基于扩展描述逻辑和逻辑程序的事件动作形式化表示与推理

刘 炜[1] 徐文杰[1] 唐英英[1] 付剑锋[2] 张旭洁[1] 刘宗田[1]

(上海大学计算机工程与科学学院 上海 200072)[1] (上海立信会计学院数学与信息学院 上海 201620)[2]

摘 要 "事件"是比"概念"粒度更大的人类知识表示单元,更接近于人类的认知过程。动作作为事件的一个重要要素,表示事件中对象的状态的变化过程。在状态的变化过程中加入时间信息,将动作表示为对象的状态随时间变化而变化的过程,使得动作描述得更加具体。运用事件中的动作、对象和时间要素构建了一个动作形式化体系,研究了事件知识中确定性动作和不确定性动作的语法表示和语义解释。该形式化体系将扩展的带时间维的描述逻辑 T-ALC 和逻辑程序设计进行整合,增强了动作的表达能力;在动作的推理中,将确定性动作转化为逻辑程序 Datalog 规则实现动态推理,将不确定性动作转化为 Datalog¬ 规则实现不确定推理。最后通过银行服务系统实例对动作的形式化表示和推理进行了验证。

关键词 T-ALC,逻辑程序,事件动作,动作推理,不确定性推理

中图法分类号 TP391 **文献标识码** A

Formalized Representation and Reasoning of Event Action Based on Extended Description Logic and Logic Program

LIU Wei[1] XU Wen-jie[1] TANG Ying-ying[1] FU Jian-feng[2] ZHANG Xu-jie[1] LIU Zong-tian[1]

(School of Computer Engineering and Science, Shanghai University, Shanghai 200072, China)[1]

(School of Mathematics and Information, Shanghai Lixin University of Commerce, Shanghai 201620, China)[2]

Abstract Event is a human knowledge unit with larger-grained than concept. It is closer to the human cognitive process. As an important element of the event, action describes the process of status change of the objects in the event. Adding time to the process of status change, and describing action as the process of object's status change with time, lead to a more specific description of action. This paper constructed an action formalism through actions, objects and time in the event, and studieds the syntax and semantics of certainty action and uncertain action. This action formalism combines an extended description logic with time (T-ALC) with logic program, thus to enhance the expressive ability of action. In this action formalism, the certainty actions forms are transformed to Datalog rules based forms to realize dynamical reasoning, and uncertainty action forms are transformed to Datalog¬ rules based forms to realize uncertainty reasoning. At last, a case study of bank service was described to verify the feasibility of the action formalism and reasoning method.

Keywords T-ALC, Logic program, Event action, Action reasoning, Uncertainty reasoning

1 引言

近年来,"事件"概念逐渐被计算语言学、信息检索、信息抽取和自然语言处理等知识处理领域所采用[1-3]。而互联网上的大量文本信息也是以事件为单位进行描述的。因此,将事件概念运用到语义 Web 技术,实现计算机对互联网信息基于事件的语义理解显得非常自然[4]。在众多关于"事件"的研究中,其动作要素是必不可忽略的重要要素,它描述了事件的变化过程及其特征。因此,研究事件动作对于实现事件信息的自动理解非常重要。事件动作是随时间不断变化的,其变化过程多种多样,也就是说事件中的动作在执行过程中存在动态性和不确定性。一个动作执行之后,必然产生某些结果,期望的结果也可以通过一连串动作的执行而获得。人们通过对动作的思考来预测动作的可能结果,进而通过对动作可能结果的分析和评价来决定是否真正采取该动作,并确定下一步动作[5,6]。因此,关于动作的表示与推理一直是人工智能

到稿日期:2013-05-04 返修日期:2013-07-03 本文受国家自然科学基金(61305053,61273328,61074135),上海市自然科学基金(12ZR1410900)资助。

刘 炜(1978—),男,博士,副研究员,主要研究方向为知识表示、语义本体,E-mail:liuw@shu.edu.cn;徐文杰(1988—),女,硕士,主要研究方向为知识表示和推理;唐英英(1988—),女,硕士生,主要研究方向为知识表示和推理;付剑锋(1978—),男,博士,讲师,主要研究方向为自然语言处理、机器学习;张旭洁(1980—),女,博士生,主要研究方向为知识表示、知识推理;刘宗田(1946—),男,教授,博士生导师,主要研究方向为人工智能、软件工程等。

研究所关注的重要领域。目前，国内外学者已提出一些针对动作表示和推理的形式体系，如情景演算[7]、事件演算[8]、动态描述逻辑(DDL)[9-11]、整合描述逻辑和动作的形式体系[12]等。这些形式体系并不都以动作作为基本的表示单元，但都以各种形式来表示系统中的动态知识，并且都主要从动态知识的表示能力和推理的可判定性两个方面进行研究。由于知识表示与知识推理存在相互的制约，以上这些关于动态知识的形式体系用于事件动作的表示和推理普遍存在不足，比如：为了支持可判定的推理，知识表示能力太弱；或者知识表示能力太强而导致不可判定的推理；此外，大多数方法忽略了时间信息，使知识表示得不够具体，也不能处理时间相关的推理。针对这些不足，本文试图结合事件中的动作、对象和时间要素，通过集成描述逻辑和逻辑程序构建一个具有较强表达能力和可判定推理的动作形式体系。前期的工作提出了一种带时间维的描述逻辑 T-ALC[13]，它避免了一般融合时态算子的描述逻辑在推理上的困难。本文首先集成了描述逻辑和动作理论，将动作解释为扩展的描述逻辑 T-ALC 中个体实例的状态随时间变化而变化的过程，使 T-ALC 描述的静态知识和动作描述的动态知识的表示与推理有机地整合，形成一种简单的动作形式化体系 A1；然后，针对描述逻辑只含有一元和二元谓词、不支持过程性知识的表示与非单调推理等不足进一步将 T-ALC 与逻辑程序相结合，建立一种可实现不确定动作推理的形式化体系 A2。

对于描述逻辑与逻辑程序的集成，由于各自的表达能力和推理方法都不同，将两者相结合可提高本体的描述能力和推理能力，但它们都是单调性的，不支持常识推理(非单调推理)，而常识推理被认为是实现人类级别人工智能的重要解决途径。因此，对描述逻辑进行非单调扩展也是语义 Web 领域的研究课题之一。将描述逻辑和规则进行集成，主要存在以下需解决的问题：①描述逻辑与规则的表示形式不同。在描述逻辑中，只含一元谓词和二元谓词，即概念 $C(x)$，角色 $R(x,y)$，个体 a，实例 $C(a)$ 或 $R(a,b)$；在规则中，可含有任意元谓词 $P(x_1,\cdots,x_n)$。②语义假设不一致，即开放世界假说和封闭世界假说。③集成后知识表示的可判定性问题等。针对这些问题，本文以事件作为动态知识表示的单元，首先以扩展描述逻辑 T-ALC 为基础构建基于描述逻辑的知识库；其次以逻辑程序为基础构建基于规则的规则知识库、知识库；再次根据动作理论封装描述逻辑知识库和规则知识库形成确定性动作和不确定性动作的表示，并在推理时根据确定性动作语义将确定性动作转化为整合 T-ALC 与逻辑程序 Datalog 的推理，根据不确定性动作语义，整合描述逻辑开放世界假说的语义与非单调规则封闭世界假说的语义将不确定性动作转化为整合 T-ALC 与逻辑程序 Datalog$^{\neg}$ 的推理；最后针对两类动作推理给出其可执行性和可投影的推理。本文最后给出了一个动作形式体系的应用实例——银行服务(BankService)对动作的形式化推理进行验证。

2 相关研究工作

在语义 Web 新的系统结构中，本体(Ontology：OWL)和规则(Rules：RIF)放置在了相同的位置，本体层和规则层的整合已成为语义 Web 领域的研究热点之一[14]。

目前，描述逻辑与单调规则的集成主要有：1)同构法，即将本体与规则嵌入特定的逻辑语言，使用统一的知识表示与推理，语法共享语义兼容，如描述逻辑程序(Description Logic Program，DLP)[15]和语义网规则语言(Semantic Web Rule Language，SWRL)[16]，引入“角色与形式”(Role Conjunction Form)扩展描述逻辑，从而使 OWL 能够描述一阶谓词规则[14]等；2)异构法，即将本体与规则视为独立组件，保持各自的知识表示与推理，通过良好的接口如 AL-Log[17]、CARIN[18]等相连。描述逻辑与非单调规则的集成典型的工作有 DL-Log[19]、r-Hybird 系统[20]、ALCp^{u}[21]等。

描述逻辑程序(DLP)是 DL 和 LP 的一个可判定交集，两者相互转换来实现推理。然而，DLP 既限制了 DL 又限制了 LP，其知识表示能力是极其有限的。语义网规则语言(SWRL)是 OWL DL 与 Datalog 的并集，其知识表示能力远远超过 DLP，但文献[22]已证明 SWRL 的不可判定性，从而消弱了 SWRL 的应用。AL-Log 将知识库分为两部分：用 ALC 描述的结构化部分，用一组受限的 Datalog 规则集描述的关系化部分，并使用 DL-Safe 安全限制来约束规则，保证推理的可判定性。CARIN 从可判定的 DL 子语言和 Datalog 规则出发，通过限制各部分的知识表达能力来获得可判定性的方法。

DL-log 扩展了 AL-log 的基本思想，即将 Datalog 规则扩展为 Datalog 规则，并在关系化部分的规则约束中增加了角色约束。r-Hybird 系统是在 SWRL 的基础上，通过在规则中增加 NAF 原子对 SWRL 的表达能力进行扩展而得到的。为了保证 r-Hybird 系统的可判定性，Roasti 进一步提出在 r-Hybird 知识库中加以 DL-安全性或弱 DL-安全限制。ALCp^{u} 将描述逻辑的表达能力限制在 ALC 语言上，规则中只允许出现一元和二元谓词，降低了规则原有的知识表示能力。

以上这些工作从各个角度研究了语义 Web 中本体层和规则层相结合的方法，针对描述逻辑和规则表示形式的不一致、集成后知识表示的不可判定性等问题给出了相应的解决策略，如：语法都采用一阶谓词逻辑形式的语法；使用同构或异构的方法整合描述逻辑的开放世界假设和规则的封闭世界假设；对描述逻辑和规则分别进行限制，给出可判定的推理算法等。这些方法为本文集成描述逻辑与逻辑程序对事件动作进行表示和推理提供了借鉴和参考思路。综观目前语义 Web 领域众多的动态知识表示和推理方法，我们发现既有结合描述逻辑和动作理论进行动作表示和简单推理的工作[23,24]，也有将描述逻辑和规则相结合实现知识的非单调推理的工作[25]，但很少有将这 3 者结合来实现事件动作表示和非单调推理的方法，这也是本文最主要的研究动机之一。

3 基于 T-ALC 的动作表示与推理

3.1 预备知识

3.1.1 事件与动作

动作的表示和推理，不仅仅要考虑动作本身的变化，也应该考虑动作发生的时间和环境，即在事件背景下对动作进行推理。事件演算[8]虽然加入了时间点的表示方法，但仍然无法描述动作的发生过程。以下将简要介绍以事件为动作背景的动作表示方法和带时间维的描述逻辑 T-ALC 及其在此基础上的简单动作推理的相关内容和概念。

定义 1 事件(Event)是指在某个特定的时间和环境下

发生的，由若干角色参与，表现出若干动作特征的一件事情[4]，形式上用 e 来表示，被定义为一个六元组的结构：

$e::=_{def}(A,O,T,V,P,L)$

其中，事件六元组中的元素称为事件要素，分别表示动作、对象、时间、环境、断言和语言表现，关于事件定义的详细描述见文献[4]。

定义 2 动作(Action)表示事件的变化过程及其特征，可形式化定义为：

$A(x_1,\cdots,x_n,T)=(Pre,Post)$

其中：(1)A 是动作名；(2)$x_1,\cdots,x_n$ 是个体变量，表示动作作用的对象；(3)T 是动作发生的时间；(4)Pre 表示前置条件，是动作发生前各对象状态的有限集合；(5)$Post$ 表示后置结果，是形如条件表达式 φ/ψ 的有限集合，表示在某些条件下动作执行后产生的结果，可以表示动作的所有可能结果。

这个定义中包含了动作的时间、动作对象，以及动作发生前后状态的改变。其中 Pre 和 $Post$ 集合都使用谓词来表示，因此，Pre 和 $Post$ 集合中允许出现的谓词的形式及其构造算子的表达能力决定了动作的表达能力。这种动作中允许出现谓词的不同限制，导致了动作表达能力的不同。

对于定义 2 中动作 A 的 $Post$ 集合的条件表达式，φ/ψ 的含义如下：(1)如果 A 是可执行的，那么 ψ 为真，当且仅当 φ 为真。(2)如果 $Post$ 集合中所有的 φ/ψ 中 φ 都恒为真，即形如 true/ψ 的形式，那么可以将 φ/ψ 写为 ψ。(3)设 $Post$ 集合中 φ/ψ 的个数为 n，那么根据 φ 为真或假的不同，可以得到 2^n 个不同组合的 ψ 的结果的集合。

定义 3 实例代换(Instance of Substitution)是指动作 A 中的所有变量$\{x_1,\cdots,x_n\}$除时间 T 外都被常量$\{a_1,\cdots,a_n\}$代换，称$\{a_1/x_1,\cdots,a_n/x_n\}$为 A 的一个实例代换，其中 $x_i\,x_j$，i,j $\{1,\cdots,n\}$，$i\neq j$。时间 T 只有在动作被真正执行之后，才能有具体的表示处理。

对于定义 2 中的动作 $A(x_1,\cdots,x_n,T)=(Pre,Post)$，未实例代换之前称之为抽象动作。根据具体情况的不同，可以表示 3 类动作。设存在动作 A 的一个实例代换$\{a_1/x_1,\cdots,a_n/x_n\}$，将 A 中的所有变量$\{x_1,\cdots,x_n\}$都用常量$\{a_1,\cdots,a_n\}$代换：

(1)不确定性动作(action not be executed with uncertain effects)：A 未执行，并且其 $Post$ 集合至少有一个为 φ/ψ；

(2)确定性动作(action not be executed with certain effects)：A 未执行，并且其 $Post$ 集合中所有的 φ/ψ 都为 ψ；

(3)已执行动作(action be executed)：A 已执行，那么其 $Post$ 集合已经过处理，则可得到唯一的结果集合。

因此，如何处理 $Post$ 集合得到动作的执行结果是本文关于动作推理的主要推理问题之一。

在基于 T-ALC 和逻辑程序的动作形式体系中，其原子动作中允许出现描述逻辑谓词和规则谓词。而根据原子动作中对 $Post$ 集合的条件表达式的限制，将未执行的动作分为了两类，即不确定性动作和确定性动作。因此，根据动作的 Pre 和 $Post$ 集合中允许出现谓词的表现形式及动作的执行结果，可以将确定性动作分解为若干条 Datalog 的集合，将不确定性动作分解为若干条 Datalog 规则和 Datalog¬规则的集合，将这些分解出来的规则称为动作分解规则。这些动作分解规则既包含描述逻辑谓词，又包含规则谓词，因此，可以将动作的执行结果转化为对动作分解规则的推理，即描述逻辑与逻辑程序规则集成的推理。

3.1.2 T-ALC 简介

T-ALC 以描述逻辑为主要框架，对 ABox 中的概念断言和角色断言加入两类时间范围的限制，即时间点和时间段，使 ABox 描述的实例在这个时间范围内成立。T-ALC 不仅继承了描述逻辑的特点和优点，同时又具有对时间信息的处理能力，从而避免了一般融合时态算子的描述逻辑在推理上的困难。

基于 T-ALC 的知识库包含 TBox 和 ABox 两个部分，TBox 描述的是应用领域结构的公理的集合，而应用领域的知识一般保持不变，例如 $Father\doteq Man\sqcap\exists haschild.\,Human$。因此 T-ALC 的 TBox 与 ALC 的 TBox 完全等同。ABox 是描述实例断言的集合，随着时间的变化，个体实例的属性可能会发生变化、增加或减少，个体实例之间的关系也会发生变化，例如在时间点 t_1，$oneChildFahter(x)$ 成立，在时间点 t_2，存在 $haschild(x,z)$，则 $oneChildFahter(x)$ 不成立。因此，在 T-ALC 的 ABox 中将时间信息作为一种限制条件加入概念断言和角色断言中，表示这个实例断言在这个时间范围内是成立的，在其他时间未知。T-ALC 由于未增加构造算子，只增加了一种时间范围的限制，因此具有良好的扩展性，可以将时间信息扩展到其他表示能力更强的描述逻辑之上。

下面简单介绍 T-ALC 的语法及语义。

定义 4 时间 T 表示一个时间范围，用序偶$[t_1,t_2]$(或$(t_1,t_2]$，$[t_1,t_2)$，(t_1,t_2))($t_1\leqslant t_2$)表示，其中，$t_1=start(T)$和 $t_2=end(T)$是时间点，分别表示 T 的开始时间和结束时间。当 $t_1=t_2$ 时，T 表示一个时间点。

定义 5 时间$|T|$表示时间 T 的持续时间，由一个时间单位和持续的数字表示。

如$|[14:00,16:00]|=2$ hours，其中 2 是持续的数字，hours 是时间单位。

定义 6 两个时间 T_1，T_2 之间主要有以下 7 种时序关系[26]：*Meets*，*Before*，*Overlaps*，*Starts*，*Ends*，*During* 和 *Equals*。它们的含义如下：

$Meets(T_1,T_2),\{T_1\ Meets\ T_2\mid t_{12}=t_{21}\}$

$Before(T_1,T_2),\{T_1\ Before\ T_2\mid t_{12}<t_{21}\}$

$Overlaps(T_1,T_2),\{T_1 Overlaps\ T_2\mid t_{11}<t_{21}\wedge t_{12}<t_{22}\}$

$Starts(T_1,T_2),\{\ T_1\ Starts\ T_2\mid t_{11}=t_{21}\wedge t_{12}<t_{22}\}$

$Ends(T_1,T_2),\{\ T_1 Ends\ T_2\mid t_{11}>t_{21}\wedge t_{12}<t_{22}\}$

$During(T_1,T_2),\{T_1\ During\ T_2\mid t_{11}>t_{21}\wedge t_{12}<t_{22}\}$

$Equals(T_1,T_2),\{T_1\ Equals\ T_2\mid\ t_{11}=t_{21}\wedge t_{12}=t_{22}\}$

其中，$t_{11}=start(T_1)$是 T_1 的开始时间，$t_{12}=end(T_1)$是 T_1 的结束时间，并且 $t_{11}\leqslant t_{12}$。$t_{21}=start(T_2)$是 T_2 的开始时间，$t_{22}=end(T_2)$是 T_2 的结束时间，并且 $t_{21}\leqslant t_{22}$。

设 N_C 表示概念名的集合，N_R 表示角色名的集合，N_O 表示个体名的集合。

定义 7 ABox $\mathcal{A}_T$ 是由有限个带时间信息的实例断言组成的集合。带时间信息的实例断言包含形如 $a^{[t_1,t_2]}:C$ 或 $a^t:C$ 的概念断言，和形如$(a,b)^{[t_1,t_2]}:R$ 或$(a,b)^t:R$ 的角色断言，其中 $a,b\in N_O$，$C\in N_C$，$R\in N_R$，t_1，t_2，t 都是时间点。

定义 8 知识库 $K_{T\text{-}ALC}=\langle\mathcal{T},\mathcal{A}_T\rangle$ 由两个部分组成，一个是经典的 TBox $\mathcal{T}$，TBox $\mathcal{T}$ 是由有限个一般概念蕴含式 GCIs

(general concept inclusions)组成的集合；一个是定义 7 中的 ABox $\mathcal{A}_T$。

定义 9 ABox $\mathcal{A}_T$ 中的解释是形如 $\mathcal{I}(t)=(\overset{\mathcal{I}(t)}{\Delta},\overset{\mathcal{I}(t)}{\cdot})$的二元组。其中 $\overset{\mathcal{I}(t)}{\Delta}$ 为论域，指在时间点 t 的一个非空集合。$\overset{\mathcal{I}(t)}{\cdot}$ 为在时间点 t 的一个映射函数。

定义 10 TBox $\mathcal{T}$ 中的解释是形如 $\mathcal{K}(t)=(\overset{\mathcal{K}(t)}{\Delta},\overset{\mathcal{K}(t)}{\cdot})$的二元组。其中$\overset{\mathcal{K}(t)}{\Delta}$为论域，是在时间点 t 的一个非空的集合，其中 $t\in[0,V]$，表示知识库存在的所有有效时间范围，即$\overset{\mathcal{K}(t)}{\Delta}$不受时间限制，可以简化为 $\overset{\mathcal{I}}{\Delta}$。$\overset{\mathcal{K}(t)}{\cdot}$ 为在时间点 t 的一个映射函数，同理$\overset{\mathcal{K}(t)}{\cdot}$ 也不受时间限制，可以简化为 $\overset{\mathcal{I}}{\cdot}$，即将每个概念 $C\in N_C$ 映射为$\overset{\mathcal{K}(t)}{\Delta}$ ($\overset{\mathcal{I}}{\Delta}$)的一个子集 $C^{\mathcal{K}(t)}$ ($C^{\mathcal{I}}$)，将每个角色 $R\in N_R$映射为$\overset{\mathcal{K}(t)}{\Delta}\times\overset{\mathcal{K}(t)}{\Delta}$ ($\overset{\mathcal{I}}{\Delta}\times\overset{\mathcal{I}}{\Delta}$)的一个二元关系 $R^{\mathcal{K}(t)}$ ($R^{\mathcal{I}}$)。

在知识库 $K_{T\text{-}ALC}$ 中，虽然分别定义了 ABox $\mathcal{A}_T$ 和 TBox $\mathcal{T}$ 中的解释，但由于时间信息在 ABox 中表示的是一种限制范围，TBox 相对于 ABox 没有时间限制，换个角度即可以将 TBox 的时间限制看作在知识库存在的所有有效时间内都成立，这种特性由知识库结构的不变性决定。因此，可以将两者的解释统一起来，共同表示知识库 $K_{T\text{-}ALC}$ 的解释。

存在知识库 $K_{T\text{-}ALC}=\langle\mathcal{T},\mathcal{A}_T\rangle$及一个在时间 t 的解释 $\mathcal{K}(t)=(\overset{\mathcal{K}(t)}{\Delta},\overset{\mathcal{K}(t)}{\cdot})$。那么，在时间 t：

(1)在 TBox $\mathcal{T}$ 中，如果 $C^{\mathcal{K}(t)}\subseteq D^{\mathcal{K}(t)}$ 成立，那么称 $\mathcal{K}(t)$ 满足概念蕴含式 $C\sqsubseteq D$。如果 $C^{\mathcal{K}(t)}\subseteq D^{\mathcal{K}(t)}$ 和 $D^{\mathcal{K}(t)}\subseteq C^{\mathcal{K}(t)}$ 同时成立，那么称 $\mathcal{K}(t)$ 满足概念等式 $C\equiv D$。如果 $\mathcal{K}(t)$ 满足 $\mathcal{T}$ 中所有公理，那么称 $\mathcal{K}(t)$ 满足术语集 $\mathcal{T}$。

(2)在 ABox $\mathcal{A}_T$ 中，对概念断言 $a^t:C$ 和 $a^{[t_1,t_2]}:C(t\ [t_1,t_2])$：如果概念 C 关于 TBox 是可满足的，并且 $a^{\mathcal{K}(t)}\in C^{\mathcal{K}(t)}$，那么称 $\mathcal{K}(t)$ 满足概念断言 $a^t:C(\mathcal{K}(t)\mid=a^t:C)$。如果在$[t_1,t_2]$中的任一时间点 $t'\in[t_1,t_2]$，都存在一个解释 $\mathcal{K}(t')$ 使得 $a^{\mathcal{K}(t')}\in C^{\mathcal{K}(t')}$，那么称概念断言 $a^{[t_1,t_2]}:C$ 在时间$[t_1,t_2]$内是可满足的。对角色断言$(a,b)^t:R$ 和$(a,b)^{[t_1,t_2]}:R(t\in[t_1,t_2])$：如果角色 R 关于 TBox 是可满足的，并且$(a,b)^{\mathcal{K}(t)}\in R^{\mathcal{K}(t)}$，那么称 $\mathcal{I}(t)$满足角色断言$(a,b)^t:R(\mathcal{K}(t)\mid=(a,b)^t:R)$。如果在$[t_1,t_2]$中的任一时间点 $t'\in[t_1,t_2]$，都存在一个解释 $\mathcal{K}(t')$ 使得$(a,b)^{\mathcal{K}(t')}\in R^{\mathcal{K}(t')}$，那么称角色断言$(a,b)^{[t_1,t_2]}:R$ 在时间$[t_1,t_2]$内是可满足的。如果 $\mathcal{K}(t)$ 满足 $\mathcal{A}_T$ 中所有实例断言，那么称 $\mathcal{K}(t)$满足断言集 $\mathcal{A}_T$。

(3)如果 $\mathcal{K}(t)$同时满足术语集 $\mathcal{T}$和断言集 $\mathcal{A}_T$，那么称 $\mathcal{K}(t)$为知识库 $K_{T\text{-}ALC}$的一个模型。

3.2 基于 T-ALC 的动作形式化表示

动作中允许出现的谓词的形式及其构造算子决定了动作的表达能力。基于 T-ALC 的动作形式化表示中，动作中使用的谓词为 T-ALC 谓词，用于描述 TBox 中出现的概念和角色。我们把这种基于 T-ALC 的动作形式体系命名为 A1。

定义 11 基于 T-ALC 的动作形式体系 A1 中的原子动作是指定义 2 中动作中的 *Pre* 集合和 *Post* 集合中的谓词都为 T-ALC 谓词。

通常，我们使用状态的集合来表示静态的知识，使用状态集合的改变过程来表示动态的知识，而这些改变过程一般是由动作引起的。如果引入时间信息，动态知识可以更准确地解释为，在不同时间，随着动作的发生而引起的状态随时间变化而变化的过程。在基于 T-ALC 的动作形式化系统中，动作定义了这些变化的过程，并随着时间在 ABox 中反映出来。因此，在解释动作的语义时，假设世界的某些状态在某时间点 t 对应于 T-ALC 的一个解释 $\mathcal{K}(t)$，也就是说，一个解释 $\mathcal{K}(t)$能够描述时间点 t 世界的某些状态。这样，动作的语义可以描述为如何将一个解释 $\mathcal{K}(t)$转变为另一个解释 $\mathcal{I}'(t')$的过程。

定义 12 在一个基于 T-ALC 的动作形式体系中，惯性原则(Principle of Inertia，PoI)是指，系统中的谓词发生改变当且仅当它由动作触发而发生，否则一直成立保持不变。*PoI* 的形式化理解为，在时间 t 时的 ABox $\mathcal{A}_T$ 及一个解释 $\mathcal{K}(t)$，$C(a)$和 $R(a,b)$是它们中的概念断言和角色断言：

(1)对形如 $C(a)^t$ 和 $R(a,b)^t$ 形式的概念断言和角色断言：从时间点 t 到时间点 t'，如果在这段时间内未发生动作使 $C(a)$和 $R(a,b)$发生变化，那么在时间 t'，$C(a)^{t'}$和 $R(a,b)^{t'}$也是成立的。对形如 $C(a)^t$ 和 $R(a,b)^t$ 的概念断言和角色断言，在时间点 t'，解释 $\mathcal{K}(t)$转化为 $\mathcal{K}(t)=\{\mathcal{K}(t)\cup C(a)^{t'}\cup R(a,b)^{t'}\}\backslash\{C(a)^t\cup R(a,b)^t\}$；对 $\mathcal{A}_T$ 中所有形如 $C(a)^t$ 和 $R(a,b)^t$ 的概念断言和角色断言，在时间点 t'，ABox $\mathcal{A}_T$ 转化为 $\mathcal{A}_T'=\{\mathcal{A}_T\cup C(a)^{[t,t']}\cup R(a,b)^{[t,t']}\}\backslash\{C(a)^t\cup R(a,b)^t\}$。

(2)对形如 $C(a)^{[t_1,t]}$和 $R(a,b)^{[t_1,t]}$形式的概念断言和角色断言：从时间点 t 到时间点 t'，如果在这段时间内未发生动作使 $C(a)$和 $R(a,b)$发生变化，那么在时间点 t'，$C(a)^{t'}$和 $R(a,b)^{t'}$也是成立的。$C(a)^{[t_1,t]}$和 $R(a,b)^{[t_1,t]}$形式的概念断言和角色断言只在 $\mathcal{A}_T$ 中出现，即对 $\mathcal{A}_T$ 中所有形如$C(a)^{[t_1,t]}$和 $R(a,b)^{[t_1,t]}$形式的概念断言和角色断言，在时间点 t'，ABox $\mathcal{A}_T$ 转化为 $\mathcal{A}_T'=\{\mathcal{A}_T\cup C(a)^{[t_1,t']}\cup R(a,b)^{[t_1,t']}\}\backslash\{C(a)^{[t_1,t]}\cup R(a,b)^{[t_1,t]}\}$。

这个惯性原则表示，ABox 中谓词发生改变当且仅当它由动作触发而发生变化。否则一直保持不变，并且在 ABox 中需要与原来的谓词进行合并。

定义 13 设存在一个非循环的 TBox $\mathcal{T}$和在时间点 t 的 ABox $\mathcal{A}_T$ 和解释 $\mathcal{K}(t)$，并且 $\mathcal{K}(t)\mid=\mathcal{T},\mathcal{A}_T$。动作 A 在时间点 t 未执行，A 对 $\mathcal{T}$和 $\mathcal{A}_T$ 是可执行的。原子动作的语义表示：当动作 A 执行之后，它会改变一些状态或产生一些新的结果，即 $\mathcal{A}_T$ 会发生一些改变，转化为 $\mathcal{A}_T'$，解释 $\mathcal{K}(t)$也会转化为另一个时间点的解释 $\mathcal{I}'(t')(t'>t)$，并且 $\mathcal{I}'(t')\mid=\mathcal{T},\mathcal{A}_T'$。这些可以写为 $\mathcal{K}(t)\Rightarrow_A^{\mathcal{T},\mathcal{A}_T}$，$\mathcal{A}_T\Rightarrow_A^{\mathcal{T}}\mathcal{A}_T'$，动作 A 的执行时间为$[t,t']$。

在这个定义中，原子动作的语义表示为，在时间上从一个解释向另外一个解释的转化过程，并且伴随着 ABox 的变化。设 A 为一个原子动作，其执行时间为$[t,t']$，根据 A 中 *Post* 集合的每个条件表达式 φ/ψ，当 A 执行之后，新的解释 $\mathcal{I}'(t')$和新的 ABox $\mathcal{A}_T'$都可以通过如下方法得出，其中 $a,b\in N_O$是个体，$C,D\in N_C$ 是概念，$R,S\in N_R$ 是角色，并且 $t\leqslant t'$，t,t'都是时间点：

(1)对 ψ中出现的每个原子概念 $C(a)$或$\neg C(a)$，如果在时间点 t，集合$(Pre\cup\varphi)$是可满足和一致的，那么 A 是可执行的。当 A 执行之后，首先，

$\mathcal{K}(t')=\{\mathcal{K}(t)\cup C(a)^{t'}\}\backslash\{\neg C(a)^t\cup D(a)^t\}$或 $\mathcal{K}(t')=\{\mathcal{I}(t)\cup\neg C(a)^{t'}\}\backslash\{C(a)^t\cup D(a)^t\}$

$\mathcal{A}_{T'}=\{\mathcal{A}_T\cup\varphi\cup C(a)^{t'}\}\backslash\{D(a)^t\}$或$\mathcal{A}_{T'}=\{\mathcal{A}_T\cup\varphi\cup\neg C(a)^{t'}\}\backslash\{D(a)^t\}$

其中，D是隐含的与C相冲突的概念，当且仅当集合$\{\mathcal{K}(t)\cup C(a)^t\}$和$\{\mathcal{A}_T\cup\varphi\cup C(a)^t\}$出现不一致时；否则$D(a)^t=\emptyset$。然后，对$\mathcal{K}(t')$和$\mathcal{A}_T$使用惯性原则，得到$\mathcal{K}(t')\xrightarrow{PoI}\mathcal{I}'(t')$，$\mathcal{A}_T\xrightarrow{PoI}\mathcal{A}_T'$。

(2)对ψ中出现的每个原子角色$R(a,b)$或$\neg R(a,b)$，如果在时间点t，集合$(Pre\cup\varphi)$是可满足和一致的，那么A是可执行的。当A执行之后，首先：

$\mathcal{K}(t')=\{\mathcal{K}(t)\cup R(a,b)^{t'}\}\backslash\{\neg R(a,b)^t\cup S(a,b)^t\}$或$\mathcal{K}(t')=\{\mathcal{K}(t)\cup\neg R(a,b)^{t'}\}\backslash\{R(a,b)^t\cup S(a,b)^t\}$

$\mathcal{A}_{T'}=\{\mathcal{A}_T\cup\varphi\cup R(a,b)^{t'}\}\backslash\{\neg R(a,b)^t\cup S(a,b)^t\}$或$\mathcal{A}_{T'}=\{\mathcal{A}_T\cup\varphi\cup\neg R(a,b)^{t'}\}\backslash\{S(a,b)^t\}$

其中，S是隐含的与R相冲突的角色，当且仅当集合$\{\mathcal{K}(t)\cup R(a,b)^t\}$和$\{\mathcal{A}_T\cup\varphi\cup R(a,b)^t\}$出现不一致时；否则$S(a,b)^t=\emptyset$。然后，对$\mathcal{I}(t')$和$\mathcal{A}_T$使用惯性原则，得到$\mathcal{I}(t')\xrightarrow{PoI}\mathcal{I}'(t')$，$\mathcal{A}_T\xrightarrow{PoI}\mathcal{A}_T'$。

(3)对ψ中出现的其他的概念$C(a)$和角色$R(a,b)$，根据$\mathcal{T}$的非循环性，C和R都可以分解为有限个原子概念和原子角色组成的集合。那么$\mathcal{I}'(t')$和$\mathcal{A}_T'$也可以通过以上方法得到。

以上给出了基于T-ALC的动作形式体系中动作的语法和语义，并给出了动作执行之后，新的解释$\mathcal{I}'(t')$和新的ABox $\mathcal{A}_T'$的计算方法，它们可以通过$\mathcal{A}_T'$，$\mathcal{T}$和A唯一得出，同时，$\mathcal{I}'(t')$描述了时间点t'的状态，而$\mathcal{A}_T'$描述了在时间段$[0,t']$的状态及状态的改变过程。

3.3 基于T-ALC的动作推理

动作的推理问题主要包括在动作执行之前动作的一致性检测，动作的可执行性和可投影问题、动作的执行结果即对动作后置集合$Post$集合的处理过程，以及动作相关的一些查询问题等。

a)动作的一致性

定义14 ABox $\mathcal{A}_T$中的一个冲突包含以下的3种情况，其中$C\in N_C$为任意概念，$R\in N_R$为任意角色，$a,b\in N_O$为任意个体：

(1)在当前时间点t：对每个形如$C(a)^t$的概念和$R(a,b)^t$的角色，一个冲突是含有如下情形之一的集合，$\{\perp(a)^t\}$，$\{C(a)^t,\neg C(a)^t\}$，$\{R(a,b)^t,\neg R(a,b)^t\}$。

(2)在过去某个时间点t'：对每个形如$C(a)^{t'}$或$C(a)^{[t_1,t_2]}$($t'\in[t_1,t_2]$)的概念和形如$R(a,b)^{t'}$或$R(a,b)^{[t_3,t_4]}$($t'\in[t_3,t_4]$)的角色，一个冲突是含有如下情形之一的集合，$\{\perp(a)^{t'}\}$，$\{C(a)^{t'},\neg C(a)^{t'}\}$，$\{R(a,b)^{t'},\neg R(a,b)^{t'}\}$。

定义15 设$\mathcal{T}$为一个非循环的TBox，动作A是关于$\mathcal{T}$的一个原子动作，$\mathcal{A}_T$为A执行之前的ABox，$\mathcal{K}(t)$是一个解释并且$\mathcal{K}(t)\models\mathcal{T},\mathcal{A}_T$，原子动作的一致性是指：

(1)对$\mathcal{K}(t)$和$\mathcal{A}_T$，如果A的Pre集合不含冲突，那么Pre是一致的，称A为前提Pre一致。

(2)对$\mathcal{K}(t)$和$\mathcal{A}_T$，如果A的$Post$集合的条件表达式包含形如$\varphi_1/\psi\in Post$，$\varphi_2/\neg\psi\in Post$的形式，并且$\varphi_1$和$\varphi_2$对$\mathcal{K}(t)$和$\mathcal{A}_T$是可满足的，那么动作$A$执行之后会包含冲突，称$A$为后置$Post$不一致。

(3)如果A为前提Pre一致及后置$Post$一致，那么称A关于$\mathcal{T}$、$\mathcal{K}(t)$和$\mathcal{A}_T$是一致的。

b)动作的基本推理问题

在执行一个动作之前，总是要检测该动作是否是可执行的，即检测一下该动作在当前的状态下其前置条件集合是否是可满足的。如果该动作是可执行的，那么执行该动作将达到需要的结果，及检测一下某个需要的断言在动作执行之后是否成立。这两个问题称为动作的可执行性问题和可投影问题。

定义16 设$\mathcal{T}$为一个非循环的TBox，动作$A=(Pre,Post)$是关于$\mathcal{T}$的一个原子动作，A执行之前，$\mathcal{A}_T$是时间点t的ABox，$\mathcal{K}(t)$是一个解释并且$\mathcal{K}(t)\models\mathcal{T},\mathcal{A}_T$，原子动作的基本推理问题是指：

(1)动作的可执行性：A是可执行的当且仅在时间点t，A的Pre集合中所有的概念和角色都是可满足的，即$\mathcal{K}(t)\models Pre$。

(2)动作的投影：如果断言φ是动作A的一个执行结果，当且仅当对$\mathcal{A}_T$和$\mathcal{T}$的所有模型$\mathcal{K}(t)$，根据动作的语义，存在$\mathcal{I}'(t')$和$\mathcal{A}_T'(\mathcal{K}(t)\Rightarrow_A^{\mathcal{T},\mathcal{A}_T}\mathcal{I}'(t'),\mathcal{A}_T\Rightarrow_A^{\mathcal{T}}\mathcal{A}_T')$，$A$的执行时间为$T=[t,t']$，有$\mathcal{I}'(t')\models\varphi$。

引理1 原子动作的可执行性可以转化为T-ALC中的可满足性问题。

证明：由定义16中原子动作的可执行性问题的定义可知，原子动作$\mathcal{A}$的可执行性问题可以理解为存在$\mathcal{K}(t)\models\mathcal{T}$，使$\mathcal{K}(t)\models Pre$。由定义13原子动作的定义可知，$A$的$Pre$集合为动作发生之前世界的状态，并且都为T-ALC谓词，即在时间点t，其是$\mathcal{A}_T$中关于$\mathcal{T}$的概念断言和角色断言的集合。因此，A的可执行性可以转化为T-ALC中概念断言和角色断言的可满足性问题。

引理2 原子动作的投影问题可以在多项式时间内转化为原子动作的可执行性问题。

证明：由定义16中原子动作的投影问题的定义可知，对原子动作A的投影问题可以理解为在A执行之后存在$\mathcal{I}'(t')\models\mathcal{T}$，给定的断言$\varphi$使得$\mathcal{I}'(t')\models\varphi$。现假设存在一个无结果的原子动作$A'=(\{\varphi'\},\emptyset)$发生在$A$之后，其中$\varphi$为投影问题中这个给定的断言。如果$A$是可执行的，则$A$的投影问题也可以描述为在$A$执行之后的时间$t'$内，得到一个新的解释$\mathcal{I}'(t')$，$\mathcal{I}'(t')=\mathcal{T},\varphi$。对动作$A'=(\{\varphi'\},\emptyset)$，由于在当前解释$\mathcal{I}'(t')$下，$\mathcal{I}'(t')=\varphi$成立，则$A'$在$t'$是可执行的，即动作$A$的投影问题转化为动作$A'$的可执行性问题。因此，原子动作的投影问题转化为了原子动作的可执行性问题。

定理1 原子动作的可执行性和投影问题都可以转化为T-ALC的可满足性问题。

因此，由以上引理和定理可知，定义16中原子动作的基本推理问题，即原子动作的可执行性和投影问题都是可判定的。

c)动作的执行结果

定义2中动作的定义可以表示3类动作：不确定性动作、确定性动作和已执行动作。广义地讲，动作的执行结果的推理分以下4步：①执行之前，对知识库进行一致性检测；②动作的一致性和可执行性检测；③动作的执行结果的推理；④新的解释$\mathcal{I}'(t')$与新的ABox $\mathcal{A}_T'$的转变过程。其中，对动作的

执行结果可根据3类动作分3种情况进行讨论：

(1)对已执行的动作 A：由于动作 A 已经执行，其 *Post* 集合已经过处理，因此无以上①②的检测，对③动作的执行结果的推理即为 A 的 *Post* 集合。

(2)对确定性动作 A：在动作执行前，经过以上①②的检测后，如果知识库是一致的，并且动作 A 是一致性和可执行的，那么对③动作 A 的执行结果的推理即为 A 的 *Post* 集合经过一个实例代换$\{a_1/x_1,\cdots,a_n/x_n\}$得到的集合；否则动作 A 不可执行。

(3)对不确定性动作 A：在动作执行之前，经过以上①②的检测后，如果知识库是一致的，并且动作 A 是一致性和可执行的，对③动作 A 的执行结果的推理即为对动作 A 的 *Post* 集合的条件表达式 φ/ψ 逐个进行检查，如果 φ 是可满足的，则 ψ 为真，A 的执行结果即为根据$\{\varphi/\psi\}$中 ψ 的可满足性得到的$\{\psi\}$的集合；否则该动作不可执行。

在动作执行之后，对④新的解释 $\mathcal{I}'(t')$与新的 ABox $\mathcal{A}_T'$ 进行计算，根据定义13中动作的语义，可以得到新的解释 $\mathcal{I}'(t')$与新的 ABox $\mathcal{A}_T'$。

对动作的执行结果的推理，①②都是 T-ALC 的基本推理问题，都是可判定的；③主要是进行实例代换及判定条件表达式 φ/ψ 中 φ 的可满足性检测，因此也是可判定的；④中新的解释和新的 ABox 的计算过程可通过定义13中的方法得到，因此动作的执行结果的推理也是可判定的。

4 基于 T-ALC 与逻辑程序的动作表示与推理

4.1 集成 T-ALC 和逻辑程序的动作表示

逻辑程序的基本语句属于一阶谓词逻辑的一个子集，它的每一条规则可以看作是由前提 B 和结论 H 组成的序对 $H\leftarrow B$，表示如果前提 B 成立，则结论 H 也成立。逻辑程序根据其前提和结论中所允许的谓词或操作符的不同，逻辑程序的知识表达能力也不同。

若在逻辑程序中的任何一条规则中都不包含函数符号，在规则体中不允许出现否定即失败词 not，并且规则头由单一的正文字构成，则称逻辑程序为 Datalog 程序。在此基础上，若允许在规则头中出现多个原子的析取式，则称之为 Datalog∨ 程序；若允许在规则体中出现否定即失败词 not，则称之为 Datalog¬ 程序；若同时允许在规则头中出现多个原子析取式以及在规则体中出现否定即失败词 not，则称之为 Datalog¬,∨ 程序。它们的定义如下：

定义 17 Datalog 程序是由有限个形如 $H(Y)\leftarrow B_1(X_1),\cdots,B_n(X_n)$的 Datalog 规则所组成的集合。其中：

(1)$H(Y)$为规则头；

(2)$B_1(X_1),\cdots,B_n(X_n)$为规则体；

(3)H、$B_1,\cdots,B_n$ 为谓词符号；

(4)Y、$X_1,\cdots,X_k$ 为项。

定义 18 Datalog¬ 程序是由有限个形如 $H(Y)\leftarrow B_1(X_1),\cdots,B_m(X_m)$,not $B_{m+1}(X_{m+1}),\cdots$, not $B_n(X_n)$的 Datalog¬ 规则所组成的集合。其中：

(1)$H(Y)$为规则头；

(2)$B_1(X_1),\cdots,B_m(X_m)$, not $B_{m+1}(X_{m+1}),\cdots$, not $B_n(X_n)$为规则体；

(3)H、$B_1,\cdots,B_n$ 为谓词符合；

(4)$B_1(X_1),\cdots,B_m(X_m)$ 为规则中的正文字，$B_{m+1}(X_{m+1}),\cdots,B_n(X_n)$为规则中的负文字，也称 NAF 原子，not 为“否定即失败”语义；

(5)Y、$X_1,\cdots,X_k$ 为项。

由于动作的 *Pre* 和 *Post* 集合中允许出现的谓词的形式及其中的构造算子的表达能力决定了动作的表达能力；又由于描述逻辑自身表达能力的限制，如只含有一元和二元谓词、只适合表示结构化的确定的知识、不支持过程性知识的表示、不支持非单调的推理等，在前面基于 T-ALC 的动作形式体系 A1 中，描述逻辑即使扩展为 T-ALC，动作中的 *Pre* 和 *Post* 集合也仅由 T-ALC 谓词来描述，因此，A1 的知识表达能力是有限的。为了构建具有强表达能力的动作形式体系，有必要将 T-ALC 与逻辑程序进行集成，共同表示动态知识，称之为动作形式体系 A2。根据定义2中动作的定义，基于 T-ALC 和逻辑程序的动作形式体系中的原子动作定义如下：

定义 19 基于 T-ALC 与逻辑程序的动作形式体系中的原子动作是指对定义2中定义的动作做如下限制：

(1)*Pre* 集合出现的谓词都为 T-ALC 谓词；

(2)对于 *Post* 集合中的每个条件表达式 φ/ψ：①ψ 和 φ 可同为描述逻辑谓词(即定义11中定义的原子动作的表示形式)；②ψ 为描述逻辑谓词，φ 为规则谓词；③ψ 为规则谓词，φ 为描述逻辑谓词或规则谓词，并允许失败即否定 not 出现在 φ 中规则谓词前。

定义13给出了基于 T-ALC 的原子动作的语义。动作的语义除了可以理解为 T-ALC 中一个解释向另一个解释的转化过程外，也可以广义地理解为应用领域中一个时间点的知识向另一个时间点的知识的转化过程。因此，在基于 T-ALC 与逻辑程序的动作体系中，动作的语义则可以理解为动作执行之后知识的改变过程。在应用领域中，动作的执行导致了应用领域中知识的改变，即动作的语义；动作的语义又指导着动作的执行结果的推理，因此动作的语义与动作的执行结果的推理是一致的。

由于基于 T-ALC 和逻辑程序的动作形式体系中出现了含有多元谓词的规则谓词，因此不能直接将动作的语义理解为 T-ALC 中一个解释向另一个解释的转化的过程，需要对解释进行修正。

定义 20 设存在一个知识库为 $K=\langle K_{T\text{-}ALC},K_R\rangle$，其中 $K_{T\text{-}ALC}$表示基于 T-ALC 的知识；K_R 表示基于逻辑程序的知识，其中 K_R 中的事实也包含时间信息，是形如 $H(a_1,a_2,\cdots,a_n)^t$ 或 $H(a_1,a_2,\cdots,a_m)^{[t_1,t_2]}$形式的集合，记为 F_T。设存在 K 的一个解释 $\mathcal{A}(t)$是形如 $\mathcal{A}(t)=\langle\mathcal{I}_{T\text{-}ALC}(t),\bullet^{\mathcal{I}_R}(t)\rangle$的二元组，其中 $\mathcal{I}_{T\text{-}ALC}(t)=(\Delta^{\mathcal{I}_{T\text{-}ALC}(t)},\bullet^{\mathcal{I}_{T\text{-}ALC}(t)})$是 $K_{T\text{-}ALC}$的解释；$\bullet^{\mathcal{I}_R}(t)$为一个映射函数，将 K_R 中的每个规则谓词映射为 $\Delta^{\mathcal{I}_{T\text{-}ALC}(t)}\times\cdots\times\Delta^{\mathcal{I}_{T\text{-}ALC}(t)}$的一个任意元关系的子集，并且需要去除在 K_R 中此规则谓词的例外情况，即需对这个子集进行缩减。缩减子集的含义及过程如例1所示。

例1 设存在规则“$fly(x)\leftarrow bird(x)$，$bird(x)\leftarrow penguin(x)$”，则根据 $penguin(a)$可以得出 $fly(a)$。根据常识，这是个错误的结论。实际上，这是一种例外的情况，如果增加另一条规则“$\neg fly(x)\leftarrow penguin(x)$”，并优先使用此规则，则可得

到正确的推理$\neg fly(a)$。通过$\mathcal{A}(t)$对$fly(x)$进行解释后，可以得出它的一个子集可为$\{bird(x)\}$，但还需根据负规则进行缩减，即将其子集缩减为$\{bird(x)\backslash penguin(x)\}$。若还存在$fly(x)$的规则"$\neg fly(x)\leftarrow ostrich(x), bird(x)\leftarrow ostrich(x)$"，则其解释的子集继续缩减为$\{bird(x)\backslash\{penguin(x), ostrich(x)\}\}$。

对K_R中的一条规则$R: H\leftarrow B$，解释$\mathcal{A}(t)$是规则R的模型，对任意赋值$\vec{X}$（赋值$\vec{X}$是将R的所有变元映射到$\Delta^{\mathcal{A}_{T\text{-}ALC}(t)}$），如果规则$R$的规则头$B(\vec{X})$中的每一个谓词都是可满足的，则$R$的规则体$H(\vec{X})$也是可满足的。如果解释$\mathcal{A}(t)$满足$K_R$中的每条规则$R$和事实，则解释$\mathcal{A}(t)$满足该规则知识库$K_R$，记为$\mathcal{A}(t)|=K_R$。

定义 21 设存在一个动作形式体系 A2 的知识库为$K=\langle K_{T\text{-}ALC}, K_R\rangle$。给定$K_{T\text{-}ALC}$中一个非循环的 TBox $\mathcal{T}$，及在时间点t的 ABox $\mathcal{A}_T$、K_R中的事实F_T和解释$\mathcal{A}(t)$，并且$\mathcal{A}(t)|=\mathcal{T}, \mathcal{A}_T, K_R$。动作$A$在时间点$t$未执行，$A$是可执行的。原子动作的语义是指当动作$A$执行之后，它会改变一下状态并产生一些新的状态，即$\mathcal{A}_T$会发生改变转化为$\mathcal{A}_T'$，$F_T$会发生改变转化为$F_{T'}$，解释$\mathcal{A}(t)$也会转化为另一个时间点的解释$\mathcal{I}'(t')(t'>t)$，并且$\mathcal{I}'(t')|=\mathcal{T}, \mathcal{A}_T', K_{R'}$。这些可以写为$\mathcal{A}(t)\Rightarrow_A^{\mathcal{T},\mathcal{A}_T,K_R}\mathcal{I}'(t')$，$\mathcal{A}_T\Rightarrow_A^{\mathcal{T}}\mathcal{A}_T'$，$F_T\Rightarrow_A^{\mathcal{A}_T}F_{T'}$，动作$A$的执行时间为$[t,t']$。

在这个定义中，原子动作的语义表示为在时间上从一个解释向另外一个解释的转化过程，并伴随着 ABox 和F_R发生变化，且是动作的执行导致了 ABox 和F_T的变化。下面详细介绍基于 T-ALC 和逻辑程序的动作形式化表示体系中的动作根据动作的语义转化为 T-ALC 与 Datalog 规则和 Datalog$\neg$规则的过程。

定义 22 在 A2 中确定性动作是指其 *Post* 集合中所有的φ/ψ都为ψ，并且ψ为描述逻辑谓词或规则谓词。对一个确定性原子动作，根据其ψ中的规则谓词可以将此动作转化为有关$K_{T\text{-}ALC}$和K_R的形如$H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n)$的 Datalog 规则组成的集合，称其为确定性原子动作的分解。其中：

(1)$H(Y)$称为规则头，是规则谓词，属于$\{\psi\}$；

(2)$D_1(Z_1),\cdots,D_n(Z_n)$为描述逻辑谓词，属于 *Pre* 集合；

同时，为了保证知识库的可判定性，需对规则增加安全限制：规则头中出现的所有变量必须出现在规则体的谓词中。

定义 23 在 A2 中不确定性动作是指其 *Post* 集合至少有一个为φ/ψ，并且φ为规则谓词，允许失败即否定 not 出现在规则谓词φ之前，ψ为描述逻辑谓词或规则谓词。对一个不确定性原子动作，根据其ψ中的规则谓词可以将此动作转化为有关$K_{T\text{-}ALC}$和K_R的形如$H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n)$或$H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n),B_1(X_1),\cdots,B_m(X_m)$的 Datalog 规则，或形如$H(Y)\leftarrow B_1(X_1),\cdots,B_k(X_k),D_1(Z_1),\cdots,D_n(Z_n),\text{not } B_{k+1}(X_{k+1}),\cdots,\text{not } B_m(X_m)$的 Datalog$\neg$规则组成的集合，称其为不确定性原子动作的分解。其中：

(1)$H(Y)$称为规则头，是规则谓词，属于$\{\psi\}$；

(2)$B_1(X_1),\cdots,B_m(X_m)$为任意元谓词，称为规则谓词，属于$\{\varphi\}$；

(3)$D_1(Z_1),\cdots,D_n(Z_n)$为描述逻辑谓词，属于 *Pre* 集合或$\{\varphi\}$；

(4)$B_{k+1}(X_{k+1}),\cdots,B_m(X_m)$是其前出现失败即否定 not 的规则谓词；

(5)$Y, X_1,\cdots,X_m$为任意元项的序列，而$Z_1,\cdots,Z_n$为一元或二元项的序列。

同时，为了保证推理的可判定性，需对规则增加安全限制：规则头中出现的所有变量必须出现在规则体中谓词的变量中。

动作的语法（定义 2 和定义 19）给出了动作的形式化表示，而动作的分解原则（定义 22 和定义 23）把动作的推理分解为若干条单调规则和非单调规则的形式进行推理。对某个原子动作，根据其语义可以分解为如下的规则：$\leftarrow\{Pre,\varphi_1\}$，$\psi_2\leftarrow\{Pre,\varphi_2\},\cdots,\psi_n\leftarrow\{Pre,\varphi_n\}$，并且$\psi_1\vee\psi_2\vee\cdots\vee\psi_n=\psi$，$\varphi_1\vee\varphi_2\vee\cdots\vee\varphi_n=\varphi$，这些分解出来的规则可以命名为动作分解规则。在应用领域中，这些动作分解规则通过一定的推理算法得到的该应用领域中知识的改变构成了基于 T-ALC 和逻辑程序的动作的语义。

4.2 集成 T-ALC 与逻辑程序的动作推理

在基于 T-ALC 与逻辑程序的动作形式体系中，首先对原子动作的 *Pre* 和 *Post* 集合中允许出现的谓词的形式和构造算子进行限制，使其中的动作具有很强的表达能力；然后根据动作的执行对应用领域的影响，即原子动作的语义，将其中确定性的动作转化为若干条 Datalog 规则的形式，不确定性动作转化为若干条 Datalog 规则或 Datalog$\neg$规则的形式。因此，在该系统中动作推理的主要任务是集成描述逻辑 T-ALC 与规则对动作分解规则的推理，共同得到动作的执行结果。

除此之外，动作的推理还包括在动作执行之前动作的一致性检测、动作的可执行性和可投影问题等，这些推理任务与基于 T-ALC 的动作形式体系中动作的推理类似。

4.2.1 确定性动作的执行结果

根据定义 22 的确定性动作的分解、某个确定性动作A的 *Post* 集合中的ψ，对ψ中出现的规则谓词进行分解得出的动作的分解规则主要存在形如$H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n)$形式的 Datalog 规则。由于这种形式规则的规则头都为描述逻辑谓词，则根据描述逻辑知识库可以判定其规则头中各个谓词的可满足性，从而得到其规则体中的结果。对文中出现的描述逻辑谓词，其前提条件和后置结果都为描述逻辑谓词，也可根据 A1 中的推理方法得到A执行之后的部分结果。

因此，以 T-ALC 为例，若存在 A2 的一个知识库$K=\langle K_{T\text{-}ALC}, K_R\rangle$，确定性原子动作$A$在时间$[t,t']$内执行。若在时间$t$，$A$是可执行的，即$A$的前置条件在当前知识是可满足的$(K|=Pre)$，$A$执行之后，$A$的执行结果的步骤为：

(1)A的 *Post* 集合中的ψ，对ψ中出现的描述谓词的集合，采用 A1 中动作的执行结果的推理方法得到A执行之后在时间t'的部分结果，记为Res_1，并将Res_1加入$K_{T\text{-}ALC}$中；

(2)A的 *Post* 集合中的ψ，对ψ中出现的规则谓词的集合，使用定义 22 对A进行分解，得到若干条A的动作分解规则的集合，记为R。由于R中所有规则的规则头都为A的 *Pre* 集合，若$K|=Pre$，所有的规则都成立，则可得到A执行之后在时间t'的剩余结果，记为Res_2，并将Res_2加入K_R的事实F_T中；

(3)对(1)中得到的$K_{T\text{-}ALC}$，采用定义 14 中的惯性原则

PoI，得到动作 A 执行之后合并的新描述逻辑知识 $K_{T\text{-}ALC'}$；对(2)中得到事实 F_T，采用类似的惯性原则，得到动作 A 执行之后合并的新的事实 F_T'；

(4)$Res_1 \cup Res_2$ 即为确定性原子动作 A 的执行结果，$K'=\langle K_{T\text{-}ALC'}, K_{R'}\rangle$即为 A 执行之后 A2 发生改变后得到的新知识库。

4.2.2 不确定性动作的执行结果

根据定义 23 的不确定性动作的分解，对某个不确定性动作 A 的 $Post$ 集合中的 ψ 中出现的规则谓词进行分解得出的动作的分解规则主要存在以下 3 种形式：

(1)$H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n)$形式的 Datalog 规则；

(2)$H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n),B_1(X_1),\cdots,B_m(X_m)$形式的 Datalog 规则；

(3)$H(Y)\leftarrow B_1(X_1),\cdots,B_k(X_k),D_1(Z_1),\cdots,D_n(Z_n)$, not $B_{k+1}(X_{k+1}),\cdots,$not $B_m(X_m)$形式的 Datalog¬规则。

以上(1)与确定性原子动作的分解规则相同，其处理方式也相同。这里重点考虑(2)(3)两种形式的推理。由于(2)(3)中动作分解规则的规则体既包含描述逻辑谓词，又包含规则谓词，因此这两种形式的规则的推理还包含对规则知识库的推理。而文中出现的描述逻辑谓词，其前提条件和后置结果都为描述逻辑谓词，也可根据 A1 中的推理方法得到 A 执行之后的部分结果。下面针对规则知识库，详细给出规则谓词的可满足性判定算法。

定义 24 若存在基于 T-ALC 和规则的动作形式化表示体系的一个知识库 $K=\langle K_{T\text{-}ALC}, K_R\rangle$，其中规则知识库 $K_R=\langle(R,Q),F_T\rangle$，即 K_R 除了包含形如 $H(a_1,a_2,\cdots a_n)'$ 或 $H(a_1,a_2,\cdots,a_n)^{[t_1,t_2]}$形式的事实 F_T 外，还包含形如 $H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n)$或 $H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n),B_1(X_1),\cdots,B_m(X_m)$的正规则，或形如¬$H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n)$或¬$H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n),B_1(X_1),\cdots,B_m(X_m)$的负规则所组成的集合 R，及 R 中规则的优先序 Q。若 R 中的规则 R_1 的优先级大于规则 R_2，则表示为 $R_1>R_2$。因此，Q 为一个二元的结构，且不存在环。其中：

(1)$H(Y)$称为规则头，属于规则谓词；

(2)$B_1(X_1),\cdots,B_m(X_m)$为任意元谓词，属于规则谓词；

(3)$D_1(Z_1),\cdots,D_n(Z_n)$为描述逻辑谓词；

(4)$Y,X_1,\cdots,X_m$ 为任意元项的序列，而 $Z_1,\cdots,Z_n$ 为一元或二元项的序列。

同时，为了保证知识库的可判定性，需对规则增加安全限制：规则头中出现的所有变量必须出现在规则体的谓词中；还需对规则库增加非循环语义限制，其定义如下。

定义 25 若存在基于 T-ALC 和规则的动作形式化表示体系的一个知识库 $K=\langle K_{T\text{-}ALC}, K_R\rangle$，其中规则知识库为 $K_R=\langle(R,Q),F_T\rangle$，其非循环语义限制如下，即对 K_R 中的规则集 R 中的每条规则：

(1)对 R 中的所有规则头谓词 $H(Y)$，若在 R 中仅含有形如 $H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n)$或¬$H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n)$的规则，则在经过某个合一之后，$H(a_1,a_2,\cdots,a_n)$的可满足性仅由描述逻辑知识库就可以推出，称这类谓词为非循环规则头谓词。

(2)对 R 中形如 $H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n),B_1(X_1),\cdots,B_m(X_m)$或¬$H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n),B_1(X_1),\cdots,B_m(X_m)$形式的规则，若其规则体中的某个规则谓词为(1)中的非循环规则头谓词，则该谓词可以判定它的可满足性，可以在该规则体中删除此规则头谓词。

(3)在 R 中重复(1)(2)，若 R 中的所有规则谓词都为非循环规则头谓词，那么该规则知识库满足非循环语义限制；若存在某个规则谓词不满足非循环规则头谓词的条件，那么该规则知识库不满足非循环语义限制。

若规则知识库 K_R 满足安全限制和非循环语义限制，则只需根据描述逻辑知识库就可以判定 K_R 中规则谓词的可满足性。对非循环规则头谓词，存在的正规则和负规则两种形式的表示，主要是为了增加对知识的例外进行处理。对于规则知识库中正负规则使用的选择，存在以下选择算法：

算法 1

Input：T-ALC 知识库 $K_{T\text{-}ALC}=\langle\mathcal{T},\mathcal{A}_T\rangle$和规则知识库 $K_R=\langle(R,Q),F_T\rangle$，不含失败即 not 的非循环规则头谓词 H(Y)。

Output：对 H(Y)，找出可使用的规则，得到可满足的 H($\overline{Y}$)或¬H($\overline{Y}$)($\overline{Y}$ 为 Y 的一个实例化)。

Begin：

Step1 遍历 R 中的所有规则，找出规则头为 H(Y)或¬H(Y)的规则，若同时存在关于 H(Y)的正规则和负规则，则记负规则为 $R_1,\cdots,R_n(n\geqslant 0)$，记正规则为 $R_1',\cdots,R_m'(m\geqslant 0)$，优先使用负规则。

Step2 若不存在负规则(n=0)，则转 Step5。

Step3 若只存在一条负规则(n=1)，则判定在 $K_{T\text{-}ALC}$下 R_1 的规则体的可满足性。若规则体中的谓词都是可满足的，则使用该规则，返回¬H($\overline{Y}$)；若规则体中存在一个不可满足或不可判定的谓词，则删除该规则，并转 Step5。

Step4 若同时存在多条负规则(n>1)：①不存在优先序，则先逐条判定在 $K_{T\text{-}ALC}$下规则 $R_1,\cdots,R_n$ 的规则体中所有谓词的可满足性。②存在优先序，则按照优先序有序地判定在 $K_{T\text{-}ALC}$下规则 $R_1,\cdots,R_n$ 的规则体中所有谓词的可满足性。若某条规则的规则体中所有的谓词都是可满足的，则使用该规则，返回¬H($\overline{Y}$)；若某条规则的规则体中存在一个不可满足或不可判定的谓词，则删除该规则，继续判定后续规则，直到所有负规则都删除完，则转 Step5。

Step5 若不存在正规则(m=0)，转 Step8。

Step6 若只存在一条正规则(m=1)，则判定在 $K_{T\text{-}ALC}$下 R_1'的规则体的可满足性。若规则体中的谓词都是可满足的，则使用该规则，返回 H($\overline{Y}$)；若规则体中存在一个不可满足或不可判定的谓词，则删除该规则，并转 Step8。

Step7 若同时存在多条正规则(m>1)：①不存在优先序，则先逐条判定在 $K_{T\text{-}ALC}$下规则 $R_1',\cdots,R_m'$的规则体中所有谓词的可满足性。若某条规则的规则体中所有的谓词都是可满足的，则使用该规则，得到结果 H($\overline{Y}_1$)，并继续判定后续规则；若某条规则的规则体中存在一个不可满足或不可判定的谓词，则删除该规则，继续判定后续规则；若还存在可使用的规则，继续使用该规则，得到结果 H($\overline{Y}_2$)，直到所有正规则都判定完，返回所有可满足的结果 H($\overline{Y}_1$)，H($\overline{Y}_2$)，…，H($\overline{Y}_k$)(k≤m)；若所有正规则都删除完，还没有规则可使用，则转 Step8；②存在优先序，则先按照优先序有序地判定在 $K_{T\text{-}ALC}$下存在优先级的规则 $R_1',\cdots,R_t'(t\leqslant m)$的规则体中所有谓词的可满足性。若含有优先级的某条规则的规则体中所有的谓词都是可满足的，则使用该规则，返回 H($\overline{Y}$)；若含有优先级的某条规则的规则体中的谓词存在一个不可满足或不可判定的谓词，则删除该规则，继续判定后续规则，直到所有含有优先序的正

规则都删除完，还没有可返回的结果，则按照 Step7①判定无优先级的正规则 R_{t+1}'，…，R_m'，并返回相应的结果。

Step8　所有规则都判定完，无规则可用，H(Y)不可判定。

End

以上，通过集成描述逻辑知识库 $K_{T\text{-}ALC}$ 和规则知识库 K_R，提出了针对动作分解规则中形如 $H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n),B_1(X_1),\cdots,B_m(X_m)$ 形式的 Datalog 规则中的规则谓词 $B_1(X_1),\cdots,B_m(X_m)$ 在 K_R 的可满足性推理算法。而对动作分解规则中形如 $H(Y)\leftarrow B_1(X_1),\cdots,B_k(X_k),D_1(Z_1),\cdots,D_n(Z_n),\text{not } B_{k+1}(X_{k+1}),\cdots,\text{not } B_m(X_m)$ 形式的 Datalog 规则，其规则体中还允许出现否定即失败 not。规则谓词 $B_{k+1}(X_{k+1}),\cdots,B_m(X_m)$ 在规则知识库 K_R 满足非循环语义限制的前提下，通过规则知识库 K_R 对它们的可满足性进行判定时是基于描述逻辑的开放世界假设的，而通过动作分解规则进行推理时是基于封闭世界假设的。

定义 26　对否定即失败 not，若在某条规则的规则体中存在 not $B(X)$ 项，则尝试证明 $B(X)$ 为真。若 $B(X)$ 不为真（为假或不可判定），则 not $B(X)$ 为真，可以去掉规则体中的 not $B(X)$ 项；若 $B(X)$ 为真，则转化为经典的否定 $B(X)$。

根据规则知识库 K_R 的非循环语义限制，规则谓词 $B_{k+1}(X_{k+1}),\cdots,B_m(X_m)$ 总能在 K_R 中经过有限次转换成为规则体中仅含有描述谓词的规则，而选择使用规则的选择算法则根据描述逻辑的推理及其开放世界假设，总可以判定规则谓词 $B_{k+1}(X_{k+1}),\cdots,B_m(X_m)$ 是可满足、或不可满足的、或不可判定的。这样，根据定义 25 对含有否定即失败 not 的规则谓词 not $B_{k+1}(X_{k+1}),\cdots,$ not $B_m(X_m)$ 进行处理，便可以将 $H(Y)\leftarrow B_1(X_1),\cdots,B_k(X_k),D_1(Z_1),\cdots,D_n(Z_n),\text{not } B_{k+1}(X_{k+1}),\cdots,\text{not } B_m(X_m)$ 形式的 Datalog 规则转化为 $H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n),B_1(X_1),\cdots,B_m(X_m)$ 形式的 Datalog 规则进行处理。

综上，若存在 A2 的一个知识库 $K=\langle K_{T\text{-}ALC},K_R\rangle$，$K_R$ 满足非循环语义限制，则不确定性原子动作 A 在时间 $[t,t']$ 内执行。若在时间 t，A 是可执行的，即 A 的前置条件在当前知识是可满足的 $(K|=Pre)$，A 执行之后，得到 A 的执行结果的步骤为：

(1)对 A 的 $Post$ 集合中的 ψ 中出现的描述谓词的集合，采用 A1 中动作的执行结果的推理方法得到 A 执行之后在时间 t' 的部分结果，记为 Res_1，并将 Res_1 加入 $K_{T\text{-}ALC}$ 中。

(2)对 A 的 $Post$ 集合中的 ψ 中出现的规则谓词的集合，使用定义 22 对 A 进行分解，得到若干条 A 的动作分解规则的集合，记为 R。得到的形如 $H(Y)\leftarrow D_1(Z_1),\cdots,D_n(Z_n),B_1(X_1),\cdots,B_m(X_m)$ 形式动作分解规则的集合，记为 R_1；得到的形如 $H(Y)\leftarrow B_1(X_1),\cdots,B_k(X_k),D_1(Z_1),\cdots,D_n(Z_n),\text{not } B_{k+1}(X_{k+1}),\cdots,\text{not } B_m(X_m)$ 形式动作分解规则的集合，记为 R_2。

(3)由于 R_1 中所有规则的规则头为描述逻辑谓词或非循环规则头谓词，可以根据算法 1 判定该规则头谓词的可满足性。对 R_1 中的每一条规则，若其规则体谓词都是可满足的，那么 A 执行之后，在时间 t' 其规则头谓词成立。将 R_1 的所有结果记为 Res_2，并将 Res_2 加入 K_R 的事实 F_T 中。

(4)由于 R_2 中所有规则的规则头为描述逻辑谓词或非循环规则头谓词，并允许是否即否定 not 出现在部分规则谓词之前，则可以根据以上正负规则的选择算法判定该规则头谓词是否是可满足的或不可判定的。再根据定义 26 去掉规则谓词前的 not，得到新的规则集合 R_2'，R_2' 中所有的规则的形式同 R_1，然后同(3)，针对 R_2' 中的每一条规则，若其规则体谓词都是可满足的，那么 A 执行之后，在时间 t' 其规则头谓词成立。将 R_2' 的所有结果记为 Res_3，并将 Res_3 加入 K_R 的事实 F_T 中。

(5)对(1)中得到的 $K_{T\text{-}ALC}$，采用定义 12 中的惯性原则 PoI，得到动作 A 执行之后合并的新描述逻辑知识 $K_{T\text{-}ALC}'$；对(4)中得到事实 F_T，采用类似的惯性原则，得到动作 A 执行之后合并的新的事实 F_T'。

(6)$Res_1\cup Res_2\cup Res_3$ 即为不确定性原子动作 A 的执行结果，$K'=\langle K_{T\text{-}ALC}',K_R'\rangle$ 即为 A 执行之后得到的新的知识库。

5　实例分析

下面以银行业务系统为例，对其中的对象、规则和动作进行表示，构建一个基于 T-ALC 和规则的动作形式体系 *BankService*。

首先，给出系统基于描述逻辑的知识库 $K_{T\text{-}ALC}=\langle\mathcal{T},\mathcal{A}_T\rangle$：

(1)术语集 $\mathcal{T}=\{Person=Woman\sqcup Man, hasParent=hasFather\sqcup hasMother, hasParent=hasChild^-, BankCard=DebitCard\sqcup CreditCard\}$

(2)术语集 $\mathcal{A}_T(t=0)=\{Person(John), hasAge(John,42), Person(Mary), hasAge(Mary,16), hasFather(Mary,John), Credit(Mary,\text{“}A\text{”})\}$

其次，对系统中难以用描述逻辑表示的知识使用规则知识库 $K_R=\langle(R,Q),F_T\rangle$ 来表示。

(3)规则集 $R=\{$

R_1：$allowBankCard(x,y)\leftarrow Person(x)\wedge Bank(y)\wedge hasAge(x,\geqslant 18)$（表示“办卡人的年龄在 18 岁以上允许办卡”）；

R_2：$allowBankCard(x,y)\leftarrow Person(x)\wedge hasAge(x,<18)\wedge Person(x_1)\wedge hasParent(x,x_1)\,BankCard(z)\wedge hasBankCard(x_1,z)\wedge Bank(y)\wedge CardBelongTo(z,y)$（表示“办卡人的年龄不满 18 岁，但他的父母有该银行的银行卡，则允许办卡”）；

R_3：$allowDebitCard(x,y)\leftarrow allowBankCard(x,y)$（表示“允许办卡，则允许办储蓄卡”）；

R_4：$allowCreditCard(x,y)\leftarrow Person(x)\wedge Bank(y)\wedge allowBankCard(x,y)\wedge IncomeCertificate(z)\ hasIncomeCertificate(x,z)$（表示“若允许办卡，并且有工资证明，则允许办信用卡”）；

R_5：$\neg allowDebitCard(x,y)\leftarrow Person(x)\wedge Bank(y)\ DebitCard(z)\wedge CardBelongTo(z,y)\wedge hasBankCard(x,z)$（表示“如果已经办了某银行的储蓄卡，则不允许再办同一银行的储蓄卡”）；

R_6：$\neg allowCreditCard(x,y)\leftarrow Person(x)\wedge Bank(y)\wedge CreditCard(z)\wedge CardBelongTo(z,y)\ hasBankCard(x,z)$（表示“如果已经办了某银行的信用卡，则不允许再办同一银行的信用卡”）；

R_7：$\neg allowDebitCard(x,y)\leftarrow Person(x)\wedge Bank(y)\wedge$

$CreditCard(z) \wedge CardBelongTo(z,y) \wedge hasBankCard(x,z) \wedge Credit(z,<\text{“}A\text{”})$，表示“已经办了某银行的信用卡，但信用不好，则不允许办同一银行的储蓄卡”

}。

其中的有限序为 $Q=\{R_6>R_1, R_6>R_2\}$。

最后，给出系统中所包含的动作，即 K_A。在银行服务领域，动作为某人对他的账户进行的所有操作。这里定义开户（*OpenAccount*）、存款（*DespositMoney*）、取款（*WithdrawMoney*）、转账（*TransferAccount*）、挂失（*ReportLoss*）、销户（*CloseAccount*）这几个动作，它们的表示如下：

$K_A=\{$

$OpenDebitAccount(x, y, T) = (\{Person(x)^{t_1}, Bank(y)^{t_1}\}, \{\{allowDebitCard(x,y)^{t_1}\}/\{AccountUser(x)^{t_2}, hasBankCard(x,z)^{t_2}, DebitCard(z)^{t_2}, hasBalance(z,0)^{t_2}\}\})$

$OpenCreditAccount(x, y, T) = (\{Person(x)^{t_1}, Bank(y)^{t_1}\}, \{\{allowCreditCard(x, y)^{t_1}\}/\{AccountUser(x)^{t_2}, hasBankCard(x, z)^{t_2}, CreditCard(z)^{t_2}, hasBalance(z, 0)^{t_2}\}\})$

$DespositMoney(x,y,z,T)=(\{Person(x)^{t_1}, BankCard(y)^{t_1}, hasBankCard(x,y)^{t_1}, Money(z)^{t_1}\}, \{\text{not } frozenAccount(y)^{t_1} / hasBalance(y, balance+z)^{t_2}\})$

$WithdrawMoney(x,y,z,T)=(\{Person(x)^{t_1}, BankCard(y)^{t_1}, hasBankCard(x,y)^{t_1}, Money(z)^{t_1}\}, \{\text{not } frozenAccount(y)^{t_1} / hasBalance(y, balance-z)^{t_2}\}\})$

$TransferAccount(x, y_1, y_2, z, T) = (\{Person(x)^{t_1}, BankCard(y_1)^{t_1}, BankCard(y_2)^{t_1}, hasBankCard(x, y_1)^{t_1}, Money(z)^{t_1}\}, \{\text{not } frozenAccount(y)^{t_1} / \{hasBalance(y_1, balance_1-z)^{t_2}, hasBalance(y_2, balance_2+z)^{t_2}\}\})$

$ReportLoss(x,y,T)=(\{Person(x)^{t_1}, BankCard(y)^{t_1}, hasBankCard(x,y)^{t_1}\}, \{frozenAccount(y)^{t_2}\})$

$CloseAccount(x, y, T) = (\{Person(x)^{t_1}, BankCard(y)^{t_1}, hasBankCard(x,y)^{t_1}\}, \{\neg hasBankCard(x,y)^{t_2}\})$

}

根据动作的定义，可见 *ReportLoss* 和 *CloseAccount* 为确定性动作，而 *OpenDebitAccount*、*OpenCreditAccount*、*DespositMoney*、*WithdrawMoney* 和 *TransferAccount* 为不确定性动作，这主要是因为其 *Post* 集合中包含 φ/ψ，并且 φ 不恒为真。其中，not *frozenAccount* 表示如果某个账户冻结了，那么不能正常地进行存钱、取钱和转账操作。

在时间 $t=2$，若 *Mary* 想办一张 ABC 的储蓄卡，则需执行动作 $OpenDebitAccount(x,y,T)$，先进行实例代换 $\{Mary/x, ABC/y\}$，然后根据 4.2.2 节中不确定性动作的执行步骤：

(1)在执行动作之前，其 *Pre* 集合 $\{Person(Mary)^2, Bank(ABC)^2\}$ 若是可满足的，则 $OpenDebitAccount(Mary,ABC)$ 是可执行的。

(2)对其 *Post* 集合的 φ，存在规则谓词 *allowDebitCard*，即需先判定规则谓词 $allowDebitCard(Mary,ABC)^2$ 的可满足性。

(3)根据规则知识库中正负规则的选择算法，优先使用负规则 R_5 和 R_7，但是根据 R_5 和 R_7 两条规则 $allowDebitCard(x,y)$ 是不可满足的，因此需要继续判断正规则 R_3，要判断规则谓词 $allowDebitCard(Mary,ABC)^2$，则需判断 $allowBankCard(Mary, ABC)^2$。根据 R_1，$allowBankCard(Mary, ABC)^2$ 是不可满足的，但根据 R_2，$allowBankCard(Mary, ABC)^2$ 是可满足的。因此，$allowDebitCard(Mary,ABC)^2$ 是可满足的。

(4)根据动作的分解规则，将 $OpenDebitAccount(Mary, ABC)$ 分解为若干条确定性规则 $AccountUser(Mary)$，$hasBankCard(Mary, dCard)$，$DebitCard(dCard)$，$hasBalance(dCard,0) \leftarrow Person(Mary) \wedge Bank(ABC) \wedge allowDebitCard(Mary,ABC)$。

(5)执行动作之后，$t=3$，得到其执行结果为：$AccountUser(Mary)^3$，$hasBankCard(Mary,ABC)^3$，$DebitCard(dCard)^3$，$hasBalance(dCard,0)^3$，然后更新 ABox $\mathcal{A}_T(t=3)=\{Person(John)^{[0,3]}, hasAge(John, 42)^{[0,3]}, Person(Mary)^{[0,3]}, hasAge(Mary, 16)^{[0,3]}, hasFather(Mary, John)^{[0,3]}, hasBankCard(Mary, ABC)^3, AccountUser(Mary)^3, DebitCard(dCard)^3, hasBalance(dCard,0)^3\}$，更新事实集 $F_T(t=3)=\{allowDebitCard(Mary,ABC)^{[2,3]}\}$。

若在时间 $t=10$，当前 $F_T(t=10)$ 中包含 $frozenAccount(dCard)^{10}$，则 $F_T|=frozenAccount(dCard)^{10}$。若执行动作 $DespositMoney(Mary,dCard,100)$，虽然它是可执行的，但 $frozenAccount(dCard)^{10}$ 是可满足的，那么根据定义 26，not $frozenAccount(dCard)^{10}$ 转化为经典的否定 $\neg frozenAccount(dCard)^{10}$。因此，$\neg frozenAccount(dCard)^{10}$ 是不可满足的，该动作不可执行。同理，在执行其他不确定性动作 *WithdrawMoney* 和 *TransferAccount* 时，首先也是要根据 *frozenAccount* 在规则知识库中的可满足性，去掉失败即否定 not。

在这个实例中，仅以描述逻辑或逻辑程序都无法完全表示该系统中的知识，并且对于其中出现的动作，根据实际情况的不同，即使动作是可执行的，但由于出现失败即否定 not，也会导致该动作不可执行。

结束语 基于 T-ALC 和逻辑程序的动作形式化表示方法整合了描述逻辑、动作理论和逻辑程序，在动作表示的 *Post* 集合中允许在出现描述逻辑谓词、规则谓词及在规则谓词之前出现否定即失败 not，从而大大增强了动作的表示能力，增强了知识表示能力。根据动作的语义理解，将确定性动作转化为若干条 Datalog 规则，将不确定性动作转化为若干条 Datalog 规则和 Datalog¬规则进行处理，并给出了基于扩展 T-ALC 和逻辑程序的动作的语法和语义。重点研究了确定性动作和不确定性动作的执行结果的推理，并将它们转化为整合扩展描述逻辑与逻辑程序的推理。然而，在计算机中将互联网中的文本信息以事件为单位表示出来，是一项富有挑战性的工作，特别是很多事件并不符合定义 1 中的事件定义，事件要素也存在缺省。本研究是在对事件知识进行抽取的基础之上，从知识表示与知识运用的角度出发，研究了明确存在对象、动作、时间、断言等要素的这一类事件中的动作表示与推理。由于事件与事件存在复杂的语义联系[4]，动作之间也存在着这些语义关联，因此，关于动作之间关系的研究将是需要进一步深入研究的方向。

（下转第 145 页）

[5] Wang Li-zhen, Bao Yu-zhen, Lu J, et al. A new join-less approach for co-location pattern mining[C]// Proceedings of the IEEE 8th International Conference on Computer and Information Technology (CIT 2008). Sydney, Australia, 2008: 197-202

[6] Wang Li-zhen, Zhou Li-hua, Lu J, et al. An order-clique-based approach for mining maximal co-locations[J]. Information Sciences, 2009, 179(19): 3370-3382

[7] 陆叶,王丽珍,张晓峰. 从不确定数据集中挖掘频繁 Co-location 模式[J]. 计算机科学与探索, 2009, 3(6): 656-664

[8] Wang Li-zhen, Chen Hong-mei, Zhao Li-hong, et al. Efficiently mining co-location rules on interval data[C]// Proceedings of the 6th Int Conf on Advanced Data Mining and Applications (ADMA 2010). Chongqing, China, Part I, LNCS 6440, 2010: 477-488

[9] Altman D. Fuzzy set theoretic approaches for handling imprecision in spatial analysis[J]. International Journal of Geographical Information Science, 1994, 8(3): 271-289

[10] Zheng Kai, Fung Pui-cheong, Zhou Xiao-fang. K-nearest neighbor search for fuzzy objects[C]// Proceedings of the Special Interest Group on Management of Data (SIGMOD'10). Indiana, USA, 2010: 699-710

[11] 欧阳志平,王丽珍,陈红梅. 模糊对象的空间 co-location 模式挖掘研究[J]. 计算机学报, 2011, 34(10): 1947-1955

[12] 欧阳志平,王丽珍,周丽华. 实例位置模糊的空间 co-location 模式挖掘研究[J]. 计算机科学与探索, 2012, 6(12): 1144-1152

[13] 吴萍萍. 模糊 Co-Location 模式挖掘[D]. 昆明:云南大学, 2012

[14] Hirate Y, Iwahashi E, Yamana H. An Efficient Algorithm for Mining Frequent Patterns without any Thresholds[C]// Proc. of Workshop on Alternative Techniques for Date Mining and Knowledge Discovery. 2004

[15] Bow M. Compact Co-location Pattern Mining [D]. Indiana: Indiana University, 2011

(上接第 125 页)

参 考 文 献

[1] Zacks J M, Tversky B. Event Structure in Perception and Conception [J]. Psychological Bulletin, 2001, 127(1): 3-21

[2] Nelson K, Gruendel J. Event knowledge: Structure and function in development [M]. Hilladale, NJ: Erlbaum, 1986

[3] ACE (Automatic Content Extraction). Chinese Annotation Guidelines for Events[S]. National Institute of Standards and Technology, 2005

[4] 刘宗田,黄美丽,周文,等. 面向事件的本体研究[J]. 计算机科学, 2009, 6(11): 189-192

[5] 戈也挺,朱朝晖,陈世福. 行动推理中若干问题的研究[J]. 计算机科学, 2000, 27: 85-89

[6] 黄智生. 关于行动的推理[J]. 计算机科学, 1993, 20: 7-13

[7] McCarthy J. Situations, actions, and causal laws[R]. Stanford, California: Stanford University Artificial Intelligence Project, 1963

[8] Shanahan M. The event calculus explained [M]. Artificial Intelligence Today, Spring Berlin Heidelberg, 1999: 409-430

[9] 史忠植,董明楷,蒋运承,等. 语义 Web 的逻辑基础[J]. 中国科学:E 辑, 2004, 34(10): 1123-1138

[10] 常亮,史忠植,陈立民,等. 一类扩展的动态描述逻辑[J]. 软件学报, 2010, 21(1): 1-13

[11] Chang Liang, Shi Zhong-zhi, Qiu Li-rong, et al. Dynamic description logic: embracing actions into description logic[C]// Proc. of the 20th International Workshop on Description Logics (DL'07). Italy: 2007: 243-253

[12] Baader F, Lippmann M, Liu Hong-kai. Using causal relationships to deal with the ramification problem in action formalisms based on description logics[C]// Logic for Programming, Artificial Intelligence, and Reasoning. Springer Berlin Heidelberg, 2010, 6397: 82-96

[13] Liu Wei, Xu Wen-jie, Wang Dong, et al. An extending description logic for action formalism in event ontology [J]. International Journal of Computational Science and Engineering, 2010, 6104: 471-481

[14] Martinez DC, Hitzler P. Extending Description Logic Rules[C]// Proceedings of 9th Extended Semantic Web Conference. Heraklion, 2012: 345-359

[15] Grosof B N, Horrocks I, Volz R, et al. Description logic programs: Combining logic programs with description logic[C]// Proceedings of the 12th International Conference on World Wide Web. ACM, 2003: 48-57

[16] Horrocks I, Patel-SchneiderP F. A proposal for an OWL rules language[C]// Proceedings of the 13th international conference on World Wide Web. ACM, 2004: 723-731

[17] Donini F M, Lenzerini M, Nardi D, et al. AL-log: Integrating datalog and description logics [J]. Journal of Intelligent Information Systems, 1998, 10(3): 227-252

[18] Levy A Y, Rousset M C. CARIN: a representation language combining horn rules and description logics[C]// ECAI. Pitman, 1996: 323-327

[19] Rosati R. Towards expressive KR systems integrating Datalog and description logics: Preliminary report [C]// Proc. of DL'99. 1999: 160-164

[20] Rosati R. On the decidability and complexity of integrating ontologies and rules [J]. Web Semantics: Science, Services and Agents on the World Wide Web, 2005, 3(1): 61-73

[21] Mei Jing, Lin Zuo-quan, Boley H. ALCP^{u}: An Integration of Description Logic and General Rules[C]// Proceedings of the 1st International Conference on Web Reasoning and Rule Systems. Heidelberg: Springer-verlag Berlin, 2007: 163-177

[22] Schmidt-Schauβ M. Subsumption in KL-ONE is Undecidable. Principles of Knowledge Representation [M]. Stanford, CA, US: CSLI Publications, 1989: 421-431

[23] Yehia W, Liu H K, Lippmann M, et al. Experimental results on solving the projection problem in action formalisms based on description logics[C]// Proc. of the 25th Intern. Workshop on Description Logics. 2012

[24] Coelho H. Verifying properties of infinite sequences of description logic actions[C]// ECAI 2010: 19th European Conference on Artificial Intelligence. IOS Press, Incorporated, 2010: 53-58

[25] Drabent W, Eiter T, Ianni G, et al. Hybrid reasoning with rules and ontologies [M]. Semantic techniques for the Web, Springer Berlin Heidelberg, 2009: 1-49

[26] Allen J F. Temporal reasoning and planning[C]// Reasoning about Plans. Morgan Kaufmann, 1991: 1-67