从规划解中学习一阶派生谓词规则

饶东宁[1),2)] 蒋志华[3)] 姜云飞[2)] 刘 强[2)]

[1)](广东工业大学计算机学院 广州 510090)

[2)](中山大学信息科技学院软件研究所 广州 510275)

[3)](暨南大学计算机科学系 广州 510632)

摘 要 派生谓词是描述动作非直接效果的主要方式.但是由人类专家设计的派生谓词规则(即领域理论)不能保证总是正确或者完备的,因此有时很难解释一个观察到的规划解为什么是有效的.结合归纳学习与分析学习的优点,文中提出一种称为 FODRL(First-Order Derived Rules Learning) 的算法,在不完美的初始领域理论的引导下从观察到的规划解中学习一阶派生谓词规则.FODRL 基于归纳学习算法 FOIL(First-Order Inductive Learning),最主要的改进是可以使用派生谓词的激活集来扩大搜索步,从而提高学习到的规则的精确度.学习过程分为两个步骤:先从规划解中提取训练例,然后学习能够最好拟合训练例和初始领域理论的一阶规则集.在 PSR 和 PROMELA 两个派生规划领域进行实验,结果表明,在大部分情况下 FODRL 比 FOIL(甚至包括其变型算法 FOCL)学习到的规则的精确度都要高.

关键词 人工智能;智能规划;派生谓词规则;归纳学习;激活集

中图法分类号 TP182 **DOI 号**: 10.3724/SP.J.1016.2009.00251

Learning First-Order Rules for Derived Predicates from Plan Examples

RAO Dong-Ning[1),2)] JIANG Zhi-Hua[3)] JIANG Yun-Fei[2)] LIU Qiang[2)]

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

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

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

Abstract Derived predicates are a natural way to depict indirect effects of domain actions, and their truth values in the current state are inferred from that of other predicates via domain rules. However, domain rules designed by human experts cannot be guaranteed to be correct or complete. So it is often difficult to explain why an observed plan is valid under imperfect domain rules. Combining inductive learning with analytical learning, in this paper, we develop an algorithm called FODRL (First-Order Derived Rules Learning) to automatically discover first-order rules for derived predicates from observed plans under an initial domain theory. FODRL is based on the pure inductive learning system FOIL (First-Order Inductive Learning), which learns a new rule that covers partial positive examples but avoids all negative examples once a time, until all positive examples are covered. However, better than FOIL, FODRL uses activation sets of derived predicates to expand search steps so as to improve the accuracy of learned rules. An activation set is a minimal set of basic facts or predicates which can make a derived predicate hold true under domain rules. The learning process is divided into two steps: first, extract training examples from observed plans; then, learn first-order rules for derived predicates which can best fit training examples and the initial domain theory. We experiment in two derived planning domains,

收稿日期:2008-07-13;最终修改稿收到日期:2009-03-23.本课题得到国家自然科学基金(60173039)和广东工业大学博士启动基金项目(093032)资助.**饶东宁**,男,1977 年生,博士,主要研究方向为智能规划和图论. E-mail: rdn2006@163.com. **蒋志华**,女,1978 年生,博士,讲师,主要研究方向为智能规划和知识推理. **姜云飞**,男,1945 年生,教授,博士生导师,主要研究方向为智能规划、知识推理以及模型诊断. **刘 强**,男,1978 年生,博士,主要研究方向为智能规划.

PSR and PROMELA. The results show that, with the guidance of an initial domain theory, the rules learned by FODRL are more accurate than those from FOIL, even FOCL (a descendant of FOIL).

Keywords artificial intelligence; automated planning; derived predicate rules; inductive learning; activation set

1 引 言

规划是关于动作的推理.在这个过程中,首先要预期各种动作的效果,然后制定满足某些目标的动作执行策略.一般地,在对规划领域建模的时候,动作模型越简洁,越有利于规划问题的求解和执行得到的规划解.根据动作发生的对象,动作的效果可以分为两类:直接效果和非直接效果.每次执行一个动作,动作的直接效果通过动作模型在当前状态下发生,而动作的非直接效果通过推理规则在改变后的状态下进行推导.这样的推理规则独立于任何动作模型,使得规划领域的描述变得清晰和简洁.

为了建立表示动作非直接效果的推理规则,在规划领域描述中,谓词分为两类:基本谓词和派生谓词[1].二者的差别是,基本谓词可以出现在领域动作的效果中,而派生谓词不能,它们在当前状态下的真值是由封闭世界假设中其它谓词的真值通过领域规则推导出来.派生谓词不能作为动作的直接效果,但是可以出现在动作的前提(称为派生前提)和目标状态(称为派生目标)中.有了派生谓词,动作模型只需要描述动作的最直接效果,而事物之间千丝万缕的因果联系可以用派生谓词规则来表示,很好地解决了规划领域描述的框架问题[2]和动作衍生问题[3].

一般地,某个领域的派生谓词规则是由熟悉该领域的人类专家所设定的.这些规则集合称为领域理论.然而,由于领域的复杂性,在大多数情况下这些规则很难保证是正确而且完备的.例如,由于信息不足某些规则的条件被遗漏了,或者由于错误的理解某些规则包含不必要的条件.如果预先给定的领域规则是错误的,可能会导致错误的规划解或者原问题无解.如果给定的规则是不完备的,可能无法解释一个 Agent 观察到的规划解为什么是有效的.比如,动作的某个派生前提为什么在当前状态下是成立的,再比如,一个期待的目标在最终状态如何通过规则推导出来?因此,从实际观察到的规划解中学习派生谓词的规则,一方面可以验证初始的领域理论是否是完美的,另一方面可以挖掘领域中隐含的因果联系以完善领域理论.这是一个有趣而且有前景的研究课题.

规划和学习的结合已经成为人工智能研究领域的一大热点.在最近几年的国际会议 IJCAI、AAAI、ICML、ICAPS 以及权威期刊 AI、JAIR 上已有不少文章讨论将学习技术应用到规划研究领域中,例如学习动作模型[4-6]、学习控制知识[7-8]、学习启发式函数[9]、学习动作策略[10]等.动作模型的学习是从规划解中寻找状态之间、动作之间的各种关联,以提取动作的前提和效果.控制知识的学习是从动作序列中提取一般规律,用于引导规划解的搜索过程.启发式函数的学习是调整各种启发项的权值,用以加速启发式搜索过程.动作策略的学习从规划解中寻找针对某领域的状态-动作规则,使得应用这些规则无需搜索就可找到规划解.在最新的进展上,在今年即将举行的国际规划竞赛 IPC 2008 上,将首次增加一个 learning track,以测试规划系统的学习性能和规范各种学习问题描述.

虽然已经有不少方法用于学习领域规则,例如遗传算法、决策树学习、FOIL 或者 EBL 等,但是至今还没有研究工作学习规划领域的派生谓词规则.由于一个规划领域可能包含多个派生谓词,因此派生谓词规则学习属于多谓词学习[11-12],并且学习的是一阶规则,是比较复杂的学习任务.本文从规划解中学习派生谓词规则分为两个步骤:首先,在假定动作模型已知的前提下,从规划解中提取学习问题所需要的训练例.规划解不能直接作为学习问题的训练例,但是可以将规划解应用于初始状态产生的〈状态-事实〉序对来作为规则学习的训练例;其次,结合归纳学习和分析学习的优点,使用初始领域理论来推导派生谓词的激活集,这些激活集可以作为构造规则前件的候选式,使得归纳学习算法在假设空间中一次移动多步,极大地改变搜索的方向.本文提出的方法在两个派生规划领域 PSR 和 PROMELA 进行实验,从而验证方法的可行性和有效性.

本文第 2 节介绍本文的研究背景,包括各种学习规则的方法和派生规划问题的综述;第 3 节从训练例的构造和初始领域理论等方面来具体地描述要

处理的学习问题;第 4 节给出学习的主算法 FODRL 以及如何根据初始领域理论建立规则图和在规则图上计算用于扩展搜索步的派生谓词的激活集;第 5 节是实验和分析部分;最后一节是结语以及对未来工作的展望.

2　研究背景

本节从机器学习领域的各种学习规则的方法以及智能规划领域的派生规划问题描述等方面,来介绍本文的研究背景.

2.1　学习规则集合

基本的规则学习方法分为归纳学习和分析学习.纯粹的归纳学习方法通过在训练样例中寻找经验化的规律来形成一般假设,如决策树归纳(Decision Tree)、遗传算法(Genetic Algorithm)、序列覆盖算法(Sequential Covering Algorithm)和 FOIL(First-Order Inductive Learning)[13]等.纯粹的分析方法使用先验知识演绎推导一般假设,如基于解释的学习 EBL[14].在 EBL 中,先验知识用于分析观察到的学习样例怎样满足目标概念,然后这个解释被用于区分训练样例中相关的特征.EBL 已被成功地用于在各种规划和调度任务中学习控制知识,如规划系统 PRODIGY 和 SOAR 等.为了获得有先验知识时更高的泛化精度和依赖训练数据克服先验知识的不足,应该将归纳和分析两种机制结合起来,如 KBANN、EBNN、FOCL[15]等算法.

FOCL 基于归纳系统 FOIL,在修正假设时使用领域理论来扩展可用的文字集合,在单个搜索步中一次加入多个文字,极大地改变搜索方向.本文的方法与 FOCL 相似,区别在于:(1) FOCL 所接受的初始领域理论不允许包含递归的规则,而本文的 FODRL 算法允许递归规则;(2) FOCL 学习的是某个概念的分类规则,而本文学习的是派生谓词规则,表示动作的各种效果之间的因果联系;(3) FOCL 的非操作型文字(即不出现在训练例中的谓词)的作用仅仅是使得领域理论具有层次结构,而规划领域中的派生谓词可以作为动作的前提和规划问题的目标.

以上的规则学习方法都是单谓词学习,一次只学习一个概念,采用了强假设:要学习的概念之间是彼此独立的.因此,学习概念的顺序可能会影响学习的结果.多谓词学习同时学习几个概念,在学习过程中要不断检测学习到的规则是否满足谓词之间的依赖关系,因此比单谓词学习要复杂得多.多谓词学习一般显式地表示谓词之间的依赖关系,使用依赖层次来引导学习过程,如 MPL[11]和 RTL[12]等算法.本文的学习主算法采用了多谓词学习方法的主体框架.

相关工作还包括:Zettlemoyer 等提出的一种算法在有噪音的随机环境下学习动作的效果模型[16],将预先定义的操作子应用到各个文字上来迭代地产生新的谓词,并且在学习到的动作模型上测试其有用性.但是,他们的工作是学习动作模型中的派生效果而非派生谓词的规则.

2.2　派生规划问题

简单地说,包含派生谓词的规划问题称为派生规划问题.派生谓词规则实质上是 PDDL 公理.McDermott 最早将"公理"一词引入到规划问题描述语言 PDDL 中,称为 PDDL 公理[17].之后,Thiebaux 等给出了 PDDL 公理的精确语义,并且证明在限定规划长度和领域描述是多项式级的条件下,公理是规划描述语言的基本要素,不可能用其它的语言要素来完全替换[1].设 B 是基本谓词符号集合,D 是派生谓词符号集合,$B\cap D=\varnothing$.R 是派生谓词规则(或简称为规则)集合,R 中的规则具有如下形式:(:derived $(d?\boldsymbol{x})(f?\boldsymbol{x})$),其中 $d\in D$,f 是由集合 $B\cup D$中的谓词符号所构建的一阶公式,f 中的自由变量与 d 中的相同,均为 $\boldsymbol{x}$.直观地,规则(:derived $(d?\boldsymbol{x})(f?\boldsymbol{x})$)意味着当 $(f?\boldsymbol{x})$在某个状态为真的时候,应该推导出具有相同参数的 $(d?\boldsymbol{x})$在该状态下也为真.在本文中,派生规则集合 R 称为领域理论.

规则中一阶公式 f 的否定范式 NNF① 如果使用了某个谓词的否定形式,则该谓词应该首先被定义.因此,规则集的层次结构定义如下.

定义 1.　规则集 R 是分层的,当且仅当存在着派生谓词集合 D 的一个划分 $\{D_i,1\leq i\leq n\}$,对于任意 $d_i\in D_i$及每个规则(:derived$(d_i?\boldsymbol{x})(f?\boldsymbol{x})$)$\in R$,满足下列条件:

(1) 如果 $d_j\in D_j$ 出现在 $NNF(f?\boldsymbol{x})$中,那么 $j\leq i$;

(2) 如果 $d_j\in D_j$在 $NNF(f?\boldsymbol{x})$中是以否定的形式出现的,那么 $j<i$.

例 1.　在用 PDDL 2.2 语言[18]描述的 PSR 领域(处理供电系统的重新配置问题)的 DerivedADL 版本②(附录 A)中,$D_1=\{\text{unsafe},\text{upstream}\}$,$D_2=$

① 任何一个谓词公式可以通过以下的"否定符号内移"的转化规则变成其 NNF:①$\neg\ \exists x:\Phi\rightarrow\forall x:\neg\Phi$;②$\neg\ \forall x:\Phi\rightarrow\exists x:\neg\Phi$;③$\neg\ \wedge\Phi_i\rightarrow\vee\neg\Phi_i$;④$\neg\ \vee\Phi_i\rightarrow\wedge\neg\Phi_i$.

② http://ls5-www.cs.uni-dortmund.de/~edelkamp/ipc-4/可下载 PSR 的领域描述.

{affected,fed}.

定义 2. D 上的任何一种层次结构$\{D_i,1\leq i\leq n\}$自然导致了 R 上的层次结构$\{R_i,1\leq i\leq n\}$：

$$R_i=\{(:\text{derived}(d_i?\boldsymbol{x})(f?\boldsymbol{x}))\in R\mid d_i\in D_i\}.$$

不管选用哪一种层次结构，从最低层开始以任意顺序应用规则直到到达一个固定点，然后前进到下一层，以任意顺序应用规则直到到达一个固定点，直至最后一层. 这样的推理顺序将导致相同的最终固定点[19]，而这个最终的固定点即为规则集的逻辑结论集合.

在下列定义中，$\boldsymbol{x}$ 表示变量向量，$\boldsymbol{t}$ 表示基原子向量. 对任意 $b\in B$，$(b\ \boldsymbol{t})$称为基本事实. 对任意$d\in D$，$(d\ \boldsymbol{t})$称为派生事实. 设状态 s 是由基本事实组成的集合. 在状态 s 中，应用规则 $r=(:\text{derived}(d?\boldsymbol{x})(f?\boldsymbol{x}))$将推导出派生事实集合$[\![r]\!](s)$，即规则 r 在 s 中的逻辑结论.

定义 3. $[\![r]\!](s)=\{(d\ \boldsymbol{t})\mid s\models(f\ \boldsymbol{t}),\boldsymbol{t}$ 是基向量$\}$. 其中，$\models$是逻辑蕴涵符号.

定义 4. 对一个分层的规则集 R，设$\{R_i,1\leq i\leq n\}$是任意的一种分层结构. 对于每个状态 s，有

(1) $[\![R]\!]_0(s)=\varnothing$；

(2) $[\![R]\!]_i(s)=\bigcap\{DF\mid \bigcup_{r\in R_i}[\![r]\!](s)\cup[\![R]\!]_{i-1}(s)\subseteq DF\},1\leq i\leq n$；

则 $[\![R]\!](s)=[\![R]\!]_n(s)$

在定义 4 中，$[\![R]\!](s)$是包含$[\![R]\!]_{i-1}(s)$及第 i 层规则的逻辑结论的最小派生事实集合. 如果一个由基本谓词和派生谓词组成的公式 f 在状态 s 中成立，有以下定义：

定义 5. $s\models_R f$ 当且仅当 $s\cup[\![R]\!](s)\models f$.

为了方便规划问题描述，将状态 s 进行扩充.

定义 6. 函数 $\mathcal{D}$ 将状态 s 映射为在规则集 R 下的扩充状态：

$$\mathcal{D}(s)=s\cup[\![R]\!](s)$$

根据规则集的语义，按照任何一种层次结构在 s 上应用规则集 R 都产生相同的$\mathcal{D}(s)$. 在派生规划问题描述中，动作模型可以如下定义：

定义 7. 动作 a 是三元组$\langle pre(a),add(a),del(a)\rangle$，$pre(a)$是动作 a 的前提集合，$add(a)$是动作 a 的增加效果集合，$del(a)$是动作 a 的删除效果集合. 派生谓词集合 D 中的符号不出现在 $add(a)$和 $del(a)$中.

对任意 $b\in B$，如果 b 出现在 $pre(a)$，则 b 称为 a 的基本前提. 对任意 $d\in D$，如果 d 出现在 $pre(a)$，则 d 称为 a 的派生前提. 一个动作在某个状态中是可应用的，当且仅当该动作的所有前提在该状态中是成立的.

定义 8. 动作 a 在状态 s 中是可应用的，当且仅当 $s\models_R pre(a)$.

如果动作模型中不包含任何变量，则该动作称为原子动作. 在确定性的派生规划问题中，原子动作应用于当前状态将产生唯一的后继状态.

定义 9. 在状态 s 下应用原子动作 a 的函数 $apply$ 定义如下：

$$s'=apply(a,s)=(s\backslash del(a))\cup add(a),$$

s'称为后继状态.

下面，给出派生规划领域的形式化定义.

定义 10. 派生规划领域是一个六元组 $\Sigma=\langle B,D,R,S,A,X\rangle$：

- B 是基本谓词符号集合；
- D 是派生谓词符号集合，且 $B\cap D=\varnothing$；
- R 是派生谓词规则集合，$R=\{r\mid r=(:\text{derived}(d?\boldsymbol{x})(f?\boldsymbol{x}))\}$.
- S 是状态集合；
- A 是动作集合，$A=\{a\mid a=\langle pre(a),add(a),del(a)\rangle\}$；
- X 是转换函数，$X(s,a)=\mathcal{D}(apply(a,s))$，当动作 a 在 s 中是可应用的；

例 2. PSR 问题的领域描述见附录 A，要处理的是在某些线路出故障的情况下，供电系统要调整相应的设备开关来给需要供电的线路恢复供电. 基本谓词集合 $B=\{\text{ext},\text{breaker},\text{closed},\text{faulty},\text{con}\}$，派生谓词集合 $D=\{\text{upstream},\text{unsafe},\text{affected},\text{fed}\}$，带有条件效果的动作集合 $A=\{\text{wait},\text{open},\text{close}\}$. 规则集合 R 中共有 4 条派生谓词规则，规则条件中的一阶公式允许逻辑运算(and，or，not)、存在量词(exist)及全称量词(forall)等. 谓词集合中各个谓词符号的含义如下：

(ext ?l ?x ?s)：线路?l 连接设备?x 的某一边?s；

(breaker ?x)：设备?x 是一个电源；

(closed ?x)：设备?x 是闭合的；

(faulty ?l)：线路?l 有故障；

(con ?x ?sx ?y ?sy)：设备?x 的?sx 边与设备?y 的?sy 边相连；

(upstream ?x ?sx ?y ?sy)：从设备?x 的?sx 边有电流流向设备?y 的?sy 边；

(unsafe ?x ?sx)：设备?x 的?sx 边是不安全的；

(affected ?x)：设备?x 受到故障线路的影响；

(fed ?l)：线路?l 需要供电.

定义 11. 一个派生规划问题是一个三元组 $\Sigma=$

$\langle \Sigma, I, G\rangle$,其中,$\Sigma$ 是规划领域描述;I 是初始状态;G 是目标状态.

定义 12. 派生规划问题 $\Pi=\langle \Sigma, I, G\rangle$ 的解是一个原子动作序列 $\langle a_1, a_2, \cdots, a_n\rangle$,该动作序列通过一系列的状态变换 $\langle s_0, s_1, s_2, \cdots, s_n\rangle$ 将初始状态 I 转化为目标状态 G:

(1) $s_0=I$;

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

(3) $G\subseteq s_n$.

3 学习问题描述

初始的领域理论可能是不正确的,或者不完备的.不管是哪一种情况,均称为不完美的领域理论.不完美的领域理论可能会导致原问题得到错误的规划解或者无解.初始领域理论可以由专家根据经验提供,也可以通过更抽象的背景知识来构建[20].不完美的领域理论可能包含以下四种错误:

(1) 多余规则;

(2) 缺少规则;

(3) 多余的规则条件;

(4) 缺少的规则条件.

多余规则是指领域理论中定义派生谓词的某个规则整个是错误的.缺少规则是指领域理论缺少某个派生谓词的规则,因而领域理论是不完备的.多余的规则条件是指某个规则的条件部分包含错误的谓词.缺少的规则条件是指某个规则的条件部分缺少某个必要的谓词.一般地,多余的规则或者规则中缺少必要的条件,会使得领域理论过于一般,而缺少规则或者规则中包含多余的条件,会使得领域理论过于特殊.但是不管包含何种错误的初始理论,都要求每个谓词所有参数的类型信息是已知的.第 5 节的实验部分具体分析了各类领域理论错误对于学习到的规则的精确度的影响.

例 3. 构造 PSR 领域的包含四类错误(已用粗体表示)的初始领域理论,如表 1 所示.其中,side1 和 side2 是理论常量,可以出现在规则中.各个谓词符号的含义见例 2.与附录 A 中完美的领域理论相比,表 1 中的规则包含如下错误:规则 $r1$ 的条件部分包含了多余的谓词 break,规则 $r2$ 的条件部分为空,即缺少关于派生谓词 unsafe 的规则,规则 $r4$ 是关于派生谓词 fed 错误的规则,规则 $r5$ 的条件部分缺少必要的谓词 closed.此外,规则 $r3$ 没有错误,是完全正确的.

表 1　构造 PSR 领域的包含四类错误的初始领域理论描述

```
r1: (:derived (upstream ?x-DEVICE
      ?sx-SIDE ?y-DEVICE ?sy-SIDE)
      (and (closed ?x)
          多余条件(break ?x)
          (or (and (=?sx side1) (con ?x side2 ?y ?sy))
              (and (=?sx side2) (con ?x side1 ?y ?sy))
              (ex ists(?z-DEVICE)
                  (and (closed ?z)
                    (or (and (con ?z side1 ?y ?sy)
                             (upstream ?x ?sx ?z side2))
                        (and (con ?z side2 ?y ?sy)
                             (upstream ?x ?sx ?z side1)))))))))
r2: (:derived (unsafe ?x-DEVICE ?sx-SIDE))   缺少规则
r3: (:derived (affected ?x-DEVICE)
          (and (breaker ?x)
               (exists (?sx-SIDE)(unsafe ?x?sx))))
r4: (:derived (fed ?l-LINE)(faulty ?l)) 多余规则
r5: (:derived (fed ?l-LINE)
      (exists (?x-DEVICE)
        (and 缺少条件(closed ?x)
          (or (and (ext ?l ?x side1)
              (or (breaker ?x)
                  (exists (?y-DEVICE)
                  (exists (?sy-SIDE)
                          (and (breaker ?y)
                            (upstream ?y ?sy ?x side2))))))
              (and (ext ?l ?x side2)
                   (or (breaker ?x)
                       (exists (?y-DEVICE)
                       (exists (?sy-SIDE)
                         (and (breaker ?y)
                         (upstream ?y ?sy ?x side1))))))))))
```

规划解可以由在实验环境中活动的 Agent 观察得到.但是一个规划解只包含动作序列,派生谓词只在其中充当动作前提,因此规划解不能直接作为学习问题的训练例.对于一个状态 s 而言,派生事实 $(d\ \boldsymbol{t})$ 只存在成立或不成立两种情况.如果 $(d\ \boldsymbol{t})$ 在 s 中成立,则说明 $s\models_R(d\ \boldsymbol{t})$,该状态可以作为学习规则的正例;反之,则为反例.因此,在动作模型完全已知的情况下,可以把规划解应用于初始状态,从而得到状态转化序列,这些状态可以作为学习规则的训练例.

定义 13. 设训练例为序对 $\langle s, d\rangle$,其中 s 是状态,d 是派生事实.如果有 $s\models_R d$,则称 $\langle s, d\rangle$ 为关于 d 的正例,反之,则称 $\langle s, d\rangle$ 为关于 d 的反例.

下面给出从规划解中产生训练例的 GTE(Generating Training Examples)算法.该算法产生学习所需要的正例集合 T_p 和反例集合 T_N.在该算法中,$|P|$ 表示规划解的长度,即所包含的动作个数.由于 PDDL 动作中允许否定前提,因此派生前提如果是以肯定形式出现的,则产生一个正例,如果是以否定形式出现的,则产生一个反例.s_i 是在状态 s_{i-1} 下应

用动作 a_i 产生的后继状态($Apply$ 函数见定义 9). 对于每个派生目标,在最终状态下是成立的,因而产生一个正例. 为了增加训练例的个数且简化学习过程,对于所有非最终状态,均认为目标不成立. 对每个规划解分别应用 GTE 算法,再将所有正例集合合并在一起,所有反例集合合并在一起,则可以得到最终的训练集合.

算法 1. GTE(Generating Training Examples)算法.

输入:派生规划问题$\langle \Sigma, I, G\rangle$,规划解 P

输出:正例集合 T_p,反例集合 T_N

Begin

1. $T_p \leftarrow \varnothing$, $T_N \leftarrow \varnothing$;
2. $s_0 \leftarrow I$;
3. For $i=1$ to $|P|$ do
4. 对于动作 a_i 的每个派生前提 d
5. 如果 d 是 a_i 的肯定前提,则
6. $T_p \leftarrow T_p \cup \{\langle s_{i-1}, d\rangle\}$;
7. 否则
8. $T_N \leftarrow T_N \cup \{\langle s_{i-1}, d\rangle\}$;
9. $s_i = Apply(a_i, s_{i-1})$;
10. 对于每个派生目标 $g \in G$
11. For $i=0$ to $|P|-1$ do
12. $T_N \leftarrow T_N \cup \{\langle s_j, g\rangle\}$;
13. $T_p \leftarrow T_p \cup \{\langle s_{|P|}, g\rangle\}$;

End.

例 4. 设一个 PSR 规划问题的解为动作序列〈WAIT,OPEN(SD8),CLOSE(CB2)〉,对应的状态序列为$\langle s_0, s_1, s_2, s_3\rangle$,其中,$s_0$ 是初始状态,s_3 是目标状态. 应用 GTE 算法,产生了如表 2 所示的训练集合.

表 2 〈状态-事实〉序对形式的训练例

T_p	T_N
$\langle s0$,(affected $cb1)\rangle$	$\langle s1$,(affected $cb1)\rangle$
$\langle s0$,(affected $cb2)\rangle$	$\langle s1$,(affected $cb2)\rangle$
$\langle s3$,(fed $l5)\rangle$	$\langle s2$,(affected $cb1)\rangle$
$\langle s3$,(fed $l6)\rangle$	$\langle s2$,(affected $cb2)\rangle$
	$\langle s3$,(affected $cb1)\rangle$
	$\langle s3$,(affected $cb2)\rangle$
	$\langle s0$,(fed $l5)\rangle$
	$\langle s0$,(fed $l6)\rangle$
	$\langle s1$,(fed $l5)\rangle$
	$\langle s1$,(fed $l6)\rangle$
	$\langle s2$,(fed $l5)\rangle$
	$\langle s2$,(fed $l6)\rangle$

从规划解中产生训练例之后,本文的学习任务为:

已知:

- 一个训练样例集合 T,可能包含错误;
- 一个领域理论 R,可能包含错误;
- 假设空间 H;

求解:一个最好地拟合训练样例和领域理论的假设 h.

在上述学习任务中,每个假设 h 是类似 Horn 子句的子句集合,区别在于:文字中不允许包含函数符号,因而比 Horn 子句更受限制;子句体中允许出现负文字,因而比 Horn 子句更有表征力. 每个子句是一个一阶的派生谓词规则,子句头是单个派生谓词 $d \in D$,子句体是由 $B \cup D$ 中的谓词构成的文字集合. 要求学习到的假设最好地拟合训练样例和领域理论,是指使假设在数据和领域理论上的错误率的某种综合度量最小化. 定义 h 关于训练集 T 的错误率 $error_T(h)$ 为 T 中被 h 误分类的样例所占的比例,同样地,定义 h 关于领域理论 R 的错误率 $error_R(h)$ 为 h 与 R 在分类一个随机实例时不一致的概率. 希望输出的假设是使得上述错误率的某种综合度量最小化,如

$$\arg\min_{h \in H}(k_T \times error_T(h) + k_R \times error_R(h)),$$

K_T 和 K_R 分别表示拟合数据和拟合理论两者的相对重要程度. 如果初始领域理论的精确度较差,却有大量可靠数据,则可以选择较大的 K_T 值. 如果理论的精确度较高,数据样本很小并且存在大量噪声,则可以选择较大的 K_R 值.

4 学习算法

在定义了上述学习任务之后,下面先来看看学习规则的整个过程(如图 1 所示). 首先使用 GTE 算法从规划解和规划问题描述中产生学习问题所需的训练例集合. 其次,规划问题描述中包含初始的领域理论,可以用来建立规则图,规则图描述了谓词之间的依赖关系. 学习算法 FODRL 基于基本归纳系统 FOIL,在假设空间寻找拟合训练例的最佳假设. 通过 ASS(Activation Set Searching)算法得到派生谓词的激活集,用来扩大 FOIL 的搜索步,使得一次可以考虑多个候选文字. 简单地说,派生谓词的激活集指的是在领域理论下能够推导出该派生谓词的基本事实或谓词集合. 引入激活集的原因在于,GTE 算法所产生的训练例中的状态只包含基本谓词的实例,因此当规则前件的候选文字是派生谓词时,派生谓词的规则尚未知道,只有将派生谓词转换为对应的基本谓词集合(即激活集)才能判断新构造的规则对训练例的拟合程度. 最后输出的是针对规划领域的一阶派生谓词规则集合.

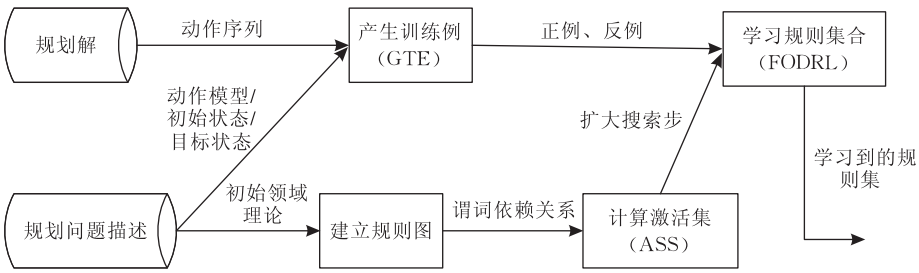


图 1　学习派生谓词规则的过程

根据图 1 所描述的规则学习过程,下面分别处理如下问题:(1) 建立学习规则的主算法;(2) 从初始领域理论建立规则图;(3) 在规则图上计算派生谓词的激活集.

4.1　学习算法 FODRL

学习派生谓词规则的主算法称为 FODRL (First-Order Derived Rules Learning),基于 RTL 的多谓词学习框架和 FOIL 的归纳学习算法. FODRL 算法描述如下.

算法 2.　FODRL(First-Order Derived Rules Learning)算法.

输入:要学习的谓词集合 Learned_predicates, 谓词集 Predicates, 训练集 Examples

输出:规则集 Learned_rules

Begin

1. Learned_predicates 中的谓词形成队列;
2. Learned_rules←{};
3. 当队列不空
4. 　取队列中的第一个谓词 Target_predicate
5. 　Pos←Examples 中 Target_predicate 的正例;
6. 　Neg←Examples 中 Target_predicate 的反例;
7. 　NewRule←没有前件的谓词 Target_predicate 的规则;
8. 　NewRuleNeg←Neg;
9. 　当 NewRuleNeg 不空,增加新候选前件以特化 NewRule;
10. 　基于谓词集 Predicates 对 NewRule 生成候选新文字 NewLiteral,
11. 　如果 NewLiteral 是派生谓词,则
12. 　Candidates←基于规则图通过 ASS 算法求 NewLiteral 的激活集;
13. 　否则,Candidates←NewLiteral;
14. 　Best_candidate←argmax Fodrl_Gain(L, NewRule);
15. 　把 Best_ Candidate 加入到 NewRule 的前件;
16. 　NewRuleNeg←NewRuleNeg 中满足 NewRule 前件的子集;
17. 　Learned_rules←Learned_rules+NewRule;
18. 　NewRule 加入到规则图中;
19. 　Pos←Pos-{被 NewRule 覆盖的 Pos 成员};
20. 　如果 Pos 非空,则 Target_predicate 加入队列末尾
21. 　Examples←Examples-{被 NewRule 覆盖的 Examples 成员};
22. 返回 Learned_rules;

End.

FODRL 由两层循环来进行控制. 外层循环与 RTL 类似,将要学习的谓词排成队列,每次选取排在最前面的谓词 Target_predicate 来学习规则. 如果学习到的规则仍有未覆盖的正例,则将 Target_predicate 放在队列的末尾留待以后再学习. 交叉学习不同谓词的规则,可以利用谓词之间的依赖关系来构建规则的前件,是一种多谓词学习过程.

FODRL 的内层循环与 FOIL 类似,每次学习一个覆盖部分正例而避开所有反例的规则. 内层循环执行一般到特殊的爬山搜索,开始于最一般的前件,然后增加文字以使规则特化直到避开所有的反例. 然而,与 FOIL 算法不同的是,FODRL 算法在生成候选式 Candidates 时使用了初始领域理论来扩大搜索步(步 11~12). FODRL 生成两类候选式:单个文字,被单独加到规则前件中;文字集合,被整个加到规则前件中. 单个文字为谓词集 Predicates 中出现的任意基本谓词或其否定,但是要求至少包含当前规则中一个已有的变量. 如果候选新文字是派生谓词,则需在领域理论下推导其激活集(文字集合)来代替 FODRL 通过在训练数据上测量规则的性能,每一步从候选式中选择最有希望的文字或者文字集合. FODRL 使用评估函数 Fodrl_gain 以估计新候选式的效用,它基于加入候选式之后的正例和反例的约束数目. 设 R′为加入候选式 L 到规则 R 后生成的规则,则

$$Fodrl\_Gain(L,R)=t\left(\log_2\frac{p_1}{p_1+n_1}-\log_2\frac{p_0}{p_0+n_0}\right),$$

其中,p_0 为规则 R 的正例约束数目,n_0 为规则 R 的反例约束数目,p_1 为规则 R' 的正例约束数目,n_1 为规则 R' 的反例约束数目,t 是在加入文字 L 到 R 后仍旧能覆盖的规则 R 的正例约束数目. 根据信息论的理论,$Fodrl\_gain(L,\ R)$ 的意义为:为了编码 R

的所有正例约束的分类所需的全部位数由于 L 带来的减少.特别地,如果 L 为文字集合,要对 L 中的文字进行修剪,移去对于信息增益无用的文字.

4.2 构造规则图

为了从初始领域理论推导出谓词之间的依赖关系,需要基于初始领域理论建立规则图.规则图是与或图,由两类结点组成:与结点(谓词结点)和或结点(规则结点),与结点是由基本谓词标记的叶子结点或者由派生谓词标记的内部结点,而或结点是由一阶规则标记的内部结点.对于一个规则结点来说,它的入边(只有一条)来自于由该规则所推导的派生谓词,而它的出边(若干条)则指向该规则的所有触发条件(由基本谓词或派生谓词组成).如果规则图中的结点是用谓词来标记的,这样的规则图称为一阶规则图.

由于 PDDL 语言允许派生谓词规则的条件部分包含逻辑运算和量词,而规则图中的规则条件是文字的合取,因此在建立规则图之前要对初始领域理论进行一些转换.由于规则条件是文字的合取,因此逻辑与运算和非运算不需要处理,而对于逻辑或运算,则将规则进行拆分,每个或条件对应一个新的规则.全称量词是隐式的,不需要处理,而存在量词是显式的,可以直接去掉而保留量词所约束的变量.

例 5. 对于表 1 中关于谓词 fed 的规则 $r5$,经过上述调整方法,拆分成如下 4 条只包含文字合取的规则:

$r5\_1$:(breaker $?x$) ∧ (ext $?l$ $?x$ side1) → (fed $?l$);

$r5\_2$:(breaker $?x$) ∧ (ext $?l$ $?x$ side2) → (fed $?l$);

$r5\_3$:(ext $?l$ $?x$ side1) ∧ (breaker $?y$) ∧ (upstream $?y$ $?sy$ $?x$ side2) → (fed $?l$);

$r5\_4$:(ext $?l$ $?x$ side2) ∧ (breaker $?y$) ∧ (upstream $?y$ $?sy$ $?x$ side1) → (fed $?l$).

根据派生谓词集合上的层次关系,高层的谓词定义依赖于低层的谓词定义.为了反映这些依赖关系,规则图中必须通过变量替换将各种规则衔接起来.例如,例 5 中规则 $r5\_4$ 的条件部分包含派生谓词(upstream $?y$ $?sy$ $?x$ side1),而表 1 的初始领域理论中只包含(upstream $?x$ $?sx$ $?y$ $?sy$)的规则,可以通过替换{$?x/?y$, $?sx/?sy$, $?y/?x$, $?sy/side1$}来进一步扩展规则图.如果初始理论包含的规则是递归的,这样的变量替换过程会导致规则图的无限增长.因此,需要限制规则图扩展的最大层次 L.

例 6. 由表 1 的初始领域理论可以得到如图 2 所示的规则图子图($L=2$).方框表示由基本谓词或者派生谓词标记的与结点,椭圆表示由规则编号标记的或结点.(fed $?l$)是该子图的根结点,层次为 0.推导出(fed $?l$)的规则条件中的各个谓词层次为 1,例如(upstream $?y$ $?sy$ $?x$ side1)的层次为 1.以此类推,推导出(upstream $?y$ $?sy$ $?x$ side1)的规则条件中的各个谓词层次为 2,例如(upstream $?y$ $?sy$ $?w$ side1)的层次为 2.

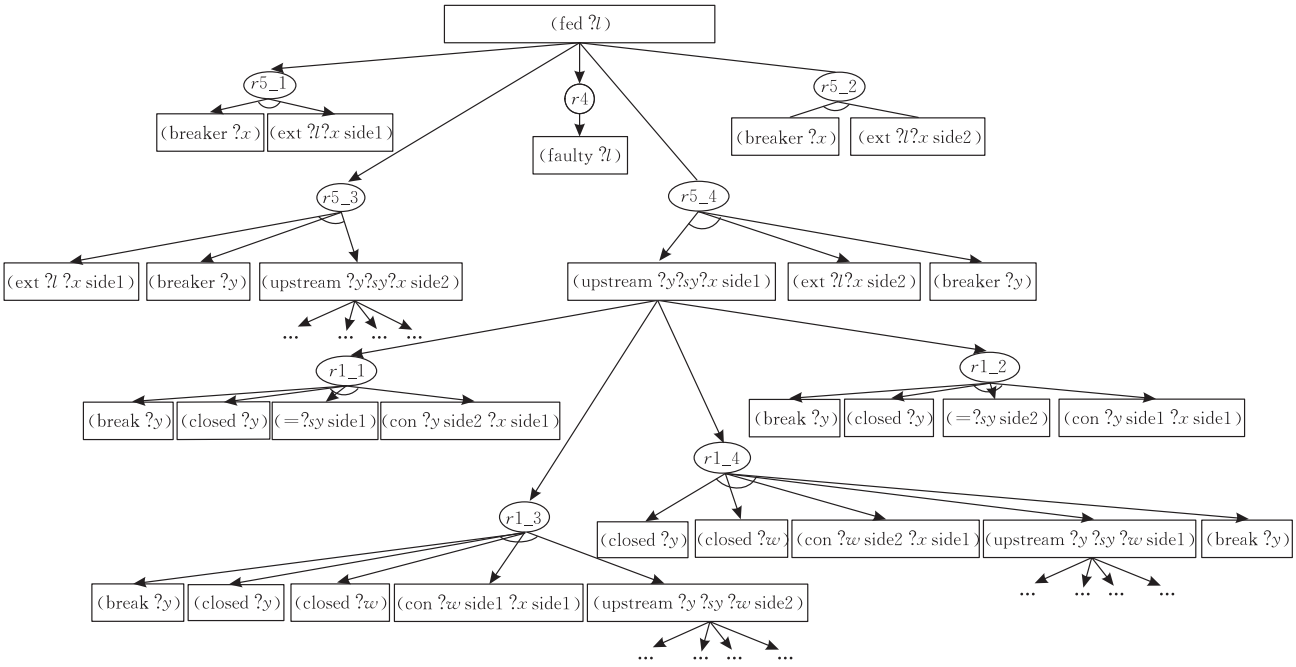


图 2 基于初始领域理论构造的规则图

4.3 计算激活集

激活集的定义来自于自然演绎推理,由 Gerevini 等[21]引入到规划系统 LPG-td 来求解派生规划问题.激活集描述的是在领域理论下能够推导出派生谓词的一组条件集合.然而,由 Gerevini 等定义的激活集存在着两种局限:(1)激活集的计算与当前状态相关,一旦状态发生变化,需重新计算激活集;(2)激活集仅限于命题集合.我们早期的研究工作定义了与状态无关的激活集,并且提出了一种高效的算法来计算激活集[22],以及在完美的领域理论下学习与激活集对应的命题规则等[23].

一个命题激活集可以定义如下[22].

定义 14. 设 d 是派生事实,d 的一个命题激活集 F 是满足 $F \models_R d$ 的由基本事实组成的最小集合.

由以上定义可知,对于一个派生事实,其激活集是一个由基本事实组成的命题集合,该集合可以通过领域理论推导出该派生事实.这样的集合要求是最小化的,即不包含任何与推导无关的其它命题.例如,通过变量代换$\{?l/l5, ?x/cb2\}$,根据规则 $r5\_2$,有{(breaker cb2), (closed $cb2$), (ext $l5$ $cb2$ side2)} $\models_R$ (fed $l5$),则 F = {(breaker $cb2$), (closed $cb2$), (ext $l5$ $cb2$ side2)}是派生事实(fed $l5$)的一个命题激活集.

为了在一阶规则的学习过程中用激活集来扩展搜索步,这里将命题激活集的定义扩展为一阶激活集.

定义 15. 设 d 是派生谓词,d 的一阶激活集 F 是满足 $F \models_R d$ 的由基本谓词组成的最小集合.

由定义 15 可知,一阶激活集是可以通过领域理论推导出某个派生谓词的基本谓词组成的集合.同样地,这样的集合也要求是最小的,即不包含任何与推导无关的其它谓词.把一阶激活集中的变量全部用常量来进行替换,可以得到对应派生事实的命题激活集.一阶激活集可以在一阶规则图上进行查找,而命题激活集需要将规则图用具体问题所包含的对象进行实例化才能进行查找.相对于实例化空间,层次受限的一阶规则图的搜索空间显然要小得多.在不正确或者不完备的领域理论下,可能会产生不正确的一阶激活集.这些激活集仍然可以用于扩展学习过程的搜索步,因为扩展的搜索步是一种多步探测,它的去留仅仅取决于对训练数据的拟合程度.因此,下面给出在可能包含错误的规则图上计算一阶激活集的 ASS 算法.一个派生谓词可能有多个不同的激活集,这些激活集构成了激活集集合 Σ.给定一个派生谓词和基于初始领域理论建立的规则图,该算法返回派生谓词的激活集集合 Σ.ASS 算法描述如下.

算法 3. ASS(Activation Set Searching)算法.

Procedure And-Search(n,A,$PathNodes$,$Open$)

输入:规则图 R 上的与结点 n,构造中的激活集 A,从搜索树的树根到结点 n 的搜索路径上的与结点集合 PathNodes,待访问的结点集合 Open

输出:A 中的元素,逻辑值 false 或空集

Begin

1. 如果 $n \in PathNodes$,则返回 false;
2. 否则,如果 n 是一个基本谓词,则返回 n;
3. 如果 n 没有任何后继规则结点,则返回 false;
4. 对于 n 在规则图 R 中的任何一个后继规则结点,调用
5. Or-Search(n',A,$PathNodes \cup \{n\}$,$Open$);
6. 返回空集.

End.

Procedure Or-Search(n,A,$PathNodes$,$Open$)

输入:规则图 R 上的或结点 n,构造中的激活集 A,从搜索树的树根到结点 n 的搜索路径上的与结点集合 PathNodes,待访问的结点集合 Open

副效果:更新激活集集合 Σ

Begin

1. Open←Open$\cup$$\{n' | n'$是 n 在规则图 R 上的后继谓词结点$\}$;
2. 对于 Open 表中所有元素 t
3. $Open \leftarrow Open \setminus \{t\}$;
4. $n' \leftarrow$And-Search(t,A,$PathNodes$,$Open$);
5. 如果 n'=false 则返回;
6. 否则 $A \leftarrow A \cup \{n'\}$;
7. $\Sigma \leftarrow \Sigma \cup \{A\}$.

End.

在规则图上计算某个派生谓词的激活集,是两个过程的交替搜索:与搜索过程和或搜索过程.在与搜索过程中,如果当前与结点是一个基本谓词,则放到正在构造的激活集 A 中.如果是一个派生谓词,则对它的每一个规则结点(或结点)调用或搜索过程.在或搜索过程中,对于规则结点的每一个触发条件(与结点)进行与搜索过程.为了避免无限循环,用 PathNodes 表来记录已经搜索过的派生谓词结点,若待扩展的结点是已经访问过的(与 PathNodes 表上某个谓词的名字和参数均相同),则停止对该结点的扩展.由于规则图的层次是受限的,因此规则图的规模是有限的,在遍历规则图中的所有结点之后,上述的搜索算法最终会停止.

例 7. 基于图 2 的规则图，应用 ASS 算法计算出 fed(?l)的一阶激活集集合 $\Sigma_{\text{fed}(?l)}$ 如下：

$$\begin{cases} F_1=\{(\text{faulty } ?l)\} \\ F_2=\{(\text{breaker } ?x),(\text{ext } ?l\ ?x\ \text{side1})\} \\ F_3=\{(\text{breaker } ?x),(\text{ext } ?l\ ?x\ \text{side2})\} \\ F_4=\{(\text{breaker } ?y),(\text{ext } ?l\ ?x\ \text{side2}), \\ \quad (\text{closed } ?y),(=?s y\ \text{side1}), \\ \quad (\text{con } ?y\ \text{side2}\ ?x\ \text{side1}),(\text{break } ?y)\} \\ F_5=\{(\text{breaker } ?y),(\text{ext } ?l\ ?x\ \text{side2}), \\ \quad (\text{closed } ?y),(=?s y\ \text{side2}), \\ \quad (\text{con } ?y\ \text{side1}\ ?x\ \text{side1}),(\text{break } ?y)\} \\ F_6=\{(\text{breaker } ?y),(\text{ext } ?l\ ?x\ \text{side1}), \\ \quad (\text{closed } ?y),(=?s y\ \text{side1}), \\ \quad (\text{con } ?y\ \text{side2}\ ?x\ \text{side2}),(\text{break } ?y)\} \\ F_7=\{(\text{breaker } ?y),(\text{ext } ?l\ ?x\ \text{side1}), \\ \quad (\text{closed } ?y),(=?s y\ \text{side2}), \\ \quad (\text{con } ?y\ \text{side1}\ ?x\ \text{side2}),(\text{break } ?y)\} \end{cases}$$

下面举例说明激活集对搜索步的扩展.

例 8. 学习派生谓词 fed(?l)的规则，该谓词在训练集中有 2 个正例和 6 个反例(见表 2). 用[$n+$, $n-$]来表示一个新规则所约束的正例和反例数目. 从最一般前件的规则开始，可选的候选式有如下文字(候选文字必须至少包含已有规则中的一个旧变量)及例 7 中的 7 个激活集：

(faulty ?l)；

not (faulty ?l)；

(ext ?l ?x ?s)；

not (ext ?l ?x ?s).

各个候选式的正例和反例约束数目如图 3 所示.

由图 3 可以看出，由 fed(?l)的激活集 F_4 扩展的搜索步(错误文字(break ?y)已被修剪)带来最大的信息增益，因此搜索方向沿着该搜索步进行，直至找出只覆盖正例的规则. 同样地，由 fed(?l)的激活集 F_3 扩展的搜索步带来的信息增益次之，当仍有正例未被学习到的规则覆盖时，搜索方向沿着该搜索步进行，直至找出只覆盖正例的规则. 最后，学习到的派生谓词规则如下：

①(breaker ?y)∧(closed ?y)∧(con ?y side2 ?x side1)∧(ext ?l ?x side2)∧(ext ?l ?w side1)∧(not (closed ?w))→fed(?l)；

②(breaker ?x)∧(close ?x)∧(ext ?l ?x side2)∧(ext ?l ?y side1)∧(con ?y side2 ?w side1)∧(not (closed ?w))→fed(?l).

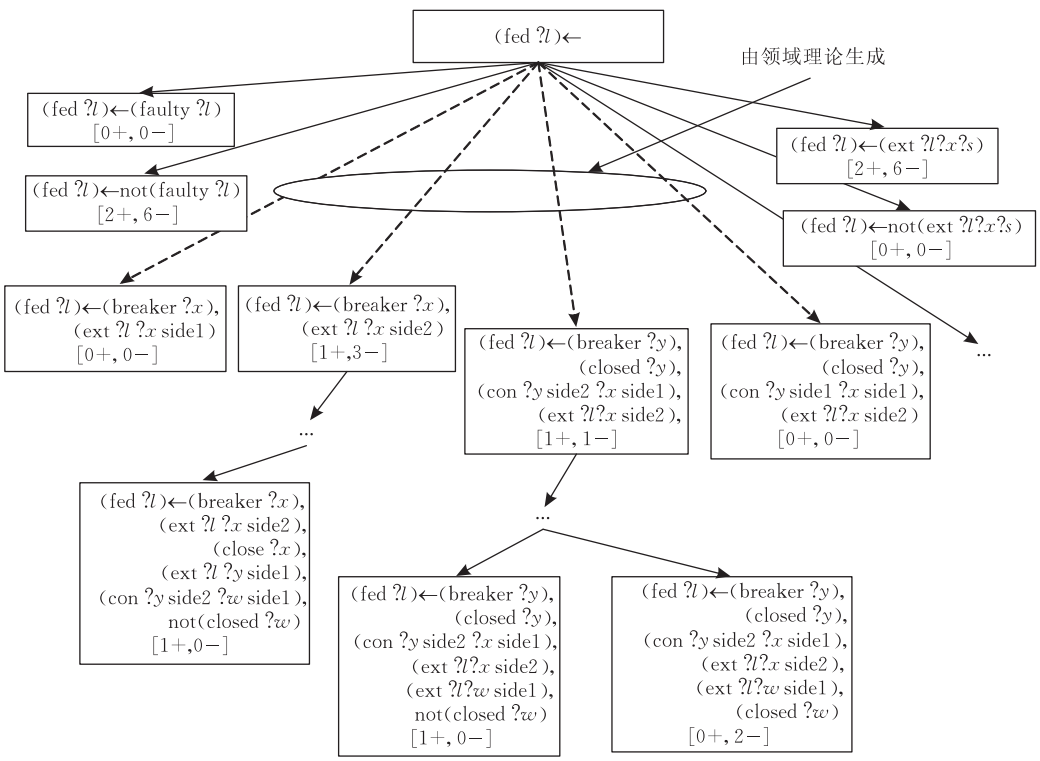


图 3 派生谓词的激活集扩展归纳学习的搜索步

由于规则①覆盖正例⟨$s3$,(fed $l6$)⟩，规则②覆盖正例⟨$s3$,(fed $l5$)⟩，且这两个规则均不覆盖任何反例，因此关于谓词 fed(?l)的学习过程结束. 但是可以看到，学习到的规则前件只包含基本谓词，使得

规则过于冗长.因此,更进一步的工作可以根据初始领域理论所包含的谓词依赖关系,引入中间派生谓词对学习到的规则集进行调整,使其包含可能的层次结构.

5　实验与分析

为了验证学习算法 FODRL 的有效性,基于开放源码的 FOIL 6.4 系统[①](C 语言编写),我们实现了一个同样称为 FODRL 的学习程序.该学习程序的输入是规划领域描述(DOMAIN.PDDL)、一组规划问题(PROBLEMS.PDDL)和对应的规划解(PLANS.txt),输出是学习到的规则集(RULES.txt).

本文的实验工作包括以下两个方面:(1)初始领域理论对于学习到的规则的精确度的影响;(2)四类领域理论错误分别对规则精确度的影响.测试数据来自于国际规划大赛 IPC'04 上的两个包含派生谓词的竞赛领域[②]:PSR(处理供电系统的重新配置问题)和 PROMELA(处理通信协议的错误检测问题).

5.1　初始领域理论对规则精确度的影响

为了考察初始领域理论对学习到的规则的精确度的影响,我们将 FODRL 与 FOIL、FOCL 进行比较.FOIL 是不能处理初始领域理论的纯归纳学习系统,而 FODRL 和 FOCL 都是能够接受初始领域理论的分析与归纳结合的学习系统,但是 FOCL 不能处理递归规则.由于两个测试领域都包含递归规则,因此这里 FOCL 只能作为纯归纳系统来学习规则.FODRL(FODRL 1.0 版本)和 FOIL(FOIL 6.4 版本)的实验环境均为 CPU(Pentium Processor 1.33GHz)+RAM(384MB)+FreeBSD 6.2+gcc 4.3.0,FOCL(FOCL-1-2-3 版本[③])的实验环境为 CPU(Intel celeron 2.6GHz)+RAM (1GB)+Window 2000+GNU Common Lisp -2.6.1.

要说明的是,FODRL 系统要将 GTE 算法产生的〈状态-派生事实〉序对形式的训练例转化为 FOIL 系统的关系元组形式的训练例.转化方法是:每个谓词(即关系)增加一个状态参数,每个谓词的实例(即在某个状态下成立的命题)增加一个状态标签.这样可以把在不同状态下成立的相同的谓词实例转化为关系元组形式的训练例.学习过程采用 k 次交叉-验证法,即训练样例分为 k 个子集,$k-1$ 个子集用于学习规则,1 个子集用于验证规则的精确度,总共进行 k 次.规则的精确度定义为在验证集上规则能够正确分类的训练例比例.具体地说,当一个正例被规则覆盖或者一个反例不被规则覆盖,均称为规则能够正确分类的情况.当一个反例被规则覆盖或者一个正例不被规则覆盖,则称为规则错误分类的情况.

对于表 2 所示的训练集,FODRL(包含表 1 的初始领域理论)、FOIL 及 FOCL 学习到的规则集如表 3 所示.在表 3 中,要说明的是,FOIL 没有输出派生谓词 affected 的规则,是因为所有候选式产生的规则在训练集上的精确度过低[④]而使得学习过程终止.另外,没有学习到 affected 的规则,但是 affected 又作为 fed 的条件出现,这是明显的不完备的领域理论.在 FODRL 学习到的规则中,{breaker(A,C), ext(A,B,C,D)}是谓词 fed 的一阶激活集所产生的候选式,可以看到初始领域理论对规则的学习过程的引导作用.如果初始领域理论的精确度不差的话,由它引导学习到的规则更具有一般性.最后,FOCL 学习到关于谓词 fed 的规则实际上包含错误,因为?0 和?3 是不同的状态变量.另外,FOCL 的两个异常现象是:(1)会存储学习到的针对某个领域的最佳规则,如果在新的训练集上学不到更好的规则,则会输出保存的最佳规则;(2)另外,对于本文的学习问题,即使增加的是非递归的初始领域理论,也学不到规则.可能是本文学习问题的初始领域理论导致 FOCL 的搜索空间迅猛增大而超出其限制,更准确的原因有待于进一步的代码分析.

表 3　FODRL、FOIL、FOCL 3 个系统学习到的规则集

系统	规则集
FODRL	affected(A,B):-closed(A,B). fed(A,B):-breaker(A,C),ext(A,B,C,D),closed(A,C).
FOIL	fed(A,B):-not(affected(A,C)), ext(A,B,C,D).
FOCL	(affected ?0 ?1):-(closed ?0 ?1). (fed ?0 ?1):-(closed ?0 ?2), (breaker ?3 ?2).

为了比较在不同的训练集上学习到的规则的精确度,我们选取 PSR-Middle-ADLDerived 版本下的 5 个规划问题(P01.PDDL～P05.PDDL)及其规划解(LPG-td 规划系统在 IPC'04 比赛中得到的规划

① http://www.rulequest.com/Personal/可下载 FOIL 的源代码.

② http://ls5-www.cs.uni-dortmund.de/～edelkamp/ipc-4/可下载 PSR 和 PROMELA 的领域描述.

③ ftp://ics.uci.edu/pub/machine-learning-programs/可下载 FOCL 源代码.

④ 候选式产生的规则在训练集中的精确度不能低于缺省值 80%,否则候选规则被抛弃.此时,FOIL 输出提示信息:"Clause too inaccurate".

解)来分别构造训练例.学习过程采用 3 次交叉-验证法.3 个系统学习到的规则精确度见图 4.

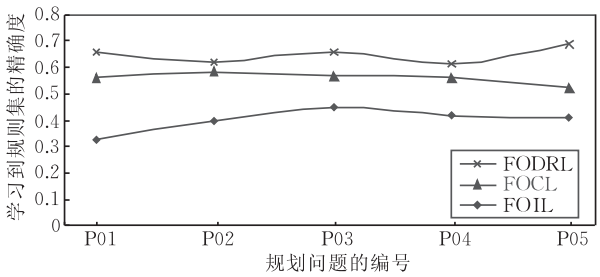


图 4 FODRL、FOIL、FOCL 学习到的规则精确度

在图 4 中,可以看到 3 个系统在不同的训练集上学习到的规则的精确度均有所差异.FODRL 学习到的规则的精确度最高,平均值在 0.6～0.7 之间.大部分规则条件中都包含激活集文字,较好地拟合了初始领域理论.将每组学习到的规则在要学习的谓词的所有训练例上进行验证,选择精确度最高的一组规则作为 PSR 领域的最后学习到的领域理论.FOCL 学习到的规则的精确度次之,平均值在 0.5～0.6 之间,但是很多规则中都包含上述所说的错误(状态参数不同),因而在语义上很难解释.FOIL 学习到的规则的精确度最低,平均值在 0.3～0.45 之间.在大部分测试组中都学不到派生谓词 affected 的规则,因此整体规则精确度降低了.

此外,由于 FODRL 和 FOCL 均接受初始领域理论,为了实验比较的全面性,我们在一个简单的包含非递归初始规则的实验领域上比较这两个混合学习算法的性能.该实验领域是学习谓词 member 的定义规则.目标谓词 member(A, B)表示元素 A 是表 B 的成员,可用的背景谓词是 component(C,D,E),表示表 C 是由表头元素 D 和表尾 E 组成的.为方便操作,训练例随机产生,训练集由相同数目的谓词的正例和反例组成,其中 member 的 1/4 的训练例做为验证集.构造 2 个不完美的初始领域理论如下:

(1) member(A,B):－components(B,A,C).

(2) member(A,B):－components(B,C,D).

这两个初始规则集均不包含递归规则.其中,第 1 个规则集是正确的,但是不完备的.例如,它不能解释 member(2, [1, 2])是正例.第 2 个规则集既不正确也不完备.例如,它把 member(3, [1, 2])错误地解释为正例.在这两个不完美的初始领域理论下,对不同的训练集规模,FODRL 和 FOCL 两个系统分别学习的效果见图 5.

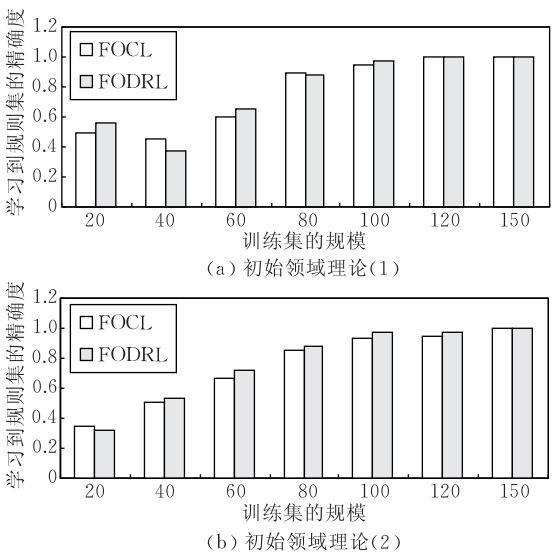


图 5 FOCL 和 FODRL 在非递归的初始领域理论下学习规则的性能比较

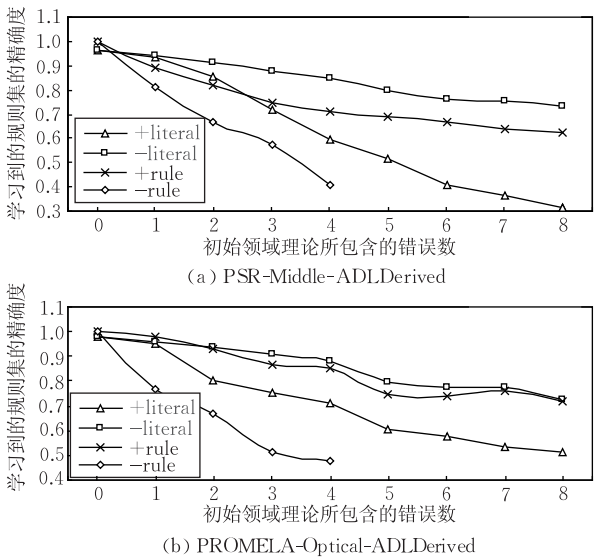
从图 5 的上/下图可以看到:首先,随着训练例的增多,两个系统的学习精度均逐步提高.在训练集规模达到 120 时,两个系统的学习精度都达到了 95%.其次,在学习目标谓词 member 的规则时,尽管预先定义了不同的初始领域理论,两个系统的学习性能大致相同,学习精度的差别不超过 10%.这表明 FOCL 和 FODRL 在简单的包含非递归初始规则的领域上的学习性能是相当的,这来自于它们均采用 FOIL 的基本学习算法以及对非递归规则进行替换的简单处理方法.

5.2 领域理论错误对规则精确度的影响

下面考察在给定训练集的情况下,分别包含四类错误的初始领域理论对 FODRL 学习规则的影响.从已发布的竞赛问题描述中构造不完美的领域理论,即产生各类错误."多余规则"类错误(简记为＋rule)的产生可以从谓词集中任意选取多个关联文字的合取来构成某个派生谓词的规则前件.直接去掉某个原有规则可以产生"缺少规则"类错误(简记为－rule).从谓词集中选取某个不在规则条件中的谓词加入到规则条件中产生"多余的规则条件"类错误(简记为＋literal).从规则中直接删除某个原有的规则条件产生"缺少的规则条件"类错误(简记为－literal).

如前,FODRL 的实验环境为 CPU (Pentium Processor 1.33GHz)＋RAM (384M)＋FreeBSD 6.2＋gcc 4.3.0.分别选取 PSR-Middle-ADL-Derived 领域的一个规划问题(P01.PDDL)和 PROMELA-Optical-ADLDerived 领域的一个规划

问题(P01_OPT2.PDDL)及各自对应的规划解来产生训练集.学习过程采用训练-验证法,2/3 的训练例用于学习规则,1/3 的训练例用于验证规则的精确度.在构造领域错误时,对给定的某类错误的数目,分别生成 3 组测试数据,每组测试数据是随机构成的包含该类错误数目的初始领域理论.例如,在生成包含 2 个-literal 类错误时,通过 3 次随机删除规则条件部分的两个文字而得到 3 组测试数据.3 组测试数据上学习到的规则精确度的平均值作为包含该类错误数目的平均规则精确度.FODRL 的学习结果见图 6.



(a) PSR-Middle-ADLDerived

(b) PROMELA-Optical-ADLDerived

图 6　领域理论错误对规则精确度的影响

因为完整的 PSR 和 PROMELA 领域描述都包含 4 条派生谓词规则,因此−rule 类错误的最大错误数为 4.从图 6 中可以看到,在两个领域,−literal 类错误对规则精确度的影响最小,是最容易容忍的领域错误,+rule 类错误其次.这是因为缺少的规则条件可以通过归纳学习过程来进行补足,多余的规则不会带来较大的信息增益,也容易被候选式选择函数所抛弃.相比较而言,在 PSR 领域,+literal 类错误是最不能容忍的错误,这类错误极大地降低了规则精确度.这是因为在派生谓词的激活集用于候选式的时候,要对其中的每个文字进行剪枝(如果去除该文字能带来更大的信息增益则该文字应该被剪枝),但某些无用的文字并不减少信息增益而被保留.当这些文字不能拟合验证集时,则降低了规则精确度.在 PROMELA 领域,−rule 类错误是最不能容忍的错误,这是因为要学习规则的派生谓词只有 1 个(PSR 领域是 2 个),中间谓词相对较多且谓词之间依赖关系比较复杂,因而减少其中的初始规则对于学习到的规则的精确度影响较大.

6　结　语

学习和规划结合是智能规划研究领域中的热点问题.与以往学习动作模型、学习控制知识等研究工作不同,本文是最早的学习派生谓词规则的研究工作.为了自动获取派生规划问题的领域理论,本文提出了从规划解中学习一阶派生谓词规则的 FODRL 算法.学习过程分为两个步骤:首先根据派生谓词的语义从规划解中提取学习过程所需要的训练例,然后在初始领域理论的引导下学习能够最好拟合训练例的一阶规则集合.

本文的主要创新点在于结合分析学习和归纳学习各自的优点来学习派生谓词规则.首先,规划解不能够直接作为学习问题的训练例,需要结合动作模型和规划问题描述来从规划解中提取谓词的训练例集合.其次,提出了一阶规则图和一阶激活集的概念,并且给出在层次受限的一阶规则图上计算派生谓词的一阶激活集的算法.激活集用于扩展搜索步,使得一次可以试探多个文字.接着,实现了学习系统 FODRL,实验结果表明,在初始领域理论的引导下 FODRL 学习到的规则精确度比纯归纳学习系统 FOIL 及组合学习系统 FOCL 的更高.最后,讨论了各类领域理论错误对学习到的规则精确度的影响.

本文迈出了学习派生谓词规则的第一步,然而在提出的方法中还有一些不足需要在未来的工作中克服.首先,学习到的规则前件中只包含基本谓词,需要制定相关的策略调整规则结构,使其包含中间派生谓词且增加层次结构.其次,在现实的规划执行的环境中存在着各种不稳定的因素,因此学习概率性的规则是很有意义的.

参　考　文　献

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

[2] Fikes R, Nilsson N. STRIPS: A new approach to the application of theorem proving to problem solving. Artificial Intelligence, 1971, 2(3/4): 189-208

[3] Thielscher M. Ramification and causality. Artificial Intelligence, 1997, 89(1/2): 317-364

[4] Yang Qing, Wu Kang-Hen, Jiang Yun-Fei. Learning action models from plan examples using weighted MAX-SAT. Artificial Intelligence, 2007, 171(2-3): 107-143

[5] Amir E. Learning partially observable deterministic action models//Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2005). Edinburgh, Scotland, UK, 2005: 1433-1439

[6] Ilghami O, Munoz-Avila H, Nau D S, Aha D W. Learning preconditions for planning from plan traces and HTN structure. Journal of Artificial Intelligence Research, 2005, 21(4): 388-413

[7] Aler R, Borrajo D, Isasi P. Using genetic programming to learn and improve control knowledge. Artificial Intelligence, 2002, 141(1/2): 29-56

[8] Huang Y C, Selman B, Kautz H A. Learning declarative control rules for constraint-based planning//Proceedings of International Conference on Machine Learning. Stanford, CA, USA, Morgan Kaufmann, 2000: 415-422

[9] Bianchi R A C, Ribeiro C H C, Costa A H R. Heuristic selection of actions in multiagent reinforcement learning//Proceedings of the IJCAI 2007. Hyderabad, 2007: 690-695

[10] Khardon R. Learning action strategies for planning domains. Artificial Intelligence, 1999, (113): 125-148

[11] De Raedt L, Lavrac N, Dzeroski S. Multiple predicate learning//Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93). Chambery, France, 1993: 1037-1042

[12] Baroglio C, Botta M. Multiple predicate learning with RTL//Proceedings of the 4th Congress of the Italian Association for Artificial Intelligence AI*IA'95. LNAI-992. Florence, Italy: Springer, 1995: 44-55

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

[14] DeJong G. Explanation-based learning//Tucker A ed. The Computer Science and Engineering Handbook. Boca Raton, FL: CRC Press, 1997: 499-520

[15] Pazzani M, Kibler D. The utility of knowledge in inductive learning. Machine Learning, 1992, 9(1): 57-94

[16] Zettlemoyer L, Pasula H, Kaelbling L. Learning planning rules in noisy stochastic worlds//Proceedings of Workshop on Planning and Learning in A Priori Unknown or Dynamic Domains of IJCAI05. Edinburgh, Scotland, 2005: 1-8

[17] McDermott D. PDDL—The planning domain definition language. Yale Center for Computational Vision and Control: Technical Report CVC TR-98-003/DCS TR-1165, 1998

[18] Edelkamp S, Hoffmann J. PDDL2.2: The language for the classical part of the 4th international planning competition. Freiburg, Germany: Technical Report 195, 2004

[19] Apt K, Blair H, Walker A. Towards a theory of declarative knowledge//Foundations of Deductive Databases and Logic Programming. San Mateo, CA: Morgan Kaufmann, 1988: 89-148

[20] Clark P, Matwin S. Learning domain theories using abstract background knowledge//Proceedings of ECML. Viennia, Austria: Springer, 1993: 360-365

[21] Gerevini A, Saetti A, Serina I, Toninelli P. Fast planning in domains with derived Predicates: An approach based on ruleaction graphs and local search//Proceedings of the 20th National Conference on Artificial Intelligence (AAAI'05). Pittsburgh, Pennsylvania, 2005: 1157-1162

[22] Jiang Zhi-Hua, Jiang Yun-Fei. An improved method for calculating activation sets of action derived preconditions. Chinese Journal of Computers, 2007, 30(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)

[23] Rao Dong-Ning, Jiang Zhi-Hua, Jiang Yun-Fei. Learning activation rules for derived predicates from plan examples//Proceedings of Workshop on Artificial Intelligence Planning and Learning of ICAPS07. Providence, Rhode Island, USA, 2007

附录 A: DerivedADL 版本的 PSR 领域描述

```
(define (domain psr)
  (:requirements :adl :derived-predicates)
  (:types DEVICE SIDE LINE)
  (:constants side1 side2-SIDE earth-DEVICE)
  (:predicates (ext ?l-LINE ?x-DEVICE ?s-SIDE)
               (breaker ?x-DEVICE)
               (closed ?x-DEVICE)
               (faulty ?l-LINE)
               (con ?x-DEVICE ?sx-SIDE ?y-DEVICE ?sy
                    -SIDE)
               (upstream ?x -DEVICE ?sx -SIDE ?y -DE-
                    VICE ?sy-SIDE)
               (unsafe ?x-DEVICE ?sx-SIDE)
               (affected ?x-DEVICE)
               (fed ?l-LINE))

  (:derived (upstream ?x-DEVICE ?sx-SIDE ?y-DEVICE
            ?sy-SIDE)
    (and (closed ?x)
         (or (and (=?sx side1) (con ?x side2 ?y ?sy))
             (and (=?sx side2) (con ?x side1 ?y ?sy))
             (exists (?z-DEVICE)
                (and (closed ?z)
                     (or (and (con ?z side1 ?y ?sy)
                              (upstream ?x ?sx ?z side2))
                         (and (con ?z side2 ?y ?sy)
                              (upstream ?x ?sx ?z side1)))))))))

  (:derived (unsafe ?x-DEVICE ?sx-SIDE)
    (and (closed ?x)
      (or (and (=?sx side1)
               (exists (?l-LINE)
                 (and (ext ?l ?x side2)
```

```
            (or (faulty ?l)
                (exists (?y-DEVICE)
                  (exists (?sy-SIDE)
                    (and (con ?x side2 ?y ?sy)
                         (unsafe ?y ?sy)))))))

        (and (=?sx side2)
          (exists (?l-LINE)
            (and (ext ?l ?x side1)
              (or (faulty ?l)
                  (exists (?y-DEVICE)
                    (exists (?sy-SIDE)
                      (and (con ?x side1 ?y ?sy)
                        (unsafe ?y ?sy))))))))))

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

(:derived (fed ?l-LINE)
  (exists (?x-DEVICE)
    (and (closed ?x)
      (or (and (ext ?l ?x side1)
               (or (breaker ?x)
                   (exists (?y-DEVICE)
                   (exists (?sy-SIDE)
                   (and (breaker ?y)
                     (upstream ?y ?sy ?x side2))))))
               (and (ext ?l ?x side2)
                 (or (breaker ?x)
                  (exists (?y-DEVICE)
                    (exists (?sy-SIDE)
                      (and (breaker ?y)
                        (upstream ?y ?sy ?x side1))))))))))

(:action open
   :parameters (?x-DEVICE)
   :precondition (and (not (=?x earth))
                      (closed ?x)
                        (forall (?b-DEVICE)
                              (and (breaker ?b)
                                   (not (affected ?b))))))
   :effect (not (closed ?x)))

(:action close
  :parameters (?x-DEVICE)
  :precondition (and (not (=?x earth))
                     (not (closed ?x))
                          (forall (?b-DEVICE)
                                (and (breaker ?b)
                                     (not (affected ?b)))))
  :effect (closed ?x))

(:action wait
   :parameters ()
   :precondition (exist (?b-DEVICE)
                        (and (breaker ?b)
                             (affected ?b)))
   :effect (forall (?b-DEVICE)(when (affected ?b)(not
          (closed ?b)))))
)
```

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

JIANG Zhi-Hua, born in 1978, Ph. D.. Her main research interests include AI planning and knowledge reasoning.

JIANG Yun-Fei, born in 1945, professor, Ph. D. supervisor, membership of China Computer Federation. His main research interests include AI planning, knowledge reasoning, and model based diagnosis.

LIU Qiang, born in 1978, Ph. D.. His main research interests focus on AI planning.

Background

The combination of planning and learning is a hot topic in the field of Artificial Intelligence. In last two decades, researchers have proposed various practical technologies for learning action models, control knowledge, heuristic functions or action strategies in automated planning. Unlike all previous work, this paper is the first attempt to learn derived predicate rules from plan examples.

Derived predicates is a natural way to depict indirect effects of domain actions, and their truth values in the current state are inferred from that of other predicates via domain rules. These rules are action-independent so that the description of a planning domain becomes more concise and clearer. However, human-designed domain rules cannot be guaranteed to be correct or complete. It is difficult to explain why an observed plan is valid under imperfect domain rules. So learning derived predicate rules from plan examples is interesting and promising.

Since 1990s, many methods have been proposed to learn first-order rules from training examples, such as FOIL (First-Order Inductive Learning), ILP (Inductive Learning

Programming), EBL (Explanation Based Learning), FOCL (First-Order Combined Learning) and so on. FOIL is a pure inductive learning algorithm, which learns a new rule which covers partial positive examples but avoids all negative examples once a time, until all positive examples are covered. ILP is a well-studied territory for learning a logic program composed of Horn clauses based on FOIL. EBL is a pure analytical learning algorithm. With EBL, a general hypothesis is inferred from prior knowledge to explain training examples. FOCL is a combined learning algorithm, which uses imperfect prior knowledge to expand search steps of FOIL. However, FOCL cannot deal with initial domain theories that contain recursive rules. To take advantages of both and overcome shortcomings like FOCL's, we propose a new combined learning algorithm called FODRL (First-Order Derived Rules Learning) to automatically discover first-order rules for derived predicates from observed plans under an initial domain theory. With natural deductive reasoning, an activation set is a minimal set of basic facts or predicates which can make a derived predicate hold true under domain rules. FODRL uses activation sets of derived predicates to expand search steps so as to extremely change the search direction. To do this, an algorithm called ASS is presented to search activation sets in the rule graph built on the initial domain theory. Finally, experiment results show that with the guidance of an initial domain theory, the rules learned by FODRL is more accurate than those from FOIL or FOCL.

The research work here is supported by the National Nature Science Foundation (60173039) and the Guangdong University of Technology Project Foundation (093032), and the project is aimed at solving all sorts of planning problems. The research team has gained a fruitful achievement in some research lines, such as non-deterministic planning (2 papers in IJCAI'07), temporal planning (1 paper in TIME'08), planning and learning (1 paper in AI journal), and so on. One of main trends of this field is to integrate multiple techniques together, such as searching and reasoning, and derived planning problems is an example of this trend. The research work here deals with derived planning problems. Our early work includes: 1) define a new concept, that is, proposition activation sets for a derived fact; 2) present an efficient algorithm to calculate such activation sets in a rule graph. Here, in order to learn first-order rules, all early works in proposition logic are expanded into first-order logic. Besides, we has proposed a method to learn proposition rules for derived predicates according to the conversion principle of activation sets in training examples. Here rules learned are expanded into first-order, too. The best advantages of doing so is that proposition rules are for a specific planning problem, while first-order ones are for a general planning domain. Besides, learning methods are greatly improved by using information gains to evaluate candidates of rule antecedents. To sum up, the research work here is a natural expansion of early works, but it is more creative and important.