采用动态描述逻辑实现 UML 状态图特性检测*)

刘亚军　康建初　吕卫锋

(北京航空航天大学计算机学院软件开发环境国家重点实验室　北京 100083)

摘　要　状态图是 UML(Unified Modeling Language)语言中刻画对象行为的重要视图,而如何对状态图模型定义的正确性和有效性进行检验一直是一个亟待解决的问题。本文提出采用动态描述逻辑 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 对 UML 状态图形式化,并利用该逻辑系统的推理能力对状态图相关静态和动态特性进行检测。我们首先将状态图描述为一个 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 形式系统。其中,状态图中的一个状态对应于该形式系统中的一个状态,状态特性及描述被表示为该形式系统中的概念和公理,事件被表示为该形式系统中的动作。然后,我们通过 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 概念测试来检验状态图状态可满足性和冗余性,通过公式可满足性测试来验证状态转移引起的对象特性变化。

关键词　UML,状态图,特性检测,动态描述逻辑

Test UML Statechart Properties by Using Dynamic Description Logic

LIU Ya-Jun　KANG Jian-Chu　LV Wei-Feng

(National Lab of Software Development Environment, School of Computer Science & Technology, Beihang University, Beijing 100083)

Abstract　The problem of how to detect the correctness and validity of a UML (Unified Modeling Language) statechart, an important view for object behaviour description, still remains unsolved up to now. In the paper, we suggest a solution by first translating the UML statechart into a dynamic description logic $\mathcal{DPDL}$-$\mathcal{ALC2I}$ system, and then testing its static and dynamic properties by utilizing the reasoning capabilities of this system. In the translation, a state in the UML statechart is denoted by a state in the $\mathcal{DPDL}$-$\mathcal{ALC2I}$ system, state properties and their descriptions are represented by concepts and axioms, events are represented by actions. Then, state satisfiability and redundancy are detected through $\mathcal{DPDL}$-$\mathcal{ALC2I}$ concept tests, and object property changes caused by state transitions are verified through formula satisfiability tests.

Keywords　UML, Statechart, Property test, Dynamic description logic

1　引言

UML(Unified Modeling Language)是目前在软件设计与分析领域应用最为广泛的面向对象标准化建模语言。然而,由于 UML 缺乏精确和足够形式化的语义,使得 CASE 工具对该语言的支持只能限制在语法正确性检查层次上,而无法对所定义模型的正确性和一致性进行严格分析和检验。这可能会导致一个含有设计错误的模型在软件过程中被进一步使用,问题严重时甚至会造成整个软件项目的失败。

为赋予 UML 以精确和形式化的语义,众多的研究者付出了努力。其中的典型代表是由 Precise UML Group 组织启动的一个名为 Precise UML 的研究计划[1]。该计划采用一种基于一阶逻辑的形式语言 Z 来刻画 UML 的语义,并通过定理证明器来分析和验证 UML 静态模型的形式特性。然而,由于一阶逻辑的不可判定性,我们无法据此开发出一个实用的模型分析工具。直到最近,Daniela Berardi 等人采用描述逻辑对 UML 类图形式化,并利用描述逻辑的可判定推理能力对 UML 类图的形式特性(一致性、冗余性等)进行分析和检验而解决了这个问题[2]。他们的研究表明,在不考虑 n 元关系、识别约束和函数依赖约束等特性的情况下,UML 类图可以被描述逻辑 $\mathcal{ALC2I}$ 形式化,并且其语义能够保证类图形式特性推理的可靠性和完全性。他们同时证明了 UML 类图推理的复杂性是指数时间完全(EXPTIME-Complete)的。目前有很多支持 $\mathcal{ALC2I}$ 的成熟推理工具,例如 FACT、RACER 等。

本文以 Daniela Berardi 等人的研究为基础,进一步研究采用动态描述逻辑对 UML 中另一个用于刻画对象行为的重要视图——状态图的形式特性进行检验。本文采用的动态描述逻辑是描述逻辑 $\mathcal{ALC2I}$ 和确定命题动态逻辑 DPDL(Deterministic Propositional Dynamic Logic)的结合,称为 $\mathcal{DPDL}$-$\mathcal{ALC2I}$。更准确地说,$\mathcal{DPDL}$-$\mathcal{ALC2I}$ 是对 DPDL 命题逻辑部分的 $\mathcal{ALC2I}$ 语言扩展,其中动态算子应用于公式。对于一个 UML 状态图,我们首先将其描述为一个 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 形式系统。其中,状态图中的一个状态对应于该形式系统中的一个状态,状态特性及描述分别被表示为该形式系统中的概念和公理,引起状态转移的事件被表示为该形式系统中的动作。然后,我们通过 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 概念测试来检验状态图状态可满足性和冗余性(静态特性),通过公式可满足性测试来验证状态转移引起的对象特性变化(动态特性)。

2　动态描述逻辑 $\mathcal{DPDL}$-$\mathcal{ALC2I}$

2.1　语法和语义

*)国家"973"重点基础研究发展规划基金项目(2005CB321905)。刘亚军　博士研究生,研究方向为软件集成、网络管理;康建初　教授,研究方向为人工智能、软件体系结构、中间件技术;吕卫锋　副教授,研究方向为网络管理、下一代网络。

定义 1(符号集) $\mathcal{DPDL}$-$\mathcal{ALC2I}$中的原始符号包括：

- 概念名 $C_0, C_1, \cdots$；
- 角色名 $R_0, R_1, \cdots$；
- 对象名 $a_0, a_1, \cdots$；
- 概念构造子 $\cap, \cup, \neg, \forall R_i.C_i, \exists R_i.C_i, \geqslant nR_i.C_i, \leqslant nR_i.C_i$，其中 R_i 代表任一角色名，C_i 代表任一概念名，n 代表正整数；
- 角色构造子 R_i^-，R_i 代表任一角色名；
- 布尔运算符 $\wedge, \neg$；
- 动作变元 $\alpha_0, \alpha_1, \cdots$；
- 动作构造子；(联接)，$\cup$(选择)，$*$(重复)。

定义 2(概念、角色、动作项、公式) $\mathcal{DPDL}$-$\mathcal{ALC2I}$中，每一个概念名以及$\top$(顶)和$\bot$(底)是一个原子概念，每一个角色名是一个原子角色，每一个动作变元是一个原子动作。设 C 和 D 为概念，R 为角色，a 和 b 为对象，φ 和 ψ 为公式，α 和 β 为动作项，则有归纳定义：

- $\neg C, C\cap D, C\cup D, \forall R.C, \exists R.C, \geqslant nR.C, \leqslant nR.C$ 是概念；
- R^- 是角色；
- $\alpha;\beta, \alpha\cup\beta, \alpha^*$ 是动作项；
- $a:C, aRb, \varphi\wedge\psi, \neg\varphi, [\alpha]\varphi, \langle\alpha\rangle\varphi$ 是公式(前两个为原子公式，即命题)。

$\mathcal{DPDL}$-$\mathcal{ALC2I}$语言的静态描述部分(概念、角色、对象)是标准描述逻辑语言 $\mathcal{ALC2I}$，其语义(*Tarski* 型)模型定义为解释

$$I=(\Delta, R_0^I, \cdots, C_0^I, \cdots, a_0^I, \cdots)$$

其中：Δ 为一个对象集合，代表论域；R_i^I 将角色名 R_i 解释为一个 Δ 上的二元关系；C_i^I 将概念名 C_i 解释为 Δ 的一个子集；a_i^I 将对象名 a_i 解释为 Δ 中的一个元素。有关 $\mathcal{ALC2I}$ 其它语构的语义解释请参见文[2]。

另外，一个 $\mathcal{ALC2I}$ 知识库 $\mathcal{K}=(\mathcal{T}, \mathcal{A})$ 由两部分组成：*TBox* $\mathcal{T}$ 和 *ABox* $\mathcal{A}$。*TBox* 是一个公理有穷集合。公理的形式为"$C\subseteq D$"，含义是 C 包容于 D，其中 C 和 D 为任意概念。此外，我们把"$C\equiv D$"看为"$C\subseteq D$ 且 $D\subseteq C$"的缩写。*ABox* 是一个实例断言有穷集合。断言有两种形式，"$C(a)$"和"$R(b, c)$"。前者也可写为"$a\in C$"，含义是对象 a 是概念 C 的一个实例。后者也可写为"bRc"，含义是对象 b 到对象 c 存在关系 R。

如果一个解释 I 满足 $\mathcal{T}$ 中的所有公理和 $\mathcal{A}$ 中的所有断言，则称该解释是知识库 $\mathcal{K}$ 的一个模型。

$\mathcal{DPDL}$-$\mathcal{ALC2I}$语言的动态描述部分(动作、公式)是确定命题动态逻辑语言 DPDL(详见文[3])，其语义(Kripke 型)模型框架形如

$$\xi=(W, T_{\alpha_0}, T_{\alpha_1}, \cdots)$$

其中 W 是一个非空状态集合，T_{α_i} 是一个 W 的上的二元关系，刻画由动作变元 α_i 引起的状态转移。由于 DPDL 具有确定性，因此对于每个状态 $w\in W$ 和每个动作变元 α_i，只可能有一个元组 $(w, w')\in T_{\alpha_i}$，其中 $w'\in W$。

通过将 $\mathcal{DPDL}$ 模型框架与 $\mathcal{ALC2I}$ 解释集合结合，我们定义 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 的语义模型如下：

定义 3(模型) 一个基于框架 $\xi=(W, T_{\alpha_0}, T_{\alpha_1}, \cdots)$ 的 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 模型是一个二元组 $M=(\xi, Z)$，其中 Z 是一个从状态到 $\mathcal{ALC2I}$ 解释集合的函数，为每个状态 $w\in W$ 关联一个 $\mathcal{ALC2I}$ 解释集合 $Z(w)$，而对于任一解释 $I\in Z(w)$ 有 $I=(\Delta, R_0^I, \cdots, C_0^I, \cdots, a_0^I, \cdots)$。

同时，我们规定在上述所有 $\mathcal{ALC2I}$ 解释之间(包括解释集合内部和集合之间)遵循"共同论域假设"和"严格项假设"。前者规定所有解释的论域是相同的，后者则规定在所有解释之间，相同的对象名对应于同一个论域元素，不同的对象名对应于不同论域元素。

定义 4(满足性) 给定一个 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 模型 $M=(\xi, Z)$(其中 $\xi=(W, T_{\alpha_0}, T_{\alpha_1}\cdots)$)和一个状态 $w\in W$，我们有

- $(M,w)\models C\subseteq D$ iff $\forall I\in Z(w)(C^I\subseteq D^I)$
- $(M,w)\models a:C$ iff $\forall I\in Z(w)(a^I\in C^I)$
- $(M,w)\models aRb$ iff $\forall I\in Z(w)(a^IR^Ib^I)$
- $(M,w)\models\varphi\wedge\psi$ iff $(M,w)\models\varphi\wedge(M,w)\models\psi$
- $(M,w)\models\neg\varphi$ iff $(M,w)\not\models\varphi$
- $(M,w)\models[\alpha]\varphi$ iff $\forall v\in W(wT_\alpha v\rightarrow(M,v)\models\varphi)$
- $(M,w)\models\langle\alpha\rangle\varphi$ iff $\exists v\in W(wT\alpha v\wedge(M,v)\models\varphi)$
- $T_{\alpha;\beta}=T_\alpha\circ T_\beta$($T_\alpha$ 和 T_β 的联接)
- $T_{\alpha\cup\beta}=T_\alpha\cup T_\beta$
- $T_{\alpha^*}=T_\alpha^*$(T_α 的自反传递闭包)

当 M 不言自明时，我们将 $(M,w)\models\varphi$ 简写为 $w\models\varphi$。另外，由于 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 的确定性，我们有：如果 $w\models\langle\alpha\rangle\varphi$，则 $w\models[\alpha]\varphi$。

2.2 公式可满足性

如果存在一个 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 模型 M 和一个 M 中的状态 w 使得 $w\models\varphi$，则称公式 φ 是可满足的。如果公式 φ 对于 M 中的所有状态都是可满足的，则称 φ 是 M 中的公理。

公式可满足性的判定问题是 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 最基本的判定问题。

由定义 2 可知，任一 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 公式都是由原子公式(命题)通过布尔运算符($\wedge, \neg$)和动态算子($[\alpha], \langle\alpha\rangle$)构造而成的。设 $atomic(\varphi)$ 为由公式 φ 中出现的所有原子公式组成的集合。给定一个框架 $\xi=(W, T_{\alpha_0}, T_{\alpha_1}, \cdots)$，我们定义 m 为一个从 $atomic(\varphi)$ 到 W 的映射 $m: atomic(\varphi)\rightarrow W$，即对于每个 $p\in atomic(\varphi)$ 有 $m(p)\subseteq W$，其中 $m(p)$ 是由所有满足 p 的状态组成的集合，也就是对于所有的 $w\in m(p)$ 有 $w\models p$。由此，我们就针对公式 φ 构造了一个标准的 DPDL 形式系统 $\Gamma=(\xi, m)$，进而 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 公式可满足性问题就转化为了 DPDL 的公式可满足性问题。而我们知道，DPDL 的公式可满足性问题是可判定的且复杂性是指数时间完全的(相关证明和算法请参见文[3,4])。因此，由以上讨论有

定理 5 给定一个 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 模型 $M=(\xi, Z)$，其中 $\xi=(W, T_{\alpha_0}, T_{\alpha_1}, \cdots)$，"公式 φ 的可满足性是可判定的"当且仅当"对于每个 $p\in atomic(\varphi)$ 和所有的 $w\in W$，$w\models p$ 是可判定的"。

定理 5 提示我们，要保证 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 公式满足性的可判定性，必须合理地定义每个解释集合 $Z(w)$ 以确保 $w\models p$ 是可判定的。

3 状态图的 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 描述

UML 状态图采用有限状态机技术描述一个特定对象的所有可能状态以及事件发生时状态的转移过程。很多情况下，状态图用于描述一个对象在其生命周期中的行为。

状态图最基本的语构包括状态、事件和转移等。由于本文的研究重点是状态图的核心特性——状态、转移以及它们之间关系，因此我们的研究对象也限定于含有简单状态、事件

和转移等基本语构的 UML 状态图，而对活动、动作、护卫条件、伪状态等其它语构暂未涉及。图 1 就是一个基本状态图的示例，其中(a)是一个 UML 类图，定义了两个类“Person”、“Job”以及它们之间的关联，而(b)是类“Person”的对象关于就业状况的一个状态图。

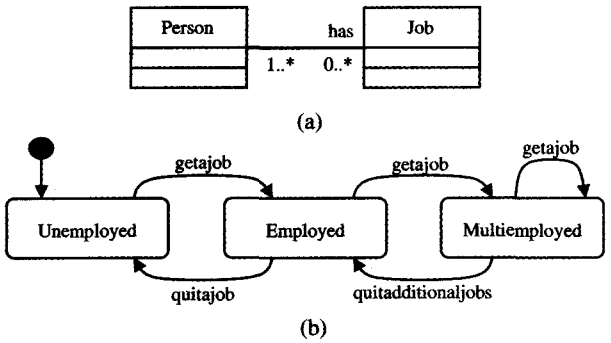

图 1　一个关于就业状况的状态图

对状态图进行 $\mathcal{DPDL}$-$\mathcal{ALCQI}$ 描述，就是要构造对应于该状态图的 $\mathcal{DPDL}$-$\mathcal{ALCQI}$ 模型 $M=(\xi,Z)$。

状态图中，状态代表对象在生命周期中一个有限的时间段内存在的一种条件或状况，事件是在时间和空间域中具有一个定位的发生事件，转移是两个状态之间的关系，表示如果规定的事件发生，则处于第一个状态的对象将进入第二个状态。对于一个给定的状态和同一个事件，最终只能产生一个转移[5]。

这样，通过把一个事件看为一个原子动作，我们很自然地定义对应于一个状态图的 DPDL 框架为 $\xi=(W,T_{a_0},T_{a_1},\cdots)$，其中 W 是状态图中所有状态组成的集合，a_i 对应于状态图的一个事件，T_{a_i} 是一个 W 的上的二元关系，刻画由事件 a_i 引起的状态转移。

例如，对于图 1(b)所示状态图，我们有 $\xi=(W,T_{getajob},T_{quitajob},T_{quitadditionaljobs})$，其中 $W=\{S_{Unemployed},S_{Employed},S_{Multiemployed}\}$。

$T_{getajob}=\{(S_{Unemployed},S_{Employed}),(S_{Employed},S_{Multiemployed}),(S_{Multiemployed},S_{Multiemployed})\}$

$T_{quitajob}=\{(S_{Employed},S_{Unemployed})\}$，$T_{quitadditionaljobs}=\{(S_{Multiemployed},S_{Employed})\}$

下面，我们讨论如何构造函数 Z，即如何为每个 $w\in W$ 关联一个 ALCQI 解释集合。

状态代表对象所处的一种状况，在这种状况下，对象满足一组特性。依据 Daniela Berardi 等人的研究[2]，我们可以采用 $\mathcal{ALCQI}$ 语言对状态特性进行描述，即把对象表示为一个 $\mathcal{ALCQI}$ 对象，把一个状态特性表示为一个 $\mathcal{ALCQI}$ 概念。对象满足一个状态特性则被表示为对象满足相应的状态特性概念。特别地，我们可以通过 $\mathcal{ALCQI}$ 公理使用“$\cap$”构造子将对象处于某个状态应满足的所有状态特性概念构造为一个对应于这个状态的更复杂的 $\mathcal{ALCQI}$ 概念，我们称这样一个概念为状态概念。例如，我们可以定义处于“失业”状态的人为没有工作的人，即有公理 $Unemployed\equiv Person\cap\neg\exists has.Job$，其中 $Unemployed$ 为状态概念。考虑到一个没有工作的人可能是一个学生，而一般意义上我们不能说一个学生是一个“失业”者，因此上述公理可以改为 $Unemployed\subseteq Person\cap\neg\exists has.Job$，即处于“失业”状态的人属于没有工作的人。以图 1 状态图为例，(同时考虑类图)则有状态特性描述公理集合：

$$\begin{array}{l}\top\subseteq\forall has^{-}.Person\cap\forall has.Job\\ Person\subseteq(\geqslant 0has.\top)\\ Job\subseteq(\geqslant 1has^{-}.\top)\\ Unemployed\subseteq Person\cap\neg\exists has.Job\\ Employed\subseteq Person\cap\exists has.Job\cap\leqslant 1has.Job\\ Multiemployed\subseteq Person\cap\geqslant 2has.Job\end{array}$$

图 2　图 1 状态图的状态特性描述公理集合

前三条公理是根据 Daniela Berardi 等人在文[2]中所介绍采用 $\mathcal{ALCQI}$ 对 UML 类图形式化的方法得到。这样，可以在状态图的状态特性描述集合中引入在类图中定义的对象所属类的静态特性描述，从而保证在状态图特性检测中保持和类图定义的一致性。

我们定义上述的状态特性描述公理集合为 $\mathcal{T}$。注意到，状态特性描述是独立于状态的，不随对象所处状态的改变而改变。换句话说，对象处于任一状态，$\mathcal{T}$ 都要得到满足。

有了 $\mathcal{T}$ 中状态概念的定义，指定对象(不失一般性，设指定对象为 a)处于某个状态 w 可以用一个 $\mathcal{ALCQI}$ 实例断言 $C_w(a)$ 即 $a\in C_w$ 表示，其中 C_w 为状态 w 对应的状态概念。

我们定义 Z 是一个从状态到 $\mathcal{ALCQI}$ 解释集合的函数，其为框架 $\xi=(W,T_{a_0},T_{a_1},\cdots)$ 中每个状态 $w\in W$ 关联一个 $\mathcal{ALCQI}$ 解释集合 $Z(w)$，$Z(w)$ 是由 $\mathcal{ALCQI}$ 知识 $\mathcal{K}_w=(\mathcal{T},\{C_w(a)\})$ 库的所有模型组成的集合，其中 T 是状态图状态特性描述公理集合，C_w 是 T 中对应于状态 w 的状态概念。

以图 2 所示状态图为例，对于状态集合 $W=\{S_{Unemployed},S_{Enployed},S_{Multiemployed}\}$，我们有 $\mathcal{ALCQI}$ 解释集合 $Z(S_{Unemployed})$、$Z(S_{Employed})$、$Z(S_{Multiemployed})$，分别是 $\mathcal{ALCQI}$ 知识库、$\mathcal{K}_{S_{Unemployed}}=(\mathcal{T},\{Unemployed(a)\})$、$\mathcal{K}_{S_{Employed}}=(\mathcal{T},\{Employed(a)\})$、$\mathcal{K}_{S_{Multiemployed}}=(\mathcal{T},\{Multiemployed(a)\})$ 的模型组成的集合，其中 $\mathcal{T}$ 如图 2 所示。

语义上，$\mathcal{ALCQI}$ 知识库 $\mathcal{K}_w=(\mathcal{T},\{C_w(a)\})$ 的一个模型是一个既满足 $\mathcal{T}$ 又满足 $C_w(a)$ 的 $\mathcal{ALCQI}$ 解释，其论域是系统中的所有对象。因此，$\mathcal{K}_w=(\mathcal{T},\{C_w(a)\})$ 的一个模型代表了指定对象 a 处于状态时系统的一种可能状况(瞬象)。而依据“共同论域假设”和“严格项假设”，知识库 $\kappa_w=(\mathcal{T},\{C_w(a)\})$ 的所有模型就代表了指定对象 a 处于状态时系统的所有可能状况，这与状态图的语义定义[6]是一致的。

由此，我们给出了对应于一个状态图(及关联类图)的 $\mathcal{DPDL}$-$\mathcal{ALCQI}$ 模型 $M=(\xi,Z)$ 的完整构造。不难看出，上述 $\mathcal{T}$ 中的每个公理都是 M 中的公理。

4　状态图特性检测

4.1　静态特性检测

状态图的静态特性是指与状态转移无关的特性，其中最重要的两个特性包括：

• 状态可满足性。状态可满足性是指状态定义必须是可满足的。逻辑上，就是要求状态定义的逻辑描述是一个可满足式，而不能是一个含有逻辑矛盾的永假式，否则状态定义是无意义的。

• 状态冗余性。直观上，状态冗余性表示状态图中某个状态可以用另一个状态代替。逻辑上，状态冗余性则是指两个状态的逻辑描述之间存在蕴涵或等价关系。状态冗余并不意味着状态定义一定存在错误，而是指出存在可能性对状态

进行合并以使状态图更简化。

在上节构造模型 $M=\langle\xi,Z\rangle$ 过程中，我们定义了关于状态图状态特性描述的 $\mathcal{ALC2I}$ 公理集合 $\mathcal{T}$。因此，对于静态特性检测有以下命题：

命题 6(状态可满足性检测) 设 C 是 M 中对应于状态 s 的状态概念，如果 C 是关于 $\mathcal{T}$ 可满足的，则状态 s 是可满足的。

命题 7(状态冗余性检测) 设 C 和 D 是 M 中分别对应于状态 s_1 和 s_2 的状态概念，如果 C 关于 $\mathcal{T}$ 包容于(或等价于) D，则状态 s_1 是冗余的。

我们知道，$\mathcal{ALC2I}$ 关于 TBox 的概念可满足性、包容性、等价性测试的复杂性是指数时间完全的[2]。因而，上述检测是可判定的且复杂性是指数时间完全的。

4.2 动态特性检测

状态图的动态特性体现在对状态机的操作层面上。人们总是对这样的问题感兴趣(问题一)：已知对象处于某个状态，如果对其施以一个(正则表达的)事件序列，能否使该对象转移到一个目标状态，且在这个目标状态中对象满足某个特性呢？或者从另外一个角度提问(问题二)：已知对象处于某个状态，是否存在一个事件序列，使得该对象转移到一个目标状态，且在这个目标状态中对象满足某个特性呢？

首先讨论问题一。设状态图对应的 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 模型为 $M=(\xi,Z)$，其中 $\xi=(W,T_{a_0},T_{a_1},\cdots)$，指定对象为 a，a 当前所处的状态为 s，需要检测的目标特性概念为 G，τ 是由事件(即原子动作) a_i 依据构造子、$\cup$、$*$ 组成的一个事件序列，则问题一可以表示为对下式的验证：

$$s\models[\tau](a:G) \tag{1}$$

式(1)的含义是公式 $[\tau](a:G)$ 在状态 s 中是满足的。因此，问题一就转化为公式 $[\tau](a:G)$ 在 M 中的可满足性判定问题。而根据定理 5，公式 $[\tau](a:G)$ 的可满足性是可判定的当且仅当对于所有的 $w\in W$，$w\models a:G$ 是可判定的。下面讨论 $w\models a:G$ 的可判定性。

在 M 中，我们通过函数 Z 为每个状态 $w\in W$ 分别关联了一个由 $\mathcal{ALC2I}$ 知识库 $\mathcal{K}_w=(\mathcal{T},\{C_w(a)\})$ 的所有模型组成的集合 $Z(w)$，其中 $\mathcal{T}$ 是状态图状态特性描述公理集合，C_w 是 $\mathcal{T}$ 中定义的对应于状态 w 的状态概念。根据定义 4，有“$w\models a:G$ 当且仅当 $\forall I\in Z(w)(a^I\in C^I)$”。再根据 $Z(w)$ 的定义，有“$\forall I\in Z(w)(a^I\in C^I)$ 当且仅当 $\mathcal{K}_w=(\mathcal{T},\{C_w(a)\})\models G(a)$”。而“$\mathcal{K}_w=(\mathcal{T},\{C_w(a)\})\models G(a)$ 当且仅当概念 $C_w\sqcap\neg G$ 是关于 $\mathcal{T}$ 不可满足的”[7]。$w\models a:G$ 从而，的判定问题最终转化为了 $\mathcal{ALC2I}$ 概念关于 TBox 的可满足性测试问题，而我们知道后者是可判定且是指数时间完全的[2]，因此 $w\models a:G$ 是可判定的。

由此，我们可以得出结论：问题一是可判定的。

对于问题二，相应地可表示为对下式的验证：

$$s\models\langle(\alpha_0\cup\alpha_1\cup\cdots\cup\alpha_n)^*\rangle(a:G) \tag{2}$$

其中，$\alpha_0,\alpha_1,\cdots,\alpha_n$ 代表了所有事件(原子动作)，因此 $(\alpha_0\cup\alpha_1\cup\cdots\cup\alpha_n)^*$ 代表了由这些事件组成的所有可能的事件序列。同问题一一样，问题二也是可判定的。

根据以上讨论，我们有如下结论：通过为状态图构造的对应 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 形式系统，可以实现状态图动态特性检测(检测算法可利用 $\mathcal{ALC2I}$ 知识库概念测试算法和 DPDL 公式可满足性判定算法分步实现)。

5 相关工作

由于状态图描述的是对象过程性行为，以往关于状态图形式化的研究侧重于对其操作语义的刻画。Diego Latella 等人的做法是首先将状态图映射为一种“扩展层次自动机”的中间形式，然后为这些自动机定义操作语义[8]。Martin Gogolla 等人将状态图表示为图形转换系统，并基于图形转换规则描述状态图操作语义[9]。还有一些研究者尝试采用一些中间语言(例如 PVS[10]、PROMELA[11] 等)表示状态图，利用该语言的检查工具进行模型检验。上述研究各有特点，但存在一个共同的缺陷：对状态图形式化的最小粒度是状态，而无法刻画状态的内在特性。因此，在对如本文所列的一些状态图重要特性进行检测时显得无能为力。

结束语 本文研究了如何采用动态描述逻辑 $\mathcal{DPDL}$-$\mathcal{ALC2I}$ 对 UML 状态图进行形式化描述，并利用该逻辑系统的可判定推理能力实现状态图特性检测，而且在检测中能够保证状态图定义与类图定义的一致性。

本文的研究只是为 UML 状态图描述逻辑形式化的研究提供了一个基础，还有很大的补充和完善空间。比如，还有一些状态图语构未被纳入进来，这些需要我们在以后的研究中加以解决。

参考文献

1 Evans A, Bruel J-M, France R, et al. Making UML Precise. In: OOPSLA'98 Workshop on "Formalizing UML, Why and How?", October 1998

2 Berardi D, Calvanese D, De Giacomo G. Reasoning on UML class diagrams. Artificial Intelligence, 2005, 168(1-2): 70～118

3 Harel D, Kozen D, Tiuryn J. Dynamic logic. In: Gabbay D M, Guenthner F, eds. Handbook of Philosophical Logic, vol2. 497～604

4 Vardi M Y, Wolper P. Automata theoretic techniques for modal logics of programs. Journal of Computer and System Sciences, 1986, 32: 183～221

5 刘超，张莉. 可视化面向对象建模技术. 北京：北京航空航天大学出版社，1999. 85～89

6 Object Management Group. OMG Unified Modeling Language Specification, Version 1.5. formal/03-03-01, March 2003

7 Baader F, Nutt W. Chapter 2: Basic Description Logics. The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, 2003. 47～100

8 Latella D, Majzik I, Massink M. Towards a formal operational semantics of UML statechart diagrams. In: 3rd International Conference on Formal Methods for Open Object-oriented Distributed Systems, Boston, 1999

9 Gogolla M, Presicce F P. State Diagrams in UML - A Formal Semantics using Graph Transformation. In: Proceedings ICSE'98 Workshop on Precise Semantics of Modeling Techniques, 1998. 55～72

10 Aredo D B. Semantics of UML Statecharts in PVS. In: Proceedings of the 12th Nordic Workshop on Programming Theory, Norway, 2000

11 Lilius J, Paltor I P. The semantics of UML state machines: [Technical Report]. No. 273. Turku Centre for Computer Science, Finland, 1999