ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.6, June 2007, pp.1328−1349
DOI: 10.1360/jos181328

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

STRIPS 规划领域中动作效果关系的研究*

吴向军[1]，姜云飞[2]，凌应标[3+]

[1](中山大学 软件学院,广东 广州 510275)

[2](中山大学 信息科学与技术学院 软件研究所,广东 广州 510275)

[3](中山大学 信息科学与技术学院 计算机科学系,广东 广州 510275)

Research on Relations of Effect of Action for STRIPS Domain

WU Xiang-Jun[1], JIANG Yun-Fei[2], LING Ying-Biao[3+]

[1](Software School, SUN YAT-SEN University, Guangzhou 510275, China)

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

[3](Department of Computer Science, School of Information Science and Technology, Sun Yat-Sen University, Guangzhou 510275, China)

+ Corresponding author: Phn: +86-20-84036776, E-mail: isslyb@mail.sysu.edu.cn

Wu XJ, Jiang YF, Ling YB. Research on relations of effect of action for STRIPS domain. *Journal of Software*, 2007,18(6):1328−1349. http://www.jos.org.cn/1000-9825/18/1328.htm

Abstract: This paper anatomizes all actions in planning domain, proposes a descriptive method for the relations between preconditions and effects of action, and defines some basic relations between preconditions and effects of action, such as direct concomitant relation, conditional concomitant relation, and direct obstructive relation. These basic relations express the domain knowledge hiding in the definition of actions. For these basic relations, this paper defines some compound operations, and obtains indirect obstructive relations and absolute obstructive relations. The indirect obstructive relation shows the transitivity of preconditions of action, and the absolute obstructive relation expresses the impact on other predications when one predication is realized by action. Finally, this paper gives some applications of these relations in the planning system. These relations are the essential theory basis for sorting predications in target state, judging the target state, and optimizing the strategy of action selection for one predication.

Key words: AI planning; planing domain; direct concomitant relation; conditional concomitant relation; direct obstructive relation; indirect obstructive relation; absolute obstructive relation

摘 要: 以规划领域中的动作为研究对象,提出了描述动作前提条件和效果之间关系的方法,定义了动作前提和效果之间的基本关系:直接伴随关系、条件伴随关系和直接阻碍关系等,这些基本关系反映了规划动作中所隐含的领域知识.对动作效果的基本关系,定义了进行关系组合的运算,产生出间接阻碍关系和绝对阻碍关系.间接阻碍关系反映出动作前提条件的传递性;绝对阻碍关系表达出实现一个谓词对其他谓词实现的影响.最后给出动作效果关系在规划求解过程中的具体运用,这些动作效果关系为目标实现顺序的排序、目标状态可解性的判

* Supported by the National Research Foundation for Doctoral Program of Ministry of Education of China (国家教育部博士点基金); the Special Foundation of SUN YAT-SET University of China (中山大学专项资金)

Received 2005-11-27; Accepted 2006-04-03

定以及动作选择策略的优化等提供了必要的理论依据.
关键词: 智能规划;规划领域;直接伴随关系;条件伴随关系;直接阻碍关系;间接阻碍关系;绝对阻碍关系
中图法分类号: TP18 文献标识码: A

智能规划是人工智能研究的一个重要分支,其所要解决的问题是:在给定的已知条件(初始状态)下,如何通过产生式规则(领域动作)获得所需要的结论(目标状态).在智能规划的早期研究[1]中,用蕴含式描述动作对状态谓词的影响,用框架公理描述状态谓词不受影响的传递关系.这样,就把规划问题转化为定理证明问题,并用归结演绎的定理证明方法来进行问题求解.虽然这种方法可以完成一些问题的求解,但大量的框架公理降低了其问题求解的效率,使得这种方法在具体的运用中受到很大的限制.

为了提高问题求解的效率,Fikes 等人提出了一种称为 STRIPS 语言的动作描述方法.该方法改进了问题的描述手段,省去了框架公理,并在该描述的基础上提出了新的求解策略.在求解过程中,把整个求解目标分解成若干个子目标.在实现子目标时,利用启发性信息来提高搜索效率.此后,研究者设计出多种不同的智能规划方法,其中之一就是基于规划问题的状态空间搜索方法.在状态空间搜索中,有前向搜索、反向搜索和双向搜索等经典的搜索策略.这些搜索策略在具体运用时,其效率会因状态空间的组合爆炸而变得十分低下.因此,在搜索方法中添加启发式搜索信息就是一个新的研究方向[2−8].在启发式信息的研究中,研究者发现:启发式信息在问题求解中所起到的作用会因问题的不同而有很大差异,构造出适用于所有问题求解的启发式函数是不可能的.所以,启发式搜索虽然在一定程度上能够改善问题求解的效率,但其时间复杂度仍是指数的.

在近期的智能规划研究中,主要研究成果是基于规划空间的搜索方法.该方法把问题空间划分成若干个抽象层次,先在最上面的层次作最一般的规划,然后利用逐步求精的方法对上层规划中的每一步进行低层次的规划并实现了相应的规划器[9−11].分层规划在每层规划求解时可避免产生庞大的搜索树,从而提高问题求解的总体效率,但部分序规划的逐步求精过程仍然是 NP 困难的,且在生成状态空间时,更难构造出有效的启发式信息.

近 10 年来,基于 STRIPS 描述的智能规划研究有了较大进展,其主要表现在图规划(GraphPlan)思想的提出和实现上[12].图规划构造了一个非常有效的规划图搜索空间,它先从规划图中产生出可选的动作集序列,然后再根据动作之间的互斥性进行分析和逆向搜索.该方法将规划的隐式约束作用于规划图的生成和互斥关系传播过程之中.基于规划图的方法使规划效率有了极大的改善,它比以前的 SNLP[13],Prodigy[14,15]和 UCPOP[16]等快了几个数量级.以往规划器在几分钟时间内只能产生出有 5~6 个动作的规划,而采用图规划方法的规划器在同样时间内可以获得 100 多个动作的规划.规划图还是 SAT 方法[17−19]和 GP-CSP[20−22]等方法的基础.

规划图是一种具有特殊结构的有向图,图中有谓词结点和动作结点,它们按层交替出现.在规划图的构造过程中,利用领域的动作定义和谓词逻辑的基本规则确定动作和动作、谓词和谓词之间的互斥关系,并用这些互斥关系对规划图进行优化,从而提高规划图的生成效率.图规划方法的主要特点就是在规划图中直观地表现出具体动作和动作、谓词和谓词之间的互斥关系,但该方法不能从这些具体的互斥关系中提取出规划领域的一般性规律,同样,两个动作(或谓词)在规划图的不同位置需重复确定其互斥关系,这不仅延长了规划图的生成时间,而且也使规划图中存在大量重复(或相似)的互斥关系.由文献[20−22]可知,即使对一些小规模的规划问题,把规划图转化成约束可满足性规划问题(CSP),也需要几兆~几百兆的编码空间.

在智能规划的研究策略方面,我们与图规划的思想有着较大的差异.在规划问题求解时,我们首先分析规划领域的动作定义,从中提取出能够反映该领域内在规律的领域知识,然后再利用这些领域知识来指导具体的规划问题求解.由于提取出来的领域知识仅与规划领域的定义有关,与规划领域的具体规划问题无关,所以,这些领域知识对该领域的所有规划问题都是有效的.这样,在同一规划领域不同规划问题求解时,可避免领域知识(如图规划中的两类互斥关系)的重新确定.这种研究策略是一种新的规划求解思路,所提取的领域知识是以一种规则的形式来表示的,其语义简洁、明了.这些都是我们在智能规划的研究方面所做的新的探索.

本文从最基本的动作定义来展开对智能规划的研究.第 1 节简单介绍 PDDL(planning domain definition language)语言定义规划动作的形式,说明动作定义的主要信息及其执行动作的含义.第 2 节提出动作效果之间

的伴随关系,分析直接伴随关系与普通逻辑的蕴涵关系之间的差异.第 3 节给出动作效果和前提条件之间的直接阻碍关系,并用多个直接阻碍关系产生出间接阻碍关系.第 4 节讨论用直接伴随关系和阻碍关系组合产生出绝对阻碍关系,证明动作效果关系对关系组合运算是封闭的.第 5 节介绍动作效果关系在规划求解中的具体运用.

1 规划领域的动作描述

智能规划研究的问题是:假设有一个规划领域的“初始状态”和“目标状态”,求规划领域的一个动作序列.通过该动作序列,可把“初始状态”转变成“目标状态”.智能规划的研究目标是求出满足要求的最短动作序列.所以,规划领域的动作在智能规划的研究中具有极其重要的作用.

本文研究的规划领域是由 PDDL 语言所描述的[23−26].PDDL 描述语言是目前用于描述规划领域的标准语言,也是国际规划大赛 IPC(international planning competition)指定的规划领域描述语言.PDDL 描述动作的一般形式如下:

(:Action *Act*
 :Parameters (P_1 P_2 … P_k)
 :Precondition (Pc_1 Pc_2 … Pc_s)
 :Effect (and Pd_1 Pd_2 … Pd_t)
)

其中:*Act* 是动作名;Pc_i 和 Pd_j 是领域中的谓词,i=1..s,j=1..t.

该动作描述的含义是:在任意规划状态下,若存在动作参数表的一个实例化置换σ,使得前提条件 $Pc_1\sigma$,$Pc_2\sigma$,…,$Pc_s\sigma$都在该规划状态中,则在该规划状态下可执行动作 *Act*,并用其动作效果谓词 $Pd_1\sigma$,$Pd_2\sigma$,…,$Pd_t\sigma$来修改当前状态,使之成为“下一状态”.

例如:在经典的规划领域 BlocksWorld 中,其动作 Unstack 的描述如下*:

(:Action Unstack
 :Parameters (?*x* ?*y*)
 :Precondition ((clear ?*x*) (on ?*x* ?*y*) (arm-empty))
 :Effect (and (not (clear ?*x*)) (not (on ?*x* ?*y*)) (not (arm-empty)) (holding ?*x*) (clear ?*y*))
)

由规划动作的描述不难看出,动作描述给出的是动作功能的一般性说明,是动作模式的一种描述,它不是一个具体的动作.只有当动作所有参数都被实例化后,该动作描述才对应一个具体的动作,如:(Unstack *A*,*B*)和(Unstack *B C*)就是动作 Unstack 的两个不同的具体动作,其中:*A*,*B* 和 *C* 是具体的积木块.

智能规划求解是求出实现“目标状态”的一系列具体的规划动作,这些动作的每个参数都是实例化的.

定义 **1**.**1**. 假设 *Pd* 是一个谓词,*S* 是一个谓词集合,σ是一个置换,

(1) $Pd\sigma$表示谓词 *Pd* 的参数表用置换σ进行变换后的结果;

(2) $S\sigma$是谓词集 *S* 中每个谓词用置换σ进行变换后所组成的集合,即:$S\sigma=\{Pd_i\sigma|Pd_i\in S\}$.

例如:*Pd*=(on ?*x* ?*y*),σ={*A*/?*x*,*B*/?*y*},那么,$Pd\sigma$=(on *A B*).

为了方便叙述规划的信息,本文所用符号和术语的含义说明如下:

1) 用 *PC*(*Act*)和 *E*(*Act*)表示规划动作 *Act* 的前提条件(precondition)和动作效果(effect)的谓词集.

对于上面动作 Unstack 的描述,有

PC(Unstack)={(clear ?*x*),(on ?*x* ?*y*),(arm-empty)}

---

* BlocksWorld 领域的详细描述信息见附录 B.本文是以 BlocksWorld 领域为例来说明有关概念,但所讨论的内容和所得出的结论,对基于 STRIPS 的规划领域都有效,并不局限于 BlocksWorld 领域.

E(Unstack)={(not (clear ?x)),(not (on ?x ?y)),(not (arm-empty)),(holding ?x),(clear ?y)}

2) 用规划状态中所有谓词的集合来代表该规划状态,并用符号 T_i 表示第 i 个规划状态,其中,T_0 表示规划问题的"初始状态".

例如:附录 C 是 BlocksWorld 规划领域中著名的规划奇异问题,其初始状态 T_0 如下:

$$T_0=\{(\text{clear } C),(\text{on } C\ A),(\text{on-table } A),(\text{clear } B),(\text{on-table } B),(\text{arm-empty})\}$$

3) 在规划状态 T_i 下执行动作 Act 是指,在该状态下,存在动作参数的一个实例化置换 σ,使得:$PC(Act)\sigma\subseteq T_i$,则用效果谓词集 $E(Act)\sigma$ 来修改状态 T_i 中的谓词集,使之变成状态 T_{i+1}.状态变化的规则为

$$T_{i+1}=(T_i-\{Pd_u\sigma|(\text{not } Pd_u)\in E(Act)\})\cup\{Pd_v\sigma|Pd_v\in E(Act)\}$$

例如:在附录 C 的"初始状态"T_0 下,存在一个实例化置换 $\sigma=\{C/?x,A/?y\}$),使得动作(Unstack C A)的所有前提条件在 T_0 中.执行该动作后,下一状态 T_1 为

$$T_1=(T_0-\{(\text{clear } C),(\text{on } C\ A),(\text{arm-empty})\})\cup\{(\text{holding } C),(\text{clear } A)\}$$
$$=\{(\text{on-table } A),(\text{clear } B),(\text{on-table } B),(\text{holding } C),(\text{clear } A)\}$$

由此可见,规划状态的改变是通过规划动作的执行来实现的.在通常情况下,执行规划动作是为了在规划状态中获得某个特定的领域谓词,但对某个特定的谓词往往可由多个动作来实现,并且在用动作实现时还会带来一些额外的效果,所以,用规划动作实现某个特定谓词时所可能带来的额外效果就值得深入研究.

定义 **1.2**. 假设当前规划状态为 T_i,谓词 Pd 中的所有参数都已实例化,若 $Pd\in T_i$,则称在当前状态下,谓词 Pd 为"真",谓词(not Pd)为"假";否则,在当前状态下,称谓词 Pd 为"假",谓词(not Pd)为"真".

定义 **1.3**. 假设当前规划状态为 T_i,谓词 Pd 的参数表中存在未实例化的参数,若存在一个实例化置换 σ,使得 $Pd\sigma\in T_i$,则称在当前状态下,谓词 Pd 为"真",谓词(not Pd)为"假";否则,称在当前状态下,谓词 Pd 为"假",谓词(not Pd)为"真".

由定义 1.2 和定义 1.3 可知,谓词的"真/假"与具体的规划状态有关.本文在不强调具体规划状态时,把"在当前状态下,谓词 Pd 为'真/假'"简写成"谓词 Pd 为'真/假'".

对于一个谓词集 pc,若 $\forall Pd_i\in pc$,都有 Pd_i 为"真",则称谓词集 pc 为"真";否则,称谓词集 pc 为"假".

2 动作效果的伴随关系

在分析规划领域的动作定义时我们发现,领域设计者在动作描述的前提条件和动作效果之间,隐含地表达了领域谓词之间的某种因果关系.本文把实现某个谓词所有动作的公共前提和公共效果之间的各种关系统称为动作效果关系,并用这种动作效果关系来表达规划领域谓词之间的内在规律.

下面,以规划领域中的动作模式为基础来介绍动作效果的伴随关系及其逻辑推导规则.

2.1 动作效果的直接伴随关系

对任意一个规划领域,用 ACT 表示该规划领域中所有动作的集合,$Act(Pd)$表示所有可实现谓词 Pd 的动作集,$CPC(Pd)$是所有实现谓词 Pd 动作的公共前提条件(common precondition),$CE(Pd)$是所有实现谓词 Pd 动作的公共效果(common effect),即

$$Act(Pd)=\{Act_i|Pd\in E(Act_i),Act_i\in ACT\},\ CPC(Pd)=\cap_{Act_i\in Act(Pd)}PC(Act_i),\ CE(Pd)=\cap_{Act_j\in Act(Pd)}E(Act_j)$$

显然,$Pd\in CE(Pd)$.

我们利用实现谓词 Pd 动作的公共效果 $CE(Pd)$来定义动作效果的直接伴随关系.

定义 **2.1**. 动作效果的伴随关系 $\xrightarrow{pc}$:

$Pd_1\xrightarrow{pc}Pd_2$ 的含义是:$\forall Act_i\in ACT$,若 $Pd_1\in E(Act_i)\Rightarrow Pd_2\in E(Act_i)$,则在用动作实现谓词 Pd_1 时,谓词 Pd_2 也一定被实现,称谓词 Pd_2 是谓词 Pd_1 的伴随结果,其中,pc 是伴随关系成立的前提条件.

定义 **2.2**. 对任意一个规划领域,Pd 是该领域的一个谓词,$\forall Pd_i\in CE(Pd)-\{Pd\}$,在用动作实现谓词 Pd 时,谓词 Pd_i 也一定被实现,则称谓词 Pd_i 是实现谓词 Pd 的直接伴随结果,或称谓词 Pd_i 和 Pd 之间具有直接伴随关系,记为 $Pd\xrightarrow{\{(\text{not } Pd)\}}Pd_i$.

例如:BlocksWorld 领域中,实现谓词(holding x)[**]的动作有(Pickup x)和(Unstack x y).

$$CE((\text{holding } x))=\{(\text{not (clear } x)),(\text{not (arm-empty)}),(\text{holding } x)\}$$

由定义 2.2 可得下列直接伴随关系:

$$(\text{holding } x)_{\{(\text{not (holding } x))\}}\rightarrow(\text{not (clear } x)),\ (\text{holding } x)_{\{(\text{not (holding } x))\}}\rightarrow(\text{not (arm-empty)})$$

由上面两个直接伴随关系可知:直接伴随关系中的条件(not (holding x))是不可缺少的,因为若谓词(holding x)已存在,那么,显然无须再用动作实现.当然,相应的直接伴随关系也就可能不成立.

2.2 动作效果的条件伴随关系

直接伴随关系反映的是被实现谓词和动作其他效果之间的附带关系.在研究规划动作时,我们还发现一些特殊的动作:它们都可以实现同一个谓词,且在它们的前提条件中仅有一个条件不同.在规划求解时,正是该不同的前提条件来确定相应动作的选择.所以,该唯一不同的前提条件可代表该特殊的动作.

定义 **2.3**. 若$\forall Act_i\in Act(Pd)$都有$|PC(Act_i)-CPC(Pd)|=1$,那么,对 $Pc\in PC(Act_j)-CPC(Pd)$,$Act_j\in Act(Pd)$,则称谓词 Pc 为动作 Act_j 实现谓词 Pd 的特征前提.

定义 2.3 中的条件"$|PC(Act_i)-CPC(Pd)|=1$"是非常严格的,它要求 $Act(Pd)$中的所有动作除公共前提外,每个动作都仅有一个不同的前提条件.因此,可用该唯一不同的前提条件来区分这些动作.

(1) 若$\exists Act_j\in Act(Pd)$,且$|PC(Act_j)-CPC(Pd)|=0$,那么,在用其他动作实现谓词 Pd 时,动作 Act_j 也一定能够实现谓词 Pd.于是,可用多个动作来实现谓词 Pd,从而使规划具有多样性.

例如:在 BlocksWorld 领域中,实现谓词(arm-empty)的动作有(Stack x y)和(Putdown x),它们的前提条件分别为{(holding x),(clear y)}和{(holding x)}.所以,在用动作(Stack x y)实现谓词(arm-empty)时,也一定可用动作(Putdown x)来实现.

(2) 若$\exists Act_j\in Act(Pd)$且$|PC(Act_j)-CPC(Pd)|>1$,那么,动作是由多个前提条件的组合来区别的.这时,定义动作的特征前提比较复杂,本文不作进一步研究.

特别地,当 $CPC(Pd)=\varnothing$时,定义动作的特征前提也就没有意义了.

定义 **2.4**. 若谓词 Pc 是动作 Act_j 实现谓词 Pd 的特征前提,则$\forall Pd_i\in E(Act_j)-CE(Pd)$,谓词 Pd_i 是用动作 Act_j 实现谓词 Pd 的条件伴随结果,记为 $Pd_{\{(\text{not } Pd),Pc\}}\rightarrow Pd_i$.

例如:在 BlocksWorld 领域中,可实现谓词(holding x)的动作有(Unstack x y)和(Pickup x).动作的定义信息整理如下:

(1) (Unstack x y)

PC: (clear x) (arm-empty) (on x y)

Effect: (not (clear x)) (not (arm-empty)) (holding x) (not (on x y)) (clear y)

(2) (Pickup x)

PC: (clear x) (arm-empty) (on-table x)

Effect: (not (clear x)) (not (arm-empty)) (holding x) (not (on-table x))

由定义 2.3 可知,谓词(on x y)和(on-table x)分别是用动作(Unstack x y)和(Pickup x)实现谓词(holding x)的特征前提,即:当(on x y)成立时,只能选用动作(Unstack x y);当(on-table x)成立时,则只能选用动作(Pickup x).因此,动作的特征前提对选择动作实现特定谓词具有决定性的作用.

由定义 2.4 可得到下列条件伴随关系:

$$(\text{holding } x)_{\{(\text{not (holding } x)),(\text{on } x\ y)\}}\rightarrow(\text{clear } y)$$

$$(\text{holding } x)_{\{(\text{not (holding } x)),(\text{on } x\ y)\}}\rightarrow(\text{not (on } x\ y))$$

$$(\text{holding } x)_{\{(\text{not (holding } x)),(\text{on-table } x)\}}\rightarrow(\text{not (on-table } x))$$

** 谓词(holding x)中的"x"可以理解为"任意一个具体的积木块",它已由动作参数进行实例化了.

对于任意一个基于 STRIPS 的规划领域,用 S_0 和 S_0' 分别表示所有直接伴随关系和条件伴随关系所组成的集合.可用定义 2.2、定义 2.4、谓词的等价关系、动作参数的置换等,从规划领域的动作定义中提取动作效果的直接伴随关系和条件伴随关系.对经典的 BlocksWorld 可得到的伴随关系见附录 D 和附录 E.

2.3 直接伴随关系的逻辑推导

上一节介绍了动作效果的两种伴随关系,本节研究两个直接伴随关系进行组合所得到的结果.

直接伴随关系进行组合的过程就是根据直接伴随关系的含义进行逻辑分析的过程,逻辑分析所得到的结果就是关系组合所得到的结果.本文用符号'$\circ$'表示直接伴随关系进行组合的运算符.

定义 **2.5**. 若 $R_1=Pd_1\ _{\{(\text{not } Pd_1)\}}\rightarrow Pd_2\in S_0, R_2=Pd_2\ _{\{(\text{not } Pd_2)\}}\rightarrow Pd_3\in S_0$,则 $R_1\circ R_2=Pd_1\ _{\{(\text{not } Pd_1)\}}\rightarrow Pd_3$.

由定义 2.5 可知,当 $Pd_1=Pd_3$ 时,$R_1\circ R_2=Pd_1\ _{\{(\text{not } Pd_1)\}}\rightarrow Pd_1$,即:用动作实现谓词 Pd_1 时,谓词 Pd_1 也被实现.显然,该直接伴随关系无意义.所以,在研究直接伴随关系组合时不考虑"$Pd_1=Pd_3$"的情况.

定理 **2.1**. 若 $R_1=Pd_1\ _{\{(\text{not } Pd_1)\}}\rightarrow Pd_2\in S_0, R_2=Pd_2\ _{\{(\text{not } Pd_2)\}}\rightarrow Pd_3\in S_0, Pd_1\neq Pd_3$,则 $R_1\circ R_2\in S_0$.

证明:假设在规划领域中,$Act(Pd_1)=\{Act_{i_1}, Act_{i_2},\ldots, Act_{i_s}\}$,$Act(Pd_2)=\{Act_{j_1}, Act_{j_2},\ldots, Act_{j_t}\}$.

在动作集 $Act(Pd_1)$中任取一个动作 Act_{i_u}.

由直接伴随关系 R_1 可知:$Pd_2\in E(Act_{i_u})$.

由动作集 $Act(Pd_2)$的含义可知, $Act_{i_u}\in Act(Pd_2)$.

由集合论的基本知识可得:$Act(Pd_1)\subseteq Act(Pd_2)$.

由直接伴随关系 R_2 可知, $Act_{j_v}\in Act(Pd_2)\Rightarrow Pd_3\in E(Act_{j_v})$.

所以,$\forall\ Act_{i_u}\in Act(Pd_1)$,都有 $Act_{i_u}\in Act(Pd_2), Pd_3\in E(Act_{i_u})$,即:$Pd_3\in \cap_{Act_{i_u}\in Act(Pd_1)} E(Act_{i_u})=CE(Pd_1)$.

由"$Pd_1\neq Pd_3$"可知,$Pd_3\in CE(Pd_1)-\{Pd_1\}$.

由定义 2.2 可得:$Pd_1\ _{\{(\text{not } Pd_1)\}}\rightarrow Pd_3\in S_0$,即 $R_1\circ R_2\in S_0$. □

例如:在 BlocksWorld 领域中,存在下列两个直接伴随关系:

$$R_1:(\text{on } x\ y)_{\{(\text{not } (\text{on } x\ y))\}}\rightarrow(\text{not } (\text{holding } x)),\ R_2:(\text{not } (\text{holding } x))_{\{(\text{holding } x)\}}\rightarrow(\text{arm-empty})$$

由附录 D 可以验证:$R_1\circ R_2=(\text{on } x\ y)_{\{(\text{not } (\text{on } x\ y))\}}\rightarrow(\text{arm-empty})\in S_0$.

定理 2.1 说明,任何两个直接伴随关系都无须再进行组合,因为它们进行组合所得到的直接伴随关系已在直接伴随关系集 S_0 之中.

2.4 直接伴随关系的含义与比较

动作效果的伴随关系反映了两个谓词是否同时被实现的关系,它表达了实现同一谓词不同动作之间的共性.这种伴随关系在一定程度上具有普通逻辑中的蕴涵特征,但它们之间有着本质的不同.下面根据动作的功能来分析直接伴随关系的含义,并说明直接伴随关系和普通逻辑中蕴涵关系之间的差异.

一、直接伴随关系的含义

直接伴随关系 $Pd_i\ _{\{(\text{not } Pd_i)\}}\rightarrow Pd_j$ 的含义可归纳为以下两个方面:

(1) 当谓词 Pd_i 为"真"时,显然无须再用动作来实现,当然也就不会产生出其伴随结果 Pd_j.所以,当谓词 Pd_i 为"真"时,该直接伴随关系不起作用;

(2) 当谓词 Pd_i 为"假"时,若不用动作实现谓词 Pd_i,该直接伴随关系也不会起作用;若用动作实现谓词 Pd_i,则谓词 Pd_j 就会被同时实现.因此,谓词 Pd_i 和 Pd_j 之间的"逻辑推导"是由规划动作来完成的.

所以,$Pd_i\ _{\{(\text{not } Pd_i)\}}\rightarrow Pd_j$ 不是反映谓词 Pd_i 和 Pd_j 之间的普通逻辑推导关系,而是表示在执行规划动作时,谓词 Pd_i 和 Pd_j 同时被实现的关系.

一般情况下,$Pd_i\ _{\{(\text{not } Pd_i)\}}\rightarrow Pd_j$ 成立时,$Pd_j\ _{\{(\text{not } Pd_j)\}}\rightarrow Pd_i$ 不一定成立.

二、直接伴随关系和蕴涵关系的差异

动作效果的伴随关系符"$_{pc}\rightarrow$"和普通逻辑中的蕴涵关系符"$\rightarrow$"很相似,但它们的含义有着明显的差异.

下面以蕴涵关系 F:$A\rightarrow B$ 和直接伴随关系 R:$Pd_i \ _{\{(\text{not } Pd_i)\}}\rightarrow Pd_j$ 为例,来说明它们的相同点和不同点.

(1) 逻辑推导的条件

由蕴涵关系 F 可知,只要 A 成立,就可以直接推导出 B 也成立;但对直接伴随关系 R 来说,当谓词 Pd_i 为“真”时,该直接伴随关系不起作用,也不会得出谓词 Pd_j 为“真”.只有当谓词 Pd_i 为“假”时,该直接伴随关系才能起作用.所以,直接伴随关系的成立是有条件的.

(2) 逻辑推导的过程

由蕴涵关系 F 可知,当 A 成立时,可用逻辑推理直接推导出 B 也成立;但直接伴随关系 R 的含义是:当用动作实现谓词 Pd_i 时,谓词 Pd_j 也被实现.谓词 Pd_i 和 Pd_j 之间的“推导过程”是由规划动作来完成的.

(3) 逻辑推导的后续结果

对蕴涵关系 F 来说,只要 A 成立,永远可用逻辑推理直接推导出 B 也成立;但直接伴随关系 R 没有这样的性质.当用规划动作同时实现谓词 Pd_i 和 Pd_j 后,对含谓词 Pd_i 和 Pd_j 的规划状态,在不删除谓词 Pd_i 的前提下,还可用其他动作删去谓词 Pd_j,从而使得谓词 Pd_j 为“假”.这样的后续状态并不与关系 R 的含义相矛盾.

(4) 蕴涵关系和直接伴随关系的直接结果

由蕴涵关系 F 可知,一旦 A 成立,则 B 也肯定成立;

由直接伴随关系 R 可知,一旦用动作使谓词 Pd_i 为“真”,则在规划次态中谓词 Pd_j 也肯定为“真”.

(5) 蕴涵关系和直接伴随关系的逻辑推导

假设有蕴涵关系:$A\rightarrow B$,$B\rightarrow C$,那么,关系 $A\rightarrow C$ 一定成立.

假设有直接伴随关系:$Pd_1 \ _{\{(\text{not } Pd_1)\}}\rightarrow Pd_2$,$Pd_2 \ _{\{(\text{not } Pd_2)\}}\rightarrow Pd_3$,由定理 2.1 可知:$Pd_1 \ _{\{(\text{not } Pd_1)\}}\rightarrow Pd_3$ 也一定成立.

综上比较可知,动作效果的直接伴随关系和普通逻辑中的蕴涵关系之间的差异是明显的,它们在关系的直接结果方面存在一点相似之处,在进行逻辑推导方面也具有相似的性质.

3 动作效果的阻碍关系

在上节中,我们定义了动作效果的伴随关系,它反映了用动作实现谓词时的附带作用.本节将研究一个谓词阻碍用动作实现另一个谓词之间的逻辑关系.

3.1 动作效果的直接阻碍关系

在智能规划的研究中,我们知道,在任何规划状态下,要执行某个规划动作,则该动作的所有前提条件在当前规划状态下都必须为“真”.若该动作的前提条件中存在某个前提条件不为“真”,则该前提条件就会阻碍此动作的执行.

由第 2.1 节中符号 $CPC(Pd)$的含义可知,它是可实现谓词 Pd 所有动作的公共前提条件集合.无论采用哪个动作来实现谓词 Pd,$CPC(Pd)$中的所有谓词都必须为“真”.若 $CPC(Pd)$中存在一个谓词在当前状态下为“假”,则在当前状态下一定不能用动作来实现谓词 Pd;但谓词集 $CPC(Pd)$中的所有谓词都为“真”,谓词 Pd 也并不一定就能被实现,因为在执行具体规划动作时,可能还需要其他前提条件.所以,谓词集 $CPC(Pd)$中的谓词是谓词 Pd 的必要条件,而不是充分条件.

与定义 2.1、定义 2.2 类似,我们利用实现谓词 Pd 的公共前提条件 $CPC(Pd)$来定义动作效果的阻碍关系.

定义 **3.1**. 动作效果的直接阻碍关系 $_{pc}$–\–>:

$(\text{not } Pd_1)_{pc}$–\–>Pd_2 的含义是:在用动作实现谓词 Pd_2 时,谓词 Pd_1 必须为“真”,称谓词(not Pd_1)直接阻碍谓词 Pd_2 的实现,其中,pc 是直接阻碍关系成立的前提条件.

定义 **3.2**. 对任意一个规划领域,Pd 是该领域的一个谓词,$\forall Pc\in CPC(Pd)-\{(\text{not } Pd)\}$,在用动作实现谓词 Pd 时,前提条件 Pc 都一定要为“真”,称谓词 Pc 是实现谓词 Pd 的前提条件,或称谓词(not Pc)直接阻碍谓词 Pd 的实现,记为:$(\text{not } Pc)_{\{(\text{not } Pd)\}}$ –\–>Pd.

例如:在 BlocksWorld 领域中,实现谓词(not (arm-empty))的动作有(Unstack x y)和(Pickup x).动作的具体信

息见第 2.2 节中的描述.

$$CPC((\text{not (arm-empty)}))=\{(\text{clear } x),(\text{arm-empty})\}$$

由定义 3.2 可知,得到直接阻碍关系$(\text{not (clear } x))_{\{(\text{arm-empty})\}}$ −\–>(not (arm-empty)).

本文用 S_1 表示任意一个基于 STRIPS 规划领域中直接阻碍关系的集合.BlocksWorld 规划领域中的直接阻碍关系见附录 F.

3.2 动作效果的间接阻碍关系

在第 3.1 节中定义了动作效果的直接阻碍关系,它反映出用一个动作实现某个特定谓词所必需的前提条件.下面,利用若干个直接阻碍关系来研究一个动作序列实现谓词的动作效果关系.

假设:当前要实现谓词为 $Pd,Pd_1\in CPC(Pd)-\{(\text{not } Pd)\},Pd_2\in CPC(Pd_1)-\{(\text{not } Pd_1)\}$.

在用动作实现谓词 Pd 时,若谓词 Pd_1 为“假”,则需先用动作实现谓词 Pd_1;在实现谓词 Pd_1 时,若谓词 Pd_2 为“假”,则谓词(not Pd_2)直接阻碍谓词 Pd_1 的实现.所以,在谓词 Pd_1 为“假”时,谓词(not Pd_2)就因直接阻碍谓词 Pd_1 的实现而间接地阻碍谓词 Pd 的实现.

定义 **3.3**. 动作效果的间接阻碍关系 $_{pc}$–\\–>:

$(\text{not } Pd_1)_{pc}$–\\–>Pd_2 的含义是:在用动作序列实现谓词 Pd_2 时,谓词 Pd_1 必须为“真”,称谓词(not Pd_1)间接阻碍谓词 Pd_2 的实现,其中,pc 是该间接阻碍关系的前提.

定义 **3.4**. 递归地定义间接阻碍关系及其阻碍级数 Obs:

(1) 若$(\text{not } Pd_1)_{\{(\text{not } Pd)\}}$–\–>$Pd$,则在(not Pd)为“真”时,谓词(not Pd_1)间接阻碍 Pd 的实现,
记为 $R=(\text{not } Pd_1)_{\{(\text{not } Pd)\}}$–\\–>$Pd$,且 $Obs(R)=1$;

(2) 若 R_1:$(\text{not } Pd_2)_{\{(\text{not } Pd_1)\}}$ −\–>Pd_1,R_2:$(\text{not } Pd_1)_{pc}$–\\–>Pd,则在(not Pd_1)和 pc 为“真”时,谓词(not Pd_2)间接阻碍 Pd 的实现,记为 $R=R_1\otimes R_2=(\text{not } Pd_2)_{\{(\text{not } Pd_1)\}\cup Pc}$ –\\–>Pd,且 $Obs(R)=1+Obs(R_2)$.

本文用 S_2 表示所有间接阻碍关系的集合.图 1 是定义 3.4 所列的生成间接阻碍关系的两种组合情况.

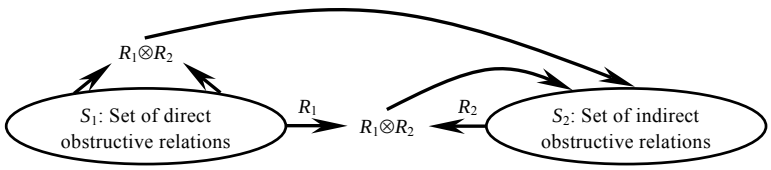

Fig.1 Two ways of generating indirect obstructive relations in definition 3.4

图 1 定义 3.4 所列生成间接阻碍关系的示意图

由定义 3.4(1)可知,“直接阻碍关系”是“间接阻碍关系”的特例,其阻碍级数为 1.

在不强调直接阻碍关系和间接阻碍关系时,我们统称它们为阻碍关系,并仍用“–\\–>”来表示阻碍关系.

阻碍关系$(\text{not } Pd_1)_{pc}$–\\–>Pd 的阻碍级数反映谓词(not Pd_1)阻碍谓词 Pd 实现的程度.阻碍级数越小的阻碍关系,前者对后者的影响就越直接,阻碍程度也就越大.

假设有阻碍关系 R_1:$(\text{not } Pd_1)_{Pc_1}$ –\\–>Pd,R_2:$(\text{not } Pd_2)_{Pc_2}$ –\\–>Pd,它们表达了不同谓词阻碍同一个谓词 Pd 的实现.若 $Obs(R_1)<Obs(R_2)$,则(not Pd_1)对实现谓词 Pd 的阻碍程度比(not Pd_2)的阻碍程度要大.

特殊地,若 $Obs(R_1)=1$,则表示(not Pd_1)直接阻碍谓词 Pd 的实现.

3.3 间接阻碍关系的生成方法

定义 3.4 给出了生成间接阻碍关系的两种组合,但并不是说这两种组合就一定能产生出新的间接阻碍关系.对一组能进行组合的阻碍关系,还需要根据规划动作的定义来分析它们能否产生出新的间接阻碍关系,从而获得它们能产生出新的间接阻碍关系所必须要满足的条件.

假设阻碍关系 R_2:$(\text{not } Pd_{i_s})_{\{(\text{not } Pd_{i_{s-1}}),\ldots,(\text{not } Pd_{i_1}),(\text{not } Pd_{i_0})\}}$ −\\–> $Pd_{i_0}\in S_1\cup S_2$.

若谓词 Pd_{i_s} 为“真”,则不存在谓词(not Pd_{i_s})阻碍谓词 Pd_{i_0} 实现的问题.只有当谓词 Pd_{i_s} 为“假”时,才存在谓词(not Pd_{i_s})阻碍谓词 Pd_{i_0} 实现的问题.这时,在谓词(not $Pd_{i_{s-1}}$),…,(not Pd_{i_2}),(not Pd_{i_1}),(not Pd_{i_0})都为“真”的前提下,为了用动作实现谓词 Pd_{i_0},就必须先用动作实现谓词 Pd_{i_s}.在用动作实现谓词 Pd_{i_s} 时,又可能需要一些必要的前提条件,以及实现它还可能产生一些相应的直接伴随结果和条件伴随结果.

下面利用与谓词 Pd_{i_s} 有关的直接阻碍关系、直接伴随关系和条件伴随关系来分析:与阻碍关系 R_2 进行组合可产生出间接阻碍关系的条件.

不妨再设:与谓词 Pd_{i_s} 有关的动作效果关系有:k 个直接阻碍关系;m 个直接伴随关系和 n 个条件伴随关系.具体的动作效果关系如下:

$$R_2:(\text{not}\ \ Pd_{i_s})_{\{(\text{not}\ Pd_{i_{s-1}}),\ldots,(\text{not}\ Pd_{i_1}),(\text{not}\ Pd_{i_0})\}} -\backslash\!\!-> Pd_{i_0}.$$

k 个直接阻碍关系: $Pd_{j_1}{}_{\{(\text{not}\ Pd_{i_s})\}} -\backslash- > Pd_{i_s}, Pd_{j_2}{}_{\{(\text{not}\ Pd_{i_s})\}} -\backslash- > Pd_{i_s},\ldots, Pd_{j_k}{}_{\{(\text{not}\ Pd_{i_s})\}} -\backslash- > Pd_{i_s}$.

m 个直接伴随关系: $Pd_{i_s}{}_{\{(\text{not}\ Pd_{i_s})\}} \to Pd_{u_1}, Pd_{i_s}{}_{\{(\text{not}\ Pd_{i_s})\}} \to Pd_{u_2},\ldots, Pd_{i_s}{}_{\{(\text{not}\ Pd_{i_s})\}} \to Pd_{u_m}$.

n 个条件伴随关系: $Pd_{i_s}{}_{\{(\text{not}\ Pd_{i_s}),pd_{k_1}\}} \to Pd_{v_1}, Pd_{i_s}{}_{\{(\text{not}\ Pd_{i_s}),pd_{k_2}\}} \to Pd_{v_2},\ldots, Pd_{i_s}{}_{\{(\text{not}\ Pd_{i_s}),pd_{k_n}\}} \to Pd_{v_n}$.

(1) $\exists(\text{not}\ \ Pd_{j_r})_{\{(\text{not}\ Pd_{i_s})\}} -\backslash-> Pd_{i_s} \in S_1$,且(not Pd_{j_r})$\in$\{(not $Pd_{i_{s-1}}$),…,(not Pd_{i_2}),(not Pd_{i_1}),(not Pd_{i_0})\}

不妨设:(not Pd_{j_r})=(not Pd_{i_t}),t=s−1..0.

(1.1) 谓词 Pd_{j_r} 为“真”

间接阻碍关系 R_2 的前提条件中,谓词(not Pd_{i_t})就为“假”.显然,间接阻碍关系 R_2 不起作用.所以,其他直接阻碍关系都无须和 R_2 进行组合.

(1.2) 谓词 Pd_{j_r} 为“假”

谓词 Pd_{j_r}, Pd_{i_s}, $Pd_{i_{s-1}}$,…, $Pd_{i_{t+1}}$, Pd_{i_t} 就是一个需用动作实现的循环谓词链.由执行动作的前提条件可知:在它们都为“假”的情况下,不可能用动作来实现该循环链中的任何一个谓词.所以,不用考虑直接阻碍谓词 Pd_{i_s} 实现的谓词间接阻碍谓词 Pd 实现的问题.

例如,在 BlocksWorld 领域中,有间接阻碍关系 R_2:(not (holding z))$_{\{(\text{not (arm-empty)}),(\text{on } x\ y)\}}-\backslash\!\!->$(not (on x y)).

在实现谓词(holding z)时,存在直接阻碍关系 R_1:(not (arm-empty))$_{\{(\text{not (holding } x))\}}-\backslash->$(holding x).

为了使关系 R_1 中的谓词(holding x)和 R_2 中的谓词(holding z)相一致,对关系 R_1 中的所有变量 x 用“z”作代入操作,得到关系 R_1 的等价表示形式 R_1' :(not (arm-empty))$_{\{(\text{not (holding } z))\}}-\backslash->$(holding z).

- 谓词(arm-empty)为“真”

间接阻碍关系 R_2 的前提条件中谓词(not (arm-empty))就为“假”.显然,间接阻碍关系 R_2 不起作用.

- 谓词(arm-empty)为“假”

由关系 R_1' 可知:要先用动作实现谓词(arm-empty).这时,就存在一个需用动作实现的循环谓词链(arm-empty),(holding z),(arm-empty).

在这些谓词都为“假”的情况下,它们是不可能用动作来实现其中任何一个谓词的.所以,不用考虑直接阻碍谓词(holding z)实现的谓词间接地阻碍谓词(not (on x y))实现的问题.

(2) $\exists\, Pd_{i_s}{}_{\{(\text{not}\ Pd_{i_s})\}} \to Pd_{u_r} \in S_0$,且(not Pd_{u_r})$\in$\{(not $Pd_{i_{s-1}}$),…,(not Pd_{i_2}),(not Pd_{i_1}),(not Pd_{i_0})\}

由直接伴随关系 $Pd_{i_s}{}_{\{(\text{not}\ Pd_{i_s})\}} \to Pd_{u_r}$ 可知:在谓词(not Pd_{i_s})为“真”,并用动作实现它时,谓词 Pd_{u_r} 也同时被实现.这时,阻碍关系 R_2 的前提条件\{(not $Pd_{i_{s-1}}$),…,(not Pd_{i_2}),(not Pd_{i_1}),(not Pd)\}中至少存在一个谓词(not Pd_{u_r})为“假”,即在用动作实现谓词 Pd_{i_s} 后,阻碍关系 R_2 就不起作用.所以,任何直接阻碍关系都不应和阻碍关系 R_2 再进行组合.

(3) $\exists\, Pd_{i_s}{}_{\{(\text{not}\ Pd_{i_s}),pd_j\}} \to Pd_{v_r} \in S_0'$,且 Pd_j,(not Pd_{v_r})$\in$\{(not $Pd_{i_{s-1}}$),…,(not Pd_{i_2}),(not Pd_{i_1}),(not Pd_{i_0})\}

由条件伴随关系“$Pd_{i_s}{}_{\{(\text{not}\ Pd_{i_s}),pd_j\}} \to Pd_{v_r}$”可知:在谓词(not Pd_{i_s})和 Pd_j 为“真”,并用动作实现谓词 Pd_{i_s} 时,谓词 Pd_{v_r} 也同时被实现.于是,在阻碍关系 R_2 的前提条件\{(not $Pd_{i_{s-1}}$),…,(not Pd_{i_2}),(not Pd_{i_1}),(not Pd_{i_0})\}中,

至少存在一个谓词(not Pd_{v_r})为“假”.所以,其他任何直接阻碍关系都不应和阻碍关系 R_2 进行组合.

综合上面 3 种情况,在用动作实现前提条件 Pd_{i_s} 时,若存在下面 3 种情况之一:

- 实现谓词 Pd_{i_s} 的前提条件与阻碍关系 R_2 的前提条件构成循环链;
- 实现谓词 Pd_{i_s} 的直接伴随结果破坏了阻碍关系 R_2 的前提条件;
- 实现谓词 Pd_{i_s} 的条件伴随结果破坏了阻碍关系 R_2 的前提条件,

则任何直接阻碍关系都不应和阻碍关系 R_2 进行组合;否则,所有直接阻碍谓词 Pd_{i_s} 实现的谓词都间接地阻碍了谓词 Pd_{i_0} 的实现,即:

对阻碍关系 R_2:(not Pd_{i_s}) $_{\{(\text{not } Pd_{i_{s-1}}),\ldots,(\text{not } Pd_{i_1}),(\text{not } Pd_{i_0})\}}$ $-\backslash\backslash->$ Pd_{i_0},$\forall R_j$:(not Pd_{j_r}) $_{\{(\text{not } Pd_{i_s})\}}$ $-\backslash->$ Pd_{i_s} $\in S_1$,都有

$$R_j \otimes R_2 = (\text{not } Pd_{i_{s+1}})\ _{\{(\text{not } Pd_{i_s}),(\text{not } Pd_{i_{s-1}}),\ldots,(\text{not } Pd_{i_1}),(\text{not } Pd_{i_0})\}}\ -\backslash\backslash->\ Pd_{i_0}.$$

对 BlocksWorld 规划领域,在直接阻碍关系集 S_1 的基础上,利用定义 3.4(2)和间接阻碍关系的生成方法可得到间接阻碍关系集合 S_2(见附录 G).

3.4 阻碍关系的含义和作用

前面给出了直接阻碍关系的定义和间接阻碍关系的生成方法,下面,我们来分析这些阻碍关系的含义以及它们在规划求解过程中的作用.

一、直接阻碍关系的含义

以直接阻碍关系(not Pd_i)$_{\{(\text{not } Pd)\}}$ $-\backslash->Pd$ 为例来说明.

(1) 当谓词 Pd 为“真”时,无须再用动作来实现谓词 Pd,当然也就不存在其他谓词阻碍其实现的问题.所以,该直接阻碍关系不起作用.

(2) 当谓词 Pd 为“假”时,该直接阻碍关系说明:用动作实现谓词 Pd 就必须要谓词 Pd_i 为“真”;否则,不可能用动作实现谓词 Pd.但谓词 Pd_i 为“真”不一定能用动作实现谓词 Pd,因为实现谓词 Pd 可能还需其他前提条件.所以,直接阻碍关系反映的是用一个规划动作实现某个谓词所需要的必要条件.

二、间接阻碍关系的含义

以间接阻碍关系(not Pd_m) $_{\{(\text{not } Pd_{m-1}),\ldots,(\text{not } Pd_1),(\text{not } Pd_0)\}}$ $-\backslash\backslash->Pd_0, m\geq 1$ 为例来说明.

(1) 当谓词 Pd_0 为“真”时,谓词 Pd_0 根本不存在用动作实现的问题.所以,该间接阻碍关系不起作用.

(2) 当谓词 Pd_0 为“假”时,由定义 3.4(2)可知:用动作实现谓词 Pd_0 就必须要谓词 Pd_1 为“真”;当谓词 Pd_1 为“假”时,用动作实现谓词 Pd_1 就必须要谓词 Pd_2 为“真”.如此反复,当谓词 Pd_{m-1} 为“假”时,用动作实现谓词 Pd_{m-1} 就必须要谓词 Pd_m 为“真”.所以,当谓词 $Pd_1,Pd_2,\ldots,Pd_{m-1}$ 都为“假”时,用动作实现谓词 Pd_0 就必须要谓词 Pd_m 为“真”.

由此可见,谓词(not Pd_m)是通过阻碍谓词(not Pd_{m-1})的实现来阻碍谓词 Pd_0 的实现.若谓词 $Pd_1,Pd_2,\ldots,Pd_{m-1}$ 之中有一个为“真”,那么,谓词(not Pd_m)就不会阻碍谓词 Pd_0 的实现.

三、阻碍关系的作用

假设阻碍关系 R_1=(not Pd_m) $_{\{(\text{not } Pd_{m-1}),\ldots,(\text{not } Pd_1),(\text{not } Pd_0)\}}$ $-\backslash\backslash->Pd_0 \in S_1 \cup S_2, m\geq 1$.

由阻碍关系的含义分析可知:在谓词 $Pd_{m-1},\ldots,Pd_2,Pd_1,Pd_0$ 都为“假”时,用动作实现谓词 Pd_0 就必须要有谓词 Pd_m 为“真”.也就是说,用阻碍关系 R_1 可向前预测 m 步来确定:必须先用动作实现谓词 Pd_m.只有先实现了谓词 Pd_m,才有可能用动作实现谓词 Pd_{m-1},然后才有可能依次用动作实现谓词 $Pd_{m-2},\ldots,Pd_2,Pd_1$ 等,从而达到最终实现谓词 Pd_0 的目的.

阻碍关系在规划求解过程的作用可归纳如下:

(1) 把实现谓词 Pd_0 所需要的一系列中间条件进行了有效的排序,从而避免产生不必要的多余动作;

(2) 在当前规划状态下可向前预测 m 步来确定所要实现的谓词(或子目标).

在当前状态下,如何确定(或预测)下一步实现的目标是所有规划求解需要解决的问题.在一些已公开的规划器中,所采用的主要策略有:构造启发函数、状态空间的智能搜索等.这些策略通过研究者的处理在一定程度

上能反映规划领域的内在规则,但在具体规划求解时,用数值的大小来反映规划领域的内在规则很难避免一些无法预计的缺陷,而且这些策略是与研究者的“人为因素”有关的.

动作效果的阻碍关系是利用领域动作的定义信息产生.直接阻碍关系反映了实现某个谓词和其他谓词之间的因果关系.间接阻碍关系是由若干个直接阻碍关系进行组合所得到的,它反映了动作序列之间的内在联系.所以,在当前状态下,用阻碍关系来预测下一步应实现的目标具有可靠的领域基础,它仅与规划领域中的动作定义有关,而与任何“人为因素”都无关.

3.5 间接阻碍关系的基本性质

第 3.2 节中的定义 3.4 说明了间接阻碍关系可由若干个直接阻碍关系进行组合而得到的,第 3.4 节叙述了间接阻碍关系的含义和作用.下面分析任意一个间接阻碍关系所具有的基本性质.

假设间接阻碍关系 $R=R_m\otimes\ldots\otimes R_2\otimes R_1=(\text{not } Pd_m)_{pc}\text{-\\-}>Pd_0\in S_2,R_m,\ldots,R_2,R_1\in S_1,m>1$,那么,$R$ 具有下列性质:

(1) 对于任何两个相邻的关系 R_{i+1} 和 $R_i,i=1..m-1$,都有

$$R_{i+1}=(\text{not } Pd_{i+1})_{\{(\text{not } Pd_i)\}}\text{-\\-}>Pd_i,\ R_i=(\text{not } Pd_i)_{\{(\text{not } Pd_{i-1})\}}\text{-\\-}>Pd_{i-1}.$$

(2) $pc=\{(\text{not } Pd_{m-1}),\ldots,(\text{not } Pd_1),(\text{not } Pd_0)\}$,且$\forall(\text{not } Pd_i),(\text{not } Pd_j)\in pc,i\neq j$,都有 $Pd_i\neq Pd_j$.

(3) R 是在 R_1 的基础上依次与直接阻碍关系 $R_2,R_3,\ldots,R_m$ 进行连续关系组合所得,且组合次序不可变换.

(4) $R_{s+k}\otimes\ldots\otimes R_{s+2}\otimes R_{s+1}\in S_1\cup S_2,0\leq s\leq m-1,1\leq k\leq m-s$.

可用第 3.3 节的“间接阻碍关系的生成条件”和反证法来证明上述性质,具体证明从略.

间接阻碍关系 R 基本性质的主要含义说明如下:

(1) 对任何两个相邻的关系 R_i 和 R_{i+1},谓词 Pd_i 和(not Pd_i)是两个互补的谓词,它们在这两个关系之间起着“承上启下”的作用;

(2) 在谓词 $Pd_0,Pd_1,\ldots,Pd_{m-1}$ 都为“假”时,谓词(not Pd_m)才间接阻碍谓词 Pd_0 的实现,并且在这 m 个谓词中不存在两个相同的谓词;

(3) 在谓词 $Pd_0,Pd_1,\ldots,Pd_{m-1}$ 都为“假”时,要用动作实现谓词 Pd_0,必须依次需要谓词 $Pd_m,Pd_{m-1},\ldots,Pd_1$ 为“真”,这隐含地确定了谓词 $Pd_m,Pd_{m-1},\ldots,Pd_1$ 的实现次序,从而可减少一些不必要的规划动作;

(4) 在 m 个直接阻碍关系 $R_m,\ldots,R_2,R_1$ 中,其任何连续 k 个关系都能进行关系组合,并可得到一个间接阻碍关系.也就是说,谓词 Pd_i 一定阻碍谓词 $Pd_{i-1},Pd_{i-2},\ldots,Pd_0$ 的实现,其中,$1<i\leq m$.

在第 3.2 和第 3.3 节中说明了直接阻碍关系与间接阻碍关系进行组合的条件和所得到的结果,但都没有讨论间接阻碍关系与其他阻碍关系进行组合的问题.下面研究任意一个间接阻碍关系与其他阻碍关系可进行关系组合的条件和所得到的结果.

定义 **3.5**. 假设 $R_u=R_m\otimes\ldots\otimes R_2\otimes R_1\in S_1\cup S_2,m\geq 1,R_v\in S_1\cup S_2$,若满足下列条件:

(1) R_1 和 R_v 能进行关系组合;

(2) $(R_m\otimes\ldots\otimes R_2)$和$(R_1\otimes R_v)$能进行关系组合.

那么称 R_u 和 R_v 能进行关系组合,并记为 $R_u\otimes R_v=(R_m\otimes\ldots\otimes R_2)\otimes(R_1\otimes R_v)$.

定义 3.5 中的条件(2)是一种递归定义,它把“R_u和 R_v能进行关系组合”转变成“$(R_m\otimes\ldots\otimes R_2)$和$(R_1\otimes R_v)$能进行关系组合”.而关系$(R_m\otimes\ldots\otimes R_2)$是由 $m-1$ 个直接阻碍关系组合所得,它比关系 R_u 少一次关系组合.所以,该递归定义会因组合关系的个数减少而终止.

定义 3.5 的本质是:对 $R_m,\ldots,R_2,R_1\in S_1,R_v\in S_1\cup S_2$,若关系 R_j 和$(R_{j-1}\otimes\ldots\otimes R_1\otimes R_v)$能进行关系组合,$j=1..m$,则称 R_u 和 R_v 能进行关系组合.

定理 **3.1**. 假设 $R_u,R_v\in S_1\cup S_2$,若 R_u 和 R_v 能进行关系组合,则 $R_u\otimes R_v\in S_2$.

证明:对于阻碍关系 R_u,由定义 3.4 可知:$R_u=R_m\otimes\ldots\otimes R_2\otimes R_1$,其中:$R_m,R_{m-1},\ldots,R_1\in S_1,m\geq 1$.

下面对生成 R_u 的直接阻碍关系个数 m 用数学归纳法来证明.

(1) 当 $m=1$ 时.

显然,$R_u\in S_1$.当 R_u 和 R_v 能进行关系组合时,由定义 3.4 可知:$R_u\otimes R_v\in S_2$.

(2) 假设 $m=n-1\geq 1$ 时,命题成立.下面证明当 $m=n$ 时,命题也同样成立.

由定义 3.5(1)和定义 3.4 可知:$R_1\otimes R_v\in S_2\subseteq S_1\cup S_2$.

由间接阻碍关系的性质(4)可知:$R_m\otimes R_{m-1}\otimes\ldots\otimes R_2\in S_1\cup S_2$.

由"R_u 和 R_v 能进行关系组合"和定义 3.5(2)可知:$(R_m\otimes R_{m-1}\otimes\ldots\otimes R_2)$和$(R_1\otimes R_v)$能进行关系组合.

由$(R_m\otimes R_{m-1}\otimes\ldots\otimes R_2),(R_1\otimes R_v)\in S_1\cup S_2$ 和归纳假设可得:$(R_m\otimes R_{m-1}\otimes\ldots\otimes R_2)\otimes(R_1\otimes R_v)\in S_2$.

由定义 3.5 可知:$R_u\otimes R_v=(R_m\otimes\ldots\otimes R_2)\otimes(R_1\otimes R_v)$,即 $R_1\otimes R_2\in S_2$. □

由定理 3.1 可知:对任何一个间接阻碍关系,它无须再与其他阻碍关系进行组合,因为这样的组合若能产生出间接阻碍关系,则该间接阻碍关系一定已在间接阻碍关系集 S_2 之中.

4 动作效果的绝对阻碍关系

第 3.2 节讨论了若干个阻碍关系进行组合的问题.下面,我们来研究直接伴随关系和阻碍关系进行组合的结果.

4.1 绝对阻碍关系的基本概念

定义 **4.1**. 动作效果的绝对阻碍关系 $_{pc}$=\\=>:

$Pd_1{}_{pc}$=\\=>Pd_2 的含义是:用动作实现谓词 Pd_1 时破坏了实现谓词 Pd_2 的前提条件,则称谓词 Pd_1 绝对阻碍谓词 Pd_2 的实现,其中,pc 是该绝对阻碍关系成立的条件.

定义 **4.2**. 假设 $R_1=Pd_{1\ \{(\text{not } Pd_1)\}}\rightarrow(\text{not } Pd_2)\in S_0, R_2=(\text{not } Pd_2)_{pc}$–\\–>$Pd_3\in S_1\cup S_2$,则用动作实现谓词 Pd_1 时一定会阻碍谓词 Pd_2 的实现,记为 $R_1\oplus R_2=Pd_{1\ \{(\text{not } Pd_1),Pd_2\}\cup Pc}$=\\=>$Pd_3, Obs(R_1\oplus R_2)=1+Obs(R_2)$.

对于定义 4.2,若谓词 Pd_2 为"假",则不可能用动作实现谓词 Pd_1,因为若用动作实现谓词 Pd_1,则无法删除谓词 Pd_2,这就与动作定义的含义相矛盾.所以,谓词 Pd_2 为"真"是 R_1 和 R_2 能进行组合的前提条件之一.

对任意一个阻碍关系 $R_2=(\text{not } Pd_2)_{pc}$–\\–>$Pd_3\in S_1\cup S_2$,下面研究与 R_2 能进行组合的直接伴随关系应具备的条件.

假设:直接伴随关系 $R_1=Pd_{1\ \{(\text{not } Pd_1)\}}\rightarrow(\text{not } Pd_2)\in S_0$.

由直接伴随关系 R_1 可知:在用动作实现谓词 Pd_1 时,伴随结果(not Pd_2)一定也被实现.

不妨再设:谓词 Pd_1 的伴随结果除谓词(not Pd_2)外,还有 m 个直接伴随结果: $Pd_{j_1}, Pd_{j_2},\ldots, Pd_{j_m}$,n 个条件伴随结果: $Pd_{v_1}, Pd_{v_2},\ldots, Pd_{v_n}$,$m\geq 0, n\geq 0$.具体的动作效果关系是

$$R_1: Pd_{1\ \{(\text{not } Pd_1)\}}\rightarrow(\text{not } Pd_2), R_2: (\text{not } Pd_2)_{pc}\text{–\\–>}Pd_3.$$

m 个直接伴随关系:$Pd_{1\ \{(\text{not } Pd_1)\}}\rightarrow Pd_{j_1}, Pd_{1\ \{(\text{not } Pd_1)\}}\rightarrow Pd_{j_2},\ldots, Pd_{1\ \{(\text{not } Pd_1)\}}\rightarrow Pd_{j_m}$.

n 个条件伴随关系:$Pd_{1\ \{(\text{not } Pd_1),Pd_{k_1}\}}\rightarrow Pd_{v_1}, Pd_{1\ \{(\text{not } Pd_1),Pd_{k_2}\}}\rightarrow Pd_{v_2},\ldots, Pd_{1\ \{(\text{not } Pd_1),Pd_{k_n}\}}\rightarrow Pd_{v_n}$.

(1) (not Pd_1)$\in pc$

在用动作实现谓词 Pd_1 时,谓词 Pd_1 就破坏阻碍关系 R_2 的前提条件.所以,R_1 和 R_2 不应再进行组合.

(2) $\exists\ Pd_{j_t}\in\{Pd_{j_1}, Pd_{j_2},\ldots, Pd_{j_m}\}\cap pc$

(2.1) Pd_{j_t} =(not Pd_3)

在用动作实现谓词 Pd_1 之前,谓词 Pd_3 已经为"真",否则,谓词 Pd_1 的直接伴随结果 Pd_{j_t} 不可能获得,即 Pd_3 不可能被删除.所以,在实现谓词 Pd_1 之前,关系 R_2 是无效的.当然,关系 R_1 和 R_2 也就不必进行关系组合.

(2.2) (not Pd_{j_t})$\in pc$

在用动作实现谓词 Pd_1 时,谓词 Pd_1 的直接伴随结果 Pd_{j_t} 就破坏阻碍关系 R_2 的前提条件.所以,关系 R_1 和 R_2 不应再进行关系组合;

(3) $\exists\ Pd_{k_r}\in pc$,使得: Pd_{k_r} =(not Pd_3)或(not Pd_{k_r})$\in pc$

类似情况 2 的分析可得:关系 R_1 和 R_2 不应进行关系组合.

(4) 不存在上述 3 种情况

由定义 4.2 可得:$R_1 \oplus R_2 = Pd_{1\ \{(\text{not } Pd_1), Pd_2\} \cup Pc} =\backslash\backslash=> Pd_3, Obs(R_1 \oplus R_2) = 1 + Obs(R_2)$.

综上可得:对于直接伴随关系 R_1 和阻碍关系 R_2,若谓词 Pd_1、谓词 Pd_1 的所有直接伴随结果和条件伴随结果都不破坏阻碍关系 R_2 的前提条件,那么,R_1 和 R_2 能进行关系组合.

对于任意一个规划领域,我们用 S_3 来表示该领域动作效果的绝对阻碍关系集合.

$$S_3 = \{R_1 \oplus R_2 | R_1 \in S_0, R_2 \in S_1 \cup S_2, \text{且 } R_1 \text{ 和 } R_2 \text{ 满足关系组合的条件}\}$$

对 BlocksWorld 规划领域,其动作效果中的所有绝对阻碍关系集 S_3 见附录 H.

4.2 绝对阻碍关系的基本性质

在研究绝对阻碍关系的性质时,需要考虑各类动作效果关系之间进行关系组合的问题.为了方便叙述,下面用符号'⊙'表示任意两个动作效果关系进行关系组合的运算符,它可以代表直接伴随关系组合运算符'°'、阻碍关系组合运算符'⊗'以及直接伴随关系和阻碍关系进行关系组合的运算符'⊕'等.

引理 **4.1**. 假设 $R_1 = (\text{not } Pd_1)_{\ \{(\text{not } Pd_2)\}} -\backslash\!\!-> Pd_2 \in S_1, R_2 = Pd_{2\ \{(\text{not } Pd_2)\}} \rightarrow Pd_3 \in S_0$,若 R_1 和 R_2 能进行关系组合,则

$$R_1 \odot R_2 = (\text{not } Pd_1)_{\ \{(\text{not } Pd_3)\}} -\backslash\!\!-> Pd_3 \in S_1.$$

证明:下面根据谓词 Pd_1 和 $CPC(Pd_3)$之间的关系来讨论.

(1) $Pd_1 \notin CPC(Pd_3)$

显然,谓词(not Pd_1)不可能阻碍谓词 Pd_3 的实现.所以,R_1 和 R_2 不能进行关系组合.

(2) $Pd_1 \in CPC(Pd_3)$

由定义 3.2 可知:$(\text{not } Pd_1)_{\ \{(\text{not } Pd_3)\}} -\backslash\!\!-> Pd_3 \in S_1$.

所以,若 R_1 和 R_2 能进行关系组合,则 $R_1 \odot R_2 = (\text{not } Pd_1)_{\ \{(\text{not } Pd_3)\}} -\backslash\!\!-> Pd_3 \in S_1$. □

引理 4.1 说明了谓词(not Pd_1)直接阻碍谓词 Pd_2 的实现,但它不一定直接阻碍谓词 Pd_3 的实现.若谓词(not Pd_1)直接阻碍谓词 Pd_3 的实现,则直接阻碍关系 $(\text{not } Pd_1)_{\ \{(\text{not } Pd_3)\}} -\backslash\!\!-> Pd_3$ 已在直接阻碍关系集合之中.所以,任何直接阻碍关系和直接伴随关系都无须再进行组合.

对于引理 4.1 证明中的情况(1),我们用 BlocksWorld 领域的情况来加以说明.

在 BlocksWorld 领域的动作效果关系中,有下列直接阻碍关系 R_1 和直接伴随关系 R_2,

$$R_1: (\text{not } (\text{holding } x))_{\{(\text{not } (\text{on } x\ y))\}} -\backslash\!\!-> (\text{on } x\ y), R_2: (\text{on } x\ y)_{\{(\text{not } (\text{on } x\ y))\}} \rightarrow (\text{clear } x).$$

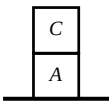

Fig.2 A state of BlocksWorld

图 2 积木世界的状态示意图

但关系$(\text{not } (\text{holding } x))_{\{(\text{clear } x)\}} -\backslash\!\!-> (\text{clear } x)$不成立,即实现(clear x)无须谓词(holding x)为"真".

例如:在如图 2 所示的状态中,要实现(clear A)就无须先实现(holding A),即(holding A)不是实现谓词(clear A)的前提条件.

下面研究绝对阻碍关系和其他动作效果关系进行组合所可能获得的结果.

定义 **4.3**. 假设 $R_u = R_m \otimes \ldots \otimes R_2 \otimes R_1 \in S_2, R_m, \ldots, R_2, R_1 \in S_1, m > 1, R_v \in S_0$,若 R_1 和 R_0,$(R_m \otimes \ldots \otimes R_2)$和$(R_1 \odot R_v)$都能进行关系组合,则称 R_u 和 R_v 能进行关系组合,并记为 $R_u \odot R_v = (R_m \otimes \ldots \otimes R_2) \otimes (R_1 \odot R_v)$.

引理 **4.2**. 假设 $R_u \in S_1 \cup S_2, R_v \in S_0$,若 R_u 和 R_v 能进行关系组合,则 $R_u \odot R_v \in S_1 \cup S_2$.

证明:下面根据 R_u 的类型来证明.

(1) $R_u \in S_1$

由定义 4.3 和引理 4.1 可知:$R_u \odot R_v \in S_1 \subseteq S_1 \cup S_2$.

(2) $R_u \in S_2$

不妨假设:$R_u = R_m \otimes \ldots \otimes R_2 \otimes R_1$,其中:$R_m, \ldots, R_2, R_1 \in S_1, m > 1$.

由 $R_1 \in S_1, R_v \in S_0$ 和引理 4.1 可知:$R_1 \odot R_v \in S_1$.

由间接阻碍关系的性质(4)可知:$R_m \otimes \ldots \otimes R_2 \in S_1 \cup S_2$.

由 $R_m \otimes \ldots \otimes R_2 \in S_1 \cup S_2, R_1 \odot R_v \in S_1$ 和定理 3.1 可知:$(R_m \otimes \ldots \otimes R_2) \otimes (R_1 \odot R_2) \in S_2 \subseteq S_1 \cup S_2$.

由定义 4.3 可知:$R_u \odot R_v=(R_m \otimes \ldots \otimes R_2)\otimes(R_1 \odot R_v)$,即 $R_u \odot R_v \in S_1 \cup S_2$.

综合上述两种情况可得:若 R_u 和 R_v 能进行关系组合,则 $R_u \odot R_v \in S_1 \cup S_2$. □

图 3 反映了直接伴随关系 S_0 与动作效果关系 $S_0 \cup S_1 \cup S_2$ 进行关系组合的各种情况;定理 2.1 和引理 4.2 分别证明了 $R_1 \circ R_2, R_4 \odot R_3$ 的结果;定义 4.2 说明了 $R_3 \oplus R_4$ 可产生绝对阻碍关系.

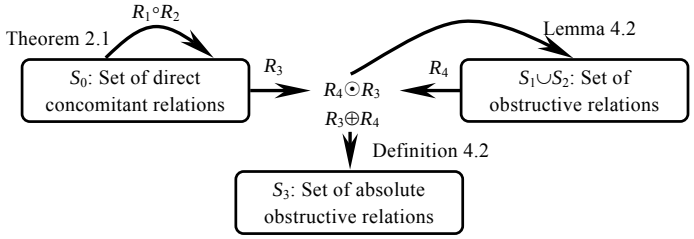

Fig.3 Results of combining direct concomitant relations with other relations of effect of actions

图 3 直接伴随关系与动作效果关系进行关系组合的结果

定义 **4.4**. 假设 $R_1 \in S_0 \cup S_1 \cup S_2, R_2=R_{20} \oplus R_{21} \in S_3, R_{20} \in S_0, R_{21} \in S_1 \cup S_2$,

(1) 若 R_1 和 R_{20},$(R_1 \odot R_{20})$和 R_{21} 都能进行关系组合,则称 R_1 和 R_2 能进行关系组合,
记为 $R_1 \odot R_2=(R_1 \odot R_{20}) \odot R_{21}$;

(2) 若 R_{21} 和 R_1, R_{20} 和$(R_{21} \odot R_1)$都能进行关系组合,则称 R_2 和 R_1 能进行关系组合,
记为 $R_2 \odot R_1=R_{20} \oplus (R_{21} \odot R_1)$.

定义 **4.5**. 假设 $R_1=R_{10} \oplus R_{11}, R_2=R_{20} \oplus R_{21}, R_1, R_2 \in S_3, R_{10}, R_{20} \in S_0, R_{11}, R_{21} \in S_1 \cup S_2$,若 R_{11} 和 R_{20},$(R_{11} \odot R_{20})$和 R_{21}, R_{10} 和$((R_{11} \odot R_{20}) \otimes R_{21})$都能进行关系组合,则称 R_1 和 R_2 能进行关系组合,记为 $R_1 \odot R_2=R_{10} \oplus ((R_{11} \odot R_{20}) \otimes R_{21})$.

定理 **4.1**. 假设 $R_1 \in S_0 \cup S_1 \cup S_2, R_2 \in S_3$,若 R_1 和 R_2 能进行组合,则 $R_1 \odot R_2 \in S_2 \cup S_3$.

定理 **4.2**. 假设 $R_1 \in S_0 \cup S_1 \cup S_2, R_2 \in S_3$,若 R_2 和 R_1 能进行组合,则 $R_2 \odot R_1 \in S_3$.

定理 **4.3**. 假设 $R_1, R_2 \in S_3$,若 R_1 和 R_2 能进行关系组合,则 $R_1 \odot R_2 \in S_3$.

定理 4.1~定理 4.3 的证明见附录 A.它们说明:任意一个绝对阻碍关系和其他动作效果关系进行组合,都不会产生新的动作效果关系.具体的关系组合情况如图 4 所示.

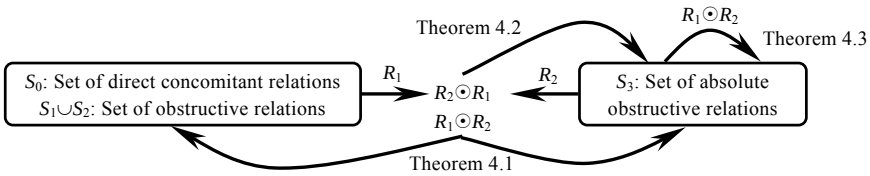

Fig.4 Results of combining absolute obstructive relations with other relations of effect of actions

图 4 绝对阻碍关系和动作效果关系进行关系组合的结果

由定理 2.1、定理 3.1、定理 4.1~定理 4.3 可知,动作效果关系集 $S_0 \cup S_1 \cup S_2 \cup S_3$ 对关系组合运算‘⊙’是封闭的.

在规划求解过程中,利用绝对阻碍关系可避免产生不必要的规划动作.有关绝对阻碍关系的作用在第 5 节有进一步的说明.

4.3 阻碍关系和绝对阻碍关系的比较

阻碍关系是由若干个直接阻碍关系进行组合所得,绝对阻碍关系是由直接伴随关系和阻碍关系进行组合所得.下面分析阻碍关系和绝对阻碍关系之间的差异.

假设:阻碍关系 R_1:(not Pd_1) $_{Pc_1}$ –\\–>$Pd_2 \in S_1 \cup S_2$,绝对阻碍关系 R_2:Pd_3 $_{Pc_2}$ =\\=>$Pd_4 \in S_3$,

(1) 阻碍关系 R_1 的含义是:当谓词集 pc_1 为“真”时,若要实现谓词 Pd_2,则必须先实现谓词 Pd_1;

(2) 绝对阻碍关系 R_2 的含义是:当谓词集 pc_2 为"真"时,若要实现谓词 Pd_4,则最好不要先用动作实现谓词 Pd_3,因为谓词 Pd_3 的伴随结果将直接阻碍谓词 Pd_4 的实现.

由此可知:阻碍关系列出了实现谓词的一系列必要的前提条件,而绝对阻碍关系描述的是一个谓词的实现将直接阻碍另一个谓词的实现.

5 动作效果关系的具体应用

前面介绍了动作效果关系的基本概念和基本性质,这些关系都是以一种规则的形式表达的,它不代表两个具体谓词之间的关系.比如:$(\text{on } x\ y)_{\{(\text{not } (\text{on } x\ y))\}}\rightarrow(\text{not } (\text{clear } y))$就是一个直接伴随关系模式.对两个具体的谓词,如(on A B)和(not (clear B)),它们之间是否符合某个动作效果关系的模式就需要进行判定.因此,该判定问题就是两个具体谓词与动作效果关系之间的模式匹配问题.

本节将研究在动作效果关系中,具体谓词和动作效果关系之间的模式匹配问题以及这些动作效果关系在规划求解中的具体作用.

5.1 动作效果关系的模式匹配

为方便叙述,在不强调具体动作效果关系时,用符号"$_{pc}\sim>$"代表"$_{pc}\rightarrow$","$_{pc}-\backslash->$","$_{pc}-\backslash\backslash->$"和"$_{pc}=\backslash\backslash=>$"等.

定义 **5.1**. 假设存在关系:$Pd_1{}_{pc}\sim>Pd_2\in S_0\cup S'_0\cup S_1\cup S_2\cup S_3$,若$\sigma$是一个置换,则关系:$Pd_1\sigma_{pc\sigma}\sim>Pd_2\sigma$也成立,并称该规则为代入规则.

动作效果关系是谓词之间的一种关系模式.对于具体的两个谓词,若它们符合动作效果关系 R 的模式,则这两个具体谓词之间就具有关系 R 的性质.

例如:在 BlocksWorld 领域中,存在直接伴随关系 $R=(\text{on } x\ y)_{\{(\text{not } (\text{on } x\ y))\}}\rightarrow(\text{not } (\text{clear } y))$,判断谓词(on A B)和(not (clear B))之间的动作效果关系.

令$\sigma=\{A/x,B/y\}$.

由定义 5.1 可知:直接伴随关系 $R'_1=(\text{on } A\ B)_{\{(\text{not } (\text{on } A\ B))\}}\rightarrow(\text{not } (\text{clear } B))$也是成立的.

由关系 R'_1 可知:在用动作实现(on A B)时,谓词(not (clear B))也被实现,即谓词(clear B)一定被删除.

5.2 子目标的排斥性

在智能规划的具体问题描述中有两个特殊的规划状态:初始状态和目标状态.本文称目标状态中的每个谓词为子目标.若目标状态中的每个子目标都实现了,显然,目标状态也就实现了.因此,我们把整个规划问题的求解分解为若干个子目标的求解.

下面利用绝对阻碍关系中的一些特殊关系来作进一步的研究.

定义 **5.2**. 假设 $R=R_1\oplus R_2=Pd_{i\ \{(\text{not } Pd_i),Pd_j\}\cup Pc}=//=>Pd_k\in S_3$,且 $R_1=Pd_{i\ \{(\text{not } Pd_i)\}}\rightarrow(\text{not } Pd_j)\in S_0$,$R_2=(\text{not } Pd_j)_{pc}-//->Pd_k\in S_1\cup S_2$,若存在 $Pd_{j\ \{(\text{not } Pd_j)\}}\rightarrow(\text{not } Pd_i)\in S_0$ 或 $Pd_{j\ \{(\text{not } Pd_j),Pd_i\}}\rightarrow(\text{not } Pd_i)\in S'_0$,则 R 为排斥性阻碍关系.

假设:当前要实现的谓词有 Pd_i 和 Pd_k,且 $R=Pd_{i\ \{(\text{not } Pd_i),Pd_j\}\cup Pc}=//=>Pd_k\in S_3$ 是排斥性阻碍关系.

由定义 5.2 可知:$R=R_1\oplus R_2$,$R_1=Pd_{i\ \{(\text{not } Pd_i)\}}\rightarrow(\text{not } Pd_j)\in S_0$,$R_2=(\text{not } Pd_j)_{pc}-//->Pd_k\in S_1\cup S_2$,且存在 $R_3=Pd_{j\ \{(\text{not } Pd_j)\}}\rightarrow(\text{not } Pd_i)\in S_0$ 或 $R_4=Pd_{j\ \{(\text{not } Pd_j),Pd_i\}}\rightarrow(\text{not } Pd_i)\in S'_0$.

不妨再设:先用规划动作实现谓词 Pd_i.

由关系 R_1 可知:在用动作实现谓词 Pd_i 时,删除了实现谓词 Pd_k 的前提条件 Pd_j.

于是,在用动作实现谓词 Pd_k 时,又要先用动作再实现谓词 Pd_j(关系 R_2 的含义).在用动作实现谓词 Pd_j 时,又删除了刚实现的谓词 Pd_i(关系 R_3 的含义).所以,先用规划动作实现谓词 Pd_i 就显得没有意义.

因此,在需要实现谓词 Pd_i 和 Pd_k 时,不能先用动作实现谓词 Pd_i.在没有实现谓词 Pd_k 的情况下,先实现谓词 Pd_i 的动作就变得多余.

在智能规划的研究中,避免产生多余的规划动作也是提高规划效率和规划质量的策略之一.

定义 **5.3**. 若绝对阻碍关系 Pd_{1pc}=//=>Pd_2 是排斥性阻碍关系,则谓词 Pd_2 的实现应先于谓词 Pd_1 的实现,记为 Pd_1<Pd_2.

例如:在 BlocksWorld 领域的具体规划问题中,存在子目标(on *A B*)和(on *B C*).

在附录 H 中,存在排斥性阻碍关系(on *z x*)$_{\{(\text{not (on } z\ x)),(\text{clear } x),(\text{not (holding } x)),(\text{not (on } x\ y))\}}$=//=>(on *x y*).

由定义 5.3 可知:(on *A B*)<(on *B C*),即:在实现谓词(on *A B*)和(on *B C*)时,应先实现谓词(on *B C*).

定理 **5.1**. 假设有两个谓词 Pd_i 和 Pd_j,若 Pd_i<Pd_j,Pd_j<Pd_i,则谓词 Pd_i 和 Pd_j 不可能都被实现.

可用定义 5.2、定义 5.3 和动作效果关系的基本含义来证明,具体证明从略.

对 BlocksWorld 领域,由附录 H 中的绝对阻碍关系和定理 5.1 可知:(holding *A*)和(holding *B*),(on *A B*)和(on *A C*),(on *A B*)和(on-table *A*)等都是不能被实现的谓词组.显然,含有它们的目标状态也就是不可实现的.

5.3 子目标的排序

在定义 5.3 中,利用排斥性阻碍关系把谓词的实现次序进行了适当的排序,获得一个合理的实现顺序,以达到减少多余动作的目的,这对提高规划的质量有着直接的指导作用.对于一组待实现的子目标,可对它们进行两两排序,然后利用顺序的传递性得到整个目标的实现顺序.若子目标的实现顺序中存在循环(或局部循环),则该目标状态一定不可实现,这样就达到了检验目标状态的合理性.

定理 **5.2**. 假设某个规划领域的目标状态中含有子目标 Pd_1,Pd_2,…,Pd_m,若 Pd_1<Pd_2<…<Pd_m<Pd_1,则该目标状态一定是不可实现的.

可用类似定理 5.1 的证明方法来证明,具体证明从略.

下面是 BlocksWorld 领域的一个基准问题(benchmark problem),其状态如图 5 所示.

```
(define (problem impossible)
(:domain     blocksworld)
(:objects    a b c)
 (:init   (on-table a) (on b a) (on c b) (clear c) (arm-empty))
(:goal    (and (on c b) (on b a) (on a c)))
)
```

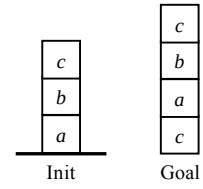

Fig.5 The state of Impossible problem

图 5 Impossible 的状态示意图

由定义 5.3 可得:(on *c b*)<(on *b a*)<(on *a c*)<(on *c b*).

由此可见,在该目标状态中存在一个目标循环链.所以,不论先实现哪个目标都会阻碍其他目标的实现,即:该规划问题是不可解的.

下面是 BlocksWorld 领域的另一个基准问题,其状态如图 6 所示.

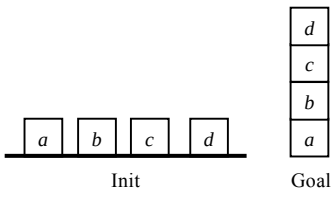

Fig.6 The state of Blocks-4-0 problem

图 6 Blocks-4-0 的状态示意图

```
(define (problem Blocks-4-0)
(:domain     blocksworld)
(:objects    a b c d)
 (:init  (clear a) (on-table a) (clear b) (on-table b) (clear c)
         (on-table c) (clear d) (on-table d) (arm-empty))
(:goal (and (on d c) (on c b) (on b a)))
)
```

由定义 5.3 可得:(on *d c*)<(on *c b*)<(on *b a*).

因此,在进行规划求解时,实现子目标的顺序是:(on *b a*),(on *c b*),(on *d c*).显然,按照该实现顺序可以快速获得实现规划问题的动作序列.

5.4 实现谓词的动作分组

在规划领域描述中,一个谓词可能由多个动作来实现,但在不同的规划状态下,只能选用少数几个特定的动作来实现.我们可用动作的特征前提来选择实现谓词的动作(见定义 2.3),也可用动作前提条件之间的互斥性来

进行分组.下面说明利用动作前提条件的互斥性进行分组的思想.

在规划过程中,谓词会因动作的执行而被增加或删除.若谓词 Pd 被增加或删除过,则称谓词 Pd 的状态发生过变化.

定义 **5.4**. 对任意两个谓词 Pd_1 和 Pd_2,若 $Pd_1 \xrightarrow{\{(\text{not } Pd_1)\}} (\text{not } Pd_2) \in S_0$,$Pd_2 \xrightarrow{\{(\text{not } Pd_2)\}} (\text{not } Pd_1) \in S_0$,则称谓词 Pd_1 和 Pd_2 是互斥的.

定义 5.4 说明,两个互斥谓词的本质就是用动作实现一个谓词时,另一个谓词肯定被删去,即在用动作改变它们的状态时,它们不可能都存在于规划次态之中.

若谓词(not Pd_1)和(not Pd_2)是互斥的,由定义 5.4 可知:用动作删除一个谓词时,另一个谓词一定被添加.即在用动作改变它们的状态时,它们不可能都不在规划次态之中.

定理 **5.3**. 假设谓词 Pd_1 和 Pd_2,(not Pd_1)和(not Pd_2)都是互斥的,若在规划求解过程中改变过它们的状态,则谓词 Pd_1 和 Pd_2 在任何规划状态中都只有一个为"真".

定理 5.3 的证明见附录 A.

例如:在 BlocksWorld 领域,由定义 5.4 可知:谓词(holding x)和(arm-empty)是互斥的,它们在任何规划状态中,有且仅有一个谓词为"真".

定义 **5.5**. 假设谓词 Pd_1 和 Pd_2 是互斥的,若 $Pd_1 \in PC(Act_i)$,$Pd_2 \in PC(Act_j)$,则动作 Act_i 和 Act_j 是互斥的.

由定义 5.5 可知:对任意两个互斥的动作,因为前提条件是在任何状态下都不可能同时得到满足,所以在任何规划状态下,它们不可能都被执行.因此,对可实现同一谓词的一组动作,我们可利用其动作前提条件的互斥性来进行分组,使得在不同的规划状态下准确地选用不同的动作,减少实现谓词的可能性,从而达到提高规划求解的效率.

例如:在 BlocksWorld 领域中,实现谓词(clear x)的动作有:(Stack x y),(Putdown x)和(Unstack y x).它们的定义信息整理如下:

(1) (Stack x y)

PC: (holding x) (clear y)

Effect: (not (holding x)) (clear x) (arm-empty) (not (clear y)) (on x y)

(2) (Putdown x)

PC: (holding x)

Effect: (not (holding x)) (clear x) (arm-empty) (on-table x)

(3) (Unstack y x)

PC: (clear y) (on y x) (arm-empty)

Effect: (not (clear y)) (not (on y x)) (not (arm-empty)) (clear x) (holding y)

由于谓词(holding x)和(arm-empty)是互斥的,在任何状态下,这两个谓词都仅有一个为"真".所以,可把上面 3 个动作分成两组,要么用动作(Stack x y)或(Putdown x)来实现谓词(clear x),要么用动作(Unstack y x)来实现.这显然减少了选择动作的可能性,提高了规划求解的效率.

6 结束语

本文从 STRIPS 领域的动作定义出发,利用实现同一谓词所有动作的公共前提和公共效果,定义了基本的动作效果关系——直接伴随关系、条件伴随关系和直接阻碍关系.在此基础上,利用 STRIPS 规划领域的动作定义,对这些基本关系进行了关系组合(或逻辑推导),得到了间接阻碍关系和绝对阻碍关系.本文所研究的内容对所有用 PDDL 语言描述的 STRIPS 领域都是有效的,所提出的动作效果关系在 Linux 环境下给予了实现.对国际规划竞赛 IPC 所公布的所有基于 STRIPS 的基准规划领域进行了测试,所获得的动作效果关系直观地表达了规划领域定义中所隐含的领域规则.

本文所介绍的动作效果关系已应用于我们开发的"与领域无关的智能规划器——StepByStep"之中,所得到

的动作效果关系直接运用于具体规划问题的求解过程.在规划求解时,用动作效果关系来决定谓词的选择策略,提高了规划效率.有关规划器 StepByStep 的设计思想和求解效率将另文介绍.

致谢 本文的部分实验由研究生魏玉虎同学完成,实验结果验证了动作效果关系的正确性和在规划求解中的实际作用.在此,对他表示感谢.

References:

[1] Nilsson NJ. Artificial Intelligence: A New Synthesis. Morgan Kaufmann Publishers, 1999. 363−404.

[2] Bonet B, Geffner H. Planning as heuristic search. Artificial Intelligence, 2001,129:5−33.

[3] Refanidis I, Vlahavas I. The GRT planning system: Backward heuristic construction in forward state-space planning. Artificial Intelligence Research, 2001,15:115−161.

[4] Refanidis I, Vlahavas I. GRT: A domain independent heuristic for strips worlds based on greedy regression tables. In: Proc. of the 5th European Conf. on Planning (ECP'99). Durham: Springer-Verlag, 1999. 347−359.

[5] Hoffmann J. FF: The fast-forward planning system. AI Magazine, 2001,22(3):57−62.

[6] Hoffmann J, Nebel B. The FF planning system: Fast plan generation through heuristic search. Artificial Intelligence Research, 2001, 14:253−302.

[7] Refanidis I, Vlahavas I. Multiobjective heuristic state-space planning. Artificial Intelligence, 2003,145(1−2):1−32.

[8] Sierra-Santibanez J. Heuristic planning: A declarative approach based on strategies for action selection. Artificial Intelligence, 2004,153:307−337.

[9] Nau DS, Smith SJJ, Erol K. Control strategies in HTN planning: Theory versus practice. In: Proc. of the AAAI'98/IAAI'98. 1998. 1127−1133.

[10] Tsuneto R, Hendler J, Nau D. Analyzing external conditions to improve the efficiency of HTN planning. In: Proc. of the AAAI'98. 1998. 913−920. http://citeseer.ist.psu.edu/650056.html

[11] Kambhampati S, Mali A, Srivastava B. Hybrid planning for partially hierarchical domains. In: Proc. of the AAAI'98/IAAI'98. 1998. 882−888.

[12] Blum A, Furst M. Fast planning through planning graph analysis. Artificial Intelligence, 1997,90(1):281−300.

[13] Mcallester D, Rosenblitt D. Systematic nonlinear planning. In: Proc. of the 9th National Conf. on AI. 1991. 634−639. http://citeseer.ist.psu.edu/mcallester91systematic.html

[14] Fink E, Veloso MM. Formalizing the PRODIGY planning algorithm. New Directions in AI Planning, 1996, 261−272. http://citeseer.ist.psu.edu/fink96formalizing.html

[15] Veloso MM. Prodigy/Analogy: Analogical reasoning in general problem solving. In: Wess S, Altho KD, Richter MM, eds. Topics on CaseBased Reasoning, Selected Papers from the 1st European Workshop on CaseBased Reasoning|EWCBR'93, LNAI 837, Springer-Verlag, 1994. 33−50.

[16] Penberthy JS, Weld DS. UCPOP: A sound, complete, partial order planner for ADL. In: Proc. of the 3rd Int'l Conf. Principles of Knowledge Representation and Reasoning. 1992. 103−114. http://citeseer.ist.psu.edu/penberthy92ucpop.html

[17] Shin JA, Davis E. Processes and continuous change in a sat-based planner. Artificial Intelligence, 2005,166:194−253.

[18] Kautz H, Selman B. BLACKBOX: A new approach to the application of theorem proving to problem solving. In: Proc. of the Workshop Planning as Combinatorial Search, AIPS-98. Pittsburgh, 1998. 58−60.

[19] Kautz H, Selman B. The role of domain-specific knowledge in the planning as satisfiability framework. In: Proc. of the 4th IJCAI. Menlo Park: AAAI Press, 1998. 181−189.

[20] Do MB, Kambhampati S. Solving Planning Graph By Compling It Into CSP. In: Proc. of the AIPS 2000. 2000.82−91.

[21] Do MB, Kambhampati S. Planning as constraint satisfaction: Solving the planning graph by compiling it into CSP. Artificial Intelligence, 2001,132:151−182.

[22] Nguyen X, Kambhampati S, Nigenda RS. Planning graph as the basis for deriving heuristics for plan synthesis by state space and CSP search. Artificial Intelligence, 2002,135:73−123.

[23] Lifschitz E. On the semantics of STRIPS. In: Reasoning about Actions and Plans. 1987. 1−9. http://citeseer.ist.psu.edu/670251.html

[24] Mcdermott D. PDDL—The planning domain definition language. Technical Report, 1998.

[25] Fox M, Long D. PDDL 2.1: An extension to PDDL for expressing temporal planning domains. Technical Report, Durham: University of Durham, 2002.

[26] Thiébaux S, Hoffmann J. Nebel B. In defense of PDDL axioms. Artificial Intelligence, 2005,168:38−69.

附录

A. 文中的证明

定理 **4.1**. 假设 $R_1\in S_0\cup S_1\cup S_2, R_2\in S_3$,若 R_1 和 R_2 能进行组合,则 $R_1\odot R_2\in S_2\cup S_3$.

证明:不妨假设 $R_2=R_{20}\oplus R_{21}, R_{20}\in S_0, R_{21}\in S_1\cup S_2$.

由定义 4.4(1)可知:$R_1\odot R_2=(R_1\odot R_{20})\odot R_{21}$.

1) $R_1\in S_0$

由 $R_1, R_{20}\in S_0$ 及定理 2.1 可知:$R_1\odot R_{20}=R_1\circ R_{20}\in S_0$.

由 $R_1\odot R_{20}\in S_0, R_{21}\in S_1\cup S_2$ 和定义 4.2 可知:$(R_1\odot R_{20})\odot R_{21}=(R_1\circ R_{20})\oplus R_{21}\in S_3\subseteq S_2\cup S_3$.

所以,$R_1\odot R_2\in S_2\cup S_3$.

2) $R_1\in S_1\cup S_2$

由 $R_1\in S_1\cup S_2, R_{20}\in S_0$ 及引理 4.2 可知:$R_1\odot R_{20}\in S_1\cup S_2$.

由 $R_1\odot R_{20}, R_{21}\in S_1\cup S_2$ 及定理 3.1 可知:$(R_1\odot R_{20})\odot R_{21}=(R_1\odot R_{20})\oplus R_{21}\in S_2\subseteq S_2\cup S_3$.

所以,$R_1\odot R_2\in S_2\cup S_3$. □

定理 **4.2**. 假设 $R_1\in S_0\cup S_1\cup S_2, R_2\in S_3$,若 R_2 和 R_1 能进行组合,则 $R_2\odot R_1\in S_3$.

证明:不妨假设 $R_2=R_{20}\oplus R_{21}, R_{20}\in S_0, R_{21}\in S_1\cup S_2$.

由定义 4.4(2)可知:$R_2\odot R_1=R_{20}\oplus(R_{21}\odot R_1)$.

1) $R_1\in S_0$

由 $R_{21}\in S_1\cup S_2, R_1\in S_0$ 及引理 4.2 可知:$R_{21}\odot R_1\in S_1\cup S_2$.

由 $R_{20}\in S_0, R_{21}\odot R_1\in S_1\cup S_2$ 及定义 4.2 可知:$R_{20}\oplus(R_{21}\odot R_1)\in S_3$.

所以,$R_2\odot R_1\in S_3$.

2) $R_1\in S_1\cup S_2$

由 $R_{21}, R_1\in S_1\cup S_2$ 及定理 3.1 可知:$R_{21}\odot R_1=R_{21}\otimes R_1\in S_2$.

由 $R_{20}\in S_0, R_{21}\odot R_1\in S_2$ 及定义 4.2 可知:$R_{20}\oplus(R_{21}\odot R_1)\in S_3$.

所以,$R_2\odot R_1\in S_3$. □

定理 **4.3**. 假设 $R_1, R_2\in S_3$,若 R_1 和 R_2 能进行关系组合,则 $R_1\odot R_2\in S_3$.

证明:不妨假设 $R_1=R_{10}\oplus R_{11}, R_2=R_{20}\oplus R_{21}, R_{10}, R_{20}\in S_0, R_{11}, R_{21}\in S_1\cup S_2$.

由定义 4.5 可知:$R_1\odot R_2=R_{10}\oplus((R_{11}\odot R_{20})\otimes R_{21})$.

由 $R_{11}\in S_1\cup S_2, R_{20}\in S_0$ 及引理 4.2 可知:$R_{11}\odot R_{20}\in S_1\cup S_2$.

由 $R_{11}\odot R_{20}, R_{21}\in S_1\cup S_2$ 及定理 3.1 可知:$(R_{11}\odot R_{20})\otimes R_{21}\in S_2$.

由 $R_{10}\in S_0, (R_{11}\odot R_{20})\otimes R_{21}\in S_2$ 及定义 4.2 可知:$R_{10}\oplus((R_{11}\odot R_{20})\otimes R_{21})\in S_3$.

所以,$R_1\odot R_2\in S_3$. □

定理 **5.3**. 假设谓词 Pd_1 和 Pd_2,(not Pd_1)和(not Pd_2)都是互斥的,若在规划求解过程中改变过它们的状态,则谓词 Pd_1 和 Pd_2 在任何规划状态中都只有一个为“真”.

证明:假设在规划求解时,其状态变化序列为 T_0(init),$T_1, T_2, \ldots, T_G$(goal).

不妨再设:从规划状态 T_u 变化到 T_{u+1} 时,谓词 Pd_1 或 Pd_2 的状态发生了变化,$u=0,\ldots,G-1$.

(1) $Pd_1, Pd_2\in T_u$

再设:$Pd_1\notin T_{u+1}$.

由“(not Pd_1)和(not Pd_2)是互斥的”和定义 5.4 可知:(not Pd_1) $_{\{Pd_1\}}\rightarrow Pd_2\in S_0$.

所以,在用动作删去谓词 Pd_1 时,谓词 Pd_2 一定会被添加到当前状态 T_u 中,使规划状态变成下一状态 T_{u+1}.

而 $Pd_2\in T_u$,这显然与规划领域动作的“添加效果”原义相矛盾.

(2) $Pd_1\notin T_u$,$Pd_2\notin T_u$

再设:$Pd_1\in T_{u+1}$.

由“Pd_1 和 Pd_2 是互斥的”和定义 5.4 可知:Pd_1 $_{\{(\text{not } Pd_1)\}}\rightarrow$(not Pd_2)$\in S_0$.

所以,在用动作添加谓词 Pd_1 时,谓词 Pd_2 一定从状态 T_u 中删去,使规划状态变成下一状态 T_{u+1}.

而 $Pd_2\notin T_u$,这显然与规划领域动作的“删除效果”原义相矛盾.

(3) 谓词 Pd_1 和 Pd_2 仅有一个在当前状态 T_u 之中

再设:$Pd_1\in T_u$,$Pd_2\notin T_u$.

(3.1) 执行规划动作,使得(not Pd_1)$\in T_{u+1}$

由“(not Pd_1)和(not Pd_2)是互斥的”和定义 5.4 可知:(not Pd_1) $_{\{Pd_1\}}\rightarrow Pd_2\in S_0$.

所以,在执行规划动作删去谓词 Pd_1 时,谓词 Pd_2 一定被添加,即 $Pd_1\notin T_{u+1}$,$Pd_2\in T_{u+1}$.

(3.2) 执行规划动作,使得 $Pd_2\in T_{u+1}$

由“Pd_1 和 Pd_2 是互斥的”和定义 5.4 可知:Pd_2 $_{\{(\text{not } Pd_2)\}}\rightarrow$(not Pd_1)$\in S_0$.

所以,在执行规划动作添加谓词 Pd_2 时,谓词 Pd_1 一定被删去,即 $Pd_1\notin T_{u+1}$,$Pd_2\in T_{u+1}$.

由情况(1)和情况(2)可知:不论哪种情况,若在用规划动作改变谓词 Pd_1 或 Pd_2 的状态时,都会得出矛盾之处.

由情况(3)可知:若谓词 Pd_1 和 Pd_2 仅有一个在当前规划状态中,则用规划动作改变谓词 Pd_1 或 Pd_2 的状态时,它们仍仅有一个在规划次态之中. □

B. **BlocksWorld**领域的描述

```
(define (domain BlocksWorld)
(:requirements    :strips)
(:predicates (clear ?x) (on-table ?x) (arm-empty) (holding ?x) (on ?x ?y))
(:action      Pickup
 :parameters     (?x)
 :precondition   (and (clear ?x) (on-table ?x) (arm-empty))
 :effect         (and (not (clear ?x)) (not (on-table ?x)) (not (arm-empty)) (holding ?x)))
(:action      Putdown
 :parameters     (?x)
 :precondition   (holding ?x)
 :effect         (and (not (holding ?x)) (clear ?x) (on-table ?x) (arm-empty)))
(:action      Unstack
 :parameters     (?x ?y)
 :precondition   (and (clear ?x) (on ?x ?y) (arm-empty))
 :effect         (and (not (clear ?x)) (not (on ?x ?y)) (not (arm-empty)) (holding ?x) (clear ?y)))
(:action      Stack
 :parameters     (?x ?y)
 :precondition   (and (clear ?y) (holding ?x))
 :effect         (and (not (clear ?y)) (not (holding ?x)) (clear ?x) (on ?x ?y) (arm-empty)))
)
```

C. **BlocksWorld**领域中**Sussman**奇异问题的描述

```
(define (problem Sussman-Anomaly)
      (:domain     Blocksworld)
      (:objects    A B C)
      (:init     (clear C) (on C A) (on-table A) (clear B) (on-table B) (arm-empty))
      (:goal     (and (on A B) (on B C)))
)
```

D. **BlocksWorld**领域的直接伴随关系集S_0

下面共有 22 个直接伴随关系,其中用 $Pd_1 \rightarrow Pd_2$ 表示 $Pd_1 \ _{\{(not\ Pd_1)\}} \rightarrow Pd_2$.

1. (on x y)→(not (holding x))
2. (on x y)→(not (clear y))
3. (on x y)→(clear x)
4. (on x y)→(arm-empty)
5. (not (on x y))→(not (clear x))
6. (not (on x y))→(not (arm-empty))
7. (not (on x y))→(holding x)
8. (not (on x y))→(clear y)
9. (on-table x)→(not (holding x))
10. (on-table x)→(clear x)
11. (on-table x)→(arm-empty)
12. (not (on-table x))→(not (clear x))
13. (not (on-table x))→(not (arm-empty))
14. (not (on-table x))→(holding x)
15. (arm-empty)→(not (holding x))
16. (arm-empty)→(clear x)
17. (not (arm-empty))→(not (clear x))
18. (not (arm-empty))→(holding x)
19. (holding x)→(not (clear x))
20. (holding x)→(not (arm-empty))
21. (not (holding x))→(clear x)
22. (not (holding x))→(arm-empty)

E. **BlocksWorld**领域的条件伴随关系集 S_0'

1. (not (arm-empty))$_{\{(\text{arm-empty}),(\text{on } x\ y)\}}$→(not (on x y))
2. (not (arm-empty))$_{\{(\text{arm-empty}),(\text{on } x\ y)\}}$→(clear y)
3. (not (arm-empty))$_{\{(\text{arm-empty}),(\text{on-table } x)\}}$→(not (on-table x))
4. (holding x)$_{\{(\text{not (holding } x)),(\text{on } x\ y)\}}$→(not (on x y))
5. (holding x)$_{\{(\text{not (holding } x)),(\text{on } x\ y)\}}$→(clear y)
6. (holding x)$_{\{(\text{not (holding } x)),(\text{on-table } x)\}}$→(not (on-table x))

F. **BlocksWorld**领域的直接阻碍关系集S_1

下面共有 11 个直接阻碍关系,其中,用(not Pd_1)–\–>Pd_2 表示(not Pd_1) $_{\{(\text{not } Pd_2)\}}$ –\–>Pd_2.

1. (not (holding x))–\–>(on x y)
2. (not (clear y))–\–>(on x y)
3. (not (clear x))–\–>(not (on x y))
4. (not (arm-empty))–\–>(not (on x y))
5. (not (holding x))–\–>(on-table x)
6. (not (clear x))–\–>(not (on-table x))
7. (not (arm-empty))–\–>(not (on-table x))
8. (not (holding x))–\–>(arm-empty)
9. (not (clear x))–\–>(not (arm-empty))
10. (not (clear x))–\–>(holding x)
11. (not (arm-empty))–\–>(holding x)

G. **BlocksWorld**领域的间接阻碍关系S_2

1. (not (clear x))$_{\{(\text{not (holding } x)),(\text{not (on } x\ y))\}}$–\\–>(on x y)
2. (not (clear x))$_{\{(\text{not (holding } x)),(\text{not (on-table } x))\}}$–\\–>(on-table x)
3. (not (arm-empty))$_{\{(\text{not (holding } x)),(\text{not (on } x\ y))\}}$–\\–>(on x y)
4. (not (arm-empty))$_{\{(\text{not (holding } x)),(\text{not (on-table } x))\}}$–\\–>(on-table x)
5. (not (holding z))$_{\{(\text{not (arm-empty)}),(\text{on } x\ y)\}}$–\\–>(not (on x y))
6. (not (holding y))$_{\{(\text{not (arm-empty)}),(\text{on-table } x)\}}$–\\–>(not (on-table x))
7. (not (holding y))$_{\{(\text{not (arm-empty)}),(\text{not (holding } x))\}}$–\\–>(holding x)
8. (not (holding z))$_{\{(\text{not (arm-empty)}),(\text{not (holding } x)),(\text{not (on } x\ y))\}}$–\\–>(on x y)
9. (not (holding y))$_{\{(\text{not (arm-empty)}),(\text{not (holding } x)),(\text{not (on-table } x))\}}$–\\–>(on-table x)

H. **BlocksWorld**领域的绝对阻碍关系S_3

1. (on x z)$_{\{(\text{not (on } x\ z)),(\text{holding } x),(\text{not (on } x\ y))\}}$=\\=>(on x y)
2. (on z y)$_{\{(\text{not (on } z\ y)),(\text{clear } y),(\text{not (on } x\ y))\}}$=\\=>(on x y)
3. (on z x)$_{\{(\text{not (on } z\ x)),(\text{clear } x),(\text{on } x\ y)\}}$=\\=>(not (on x y))
4. (on x y)$_{\{(\text{not (on } x\ y)),(\text{holding } x),(\text{not (on-table } x))\}}$=\\=>(on-table x)
5. (on y x)$_{\{(\text{not (on } y\ x)),(\text{clear } x),(\text{on-table } x)\}}$=\\=>(not (on-table x))
6. (on y x)$_{\{(\text{not (on } y\ x)),(\text{clear } x),(\text{not (holding } x))\}}$=\\=>(holding x)
7. (on z x)$_{\{(\text{not (on } z\ x)),(\text{clear } x),(\text{not (holding } x)),(\text{not (on } x\ y))\}}$=\\=>(on x y)
8. (on y x)$_{\{(\text{not (on } y\ x)),(\text{clear } x),(\text{not (holding } x)),(\text{not (on-table } x))\}}$=\\=>(on-table x)
9. (not (on y z))$_{\{(\text{on } y\ z),(\text{clear } y),(\text{not (on } x\ y))\}}$=\\=>(on x y)
10. (not (on z u))$_{\{(\text{on } z\ u),(\text{arm-empty}),(\text{on } x\ y)\}}$=\\=>(not (on x y))
11. (not (on y z))$_{\{(\text{on } y\ z),(\text{arm-empty}),(\text{on-table } x)\}}$=\\=>(not (on-table x))

12. (not (on *y z*))$_{\{(\text{on } y\ z),(\text{arm-empty}),(\text{not (holding } x))\}}$=\\=>(holding *x*)
13. (not (on *z u*))$_{\{(\text{on } z\ u),(\text{arm-empty}),(\text{not (holding } x)),(\text{not (on } x\ y))\}}$=\\=>(on *x y*)
14. (not (on *y z*))$_{\{(\text{on } y\ z),(\text{arm-empty}),(\text{not (holding } x)),(\text{not (on-table } x))\}}$=\\=>(on-table *x*)
15. (holding *y*)$_{\{(\text{not (holding } y)),(\text{clear } y),(\text{not (on } x\ y))\}}$=\\=>(on *x y*)
16. (holding *z*)$_{\{(\text{not (holding } z)),(\text{arm-empty}),(\text{on } x\ y)\}}$=\\=>(not (on *x y*))
17. (holding *y*)$_{\{(\text{not (holding } y)),(\text{arm-empty}),(\text{on-table } x)\}}$=\\=>(not (on-table *x*))
18. (holding *y*)$_{\{(\text{not (holding } y)),(\text{arm-empty}),(\text{not (holding } x))\}}$=\\=>(holding *x*)
19. (holding *z*)$_{\{(\text{not (holding } z)),(\text{arm-empty}),(\text{not (holding } x)),(\text{not (on } x\ y))\}}$=\\=>(on *x y*)
20. (holding *y*)$_{\{(\text{not (holding } y)),(\text{arm-empty}),(\text{not (holding } x)),(\text{not (on-table } x))\}}$=\\=>(on-table *x*)
21. (on-table *x*)$_{\{(\text{not (on-table } x)),(\text{holding } x),(\text{not (on } x\ y))\}}$=\\=>(on *x y*)
22. (not (on-table *y*))$_{\{(\text{on-table } y),(\text{clear } y),(\text{not (on } x\ y))\}}$=\\=>(on *x y*)
23. (not (on-table *z*))$_{\{(\text{on-table z}),(\text{arm-empty}),(\text{on } x\ y)\}}$=\\=>(not (on *x y*))
24. (not (on-table *y*))$_{\{(\text{on-table } y),(\text{arm-empty}),(\text{on-table } x)\}}$=\\=>(not (on-table *x*))
25. (not (on-table *y*))$_{\{(\text{on-table } y),(\text{arm-empty}),(\text{not (holding } x))\}}$=\\=>(holding *x*)
26. (not (on-table *z*))$_{\{(\text{on-table z}),(\text{arm-empty}),(\text{not (holding } x)),(\text{not (on } x\ y))\}}$=\\=>(on *x y*)
27. (not (on-table *y*))$_{\{(\text{on-table } y),(\text{arm-empty}),(\text{not (holding } x)),(\text{not (on-table } x))\}}$=\\=>(on-table *x*)
28. (arm-empty)$_{\{(\text{not (arm-empty)}),(\text{holding } x),(\text{not (on } x\ y))\}}$=\\=>(on *x y*)
29. (arm-empty)$_{\{(\text{not (arm-empty)}),(\text{holding } x),(\text{not (on-table } x))\}}$=\\=>(on-table *x*)
30. (not (arm-empty))$_{\{(\text{arm-empty}),(\text{clear } y),(\text{not (on } x\ y))\}}$=\\=>(on *x y*)

吴向军(1965－),男,副教授,主要研究领域为人工智能,算法设计,网络应用.

凌应标(1965－),男,博士,主要研究领域为智能规划,演化计算,智能优化.

姜云飞(1945－),男,教授,博士生导师,主要研究领域为自动推理,智能规划,基于模型诊断.