在部分观测环境下学习规划领域的派生谓词规则

饶东宁[1] 蒋志华[2] 姜云飞[3] 邓玉辉[2]

[1](广东工业大学计算机学院 广州 510090)

[2](暨南大学信息科学与技术学院计算机科学系 广州 510632)

[3](中山大学信息科学与技术学院软件研究所 广东 510275)

摘 要 文中提出了一种在部分观测环境下学习规划领域的派生谓词规则的方法.在规划领域描述语言(PDDL)中,派生谓词用来描述动作的非直接效果,是规划领域模型和搜索控制知识的重要组成部分.然而,对于大多数规划领域而言,从无到有地构造派生谓词规则是不容易的.因此,研究自动获取派生谓词的推导规则是有意义的.已有研究工作提出通过修订一个初始的不完备的领域理论来获取推导规则的方法,但是它们的主要缺点在于待学习谓词的训练例的数量非常少,这是因为训练例按照非常有限的方式来生成.而更本质的原因在于它们假设环境是不可观测的.其实,在现实生活中很多动作的非直接效果是可以观测的,或者通过简单的目测或者通过专门的工具.因此文中提出增加观测来反映动作的非直接效果,以便增加待学习谓词的训练例数目从而改善学习的精准度.此外,为了补充一些在归纳学习过程中学习不到的谓词,文中还提出了一个后处理方法来使得学习到的规则在语义上更完整.通过在派生谓词基准领域上的实验表明,文中所提出的方法是可行有效的.更深远的意义在于,文中的研究工作有利于规划领域的自动建模或者控制知识的自动获取的研究与实现.

关键词 人工智能;自动规划;派生谓词;规则学习;部分观测

中图法分类号 TP182 **DOI号** 10.11897/SP.J.1016.2015.01372

Learning Derived Predicate Rules for Planning Domains under Partial Observability

RAO Dong-Ning[1] JIANG Zhi-Hua[2] JIANG Yun-Fei[3] DENG Yu-Hui[2]

[1](*School of Computer, Guangdong University of Technology, Guangzhou* 510090)

[2](*Department of Computer Science, School of Information Science and Technology, Jinan University, Guangzhou* 510632)

[3](*Software Research Institute, School of Information Science and Technology, Sun Yat-sen University, Guangzhou* 510275)

Abstract This paper presents a method to learn derived predicate rules for planning domains under partial observability. In the PDDL (Planning Domain Description Language), derived predicates are a compact way to describe indirect effects of actions, and an important part of planning domain models or search control knowledge. However, for most planning domains, it is not easy to write derived predicate rules from scratch, even for experts. Therefore, it is worthy of studying how to automatically acquire rules for derived predicates from observed plans. There has been some research work on gaining derived rules by refining an initial and imperfect domain theory. But, their primary disadvantage was that the number of training examples for predicates to be learned was very small since training examples were produced in a very limited way. The underlying reason was that they assumed that the environment was unobservable. In fact, in the real world, many indirect effects of actions are observable by simple eye-measurement or tools. This paper uses observations to reflect actions' indirect effects in order to increase the number of training

收稿日期:2013-12-28;最终修改稿收到日期:2015-01-12.本课题得到中央高校基本科研业务费专项资金(21615438)、国家自然科学基金(61100134,61003179,61272073)及广东省自然科学基金(S2013020012865,S2011040001427)资助.**饶东宁**,男,1977年生,博士,副教授,中国计算机学会(CCF)会员,主要研究方向为智能规划、图论.E-mail:raodn@gdut.edu.cn.**蒋志华**,女,1978年生,博士,副教授,主要研究方向为智能规划.**姜云飞**,男,1945年生,硕士,教授,主要研究领域为智能规划、模型诊断.**邓玉辉**,男,1973年生,博士,教授,主要研究领域为数据存储和绿色计算.

examples and to improve the learning accuracy. Also, to complement some predicates which cannot be learned by the inductive learning method, this paper gives a post-processing algorithm to make the semantics of learned rules more perfect. Experiments on some benchmark domains show that, the method presented in this paper is feasible and effective. And further, the work in this paper is beneficial for the study on automatically modeling planning domains and automatically acquiring control knowledge.

Keywords artificial intelligence; automated planning; derived predicates; rule learning; partial observability

1 引　言

在智能规划(automated planning)[1]研究领域中,派生谓词(derived predicates)有两个主要的用途:(1)用于领域建模[2];(2)用于描述控制知识[3-4].作为领域建模的一部分,派生谓词用来描述动作的非直接效果(indirect effects),这样动作模型就变得简洁从而加速动作的搜索过程.而作为控制知识的一部分,派生谓词用来描述有利于目标求解的情境,从而引导求解过程按照期望的方向进行.不管是上述哪一种用途,派生谓词的获取要么由人类专家提供[5-6],要么由程序从规划解中自动提取[4,7-8].前一种方式耗时且易出错,而后一种方式目前还有很大的局限性,例如训练例少[7]或依赖于固定的规则模式[4,8]等.

为了同时弥补以上两种获取方式的不足,与其他领域知识的获取[9-10]一样,派生谓词的推导规则可以先通过自动获取来获得一个初始模型,然后再进行人工的调整和完善.许多领域知识的提取都是从规划解中进行的,通过应用动作来产生一系列中间状态作为学习的知识基础,派生谓词规则的提取也不例外.然而,与其他领域知识的学习不同,派生谓词关注的是同一个状态下哪些命题参与推导,而非相邻状态中哪些命题发生变化.进一步地,从一个包含众多命题的状态中找出那些与推导相关的命题,可借助归纳学习程序或者数据挖掘工具来提取规则,例如使用 FOIL[11]、ACE① 和 aleph② 等工具.但是这些自动学习的结果在很大程度上是拟合训练数据的形式,而不能体现训练数据的语义内容.比如,不随状态变化的非 fluent 谓词是不可能出现在学习到的规则中,因为它们不带来更大的信息增益,尽管它们有时候作为规则的条件在语义上是必要的.

除了在学习结果上存在着固有的局限性,派生谓词的学习还必须解决另一个重要的问题:如何产生待学习谓词的训练例?如果一个规划领域的描述中包含派生谓词,那么规划问题的状态应该由基本谓词和派生谓词组成.但是,当派生谓词的推导规则未知时,由于派生谓词不出现在动作效果中,因此应用动作所产生的一系列中间状态则成为只包含基本谓词的信念状态(belief states).信念状态的信息是不完备的,那么如何确定哪些派生谓词实例在哪些信念状态中是成立的?已有研究[7]提出了一些确立准则:在成功的规划解中,如果派生谓词出现在动作的前提中,那么它在动作应用之前的状态应该是成立的;如果派生谓词是作为目标出现的,那么它在目标状态中应该是成立的.但是,这些准则能够确定的谓词实例的成立关系非常少,因此使得待学习谓词的训练例也非常少.并且对于不出现在动作前提或者目标中的派生谓词,是无法通过上述准则来获得训练例的.解决的办法只能是增加更多的方式来产生训练例.

如前所述,在领域建模时派生谓词用来描述动作的非直接效果.其实,在现实生活中,很多动作的非直接效果是可以观测得到的.例如,打开开关时灯泡不亮,闭合开关时灯泡亮了,灯泡亮是操纵开关的非直接效果.这样的观测可以通过简单的目测或是测量仪器来进行.用观测来反映动作的非直接效果,可以确定更多的派生谓词实例在信念状态中的成立情况,从而产生更多学习的训练例.另外,在现实生活中,完全观测比部分观测更耗时间成本和费用成本,因此更一般的情况是学习在部分观测的环境下进行,更何况完全观测是部分观测的特例.综上所述,根据以上种种动机,本文提出了在部分观测的环

① The ACE Data Mining System: User's Manual. 2009: 1-82. Online proceeding: http://dtai.cs.kuleuven.be/ACE/doc/

② The Aleph Manual. 2007: 1-62. Online proceeding: http://www.cs.ox.ac.uk/activities/machlearn/Aleph/aleph_toc.html

境下学习规划领域的派生谓词的方法,其中主要贡献在于用观测反映动作的非直接效果以增加待学习谓词的训练例,并且引入通用规划(generalized plans)[12]中的角色(roles)概念来增加非 fluent 谓词在规则中出现的机会,从而使得学习的结果在语义上更完善.本文所学习的派生谓词既可用于领域建模,又可用于描述控制知识,并且有利于二者自动获取的实现.

本文第 2 节介绍本文的研究背景,包括派生谓词简介以及部分观测下规划和学习的研究现状;第 3 节陈述本文的学习问题;第 4 节介绍学习方法,包括训练例的产生、规则的学习和语义的调整;第 5 节是实验和分析部分;最后是结束语和对未来工作的展望.

2 研究背景

早期的规划是关于动作的推理,后来发展成状态空间搜索、规划空间搜索和转化为其他问题(例如 SAT 等)进行求解[1].近十年来,规划的研究逐渐地考虑更多的现实因素的影响,例如不确定动作效果、部分观测环境、外存规划、外部事件和连续时间等等.经过四五十年的发展,智能规划俨然成为人工智能领域中重要的研究分支,在每年的人工智能盛会(如 IJCAI 或 AAAI)上都占有一席之地.结合本文的学习问题,本节主要介绍派生谓词以及部分观测下的规划和学习的研究现状.

2.1 派生谓词简介

在智能规划研究领域,派生谓词用来描述事物间的因果联系,也称为领域公理(domain axioms).派生谓词由规划领域描述语言 PDDL2.2(Planning Domain Description Language 2.2)[2]正式引入.它们的真值不受动作模型的影响,而是以规则的形式来进行推导.这样的规则独立于动作模型,从而使得整个领域描述清晰而简洁.已有研究[5]证明派生谓词是规划领域描述语言必不可少的组成部分,缺少它们,容易使得领域描述和规划解长度超过多项式级别.如前所述,派生谓词可以作为领域建模的一部分,也可以作为控制知识的一部分.下面两个例子分别说明其主要用途.

例 1. 在 Blockworld 领域中,above(?x,?y)是派生谓词,定义其的规则如下:

$$\text{above}(?x,?y)\equiv\text{on}(?x,?y)\vee\exists z(\text{on}(?x,?z)\wedge\text{above}(?z,?y)).$$

该规则表明一个木块位于另一木块的上方的充要条件.该规则属于领域建模的一部分,有了它可以定义包含 above 谓词的目标.

例 2. 同样地,在 Blockworld 领域中,goodtower(?x)是派生谓词,定义其的规则如下:

$$\text{goodtower}(?x)\equiv\text{clear}(?x)\wedge\neg\,\text{GOAL}(\text{holding}(?x))\wedge\text{goodtowerbelow}(?x).$$

该规则表明一个木块处于"好塔"位置,当且仅当它位于塔顶,并且在目标状态不要求 robot 举着它,以及它下面的所有木块没有破坏任何目标条件.该规则属于控制知识的一部分,它要求后继的动作应避免去移动处于"好塔"位置的木块.

派生谓词的处理方法分成两种:间接方法和直接方法.间接方法将派生谓词转化为其他语言要素.例如,早期处理派生谓词的方法是将其转化成已有动作的条件效果或者新动作[13-14],但是这样的方法效率低,只适合求解小规模问题.直接方法基于逻辑推理机制.自从 2004 年的国际规划大赛(International Planning Competition 2004,简称 IPC-4)正式引入派生谓词的基准领域以后,不少规划系统提出了能够直接处理中等规模的包含派生谓词的规划问题的方法,其中代表性的有 LPG-td[15]、SGPlan[16]、Fast Downward[17]和 Marvin[18]等.Fast Downward 和 Marvin 采用相同的方法:每应用一个领域动作后,先删除所有原来成立的派生谓词实例,然后再由规则重新推导出新实例直至到达状态的不动点.LPG-td 在规则图上推导派生谓词实例的激活集,用来代替其在动作图上的出现以进行规划空间搜索.SGPlan 调用 LPG-td 作为底层规划器来处理包含派生谓词的子问题.但是以上方法都难以处理大规模问题,这是因为当问题对象增多时,规则的实例化空间急剧膨胀而使得推理的复杂性大大增强.为了降低复杂性,还有研究[19]改进了激活集的定义并在一定程度上简化其计算.

2.2 学习派生谓词

尽管关于派生谓词的处理方面研究得已比较成熟,但是有关派生谓词的自动提取方面的研究并不多见.原因主要来自于两方面:(1)认为派生谓词规则的提取与其他一阶规则的提取没什么不同,都是基于逻辑程序的归纳学习过程;(2)难以摆脱归纳学习本身的局限性,即只能拟合训练例的形式而无法挖掘其含义,学出来的规则往往缺乏语义上的合理解释.于是,一个令人尴尬的结果产生了:一方面认为派生谓词的学习是一件容易的事情,而另一方

面学出来的效果又总是不理想. 其实，如前所述，学习派生谓词的关键问题不在于一阶规则的归纳学习过程，而在于待学习谓词的训练例产生过程以及对学习结果在语义上的调整. 对于训练例的产生方式，已有研究或者通过谓词的预先定义[4,8]，或者通过规则语义[7]来解决. 而对于学习结果的语义调整，至今没有研究能够给出形式化的方法.

关于派生谓词学习的研究现状如下：首先，在2005 年，Zettlemoyer 等人[8]在有噪音的随机环境下学习动作的效果模型时，通过预先定义的操作应用到文字上来产生新的谓词，将它们作为动作的衍生效果加入到动作模型中. 实质上，他们将动作的所有效果都收集到动作模型中并没有形成独立的描述动作非直接效果的规则. 接着，在 2010 年，饶东宁等人[7]根据训练数据来修订一个初始的不完备的领域理论(即派生谓词规则集合). 他们的工作结合了分析学习和归纳学习，其中初始理论用于派生谓词实例的激活集来扩充规则的候选式. 尽管学习效果大大依赖于初始理论的准确程度，但是他们提出了如引言所说的一些确立准则来产生待学习谓词的训练例. 最后，在 2012 年，Rosa 等人[4]提出了采用柱状搜索和归纳学习来获取控制知识中的派生谓词. 所学习的派生谓词是为了增强控制知识的表达力，而非领域建模的一部分. 学到的谓词由固定的规则模式来定义，分为三类：组合、抽象和递归. 因此，他们的工作不是学习领域模型中的派生谓词，而是根据已有的规则模式来产生派生谓词，从而改善控制知识的描述.

2.3　部分观测下的规划和学习

在部分观测环境下，状态信息是不完备的，每个状态变成了信念状态. 因此，在部分观测下的规划问题可以形式化成在信念空间的非确定搜索问题. 这样的问题比经典规划问题更难[20-21]，因为追踪信念状态比追踪状态更难，并且要得到的是一个动作策略而不是动作序列. 根据信念状态的形式，部分观测规划问题分为两类：随机规划和 POMDP 规划. 在随机规划中，信念状态是一个状态的集合，求解方法主要是处理信念空间的搜索. 代表性的研究工作有：Hoffmann 等人[21]给出了信念空间的与或图搜索算法；Bryce 等人[22]提出了如何评估信息状态之间的距离；Cimatti 等人[23]用符号模式检测的方法来求解随机规划问题等. 而在 POMDP 规划中，信念状态由在状态集合上的概率分布来表示，求解方法主要依赖于 MDP 方法中的值迭代和策略迭代过程. 代表性的研究工作有：Kaelbling 等人[24]提出方法来估算新状态的概率分布；Bonet 等人[20]将部分观测问题转化为完全观测问题并用经典规划器来求解等等. 然而，不管是随机规划还是 POMDP 规划，在一个变得更复杂的搜索空间中如何能够快速求解是它们共有的核心问题.

在部分观测下的学习，也是在信念空间中进行[25-26]. 目前研究得较多的是在部分观测的环境下学习确定性的动作模型. 训练数据仍然由规划解产生，但此时的规划解已经从动作序列变成了动作-观测序列. 观测可以表示成一种像领域动作一样的动作[26]，也可以表示成一个由 fluent 命题组成的逻辑公式[21]. 给定用于学习的动作-观测序列，需要程序能够自动获取导致这些序列的动作模型或者转换关系. 一些有代表性的研究工作包括 Schmill 等人[25]使用决策树归纳来学习动作模型，观测信息从 agent 与环境的交互中获取；Amir 等人[26]提出了 SLAF (Simultaneous Learning and Filtering)机制，即使用领域动作来扩展信念空间和使用观测动作来过滤信念状态；Zhuo 等人[10]在不完备的信息状态下学习包含量词和蕴含关系的复杂动作模型等.

不难想象，不管是在部分观测下的规划还是学习，其结果的准确性与真实模型还是有差距的. 由于只能根据初始信息或者是观测到的信息来提取模型，因此结果的准确性不仅与求解过程或者是学习方法有关，还受到观测信息数量和质量的影响. 若没有观测到重要的信息或者观测数据不够多，则可能使得产生的模型不够准确或者不够完备. 不过，对于部分观测的两个极端——不可观测和完全观测，其学习的效果就主要由学习方法来决定了.

3　问题陈述

规划问题的描述基于一阶谓词逻辑，其中谓词分成两类：基本谓词(basic predicates)和派生谓词(derived predicates)[2]. 二者的差别是，基本谓词可以出现在动作模型的效果中，而派生谓词则不能. 这样，动作模型只需要描述动作的最直接效果，而各种衍生效果和因果联系可以通过派生谓词规则来表示. 下面给出包含派生谓词的规划领域、规划问题和规划解以及观测的形式化描述.

一个经典规划领域由谓词集合、状态集合、动作

集合和转换关系等部分组成[1]. 包含派生谓词的规划领域是在经典规划领域的基础上定义的. 与经典领域不同的是,派生谓词规划领域将谓词分为基本的和派生的,并且定义派生谓词规则,在每次应用动作后产生的新状态要进行规则集下的扩充. 引用文献[27],派生谓词规划领域、规划问题和规划解的定义如下.

定义 1. 派生谓词规划领域[29]. 设 $L=\{p_1,\cdots,p_n\}$ 是有限的谓词符号集合,派生谓词规划领域 Σ 是一个六元组 $\langle S,A,X,B,D,R\rangle$:

(1) S 是状态集合,$S\subseteq 2^L$;

(2) A 是动作集合,$A=\{a\mid a=\langle pre(a),add(a),del(a)\rangle\}$. 其中 $pre(a),add(a),del(a)\in 2^L$ 分别表示动作 a 的前提集合、增加效果集合和删除效果集合;

(3) X 是转换函数,$X(s,a)=Đ(((s-del(a))\cup add(a))-[\![R]\!](s))$,当动作 a 在 s 中是可应用的;

(4) B 是基本谓词符号集合;

(5) D 是派生谓词符号集合,且 $B\cap D=\varnothing$;

(6) R 是派生谓词规则集合,且 $R=\{r\mid r=(:\text{derived }(d?\boldsymbol{x})(f?\boldsymbol{x}))\}$.

在定义 1 中,动作 a 在状态 s 中是可应用的,当且仅当 $pre(a)\subseteq Đ(s)$. 函数Đ将状态 s 映射为在规则集 R 下的扩充状态,即 $Đ(s)=s\cup[\![R]\!](s)$,其中 $[\![R]\!](s)$ 表示在状态 s 下应用规则集 R 推导至不动点所得到出的派生谓词实例集合. 在规则集 R 中,一条派生谓词规则 r 由形如(:derived $(d?\boldsymbol{x})(f?\boldsymbol{x})$)的前缀表达式来表示,其中 $(d?\boldsymbol{x})$ 是单个派生谓词,$(f?\boldsymbol{x})$ 是由基本谓词和派生谓词组成的逻辑公式,$\boldsymbol{x}$ 是规则中的变量矢量. 规则 r 的含义为,如果 $(f?\boldsymbol{x})$ 在状态 s 下为真,则 $(d?\boldsymbol{x})$ 在 s 中也为真. 对于一个派生谓词实例而言,如果推导它的某个规则实例的条件在状态中是成立的,那么该谓词实例在状态中也是成立的;然而,如果推导它的所有规则实例的条件在状态中都不成立,那么该谓词实例在状态中也不成立. 因此,在应用动作之后(即 $(s-del(a))\cup add(a)$),有可能动作删除了原有某个派生谓词实例成立的所有条件,因而要在后继状态中先减去原有的所有派生谓词实例,然后在规则集下进行重新推导,使得新状态能够准确地持有应该成立的谓词实例. 定义 1 中状态的扩充方法与 Fast Downward[17] 和 Marvin[18] 处理派生谓词的方法是一致的.

定义 2. 派生谓词规划问题[27]. 一个派生谓词规划问题 Π 定义为三元组 $\langle\Sigma,I,G\rangle$,其中 Σ 是派生谓词规划领域描述,I 是初始状态,G 是目标条件.

定义 3. 规划解[27]. 派生谓词规划问题 $\Pi=\langle\Sigma,I,G\rangle$ 的解 π 是一个原子动作序列 $\langle a_1,a_2,\cdots,a_n\rangle$,其中 n 是动作的个数. 该动作序列产生一个状态变换序列 $\langle s_0,s_1,s_2,\cdots,s_n\rangle$,使得

(1) $s_0=Đ(I)$;

(2) $pre(a_i)\subseteq Đ(s_{i-1}),s_i=X(s_{i-1},a_i),1\leq i\leq n$;

(3) $G\subseteq s_n$.

一个派生谓词规划问题是派生谓词规划领域的实例. 遵循经典规划的假设,初始状态是唯一的,动作效果是确定的,定义 3 描述了派生谓词规划问题的解. 此外,对于一个派生谓词实例 p,如果它出现在某个动作实例的前提中,即 $p\in pre(a)$,则称 p 为派生谓词前提. 类似地,如果 p 出现在目标条件 G 中,即 $p\in G$,则称 p 为派生谓词目标.

接下来,我们定义观测. 首先,状态所包含的命题分为两类[1]:fluent 命题和非 fluent 命题. fluent 命题的真值随着状态的变化可能发生变化,例如 on(A,B) 在当前状态下为真,在应用了动作 pickup(A,B) 之后,在下一个状态下为假. 而非 fluent 命题的真值不随状态的变化而变化,例如 block(A) 在任何状态下都为真. 因此,需要观测的是 fluent 命题,一个观测变量就是一个 fluent 命题. 然后,如前所述,一个观测可以定义成动作[26],也可以定义成一个观测变量组成的逻辑公式[21]或者映射函数. 当定义成动作时,观测 o 是一个序对 $\langle pre(o),obs(o)\rangle$,其中 $pre(o)$ 和 $obs(o)$ 都是命题集合,该定义表明当 $pre(o)\subseteq s$ 成立时,可以观测到 $obs(o)$ 中命题的真值. 这样定义的好处是把观测动作当成领域动作一样来处理. 而当定义成逻辑公式或者映射函数时,观测 o 是观测变量集合 O 的一个部分映射函数 $f_o:O\times S\rightarrow\{\text{true},\text{false}\}$,它表明某个 fluent 命题 $p\in O$ 在当前状态 $s\in S$ 下的真值 $f_o(p,s)$ 或者为真或者为假①. 这样定义的好处是便于往状态中添加经过观测而确立的命题. 为了方便算法描述,本文采用观测的第 2 种定义.

定义 4. 观测. 观测 o 是一个部分映射函数 $f_o:O\times S\rightarrow\{\text{true},\text{false}\}$,其中是 O 观测变量集合,S 是状态集合,即存在着 $P\subseteq O$,对于任意 $p\in P$,$f_o(p,s)$

① 逻辑公式是映射函数中取值为真的命题组成的公式.

都存在. 并且如果 $f_o(p,s)=$true，则表明观测变量 p 在状态 s 中的真值为真，反之，如果有 $f_o(p,s)=$false，则表明观测变量 p 在状态 s 中的真值为假.

最后，我们定义本文的学习问题. 为了描述方便，本文以下内容均使用 $\overline{ao}$ 来表示由动作-观测组成的序列. 具体地说，$\overline{ao}=\langle\langle a_1,o_1\rangle,\cdots,\langle a_n,o_n\rangle\rangle$，其中 $a_i(1\leq i\leq n)$ 为领域动作，$o_i(1\leq i\leq n)$ 为观测，n 称为序列长度. 在本文问题中，动作模型是已知的，动作效果是确定的，因此观测用来反映动作的非直接效果，即观测变量全部由 fluent 形式的派生谓词实例组成. 因此，在本文余下部分，除特别说明外，观测变量均指 fluent 形式的派生谓词实例，并且所有的观测都假设为准确的.

定义 5.　在部分观测下的派生谓词学习. 若在一个规划领域描述 $\Sigma=\langle S,A,X,B,D,R\rangle$ 中，只有 R 是未知的，当给定该领域的若干规划问题 Π 及其解 $\pi_\Pi=\overline{ao}$（$\overline{ao}$ 为动作-观测组成的序列）时，能够通过学习方法得到 R 的过程，称为在部分观测下的派生谓词学习问题.

参照相关研究工作[4,7,10]的习惯，本文中派生谓词学习问题的精准度定义为学习结果在验证集上正确解释训练例的比例，而不是与一个理想模型对比的相似程度. 首先，这是因为表示相同语义的规则形式可能不是唯一的，因此当一个学习到的模型与一个理想模型在形式上有差异的时候，并不代表学习到的模型是不正确的. 其次，一阶规则的提取基于归纳学习，而学习的结果是得出拟合训练例的假设，因此当验证集合能充分代表实例空间时，在验证集上的精准度可以完全代表在实例空间上的精准度. 另外，正确解释训练例的含义指的是给定一个派生谓词的正例，经过学习到的规则能够推导出该实例，或者给定一个派生谓词的反例，经过学习到的规则不能推导出该实例.

定义 6.　学习精度. 在部分观测下的派生谓词学习问题中，给定验证集 T，学习到的规则集 R 在 T 上的精准度定义为 $A_T=n_{\text{correct}}/n_{\text{total}}$，其中 n_{total} 是 T 中训练例总数，n_{correct} 是 R 在 T 上正确解释的训练例总数.

4　学习方法

本节介绍在部分观测环境下学习派生谓词规则的具体方法，包括产生训练例、规则学习和语义调整三部分. 在产生训练例部分，首先定义什么是派生谓词学习的训练例，然后介绍如何从动作-观测序列中产生待学习谓词的训练例. 在规则学习部分，主要介绍常用的序列覆盖学习算法和柱状搜索学习算法. 在语义调整部分，首先介绍什么是角色以及角色特征，其次介绍如何将角色特征引入到学习到的规则中以加强语义完整性.

4.1　产生训练例

派生谓词的训练例采用〈状态，谓词实例，正例/反例标识〉的三元组形式. 因为动作序列会产生一系列中间状态，而派生谓词实例往往是 fluent 命题，即随着状态的变化其真值也发生变化，因此训练例的形式中应加入状态信息，以标识谓词实例在什么状态下其真值如何. 此外，对于基本谓词实例而言，如果也是 fluent 命题的，其训练例中也应加入状态信息. 因此，为了统一处理，在本文的规则学习中，所有谓词的训练例中都应该加入状态信息.

定义 7.　训练例. 在部分观测下的派生谓词学习问题中，谓词 $p(\boldsymbol{x})$ 的一个正例 t_p^+ 定义为三元组 $\langle s,p(\boldsymbol{c}),+\rangle$，其中 s 表示状态，$p(\boldsymbol{c})$ 是用常矢量 $\boldsymbol{c}$ 来替换 $\boldsymbol{x}$ 所得到的谓词实例. t_p^+ 的语义为 $p(\boldsymbol{c})$ 在 s 下真值为真. 类似地，谓词 $p(\boldsymbol{x})$ 的一个反例 t_p^+ 定义为三元组 $\langle s,p(\boldsymbol{c}),-\rangle$，其语义为 $p(\boldsymbol{c})$ 在 s 下真值为假.

下面给出从给定的问题描述和动作-观测序列中提取待学习谓词的训练例的算法.

算法 1.　产生训练例.

输入：派生谓词规划问题 $\Pi=\langle\Sigma,I,G\rangle$ 及其解 $\overline{ao}=\langle\langle a_1,o_1\rangle,\cdots,\langle a_n,o_n\rangle\rangle$

输出：正例集 T^+、反例集 T^-

1. $s_0=I$
//produce the state sequence
2. FOR $i=1$ TO n DO
3. $\quad s_i=(s_{i-1}-del(a))\cup add(a)$
//handle derived predicate preconditions
4. FOR $i=1$ TO n DO
5. $\quad$FOR all derived fact d DO
6. $\quad\quad$IF $d\in pre(a_i)$ THEN
7. $\quad\quad\quad$Add $\langle s_{i-1},d,+\rangle$ to T^+
8. $\quad\quad$ELSE IF (not d)$\in pre(a_i)$ THEN
9. $\quad\quad\quad$Add $\langle s_{i-1},d,-\rangle$ to T^-
//handle derived predicate goals
10. FOR all derived fact d DO
11. $\quad$IF $d\in G$ THEN

12. Add $\langle s_n, d, +\rangle$ to T^+
13. ELSE IF (not d)$\in G$ THEN
14. Add $\langle s_n, d, -\rangle$ to T^-
//handle observations
15. FOR i=1 to n DO
16. FOR all derived fact d DO
17. IF $f_{o_i}(d, s_i)$==true THEN
18. Add $\langle s_i, d, +\rangle$ to T^+
19. ELSE IF $f_{o_i}(d, s_i)$==false THEN
20. Add $\langle s_i, d, -\rangle$ to T^-
21.RETURN T^+, T^-

算法 1 通过 3 种方式来产生待学习谓词的训练例:(1)如果一个派生谓词实例 d(derived facts)出现在动作 a_i的前提中,那么它应该在执行 a_i之前的状态中 s_{i-1}成立,并且肯定出现产生正例,否定出现产生反例(步 4～9);(2)如果一个派生谓词实例 d 出现在目标描述 G 中,那么它必然在最终状态 s_n中成立,同样地,肯定出现产生正例,否定出现产生反例(步 10～14);(3)如果一个派生谓词实例 d 出现在观测 o_i中,由于观测 o_i是在执行完动作 a_i之后的状态 s_i中进行的,因此根据观测函数 f_{oi}的映射来判断 d 在 s_i中成立与否(步 15～20).

定理 1. 算法 1 的时间复杂性为 $O(nmk+nm^2)$,其中 n 是动作-观测序列的长度,m 是规划问题的派生谓词实例集合的大小,k 是规划问题的基本谓词实例集合的大小.

证明. 首先,由于动作效果不包含派生谓词,因此应用动作-观测序列所产生的状态全部由基本谓词实例组成.在步 2～3 中,动作 a 的增加效果和删除效果个数均不超过 k,因此所花的时间为 $O(nk)$.其次,步 4～9 包括一个两重循环,内循环总共执行 nm 次,每次执行内循环需检测一个派生谓词实例是否包含在动作前提中,由于动作前提可以包含派生谓词实例和基本谓词实例,因此所花时间为 $O(k+m)$,从而整个步 4～9 所花的时间为 $O(nm(k+m))$,即$O(nmk+nm^2)$.接着,步 10～14 只包含一个循环,所花的时间为 $O(km)$.最后,步 15～20 包含包括一个两重循环,内循环总共执行 nm 次,每次执行内循环需检测一个派生谓词实例是否存在着观测函数的映射值,所花时间为 $O(m)$,因此整个步 15～20 所花的时间为 $O(nm^2)$.综上,算法 1 所花时间为$(nmk+nm^2)$. 证毕.

另外,在产生了待学习谓词的训练例之后,还必须从规划解所产生的状态序列$\langle s_0, s_1, s_2, \cdots, s_n\rangle$中转化出基本谓词实例的训练例,作为规则学习的知识基.具体地说,如果一个基本谓词实例 $b\in s$,则产生一个正例$\langle s, b, +\rangle$;反之,如果 $b\notin s$,则产生一个反例$\langle s, b, -\rangle$.这些训练例同样加入到 T^+ 和 T^- 中,作为规则学习算法的输入.

4.2 规则学习

派生谓词的学习可以采用序列覆盖算法[11]或柱状搜索算法[4].序列覆盖算法采用信息增益来评估候选式,每次选取最佳候选式来构造规则的条件,直至其覆盖(即解释)部分正例而避开所有反例为止.在学习到一个规则后,序列覆盖算法继续在剩余的正例集合中学习其他规则.柱状搜索算法的规则扩展过程与序列覆盖算法的是类似的,二者的主要区别在于柱状搜索算法进行多点扩展,每次选取精度排在前面的 k 个规则,而序列覆盖算法进行的是单点扩展,始终围绕同一条规则来进行扩展.下面给出两个算法详细的实现过程.

算法 2. 序列覆盖算法.

输入:训练例集合 T^+ 和 T^-、谓词集合 L、要学习的派生谓词 dp
输出:规则集合 R
1. $R\leftarrow\varnothing$; $Pos\leftarrow$the positive example set of dp in T^+
2. WHILE Pos is not empty
3. $r\equiv\rightarrow dp$
4. $Neg\leftarrow$the negative example set of dp in T^-
5. WHILE Neg is not empty
6. Produce the candidate set C of r based on L
7. $BestC\leftarrow\arg\max_{c\in C} Gain(c, r)$
8. Add $BestC$ to the antecedent of r
9. $Neg\leftarrow$training examples that satisfy the antecedent of r in Neg
10. Add r to R
11. Remove the positive examples which are covered by r from Pos
12. RETURN R

算法 2 的关键在于步 6～7.在步 6 中,候选文字集合 C 的产生取决于变量的关联关系.具体地说,由谓词集所产生的候选文字至少应包含一个已在原规则中存在的变量,即候选文字与原规则是有关联的.步 7 采用信息增益函数 $Gain$ 来评估候选文字的优劣,用 $BestC$ 来记录使得 $Gain$ 函数值达到最大的候选文字.设原规则为 r,增加了候选文字 c 的规则为 r',则 $Gain$ 函数表示为编码 r 的所有正例约束的分类,加入 c 所减少的编码位数.具体定义

如下：

$$Gain(c,r)=t\left(\log_2\frac{p_1}{p_1+n_1}-\log_2\frac{p_0}{p_0+n_0}\right) \quad (1)$$

其中，p_0和n_0分别表示规则r的正例约束数目和反例约束数目，p_1和n_1分别表示规则r'的正例约束数目和反例约束数目，t是r和r'共同约束的正例数目.

定理 2.　算法 2 的时间复杂性为$O(n^3l)$，其中$n=\max\{n^+,n^-\}$，n^+和n^-分别是正例集合T^+和反例集合T^-的大小，l是谓词集合L的大小.

证明.　算法 2 包含一个双重循环. 在最坏情况下，外层循环(步 2～11)执行n^+次，内层循环(步 5～9)执行n^-次. 在执行内层循环时，候选式集合C的大小不会超过l的常数倍，即步 6 所花的时间为$O(l)$. 步 7 在训练例集合T^+和T^-上评估候选式，所花的时间为$O(l(n^++n^-))$. 同样地，规则前件的数目不会超过l的常数倍，因此步 8 所花的时间为$O(l)$. 步 9 所花的时间为$O(n^-)$. 综上，内层循环所花的时间为$O(n^-l(n^++n^-))$，外层循环所花时间为$O(n^+n^-l(n^++n^-))$. 如果记$n=\max\{n^+,n^-\}$，则算法 2 的时间复杂性为$O(n^3l)$.　　证毕.

柱状搜索算法比序列覆盖算法多了柱状宽度k. 此外，在构造规则时，柱状搜索算法没有像序列覆盖算法一样严格要求避开所有反例.

算法 3.　柱状搜索算法.

输入：训练例集合T、谓词集合L、要学习的派生谓词dp、柱状宽度k

输出：规则集合R

1. $R\leftarrow\varnothing$; $beam\leftarrow\{\rightarrow dp\}$
2. $irrelavent\leftarrow\varnothing$
3. REPEAT
4. $best\_fn=f_{acc}(\text{first}(beam))$
5. $successors\leftarrow\varnothing$
6. FOR all r in $beam$ DO
7. Produce the candidate set C of r based on L
8. FOR all $c\in C$ DO
9. Add c to the antecedent of r
10. IF $r\notin irrelavent$ THEN
11. Add r to $successors$
12. FOR all r' in $successors$ DO
13. Evaluate $f_{acc}(r',T)$
14. IF $f_{acc}(r',T)<best\_fn$ THEN
15. Add r' to $irrelavent$
16. $beam\leftarrow$first k r's in sorted($successors$, f_{acc})
17. UNTIL $f_{acc}(\text{first}(beam))\leq best\_fn$
18. RETURN first($beam$)

算法 3 每次选取精度排在前k位的规则来进行构造. f_{acc}是规则在训练集上的精准度函数，first($beam$)表示$beam$中精度排在首位的规则. 加入新的候选式的规则存放在$successors$中(步 6～11)，其中精度过低的规则被抛弃，并且为了避免重复计算精度，将它们存放于$irrelavent$中(步 12～15). sorted 函数表示将规则按照精度从高到低进行排序，排在前k位的规则存放于$beam$中作为下一轮的待扩展规则(步 16). 最后，当不再出现精度更高的规则时，算法结束，返回精度最高的规则.

定理 3.　算法 3 的时间复杂性为$O(k^2l^2+kln)$，其中n是训练例集合T的大小，l是谓词集合L的大小，k是柱状宽度.

证明.　算法 3 中 repeat-until 循环的执行次数是不固定的. 在该循环中，步 4 在训练集上评估规则精度，所花的时间为$O(n)$. 步 6～11 包含一个双重循环，由于柱状宽度为k，因此外层循环执行的次数为k. 在外层循环内部，步 7 所花的时间为$O(l)$，内层循环(步 8～11)所花的时间为$O(lkl)$. 这是因为$beam$中至多包含k个规则，每个规则的候选式集合C不超过$O(l)$，因此$successors$中的规则数不超过$O(kl)$. 而$irrelavent$中的规则均来自于$successors$，所以步 10～11 所花的时间为$O(kl)$. 综上，步 6～11 所花的时间为$O(klkl)$，即$O(k^2l^2)$. 同样地，步 12～15 所示的循环执行$O(kl)$次，每次执行评估规则精度所花的时间为$O(n)$，则总共花的时间为$O(kln)$. 步 16 需按精度进行排序，所花时间为$O(kl(\log kl))$. 综上，算法 3 的时间复杂性为$O(k^2l^2+kln+kl(\log kl))$，即$O(k^2l^2+kln)$.　　证毕.

4.3　语义调整

语义调整是一个后处理阶段，目前主要是在规则条件部分增加归纳学习过程学习不到的非 fluent 命题. 如前所述，这些非 fluent 命题的真值在状态中是保持不变的，即只要它们在初始状态中出现，就可以一直保持到终止状态. 当它们作为一阶规则学习的谓词候选式时，新规则所覆盖的正例数和反例数与原规则是一样的，即没有信息增益的增加，这些非 fluent 命题很难作为最佳候选式加入到规则中. 所以，基于信息增益的规则学习过程是学习不到非 fluent 命题的. 这些命题之所以出现在问题描述或者规则定义中，是因为它们刻画了对象的属性或类别以及对象之间的语义联系. 因此尽管在学习过程中学不到这些命题，但是在后处理阶段应尽可能地

补充它们,以增加学习结果在语义上的完整性.

例 3. PSR(Power Supply Restoration)领域是在 IPC-4 上引入的派生谓词基准领域之一,该领域描述了电源给线路供电的电路问题.在 PSR 领域中,派生谓词(affected ?x)表示电源?x 受到故障线路的影响,其定义规则(使用 PDDL2.2 语法规范)如下:

```
(:derived (affected ?x-DEVICE)
  (and (breaker ?x)
    (exists (?sx-SIDE)(unsafe ?x ?sx))))
```

其中,breaker(?x)是一个非 fluent 谓词,表示?x 是一个电源,其所有实例均为非 fluent 命题.使用 FOIL,在规则条件部分只能学习到 unsafe(?x,?sx),而不能学习到 breaker(?x).

对于一阶规则,为了在条件部分增加非 fluent 谓词,可以参考通用规划[12,28]的研究,引入角色(roles)的概念.角色表示对象所属的类别,与非 fluent 谓词的作用非常相似.角色可用抽象谓词(abstract predicates)集合来描述.对于一个规划领域而言,抽象谓词指的是领域描述中的单目谓词(即谓词只有一个参数)[12],或者是只包含一个变量而其他变量均为领域常量的多目谓词(即谓词可有多个参数)[28].在本文中,为了讨论方便,抽象谓词只局限于单目谓词,并且是非 fluent 谓词.这样,在应用规则时,如果能够识别同一参数对应的所有对象都属于一种角色,就可以引入角色的抽象谓词到规则条件中,从而实现对非 fluent 谓词的补充学习.

定义 8. 抽象谓词[12].对一个规划领域 Σ 而言,抽象谓词指的是其谓词集合 L 中所包含的单目谓词.

在本文中,抽象谓词还必须是非 fluent 谓词.

定义 9. 角色[12].对一个规划领域 Σ 而言,角色指的是对象所满足的抽象谓词集合.

对于一个规划问题而言,角色将其所包含的对象进行了划分.已有研究工作[28]证明,当一个对象属于多个角色时,总可以创建新的角色使得这个对象唯一的属于该新角色.因此,角色和对象是一对多的关系,每个对象唯一的属于一个角色.对象也称为角色成员.

例 4. 在 PSR 领域中,谓词集合 L={ext(?l,?x,?s),breaker(?x),closed(?x),faulty(?l),con(?x,?sx,?y,?sy),upstream(?x,?sx,?y,?sy),unsafe(?x,?sx),affected(?x),fed(?l)},其中非 fluent 谓词包括 ext(?l,?x,?s),breaker(?x),faulty(?l),con(?x,?sx,?y,?sy).可创建两个角色:

$BREAKER$={breaker(?x)},

$FAULTY\_LINE$={faulty(?l)}.

定义 10. 角色参数.设规划领域 Σ 的角色集合为 $ROLE$,r=(:derived (d ?$\boldsymbol{x}$)(f ?$\boldsymbol{x}$))是派生谓词规则且 $p(x_1,\cdots,x_n)\in f(\boldsymbol{x})$.对于规划问题 $\langle\Sigma,I,G\rangle$,r 存在着实例规则(即规则中所有变量均用对象常量来替换)集合 $Ground(r)$,如果 p 的某一参数 $x_i(1\leq i\leq n)$ 在 $Ground(r)$ 中对应的所有对象均属于 $ROLE$ 中的同一角色 $role$,则称 x_i 为 $role$ 的角色参数,标记为 $x_i\in parameter(role)$.

本文定义角色参数的意义在于,在归纳学习方法学习到规则以后,如果能够识别出规则中谓词的角色参数,则可将角色的特征谓词增加到规则中,即补充非 fluent 谓词从而进行语义上的完善.具体算法如下.

算法 4. 识别角色参数.

输入:规则集 R、角色集合 $ROLE$、规划问题 $\Pi=\langle\Sigma,I,G\rangle$

输出:规则集 R

1. FOR all $r\in R$ DO
2. $non\text{-}fluent\leftarrow\varnothing$
3. $X\leftarrow$ the set of variables that appear in the antecedent of r
4. Instantiate r with objects that are contained in Π to produce the set $Ground(r)$
5. FOR all $x\in X$ DO
6. IF $\exists role\in ROLE$ such that $x\in parameter(role)$ THEN
7. FOR all $q\in abstract$ predicates of $role$ DO
8. Add $q(x)$ to $non\text{-}fluent$
9. Add $non\text{-}fluent$ to the antecedent of r
10. RETURN R

定理 4. 算法 4 的时间复杂性为 $O(nk^{pl})$,其中 n 为规则集 R 的大小,l 为谓词集合 L 的大小,p 为 L 中谓词参数个数最大值,k 为规划问题所含的对象数.

证明. 首先,步 1 的循环次数为 n.其次,在该循环内部,步 3 收集规则前件中所有出现的变量,由于所产生的集合 X 的大小不超过 $O(pl)$,因此所花时间为 $O(pl)$.步 4 中,根据规划问题所包含的对象来实例化规则,由于规则中变量个数不超过 $O(pl)$,因此实例化规则的数目不超过 $O(k^{pl})$.步 5~9 包含一个双重循环,设 m 为角色集合 ROLE 的大小,其中每个角色的特征谓词集合不超过 $O(l)$,因此步 5~9 所花的时间为 $O(plml)$,即 $O(pml^2)$.最后,步 1~9 的循环所花时间为 $O(n(k^{pl}+pml^2))$,

即 $O(nk^{pl}+npml^2)$. 一般地，$k^{pl}\gg pml^2$，因此算法 4 的时间复杂性为 $O(nk^{pl})$.　　证毕.

5　实　验

基于以上算法，本文开发了两个学习工具：POLDR$^{\text{FOIL}}$（Partial Observable Learning Derived Rules based on FOIL）和 POLDR$^{\text{ICL}}$（Partial Observable Learning Derived Rules based on ICL）. 其中，POLDR$^{\text{FOIL}}$ 实现了算法 1、2 和 4，在实现算法 2 时调用了 FOIL（First Order Inductive Learning），是基于序列覆盖的归纳学习过程；POLDR$^{\text{ICL}}$ 实现了算法 1、3 和 4，在实现算法 3 时调用了 ICL（Inductive Classification Logic），是基于柱状搜索的归纳学习过程. 两个工具的组成结构图见图 1.

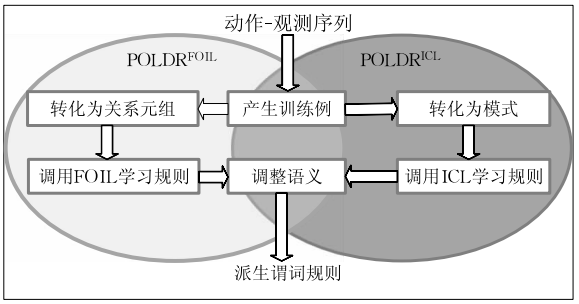

图 1　POLDR$^{\text{FOIL}}$ 和 POLDR$^{\text{ICL}}$ 的组成结构图

从图 1 可见，POLDR$^{\text{FOIL}}$ 和 POLDR$^{\text{ICL}}$ 分别由 4 个部分组成，其中产生训练例和调整语义是共有的两个部分. 由于 FOIL 和 ICL 采用不同形式的训练例，因此在调用它们之前，必须先转化为其所接受的训练例形式. FOIL 的训练例采用关系元组的形式，关系即谓词，元组即参数列表. 将算法 1 所产生的训练例〈$s, d(x_1,\cdots,x_n), +/-$〉转换为关系元组的做法如下：先将状态 s 编号，比如 i；然后将谓词 $d(x_1,\cdots,x_n)$ 转换为关系元组 $d(i,x_1,\cdots,x_n)$. 此外，FOIL 采用分号来分隔正例和反例. ICL 的训练例采用模式的形式，每个模式代表一个状态，因此不需要对状态进行编号并增加为额外的参数. ICL 实质是一个分类器，待学习谓词的正例归为正例类，反例归为反例类. 除了训练例的形式不同，二者的输入文件个数也不同. FOIL 的输入只有一个训练例文件 *.d，而 ICL 的输入有 4 个文件：*.kb、*.bg、*.l、*.s，其中，*.kb 存放训练例；*.bg 存放背景知识，比如领域规则；*.l 存放语言偏置，比如构成前件和后件的可用谓词集合以及数量范围；*.s 是设置文件. 图 2 给出了同一个 PSR 规则学习问题分别在 FOIL

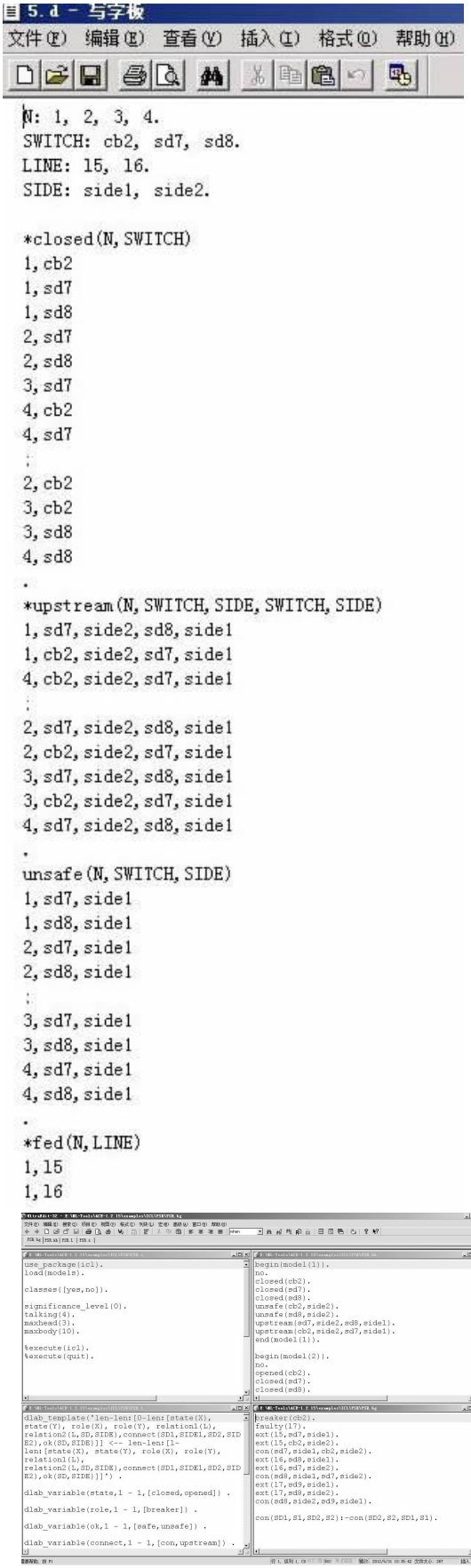

图 2　FOIL 和 ICL 中的输入示例

和 ICL 中的输入示例. $POLDR^{FOIL}$ 和 $POLDR^{ICL}$ 的各部分(除 FOIL 和 ICL 之外)用 C++语言来实现,实验环境为 Windows Server 2008 + 2.4 GHz Celeron CPU + 2GB memory + Eclipse Helios Service Release 2. 另外,FOIL 的运行环境为 Ubuntu Server 10.04.3 LTS+gcc 4.3.0+FOIL 6.4①,ICL 的运行环境为 Windows Server 2008+ACE-1.2.15②.

实验数据采用 IPC-4 公布的 PSR 规划领域的基准问题,选择 PSR-Middle-ADLDerived 版本下的规划问题以及参赛规划器 LPG-td 公布的规划解来分别产生训练例③. 由于公布的规划解不包含观测,因此为了产生部分观测的环境,对于每个规划问题 Π,可以先将其所包含的派生谓词实例化以得到实例集合 D_Π,然后设置一个观测率 λ 在 D_Π 上产生观测函数. 例如,假设 $|D_\Pi|=100$,当 $\lambda=0.2$ 时,则在每次观测时随机选取 20 个派生谓词实例来进行观测. 另外,为了避免某些派生谓词的实例特别多而造成观测集中,这些观测名额可以平均分配到每个派生谓词上. 观测加入到规划解中,构成动作-观测序列. 考虑到问题实例的代表性以及观测的随机性,选择 3 个规划问题 P01.PDDL、P03.PDDL 和 P05.PDDL 来分别产生训练例集合,即 3 个训练例集合. 对于每个训练例集合,学习过程采用 3-交叉验证方法,即 2/3 用来学习,1/3 用来评估,此学习-验证过程交叉进行 3 次. 综上,在同一观测率下,每个派生谓词学习 3×3=9 次,记录其在所有过程中出现的最高精度作为规则学习的最后精度. 实验结果由图 3 和图 4 给出. 图 3 给出观测率与学习精度之间的关系,图 4 给出相应的运行时间.

从图 3 可见,对于两个学习系统而言,学习精度总体来说是随着观测率 λ 的增加而增加. 首先,注意到 λ 的两个极值为 0 和 1. 当 $\lambda=0$ 时,即完全没有观测,派生谓词的训练例只能从动作前提和目标产生,由于 unsafe 和 upstream 不出现在这些部分,因此 $POLDR^{FOIL}$ 和 $POLDR^{ICL}$ 都无法学到它们的规则. 当 $\lambda=1$ 时,即完全可观测,在每个状态下所有派生谓词实例的真值是可知的,这时学习到的规则精度完全由 FOIL 和 ICL 的性能来决定. 其次,当 λ 分别取值 0.2、0.5 和 0.8 时,在大多数情况下两个学习系统的学习精度会增高,少数情况会降低. 这表明当待学习谓词的训练例数目增多时,学习精度不一定会持续增长,此时会受到多方面因素的影响,例如观测的随机性和有用性以及观测名额的分配比例等. 从图 3 中总体来看,$POLDR^{FOIL}$ 的学习精度曲线的走

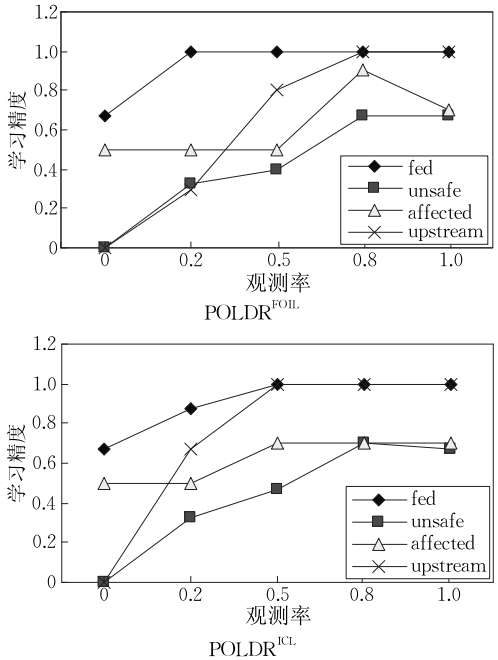

图 3 观测率与学习精度之间的关系

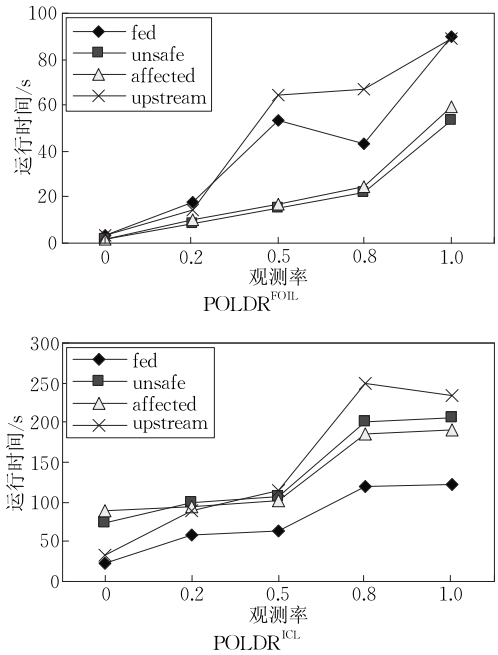

图 4 观测率与运行时间之间的关系

势与 $POLDR^{ICL}$ 的大致相同. 不过,在运行时间方面,$POLDR^{ICL}$ 多数情况下比 $POLDR^{FOIL}$ 更费时(见图 4),这主要是由于 ICL 的运行时间高于 FOIL 所导致的. 在这里还要补充说明的是,本文的工作没有与研究工作[7]进行直接的比较,其原因在于两点:

① FOIL 6.4 可在 http://www.rulequest.com/Personal/下载.
② ACE-1.2.15 可在 http://dtai.cs.kuleuven.be/ACE/doc/下载,ICL 是 ACE 的一个子工具.
③ 问题描述和公布的求解结果可在 http://www.tzi.de/~edelkamp/ipc-4/下载.

(1)研究工作[7]需要给定一个初始的大部分完备的领域理论,其学习的效果在很大程度上取决于初始理论的完备程度,而本文不需要初始理论,是纯归纳的学习;(2)当研究工作[7]没有初始领域理论引导时,就变成了本文在观测率 $\lambda=0$ 时的情形,即在不可观测的环境下学习,二者的结果是一样的.

另外,为了测试算法 4 的效果,这里借用规则学习的另一精确度定义[29]:设一个理想的标准规则为 r^*,学习到的规则为 r,如果 p 在 r^* 中出现而不在 r 出现,则称出现了一个错误.鉴于此,学习到的规则 r 的精确度定义为:(r^* 的前件数－错误数)/r^* 的前件数.另外,要说明的是,由于 FOIL 和 ICL 均学不到规则中的存在量词,因此这里在计算精确度时存在量词是被忽略的.例 5 说明了 POLDR$^{\text{FOIL}}$ 在学习谓词 affected(?x)的规则时调整语义的效果.表 1 给出了当 $\lambda=0.2$ 时两个学习系统语义调整前后的精确度比较.尽管两个系统学习到的具体规则会有些差异,但是与标准规则比较时,它们的学习精确度的变化基本一致(见例 5 和例 6).经过语义调整之后,一半的谓词规则的精准度有所提高,即更贴近于标准规则.不过,要注意的是,图 3 和表 1 中规则精准度的定义是不相同的,前者是通过验证集来得到的,后者是与标准模型比较来得到的.

表 1　当 $\lambda=0.2$ 时两个学习系统语义调整前后的精确度比较(粗斜体显示变化的部分)

	调整前			
	fed	unsafe	affected	upstream
POLDR$^{\text{FOIL}}$	**0.25**	0.33	**0.5**	0.5
POLDR$^{\text{ICL}}$	0.50	0.33	**0.5**	0.5
	调整后			
	fed	unsafe	affected	upstream
POLDR$^{\text{FOIL}}$	**0.5**	0.33	**1**	0.5
POLDR$^{\text{ICL}}$	0.5	0.33	**1**	0.5

例 5.　在 $\lambda=0.2$ 时 POLDR$^{\text{FOIL}}$ 学习到的关于 affected(?x)的规则 r 如下:

affected(A,B):－unsafe(A,B,C)!0.500 000,

其中,A 是状态参数,0.500 000 是在训练例集上的精度.经过算法 4 识别出 B 是角色参数,所属的角色为 *BREAKER*,因此引入角色抽象谓词后的规则 r' 为

affected(A,B):－breaker(A,B),unsafe(A,B,C).

已知 affected(?x)的标准规则 r^* 如下:

```
(:derived (affected ?x－DEVICE)
    (and (breaker ?x)
        (exists (?sx－SIDE)(unsafe ?x ?sx))))
```

则当与标准规则 r^* 比较时,r 的精度为 0.5,r' 的精度为 1.0.

例 6.　在 $\lambda=0.2$ 时 POLDR$^{\text{ICL}}$ 学习到的关于 affected(?x)的规则 r 如下:

```
rule((upstream(A,B,C,D),unsafe(E,F)),
    [type(dnf),cpu(93.0),heur(0.5),
    local(1,0,0,3),total(1,0,0,3)]).
```

该规则其本质是个分类规则,学习到了多余的文字 upstream,且不包含状态参数.经过算法 4,识别出 E 是角色参数,所属的角色为 *BREAKER*,因此引入角色抽象谓词后的规则 r' 为 rule(upstream(A,B,C,D),unsafe(E,F),breaker(E)).

与上述标准规则 r^* 相比,r 的精度为 0.5,r' 的精度为 1.0.

可见,语义调整阶段对于两个学习工具所学到的结果是都是有效的.

6　结束语

本文提出了在部分观测环境下学习派生谓词规则的方法,这有利于规划领域的自动建模或者控制知识的自动获取的研究与实现.学习过程由三部分组成:产生训练例、调用基本工具和语义调整.其中,训练例的产生来自于动作前提、目标和观测中出现的派生谓词实例,用于学习规则的基本工具为 FOIL 和 ICL,语义的调整基于角色参数的引入.用观测来反映动作的非直接效果,从而增加待学习谓词的训练例数目以及学习的精准度,是本文创新性的主要体现.此外,对学习到的规则的语义后处理,是本文对学习技术局限性的一种改进.在派生谓词基准领域的实验表明,本文所提出的学习方法是可行有效的.

进一步的研究工作包括以下 3 个方面:(1)在本文中观测被假定为准确无误的,然而在很多现实环境中观测是会出错的,因此可进一步研究在有噪音的观测环境下如何正确地获取派生谓词规则;(2)基于信息增益的归纳学习方法学习不到在规则前件中应出现的非 fluent 命题,本文通过引入角色的抽象谓词解决了非 fluent 命题是单目谓词的情况,但是并没有解决多目谓词的情况,因此在语义调整阶段,可参考相关工作[28]研究如何处理多目谓词之间的语义关联;(3)FOIL 和 ICL 的候选式中不包括存在量词,因此学习到的规则没有存在量词,可参考相关工作[10]在候选式中增加包含存在量词的模

式,使得可以学习到包含量词的复杂规则.

参 考 文 献

[1] Ghallab M, Nau D, Traverso P. Automated Planning Theory and Practice. Burlington, Massachusetts: Morgan Kaufmann Publishers, 2003

[2] Edelkamp S, Hoffmann J. PDDL2.2: The language for the classical part of the fourth international planning competition. Albert Ludwigs Universität, Institut fur Informatik, Freiburg, Germany: TR-195, 2004

[3] Yoon S, Fern A, Givan R. Learning control knowledge for forward search planning. Journal of Machine Learning Research, 2008, 9: 683-718

[4] Rosa T, McIlraith S. Learning domain control knowledge for TLPlan and beyond//Proceedings of the 3rd Workshop on Planning and Learning in International Conference on Automated Planning and Scheduling. Freiburg, Germany, 2011: 36-43

[5] Thiebaux S, Hoffmann J, Nebel B. In defense of PDDL axioms. Artificial Intelligence, 2005, 168(1-2): 38-69

[6] Bacchus F, Kabanza F. Using temporal logics to express search control knowledge for planning. Artificial Intelligence, 2000, 116(1-2): 123-191

[7] Rao Dong-Ning, Jiang Zhi-Hua, Jiang Yun-Fei, Liu Qiang. Learning first-order rules for derived predicates from plan examples. Chinese Journal of Computers, 2010, 33(2): 251-266(in Chinese)
(饶东宁, 蒋志华, 姜云飞, 刘强. 从规划解中学习一阶派生谓词规则. 计算机学报, 2010, 33(2): 251-266)

[8] Zettlemoyer L, Pasula H, Kaelbling L. Learning planning rules in noisy stochastic worlds//Proceedings of the 20th National Conference on Artificial Intelligence. Pittsburgh, USA, 2005: 911-918

[9] Rao Dong-Ning, Jiang Zhi-Hua, Jiang Yun-Fei, Wu Kang-Heng. Learning non-deterministic action models for Web services from WSBPEL programs. Journal of Computer Research and Development, 2010, 47(3): 445-454(in Chinese)
(饶东宁, 蒋志华, 姜云飞, 吴康恒. 从 WSBPEL 程序中学习 Web 服务的不确定动作模型. 计算机研究与发展, 2010, 47(3): 445-454)

[10] Zhuo H, Yang Q, Hu D, Li L. Learning complex action models with quantifiers and logical implications. Artificial Intelligence, 2010, 174(18): 1540-1569

[11] Quinlan J R. Learning logical definitions from relations. Machine Learning, 1990, 5(3): 239-266

[12] Srivastava S, Immerman N, Zilberstein S. A new representation and associated algorithms for generalized planning. Artificial Intelligence, 2011, 175(2): 615-647

[13] Gazen B C, Knoblock C A. Combining the expressivity of UCPOP with the efficiency of graphplan//Proceedings of the 4th European Conference on Planning. Toulouse, France, 1997: 221-233

[14] Davidson M, Garagnani M. Pre-processing planning domains containing language axioms//Proceedings of the 21st Workshop of the UK Planning and Scheduling SIG. Delft, Netherlands, 2002: 23-34

[15] Gerevini A, Saetti A, Serina I, Toninelli P. Fast planning in domains with derived predicates: An approach based on rule-action graphs and local search//Proceedings of the 20th National Conference on Artificial Intelligence. Pittsburgh, USA, 2005: 1157-1162

[16] Chen Y, Wah B, Hsu C. Temporal planning using subgoal partitioning and resolution in SGPlan. Journal of Artificial Intelligence Research, 2006, 26: 323-369

[17] Helmert M. The fast downward planning system. Journal of Artificial Intelligence Research, 2006, 26: 191-246

[18] Coles A, Smith A. Marvin: A heuristic search planner with online macro-action learning. Journal of Artificial Intelligence Research, 2007, 28: 119-156

[19] Jiang Zhi-Hua, Jiang Yun-Fei. An improved method for calculating activation sets of action derived preconditions. Chinese Journal of Computers, 2007, 29(12): 2061-2073(in Chinese)
(蒋志华, 姜云飞. 一种计算动作派生前提的激活集的改进方法. 计算机学报, 2007, 30(12): 2061-2073)

[20] Bonet B, Geffner H. Planning under partial observability by classical replanning: Theory and experiments//Proceedings of the 22nd International Joint Conference on Artificial Intelligence. Barcelona, Spain, 2011: 1936-1941

[21] Hoffmann J, Brafman R. Contingent planning via heuristic forward search with implicit belief states//Proceedings of the International Conference on Automated Planning and Scheduling. Monterey, USA, 2005: 71-80

[22] Bryce D, Kambhampati S, Smith D E. Planning graph heuristics for belief space search. Journal of Artificial Intelligence Research, 2006, 26: 35-99

[23] Cimatti A, Roveri M, Bertoli P. Conformant planning via symbolic model checking and heuristic search. Artificial Intelligence, 204, 159(1-2): 127-206

[24] Kaelbling L P, Littman M L, Cassandra A R. Planning and acting in partially observable stochastic domains. Artificial Intelligence, 1998, 101(1-2): 99-134

[25] Schmill M D, Oates T, Cohen P R. Learning planning operators in real-world, partially observable environments//Proceedings of the International Conference on Automated Planning and Scheduling, Breckenridge, USA, 2000: 246-253

[26] Amir E, Chang A. Learning partially observable deterministic action models. Journal of Artificial Intelligence Research, 2008, 33: 349-402

[27] Jiang Zhi-Hua. Research of Theory and Algorithm of Derived Planning Problems[Ph.D. dissertation]. Sun Yat-sen University, Guangzhou, 2008(in Chinese)
(蒋志华. 派生规划问题的理论与算法研究[博士学位论文]. 中山大学, 广州, 2008)

[28] Jiang Zhi-Hua, Rao Dong-Ning, Jiang Yun-Fei, Yang Tian-Qi. Acquiring automatically generalized plans for planning domains with derived predicates. Chinese Journal of Computers, 2014, 37(8): 1820-1838(in Chinese)
(蒋志华，饶东宁，姜云飞，杨天奇. 自动获取派生谓词规划领域的通用规划. 计算机学报，2014，37(8)：1820-1838)

[29] Yang Q, Wu K, Jiang Y. Learning action models from plan traces using weighted MAX-SAT. Artificial Intelligence, 2007, 171(2-3): 107-143

RAO Dong-Ning, born in 1977, Ph. D., associate professor. His main research interests include automated planning and graph theory.

JIANG Zhi-Hua, born in 1978, Ph. D., associate professor. Her main research interest is automated planning.

JIANG Yun-Fei, born in 1945, M. S., professor. His main research interests include automated planning and model-based diagnosis.

DENG Yu-Hui, born in 1973, Ph. D., professor. His main research interests include data storage and green computing.

Background

The work in this paper is devoted to learn derived predicate rules from successful plans under partial observability, and this is beneficial for the study on automatically modeling planning domains or automatically acquiring control knowledge. The research here is in the field of automated planning, which is a subfield of Artificial Intelligence. It is time-consuming and error-pruning to describe domain models or knowledge from scratch, even for experts. Therefore, since 1990s, a lot of machine learning technologies have been applied into planning to automatically acquire, for example, action models, control knowledge, heuristic functions, etc.

Learning derived predicate rules, or domain axioms, became a new issue in the direction of this research line in the last decade. The key points of this task are how to produce training examples for predicates to be learned and how to make the learned rules more correct in the aspect of semantics, since we have developed some basic tools for rule learning or mining. Some primary work on this issue is listed as follows. At first, Zettlemoyer[8] obtained new predicates by applying pre-defined operators and added them into action models as derived effects. Actually, they did not form independent rules for derived predicates. Then, Rao[7] learned derived rules by refining an initial and imperfect domain theory based on training data. They gave some principles of producing training examples for predicates to be learned, but the training set was very small and the learning accuracy was largely dependent on the perfect degree of the initial theory, instead of training data. Recently, Rosa[4] learned derived predicates in three fixed types: compound, abstract and recursive, to improve the express ability of control knowledge. They obtained rules by pre-defining rule models, instead of emulating all possibilities of combination on the predicate set.

Our work is devoted to solving the two key points of this learning task mentioned above. On one hand, we use observations to reflect actions' indirect effects so that the number of training examples for predicates to be learned is largely increased. We also experiment about the relationship between the rate of observations and the accuracy of learning. On the other hand, we define role parameters and then introduce character predicates of roles into learned rules, in order to complement those non-fluent predicates that can not be learned by the basic tools. And it is the first time to present a method to learn derived predicate rules under partial observability and the first time to improve the semantics of learned rules by role characteristics in a post-processing phrase.

Our work is supported by the Fundamental Research Funds for the Central Universities (No. 21615438), the National Natural Science Foundation of China (Nos. 61003179, 61100134, 61272073) and the Guangdong Natural Science Foundation (Nos. S2013020012865, S2011040001427). These projects are involved into developing methods and tools for automatically acquiring action models and derived predicate rules, and also applying the learned results into real applications. Both of action models and derived predicate rules are key components of planning domain models. Therefore, the research results of these projects are beneficial for automatically modeling planning domains. This will relieve people from the modeling efforts. We have developed basic methods and tools for learning action models and derived predicate rules in an unobservable environment. Now, we extend these under partial observability for it is more practical and economical in the real world.