型。在 2.1 节介绍特征图时，我们给出了产品的定义，即原产品线特征集合中符合特征约束关系的子集。本文提出投影的方法，实现了通过产品线的行为模型投影在产品上得到产品的行为模型。

设软件产品线 F 的行为模型为 BFTS M_F，W 是 F 的特征集合，P 是 F 的产品。若 M_F 中某迁移(s,a,t)的逻辑值为$\langle U,V\rangle$，则 P 的行为模型 M_p 迁移(s,a,t)的逻辑值$\langle U',V'\rangle$有以下定义：

$$U'=\begin{cases}P, & \text{当 } U\cap P\neq\varnothing\\ \varnothing, & \text{当 } U\cap P=\varnothing\end{cases}$$

$$V'=\begin{cases}P, & \text{当 } V\cap P\neq\varnothing\\ \varnothing, & \text{当 } V\cap P=\varnothing\end{cases}$$

于是，M_F 中的迁移在 M_p 中可能出现的逻辑值可分为 4 种：$\langle P,P\rangle$、$\langle P,\varnothing\rangle$、$\langle\varnothing,P\rangle$和$\langle\varnothing,\varnothing\rangle$。其中，$\langle P,\varnothing\rangle$和$\langle\varnothing,P\rangle$分别表示该迁移一定存在和不存在于 M_p 中。$\langle P,P\rangle$反映了产品 P 中出现了不一致情况，即产品 P 存在互斥的特征，该产品非有效 (invalid)，因此我们在投影的定义中不作考虑该情况。$\langle\varnothing,\varnothing\rangle$表示产品 P 中的特征与该迁移的关系并不确定，即既不需要也不排斥该迁移。由于产品线中不存在两个特征对产品线中每一条迁移持相同的态度，因此该情况下我们将这些迁移的逻辑值定义为$\langle\varnothing,P\rangle$，使它们不出现在 M_p 中，以保证产品 P 不包含除 P 中以外的特征。投影的完整定义如下。

定义 6　设一个产品线 F 的行为模型 BFTS $M_1=(S,Act,s_0,\rightarrow,B_W,R)$，若 P 是 F 的一个产品，则产品 P 的行为模型 BFTS $M_2=(S',Act',s_0,\rightarrow',B_P,R')$由 M_1 在 P 上投影得到，其定义如下：

1)$\rightarrow'=\{(s,a,t)\mid(s,a,t)\in\rightarrow\wedge\exists f\in\pi_t(R(s,a,t))\cdot f\in P\}$

2)$R'(s,a,t)=\begin{cases}\langle P,\varnothing\rangle, & \text{当}\exists f\in\pi_t(R(s,a,t))\cdot f\in P\\ \langle\varnothing,P\rangle, & \text{其他}\end{cases}$

3)$S'=\{s\mid\exists t\cdot\exists a\cdot((s,a,t)\in\rightarrow'\vee(t,a,s)\in\rightarrow')\}$

4)$Act'=\{a\mid\exists s\cdot\exists t\cdot(s,a,t)\in\rightarrow'\}$

5)$B_P=\langle P(P)\times P(P),\leqslant_i,\leqslant_t,\rightarrow\rangle$为 B_W 的一个子双格

例 3　我们以第 2 节提到的饮料机产品线为例来理解投影上迁移的逻辑值的计算。若产品 $P=\{v,b,c,r\}$，根据投影的定义，产品 P 的行为模型中的 $R'(s_1,pay,s_2)$、$R'(s_3,cappuccino,s_7)$和 $R'(s_1,coffee,s_2)$的值可按如下计算：

1. 计算 $R'(s_0,pay,s_1)$

因为有 $R(s_0,pay,s_1)=\langle\{v\},\varnothing\rangle$，且 $v\in\pi_t(R(s_0,pay,s_1))$，则 $R'(s_0,pay,s_1)=\langle\{v,b,c,r\},\varnothing\rangle$。

2. 计算 $R'(s_2,cappuccino,s_7)$

因为有 $R(s_2,cappuccino,s_6)=\langle\{o\},\{f\}\rangle$，且 $\pi_t(R(s_2,cappuccino,s_6))\wedge P=\varnothing$，则 $R'(s_2,cappuccino,s_6)=\langle\varnothing,\{v,b,c,r\}\rangle$。

3. 计算 $R'(s_1,coffee,s_2)$

因为有$(s_1,coffee,s_2)\notin\rightarrow$，即 $R(s_1,coffee,s_2)=\langle\varnothing,W\rangle$，则 $R'(s_1,coffee,s_2)=\langle\varnothing,\{v,b,c,r\}\rangle$。

最终，产品 P 的行为模型如图 3 所示。

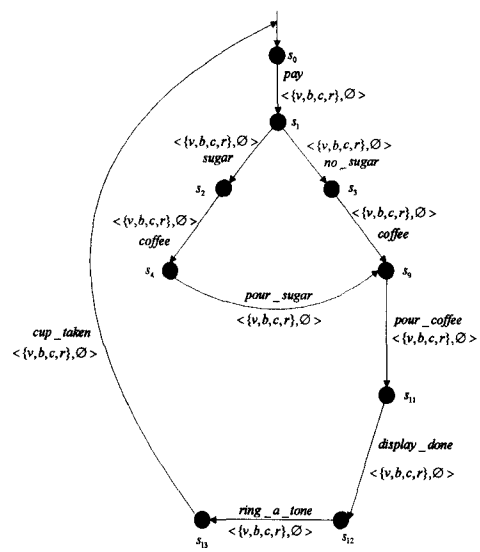

图 3　M 在产品 P 上的投影

BFTS 通过支持特征与迁移之间关系的不确定情况，以保证得到产品的正确行为模型。如图 2 中的饮料机产品线模型，迁移$(s_3,coffee,s_5)$的逻辑值为$(\{c\},\varnothing)$，表示该迁移只与特征 c 有关，而与其他特征的关系并不确定。设饮料机的产品 $P_1=\{c,b,v\}$，$P_2=\{v,b,t\}$。根据投影定义，与迁移$(s_3,coffee,s_5)$关系不确定的特征 b 和特征 v 在产品 P_1 的行为模型中确定为需要关系而在产品 P_2 的行为模型中为排斥关系；否则，若特征 b、特征 v 和迁移$(s_3,coffee,s_5)$的关系在 P_1 的行为模型中与在 P_2 的行为模型中的一样，都确定为排斥关系，这将导致特征 b、特征 v 与特征 c 互斥，造成产品 P_1 为非有效。反过来，若特征 b、特征 v 和迁移$(s_3,coffee,s_5)$的关系在 P_2 的行为模型也确定为需要关系，这将可能引起产品 P_2 中包含除特征 v、特征 t、特征 b 之外的其他特征，而这些特征根据产品的定义是禁止出现在产品中的。

4　软件产品线模型检测

多值模型检测的输入通常为一个多值状态迁移系统和一个用于描述系统属性的时序逻辑公式，其检测结果采用相应

的多值逻辑值来表示时序逻辑公式在系统上被满足的程度[14]。本文选择 ACTL 作为软件产品线模型检测的时序逻辑属性公式。由于传统的 ACTL 语义是定义在二值上的，并不适用于多值模型[15] BFTS。于是，我们定义 ACTL 在 BFTS 上的语义，其定义如下。

定义 7 设 $M=(S,Act,\rightarrow,s_0,B_W,R)$ 是一个 BFTS 模型，$s\in S$，ACTL 公式 φ 的语义 $\|\varphi\|$ 是一个映射 $S\rightarrow B_W$，根据公式的结构，归纳定义如下：

$$\|\text{true}\|=\lambda s.\langle W,\emptyset\rangle$$
$$\|\text{false}\|=\lambda s.\langle\emptyset,W\rangle$$
$$\|\neg\varphi\|=\lambda s.\neg\|\varphi\|(s)$$
$$\|\varphi\vee\varphi'\|=\lambda s.\|\varphi\|(s)\vee\|\varphi'\|(s)$$
$$\|\varphi\wedge\varphi'\|=\lambda s.\|\varphi\|(s)\wedge\|\varphi'\|(s)$$
$$\|\langle a\rangle\varphi\|=\lambda s.\bigvee_{t\in S}R(s,a,t)\wedge\|\varphi\|(t)$$
$$\|[a]\varphi\|=\lambda s.\bigwedge_{t\in S}\neg R(s,a,t)\vee\|\varphi\|(t)$$
$$\|AF\varphi\|=\mu Q.\|\varphi\|\vee(\bigwedge_{a\in Act}\|[a]Q\|)$$
$$\|EF\varphi\|=\mu Q.\|\varphi\|\vee(\bigvee_{a\in Act}\|\langle a\rangle Q\|)$$
$$\|AG\varphi\|=\nu Q.\|\varphi\|\wedge(\bigwedge_{a\in Act}\|[a]Q\|)$$
$$\|EG\varphi\|=\nu Q.\|\varphi\|\wedge(\bigvee_{a\in Act}\|\langle a\rangle Q\|)$$
$$\|A[\varphi U\varphi']\|=\mu Q.\|\varphi'\|\vee(\|\varphi\|\wedge(\bigwedge_{a\in Act}\|[a]Q\|))$$
$$\|E[\varphi U\varphi']\|=\mu Q.\|\varphi'\|\vee(\|\varphi\|\wedge(\bigwedge_{a\in Act}\|\langle a\rangle Q\|))$$

其中，符号 μ 和 ν 分别为最小不动点和最大不动点算子。

对于饮料机产品线，若想描述属性“在显示‘结束’后，系统会响铃”，可用 ACTL 公式描述为 $\|\langle display\_done\rangle\langle ring\_a\_tone\rangle\text{true}\|(s_{11})$。计算过程如下：

$$\begin{aligned}&\|\langle display\_done\rangle\langle ring\_a\_tone\rangle\text{true}\|(s_{11})\\&=R(s_{11},display\_done,s_{12})\wedge\|\langle ring\_a\_tone\rangle true\|(s_{12})\\&=R(s_{11},display\_done,s_{12})\wedge R(s_{12},ring\_a\_tone,s_{13})\\&=\langle\{r\},\{f\}\rangle\wedge\langle\{r\},\{f\}\rangle\\&=\langle\{r\},\{f\}\rangle\end{aligned}$$

对于 BFTS 上的迁移，产品线的行为模型包含的迁移是所有特征依赖的迁移的总和。产品的行为模型所包含的迁移是产品中特征依赖的迁移的总和。由于产品的特征集合是产品线特征集合的子集，产品线的行为模型所包含的迁移是所有产品所包含迁移的并集。

由此，若某属性在产品线模型上的检测的结果为 $\langle U,V\rangle$，一方面则表示 U 中的特征支持该属性，V 中的特征排斥该属性；另一方面，不考虑产品中出现的不一致情况，若产品中包含 U 中的某一特征，则该产品满足属性。若产品中包含 V 中某一特征，则该产品不满足属性。

5 实验设计与结果分析

本节主要以第 2 节提到的饮料机为例来展示模型检测方法的实验结果。

本文采用多值模型检测器 χchek 进行初步试验分析。χchek 是由加拿大多伦多大学(University of Toronto)利用 JAVA 开发的一款模型检测工具，可以支持多值模型上的模型检测，并能生成反例。我们运用 χchek 对饮料机产品线进行模型检测，属性及结果如表 1 所列。

表 1 实验设计属性及 χchek 上的计算结果 $\|\varphi\|^{BFTS}(s)$

公式 φ	验证结果
1. $\|\langle display\_done\rangle\langle ring\_a\_tone\rangle\text{true}\|(s_{11})$	$\langle\{r\},\{f\}\rangle$
2. $\|\neg(\bigvee_{act\in Act\setminus\{coffee\}}\langle act\rangle\text{true})\wedge(\langle coffee\rangle\text{true})\|(s_2)$	$\langle\emptyset,\{o,t\}\rangle$
3. $\|AF\langle pay\rangle\text{true}\|(s_0)$	$\langle\{v\},\emptyset\rangle$
4. $\|AF\langle cup\_taken\rangle\text{true}\|(s_0)$	$\langle\{v\},\emptyset\rangle$
5. $\|EF(\langle sugar\rangle\text{true}\wedge AF\langle cup\_taken\rangle\text{true})\|(s_0)$	$\langle\{v\},\emptyset\rangle$

表 1 中，式 1 表达的属性是系统处于状态 s_{11} 时，显示“结束”之后响铃。计算结果表明，饮料机产品线上的产品中，包含特征 r 并且不包含特征 f 的产品满足该属性，即产品 $\{v,b,c,r\}$，$\{v,b,t,r\}$，$\{v,b,o,r\}$。反过来，与该属性相关的特征是特征 r 和特征 f。若想产品满足这一属性，则产品中必须包含特征 r 且不能包含特征 f。同理，式 2 的计算结果表明，在饮料机产品线上，不包含特征 o 和特征 t 的产品满足该属性以及与该属性相关的特征是特征 o 和特征 t。若想满足这一属性，则产品中不能包含特征 o 和特征 t。

采用 χchek 验证式 5 描述的属性，其运行界面如图 4 所示。由于 χchek 中验证的属性采用 χCTL 公式[16]来描述，在 BFTS 模型检测中用以描述系统属性的 ACTL 公式需要转换成与其语义等价的 χCTL 公式。式 5 在语义上等价于 χCTL 公式 $EF(sugar\wedge AFcup\_taken)$。

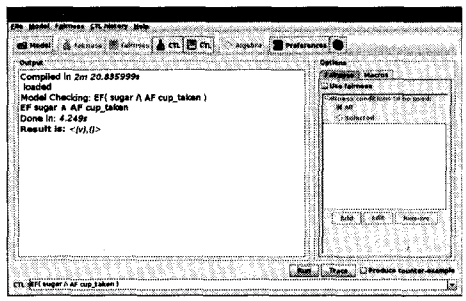

图 4 χchek 上的模型检测

6 相关工作

本节主要介绍国内外相关软件产品线建模与验证的研究。

在软件产品线建模的研究中，对可变性的建模研究越来越广泛。文献[8]选取近 20 年发表的论文进行系统综述，对可变性建模方法的分类和相关技术等展开深入探讨。文献[17]针对软件产品线的可变性建模提出一种代数规约，扩展了含有可变性的进程代数 CCS 以应用到软件产品线的建模中。

可变性建模方法主要包括两类，即特征建模(Feature Modeling)和行为建模(Behavioral Modeling)。特征建模主要侧重于描述软件产品线中的特征之间的层次关系。文献[18]总结了目前存在的众多特征建模方法，如 FODA、FORM、FeatuRSEBD 等。文献[19]利用特征模型就单个客户对产品的观点进行建模，并考虑到不同客户对产品要求的不一致性，采取暂时保留不同意见的方法，以保证产品线中存在满足不同客户要求的产品。文献[20]基于可变性的管理策略，提出了以扩展的用况模型和特征模型为表现形式的变化性建模；行为建模更倾向于描述产品线中的所有行为。文献[10]提出的广义扩展模态迁移模型(Generalized Extended Modal

Transition Systems)，利用了多值模型的特点，提高了行为建模的描述能力。

以上提到的文献只提出了软件产品线的建模方法，并没有给出软件产品线的验证方法。在软件产品线的模型检测研究方面，文献[21]基于 UML 的建模方法，通过扩展已有的 UML 类型(状态机、时序图等)来实现软件产品线上的行为建模。然而，其模型检测必须在得到单个产品的模型之后才能完成检测，这将导致随着产品数量的增多检测效率大大降低。文献[22]同样是基于 UML 的模型检测方法实现了通过产品线的模型得到单个产品的模型，但该模型检测并不能验证时序属性。

文献[5]采用行为与特征相结合的可变性建模方式，提出 FTS，并开发了相应的验证工具 SNIP[23] 和 ProveLines[24]。但该建模方式中，每个行为只能和一个特征相关联，导致其不能很好地支持软件产品线中信息不确定和不一致的情况。多值模型是传统布尔模型的扩展，与布尔模型相比，多值模型更适合对包含不确定和不一致信息的软件系统进行建模[25]。本文将 FTS 扩展到多值逻辑上，提出基于 BFTS 多值模型对软件产品线进行建模的方法，进而将特征与行为的关系从二值变为多值，解决了对软件产品线中出现的信息不确定和不一致情况的描述。

结束语 软件产品线的建模中存在不确定和不一致信息，本文利用多值模型可以有效地描述不确定和不一致信息的特性，给出基于 BFTS 多值模型的对软件产品线进行建模的方法，进而将特征与行为的关系从二值变为多值，解决了软件产品线中出现的信息不确定和不一致问题。进一步，通过将特征与迁移的关系从二值变为多值，提高了验证结果的精确度。

由于篇幅原因，本文基于双格的特征迁移系统中集合 W 单指特征集合，早期工作中将客户的观点(view)作为世界集合的研究在本文中并未涉及，实验部分运用 χchek 完成 BFTS 上的模型检测，模型和公式的转换方法也未详细说明。与传统二值模型一样，多值模型也面临着状态爆炸的问题。在后续工作中，我们将深入研究对于状态空间爆炸问题的处理。

参 考 文 献

[1] Clements P, Northrop L. Software Product Lines: Practices and Patterns [M]. Addision-Wesley Professional, 2001

[2] Baier C, Katoen J P. Principles of model checking [M]. Cambridge: MIT press, 2008

[3] Kang K C, Cohen S G, Hess J A, et al. Feature-oriented domain analysis (FODA) feasibility study[R]. Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, 1990

[4] Huth M, Jagadeesan R, Schmidt D. Modal transition systems: A foundation for three-valued program analysis[M]// Programming Languages and Systems. Springer Berlin Heidelberg, 2001: 155-169

[5] Classen A, Heymans P, Schobbens P Y, et al. Model checking lots of systems: efficient verification of temporal properties in software product lines[C]// Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1. ACM, 2010: 335-344

[6] De Nicola R, Vaandrager F. Action versus state based logics for transition systems[M]// Semantics of Systems of Concurrent Processes. Springer Berlin Heidelberg, 1990: 407-419

[7] Easterbrook S, Chechik M, Devereux B, et al. χChek: A model checker for multi-valued reasoning[C]// Proceedings of the 25th International Conference on Software Engineering. IEEE Computer Society, 2003: 804-805

[8] 聂坤明，张莉，樊志强. 软件产品线可变性建模技术系统综述[J]. 软件学报，2013，24(9)：2002-2018

[9] Apel S, Batory D, Kästner C, et al. Feature-Oriented Software Product Lines[M]. Springer, 2013

[10] Fantechi A, Gnesi S. Formal modeling for product families engineering[C]// 12th International Software Product Line Conference, 2008(SPLC'08). IEEE, 2008: 193-202

[11] Ginsberg M L. Multivalued logics: A uniform approach to reasoning in artificial intelligence[J]. Computational intelligence, 1988, 4(3): 265-316

[12] Clarke E M, Emerson E A. Design and synthesis of synchronization skeletons using branching time temporal logic[M]. Springer Berlin Heidelberg, 1982

[13] Pnueli A. The temporal logic of programs [C]// 18th Annual Symposium on Foundations of Computer Science, 1977. IEEE, 1977: 46-57

[14] 魏欧，袁泳，蔡昕烨，等. 循环对称化简及在三值模型上的扩展[J]. 软件学报，2011，22(6)：1169-1184

[15] Shoham S, Grumberg O. Multi-valued model checking games [M]// Automated Technology for Verification and Analysis. Springer Berlin Heidelberg, 2005: 354-369

[16] Chechik M, Devereux B, Easterbrook S, et al. Multi-valued symbolic model-checking[J]. ACM Transactions on Software Engineering and Methodology (TOSEM), 2003, 12(4): 371-408

[17] Leucker M, Thoma D. A formal approach to software product families[M]// Leveraging Applications of Formal Methods, Verification and Validation. Technologies for Mastering Change. Springer Berlin Heidelberg, 2012: 131-145

[18] Schobbens P, Heymans P, Trigaux J C. Feature diagrams: A survey and a formal semantics[C]// 14th IEEE International Requirements Engineering Conference. IEEE, 2006: 139-148

[19] Niu N, Savolainen J, Yu Y. Variability modeling for product line viewpoints integration[C]// 2010 IEEE 34th Annual Computer Software and Applications Conference (COMPSAC). IEEE. 2010: 337-346

[20] 邹盛享，张伟，赵海燕，等. 面向软件产品家族的变化性建模方法[J]. 软件学报，2005，16(1)：37-49

[21] Ziadi T, Hélouët L, Jézéquel J M. Towards a UML profile for software product lines[M]// Software Product-Family Engineering. Springer Berlin Heidelberg, 2004: 129-139

[22] Czarnecki K, Antkiewicz M. Mapping features to models: A template approach based on superimposed variants[C]// Generative Programming and Component Engineering. Springer Berlin Heidelberg, 2005: 422-437

[23] Classen A, Cordy M, Heymans P, et al. Model checking software product lines with SNIP[J]. International Journal on Software Tools for Technology Transfer, 2012, 14(5): 589-612

[24] Cordy M, Classen A, Heymans P, et al. ProVeLines: a product line of verifiers for software product lines[C]// Proceedings of the 17th International Software Product Line Conference co-located workshops. ACM, 2013: 141-146

[25] 陈娟娟，魏欧，黄志球，等. 基于双格的多值模型的精化关系与对称化简[J]. 计算机工程与应用，2013，49(22)：40-45