基于静态前提的谓词知识树分解策略

边 芮[1] 吴向军[2] 陈蔼祥[3]
(广东财经大学公共管理学院 广州 510320)[1] (中山大学软件学院 广州 510006)[2]
(广东财经大学数学与统计学院 广州 510320)[3]

摘 要 智能规划问题实质是一种搜索问题,通常需采用某种策略来缩小搜索空间,提高规划效率。在"以谓词为主体"的规划求解方法中,规划树的生成效率将直接影响规划求解效率。为此,提出了基于静态前提的谓词知识树分解策略,并给出了相应的分解算法。对任意一个规划领域,利用该分解算法可将知识树分解成若干个较小规模的知识子树。在规划求解的过程中,利用知识子树可有效地减少搜索空间,从而快速生成规划树,提高规划效率。同时,利用知识子树还可提取出隐含在动作描述中的领域知识。实验结果表明该分解算法是有效的。

关键词 智能规划,知识树,静态前提,分解策略
中图法分类号 TP182 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.01.044

Decomposition Strategy for Knowledge Tree of Predicate Based on Static Preconditions

BIAN Rui[1] WU Xiang-jun[2] CHEN Ai-xiang[3]
(School of Public Management, Guangdong University of Finance and Economics, Guangzhou 510320, China)[1]
(Software School, Sun Yat-sen University, Guangzhou 510006, China)[2]
(School of Mathematics and Statistics, Guangdong University of Finance and Economics, Guangzhou 510320, China)[3]

Abstract AI planning is essentially a search problem, usually using some strategies to reduce the search space, improving planning efficiency. In the planning method of taking the predicate as target, the efficiency of planning tree generation will affect planning efficiency directly. Therefore, this paper proposesd a decomposition strategy for knowledge tree of predicate based on the static preconditions, and gave the corresponding decomposition algorithm. For any domain, using the algorithm, knowledge tree of predicate can be decomposed into a number of smaller knowledge sub-trees. The use of knowledge sub-trees in planning process can reduce the search space effectively, thus generating plan tree quickly and improving the planning efficiency. At the same time, the domain knowledge is extracted from the domain with them. Finally, the experiment results show that decomposition algorithm is efficient.

Keywords AI planning, Knowledge tree, Static precondition, Decomposition strategy

智能规划(AI Planning)是人工智能领域的一个重要分支,其在机器人、企业生产调度等方面有着广泛的应用。它主要研究:如何在给定状态转换系统内搜索出一个可使系统从初始状态到达目标状态的动作序列。Erol等人[1]已经证明,经典规划问题的计算复杂性是EXPSPACE- Complete。虽然可对规划操作增加多种语法限制,以降低规划问题的复杂度,但与此同时也降低了相应领域模型的表达能力。

经过近20年的发展,智能规划系统的求解效率有了重大提高[2]。在国际规划大赛上,出现了一批高效的规划系统,例如FF,LPG,SATPLAN,SGPLAN,LAMA等。然而文献[3]表明,一般的搜索过程(例如A*算法)也必须搜索指数级数目的结点。因此,对于大规模规划问题,搜索空间约减是提高规划效率的重要方法。

目前,搜索空间约减的主要研究方向包括启发式搜索[4-7]、子目标排序[8-10]、规划领域分解、规划问题分解等。启发式搜索可以有效地指导规划解的搜索过程,基于启发式搜索的规划器在国际规划大赛上都有不俗的表现。子目标排序方法是对子目标进行排序,利用正确有效的排序关系指导规划求解,避免不必要的搜索过程,提高规划效率。规划领域分解方面的经典工作主要有层次规划方法[11,12]和分解规划方法[13,14],这两种方法分别是对规划空间进行层次分解和将原规划域分割成可重叠的子域,但由于在规划求解的过程中都需进行广泛回溯,因此通常情况下它们都不能有效处理大规模规划问题。文献[15,16]提出了一种规划领域分解方法,它们主要考虑规划领域中动作的相互依赖关系,对于规划求解过程中的任意规划状态,只需扩展可执行动作集的一个子集。该规划领域分解方法对于仅有一个层次的规划领域将失去效用。规划问题分解方面的代表工作有规划子目标划分[17-20],其主要策略是利用规划子目标之间的约束关系对规划问题中的子目标进行划分,并对每个子目标独立求解,最后对所有子

到稿日期:2016-08-09 返修日期:2016-10-16 本文受国家自然科学基金(71272084),广东高校优秀青年创新人才培养计划(育苗工程)项目(2012LYM_0065),广东财经大学自然科学研究项目(11BS52001)资助。

边 芮(1982-),女,博士,讲师,主要研究方向为知识为工程与应用、智能规划和调度;吴向军(1965-),男,博士,副教授,主要研究方向为人工智能、算法设计和网络应用,E-mail:issxjwu@mail.sysu.edu.cn;陈蔼祥(1978-),男,博士,副教授,主要研究方向为智能规划、统计机器学习。

目标的解进行组合和求精来获得原规划问题的规划解。启发式搜索和子目标排序方法也适用于分解后规划领域和规划问题的求解。但无论是基于规划领域分解的规划方法,还是基于规划问题分解的规划方法,它们都是试图对整个规划领域或者规划问题进行分解,然而实际应用中很多规划领域、规划问题的耦合度较高,不能满足上述方法的适用条件,使得这种基于分解的搜索空间约减方法失效。

本文研究一种局部规划领域分解策略,即基于静态前提的谓词知识树分解策略。在"以谓词为主体"的规划求解方法[21]中,谓词知识树存储实现同一谓词的所有动作,谓词规划树由谓词知识树递归构造而成,其生成效率将直接影响规划求解效率。文献[22]提出了基于特征谓词的谓词知识树分解策略。在研究中,我们发现另一类特殊的领域谓词(称为静态谓词),在任意规划问题求解过程中,其"真值"具有保持不变的特性,这使得此类领域谓词的真值将直接决定以其采用了前提条件的领域动作的可执行性。因此,利用静态谓词将谓词知识树分解成若干个较小规模的知识子树,在规划求解中的具体规划状态下,选用不同的知识子树来生成规划树,可避免一些不必要的动作搜索,从而提高规划求解效率。

本文第 1 节介绍谓词知识树和谓词规划树的概念;第 2 节介绍静态谓词和静态相关性,并提出谓词知识树分解原理;第 3 节介绍静态知识子树的性质;第 4 节介绍谓词知识树分解策略;第 5 节介绍谓词知识树分解算法,并选取 IPCs(International Planning Competitions)中的标准规划领域对算法的正确性和有效性进行验证;最后给出结论。

1 谓词知识树和谓词规划树

谓词知识树和谓词规划树[21]是"以谓词为主体"的规划求解方法中的两个重要概念。谓词知识树是一种特殊的树形结构,它存储了规划领域中实现同一谓词的所有领域动作。谓词规划树由在具体规划状态下递归地运用谓词知识树和置换构造而成。"以谓词为主体"的规划求解方法的思想是:在当前规划状态下,为目标状态中的每个子目标生成相应的谓词规划树,然后利用领域知识从所有规划树中选取下一个可实现的谓词,并据此更新当前规划状态,直至实现目标状态。

下面给出谓词知识树和谓词规划树的相关定义。

1.1 谓词知识树

对于 STRIPS 规划领域,文献[21]给出了谓词知识树的形式化定义,有关的符号和定义如下。

(1)用 $Para(Pd)$ 和 $Para(Act)$ 分别表示谓词 Pd 和动作 Act 的参数列表;

(2)用 $V(Para(Pd))$ 和 $V(Para(Act))$ 分别表示谓词 Pd 和动作 Act 参数列表中所有变量组成的集合;

(3)用 $PC(Act)$ 和 $E(Act)$ 分别表示动作 Act 的前提条件谓词集和动作效果谓词集;

(4)用 $Act(Pd)$ 表示可实现谓词 Pd 的动作集合,$CPC(Pd)$ 和 $CE(Pd)$ 表示动作集 $Act(Pd)$ 中所有动作的公共前提集合和公共效果集合,即:

$$CPC(Pd)=\bigcap_{a\in Act(Pd)}PC(a)$$

$$CE(Pd)=\bigcap_{a\in Act(Pd)}E(a)$$

定义 1 假设 Pd 是领域谓词,按照下面方法构造一个谓词-动作-谓词树 T:

(1)Pd 是树 T 的根结点;

(2)$\forall Act_i\in Act(Pd)$,动作 Act_i 是树 T 根结点的一个子结点,且 $PC(Act_i)$ 中的每个谓词都是动作 Act_i 的子结点,则称该谓词-动作-谓词树 T 为谓词 Pd 的知识树 KT(Knowledge Tree),称 $Act(Pd)$ 为谓词 Pd 知识树 KT 的动作集。

例如,在 Logistics 领域中,谓词(in x y)分别表示物体"x"在"y"里,根据定义 1,可构造其知识树,如图 1 所示。

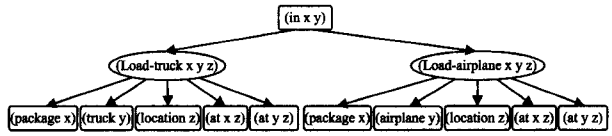


图 1 Logistics 领域中谓词(in x y)的知识树示意图

由图 1 不难看出:

$Act((\text{in } x\ y))=\{(\text{Load-truck } x\ y\ z),(\text{Load-airplane } x\ y\ z)\}$

$CPC((\text{in } x\ y))=\{(\text{package } x),(\text{location } z),(\text{at } x\ z),(\text{at } y\ z)\}$

$CE((\text{in } x\ y))=\{(\text{not } (\text{at } x\ z)),(\text{in } x\ y)\}$

定义 2 假设有谓词 Pd 和置换 σ,σPd 表示用置换 σ 对谓词 Pd 的参数列表进行置换后所得的谓词,即:

$Para(\sigma Pd)=\sigma Para(Pd)$

定义 3 假设有动作 Act 和置换 σ,σAct 表示用置换 σ 对动作 Act 进行置换后所得的动作。置换动作的具体规则如下:

(1)$Para(\sigma Act)=\sigma Para(Act)$;

(2)$\forall Pc\in PC(Act)$,有 $\sigma Pc\in PC(\sigma Act)$;

(3)$\forall Pd\in E(Act)$,有 $\sigma Pd\in E(\sigma Act)$。

定义 4 假设有动作集 A 和置换 σ,σA 表示用置换 σ 对动作集 A 中的所有动作进行置换后所得的动作集,即:

$\sigma A=\{\sigma Act_i\mid Act_i\in A\}$

定义 5 假设有谓词 Pd 的知识树 KT 和置换 σ,σKT 表示用置换 σ 对知识树 KT 进行置换后所得的 σPd 的知识树,即:

(1)σPd 是树 σKT 的根结点;

(2)$\forall Act_i\in Act(Pd)$,动作 σAct_i 是树 σKT 根结点的一个子结点,且 $PC(\sigma Act_i)$ 中的每个谓词都是 σAct_i 的子结点。

由定义 5 可知:谓词知识树是由领域谓词和可实现它的所有领域动作组成的。谓词知识树的结点中可能含有变量,在规划求解的过程中需通过置换对其进行实例化。例如,用置换 $\sigma=\{P/x,T/y\}$ 对图 1 所示的知识树进行置换可得谓词(in P T)的知识树。谓词(in P T)的知识树如图 2 所示。

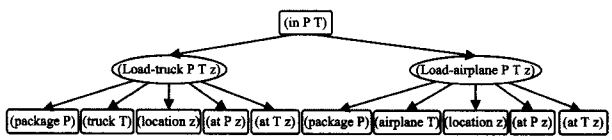


图 2 Logistics 领域中用置换 $\sigma=\{P/x,T/y\}$ 所得的谓词(in P T)的知识树示意图

因此,在规划求解过程中,需用不同的置换来获得不同的谓词知识树的实例化形式。例如:在规划求解的过程中需生成谓词(in P A)的知识树,则需用置换 $\sigma'=\{P/x,A/y\}$ 对该知识树进行实例化。

1.2 谓词规划树

谓词规划树通过递归地运用谓词知识树和置换构造而成，它可反映在当前规划状态下实现某个领域谓词的难易程度。文献[21]对谓词规划树及其树高给出了相应的定义。

假设：Pd 是领域谓词，S_i 是某个规划状态。若 $Pd \in S_i$，则称在状态 S_i 下谓词 Pd 为“真”，(not Pd)为“假”；否则，称在状态 S_i 下谓词 Pd 为“假”，(not Pd)为“真”。

定义 6 假设 Pd 是领域谓词，当前规划状态为 S_i，按照下面方法递归地构造一个谓词-动作-谓词树 T：

(1)谓词 Pd 是树 T 的根结点；

(2)若 $Pd \notin S_i$，则 T 的第一层是谓词 Pd 的知识树。对谓词 Pd 知识树的每个叶结点谓词 Pd_j 递归地构造谓词-动作-谓词树 T_j，称该谓词-动作-谓词树 T 为谓词 Pd 在状态 S_i 下的规划树 PT(Planning Tree)。

假设在 Logistics 领域的规划问题中有子目标谓词(in P T)，且当前规划状态为 S_i，如图 3 所示。

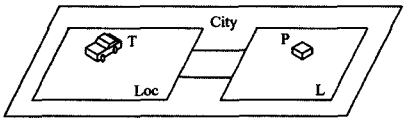


图 3　Logistics 领域中规划状态 S_i

在状态 S_i 下，用定义 6 可生成谓词(in P T)的规划树，如图 4 所示，其中虚线框表示谓词在状态 S_i 下为“真”。

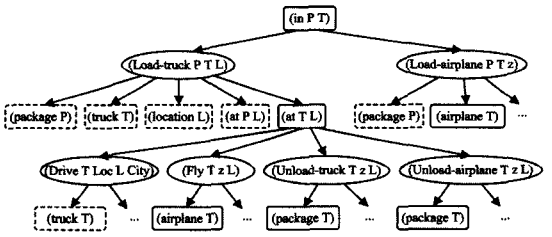


图 4　在状态 S_i 下谓词(in P T)的规划树示意图

“以谓词为主体”的规划求解方法用谓词规划树来指导规划求解，则规划树的生成效率将直接影响规划求解效率。由定义 6 可知，谓词规划树由谓词知识树递归构造而成，其规模由谓词知识树规模和规划状态决定。研究发现：对同一个谓词，在不同的规划状态下能根据特定的条件选用其知识树中某些动作，而无需考虑其余部分中的动作。因此，通过缩减谓词知识树规模来缩小搜索空间是提高规划效率的有效手段。

例如，如图 4 所示，谓词(package T)，(truck T)和(airplane T)分别为谓词(at T L)知识树中 4 个规划动作的前提条件，由于当前规划状态 S_i 下只有谓词(truck T)成立，且其他谓词不能由动作实现，则在该规划状态下可根据这 3 个谓词的真值选用谓词(at T L)知识树中的相应动作，而无需考虑其余部分中的动作。

为实现上述基本策略，本文提出基于静态前提的谓词知识树分解策略。对任意一棵谓词知识树，运用其动作的静态前提作为划分标准，将该知识树分解成若干个知识子树。在具体规划状态下，根据该“静态前提”来选用不同的知识子树生成谓词规划树，以避免遍历原知识树中的所有动作结点，从而达到提高规划树生成效率的目的。

2 谓词知识树分解原理

下面介绍静态谓词和将“静态前提”作为划分标准的谓词知识树分解原理。

2.1 静态谓词

在文献[21]中，通过分析谓词知识树的组成将领域谓词分为静态谓词、非递增谓词、非递减谓词和动态谓词等，其中静态谓词的“真值”在任意规划问题求解规划过程中是不可改变的。

定义 7 假设领域谓词 Pd 和(not Pd)的知识树分别为 KT_1 和 KT_2，若知识树 KT_1 和 KT_2 都仅含根结点，则称谓词 Pd 为静态谓词(Static Predicate)，否则称谓词 Pd 为动态谓词(Dynamic Predicate)。

静态谓词 Pd 的正负谓词知识树都仅含根结点，这表明在该规划领域中没有任何动作可改变谓词 Pd 的“真值”，因此，在该规划领域的任意规划问题求解过程中，静态谓词 Pd 的“真值”具有保持不变的特性。

例如，在规划领域 Logistics 中，其谓词(package x)和(not (package x))的知识树(见图 5)都仅含根结点，即：谓词(package x)不会被任何规划动作增加或删除。所以，在 Logistics 领域的任意规划问题求解过程中，其谓词(package x)的“真值”是保持不变的。

(a)　(b)

图 5　谓词(package x)和(not (package x))的知识树示意图

在其它规划领域中也存在类似的谓词，如 Monkey 规划领域中的谓词(location x)、Hanoi 规划领域中的谓词(smaller x y)、Gripper 领域中的谓词(room x)等。

2.2 静态相关性

由于静态谓词在任意规划问题求解过程中是不受规划动作影响的，因此其变量参数只能通过谓词的模式匹配来取值，称这种特性为动作和变量参数的静态相关性。

定义 8 若 $Pc \in PC(Act)$，且 Pc 是一个静态谓词，则称 Pc 为规划动作 Act 的静态前提。

定义 9 假设 Act 为一个规划动作，任给一个变量 x，用 $PC(Act,x)$表示动作 Act 与变量 x 相关的静态前提集，即：

$PC(Act,x)=\{Pc \mid Pc$ 是动作 Act 的静态前提，且 $x \in V(Para(Pc))\}$

(1) 若 $PC(Act,x)\neq\emptyset$，则称动作 Act 与变量 x 是静态相关的；

(2) 若 $PC(Act,x)=\emptyset$，则称动作 Act 与变量 x 是静态无关的。

定义 9 描述了动作 Act 和变量 x 之间的静态相关性。当 $PC(Act,x)\neq\emptyset$ 时，变量 x 的取值只能通过 $PC(Act,x)$ 中的静态谓词进行模式匹配来获得。

例如，在规划领域 Logistics 中，动作 Drive 是描述卡车“?t”从同一城市“?c”的地点“?s”开到不同地点“?d”的含义，即卡车只能在同一个城市的两个地点间进行移动。其动作的具体定义如下：

```
(:action Drive
:parameters (?t ?s ?d ?c)
:precondition (and (truck ?t) (at ?t ?s) (in-city ?s ?c)
(in-city ?d ?c) (not (= ?s ?d)))
:effect (and (at ?t ?d) (not (at ?t ?s))))
```

由动作 Drive 的定义可知：

$V(Para(\text{Drive}))=\{?t,?s,?d,?c\}$。

由规划领域分析和定义 8 可知：谓词(truck ?t)、(in-city ?s ?c)和(in-city ?d ?c)是动作 Drive 的静态前提。

为了说明动作 Drive 与任意变量之间的静态相关性，除动作 Drive 参数列表中的所有变量外，还引入了一个不是其参数列表中的变量"?r"，并用该变量来代表不属于动作 Drive 的任意其他变量。对于动作 Drive，根据定义 9 可计算 $PC(\text{Drive},x)$，$x\in V(Para(\text{Drive}))\cup\{?r\}$。有关结果如表 1 所列。

表 1 动作 Drive 与变量 x 相关的静态前提集

x	PC(Drive,x)
?t	{(truck ?t)}
?s	{(in-city ?s ?c)}
?d	{(in-city ?d ?c)}
?c	{(in-city ?s ?c),(in-city ?d ?c)}
?r	∅

由表 1 可知：

(1)若 $x\notin V(Para(\text{Drive}))$，则有 $PC(\text{Drive},x)=\emptyset$。

例如，当 $x=?r$ 时，有 $PC(\text{Drive},r)=\emptyset$。

(2)若 $x\in V(Para(\text{Drive}))$，且 $PC(\text{Drive},x)\neq\emptyset$。

这时，变量 x 的取值只能通过 $PC(\text{Drive},x)$ 中的静态谓词进行模式匹配来获得。

例如，当 $x=?t$ 时，有 $PC(\text{Drive},x)=\{(\text{truck }?t)\}$，则变量 x 的取值只能通过静态谓词(truck ?t)的变量"?t"进行模式匹配来获得，即它只能取卡车的符号名。

同理，变量"?s"、"?d"和"?c"的取值范围也可由相应的静态谓词来确定。

由此，对于任意动作 Act 和变量 x，可得到以下结论：

(1)若 $x\notin V(Para(Act))$，则有 $PC(Act,x)=\emptyset$，即动作 Act 与变量 x 是静态无关的；

(2)若 $x\in V(Para(Act))$，且 $PC(Act,x)=\emptyset$，则动作 Act 与变量 x 是静态无关的；

(3)若 $x\in V(Para(Act))$，且 $PC(Act,x)\neq\emptyset$，则变量 x 的取值只能通过 $PC(Act,x)$ 中的静态谓词进行模式匹配来获得。这样可缩小动作参数表的取值空间，减小动作的搜索空间。

2.3 静态知识子树

谓词知识树分解原理的基本思想：对任意谓词 Pd 的知识树，利用 $Act(Pd)$ 中所有动作与谓词 Pd 参数变量之间的静态相关性，将其动作集划分成若干个独立的动作子集，每个动作子集将构成该谓词的一棵知识子树。

定义 10 假设谓词 Pd 知识树 KT 的动作集为 A，变量 $x\in V(Para(Pd))$，对非空动作集 $A'\subseteq A$，有：

(1) 若 $\bigcap\limits_{Act\in A'}PC(Act,x)\neq\emptyset$，则动作集 A' 是动作集 A 关于变量 x 的静态相关划分块，且 $\forall Pc\in\bigcap\limits_{Act\in A'}PC(Act,x)$，$Pc$ 都是 A' 的静态前提；

(2)若 $\bigcup\limits_{Act\in A'}PC(Act,x)=\emptyset$，则动作集 A' 是动作集 A 关于变量 x 的静态无关划分块，且用永真谓词"TURE"作为 A' 的静态前提。

定义 11 假设谓词 Pd 知识树 KT 的动作集为 A，变量 $x\in V(Para(Pd))$，有 k 个关于变量 x 的划分块 $A_1,A_2,\cdots,A_k$，若满足下列条件：

(1) $\bigcup\limits_{r=1,\cdots,k}A_r=A$，$A_i\cap A_j=\emptyset$，$i,j(\{1,2,\cdots,k\},i\neq j$；

(2)对 $A_1,A_2,\cdots,A_k$，分别存在其静态前提 $Pc_1,Pc_2,\cdots,Pc_k$，且 $Pc_i\neq Pc_j$，$i,j\in\{1,2,\cdots,k\}$，$i\neq j$；

则称划分块 $A_1,A_2,\cdots,A_k$ 为动作集 A 的一个划分，Pc_i 是 A_i 的划分条件。

定义 11 的含义说明如下：

(1)条件(1)说明谓词知识树 KT 中的每个动作"属于"且"仅属于"一个划分块；

(2)条件(2)说明划分中的每个划分块都有区别于其他划分块的静态前提。

性质 1 假设谓词 Pd 知识树 KT 的动作集为 A，则动作集 A 的划分中至多存在一个静态无关划分块。

证明：

假设：动作集 A 的划分中存在两个静态无关划分块 A_i 和 A_j，Pc_i 和 Pc_j 分别为 A_i 和 A_j 的静态前提。

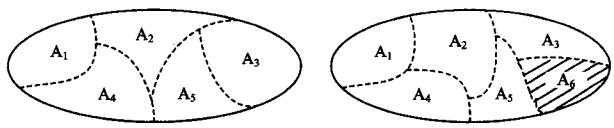
由"A_i 和 A_j 是静态无关划分块"和定义 10 可知：$Pc_i=$TURE，$Pc_j=$TURE，这与定义 11 中的条件"$Pc_i\neq Pc_j$"相矛盾。因此动作集 A 的划分中至多存在一个静态无关划分块。

定义 12 假设谓词 Pd 知识树 KT 的动作集为 A，划分块 $A_1,A_2,\cdots,A_k$ 为动作集 A 的一个划分，若任意划分块 A_i ($i=1,\cdots,k$)都是静态相关划分块，则称该划分是完全静态划分，否则称之为非完全静态划分。

定义 12 的含义说明如下：

(1)若划分中的所有划分块都是静态相关划分块，则表示利用静态相关性可将动作集 A 中的所有动作分类到划分块 $A_1,A_2,\cdots,A_k$ 中，因此称之为完全静态划分，如图 6(a)所示。

(2)若划分中存在一个静态无关划分块 A_j，则表示不能用静态相关性对动作集 A_j 中的动作进行分类，因此称之为非完全静态划分，如图 6(b)所示。



(a)完全静态划分 (b)非完全静态划分

图 6 动作集 A 的划分示意图

在图 6(a)中，划分块 $A_1,A_2,\cdots,A_5$ 构成动作集 A 的一个完全静态划分。在图 6(b)中，划分块 $A_1,A_2,\cdots,A_6$ 构成动作集 A 一个非完全静态划分，其中阴影划分块 A_6 是一个静态无关划分块，其划分条件为 TURE。

定义 13 假设谓词 Pd 知识树 KT 中的动作集为 A，划分块 $A_1,A_2,\cdots,A_k$ 是 A 的一个划分，Pc_i 是 A_i 的划分条件，则称$\langle Pc_i,A_i\rangle$为谓词 Pd 的静态知识子树，称谓词 Pc_i 为静态知识子树$\langle Pc_i,A_i\rangle$的选择条件，$i=1,\cdots,k$。

定义 14 假设有谓词 Pd 的静态知识子树$\langle Pc_i,A_i\rangle$和置换 σ。用 $\sigma\langle Pc_i,A_i\rangle$表示用置换 σ 对该知识子树进行置换后所得的 σPd 的知识子树，且有 $\sigma\langle Pc_i,A_i\rangle=\langle\sigma Pc_i,\sigma A_i\rangle$。

对于谓词 Pd 知识树 KT 的动作集 A 及其静态划分的知识子树，可得出下列结论：

(1)根据不同参数变量的静态相关性可得到动作集 A 的不同划分，在这些划分中可能存在完全静态划分和非完全静态划分。显然，完全静态划分是一种"彻底"的静态分类方式。

(2)利用动作集 A 的划分对谓词知识树进行分解，可将原知识树分解成若干个规模较小的知识子树，且每个知识子

树中的动作都拥有公共的静态前提。

由此可知，对于任意一个谓词知识树，在利用静态相关性得到其动作集的多个划分情况下，静态知识子树的生成策略是：若存在其动作集的完全静态划分，则用划分块最多的完全静态划分来构造静态知识子树；否则用划分块最多的非完全静态划分来构造静态知识子树。

3　静态知识子树的性质

下面研究静态知识子树的性质。

引理 1　假设谓词 Pd 知识树可得 k 个静态知识子树 $\langle Pc_j, A_j\rangle (j=1,\cdots,k)$。在规划状态 S 下有待实现的谓词 Pd_u，且存在置换 σ，使得：$Pd_u=\sigma Pd$。若 $Pc_v\in\{Pc_1, Pc_2,\cdots, Pc_k\}$，且 $\sigma Pc_v\notin S$，则：

(1)在状态 S 下，谓词 Pd_u 不能用静态知识子树 $\sigma\langle Pc_v, A_v\rangle$ 中的动作实现；

(2)在状态 S 的任意后继状态 S' 下，谓词 Pd_u 也不能用静态知识子树 $\sigma\langle Pc_v, A_v\rangle$ 中的动作实现。

证明：

用置换 σ 对谓词 Pd 的静态知识子树 $\langle Pc_v, A_v\rangle$ 进行置换 $\sigma\langle Pc_v, A_v\rangle=\langle\sigma Pc_v, \sigma A_v\rangle$。由静态知识子树定义可知：静态谓词 Pc_v 是静态知识子树 $\langle Pc_v, A_v\rangle$ 中所有动作的公共静态前提条件。所以谓词 σPc_v 是静态谓词，且为 $\langle\sigma Pc_v, \sigma A_v\rangle$ 中所有动作的公共静态前提条件。

(1)在状态 S 下

由已知条件“$\sigma Pc_v\notin S$”可知：在状态 S 下，$\langle\sigma Pc_v, \sigma A_v\rangle$ 中所有动作都存在不满足的前提条件。

所以，在状态 S 下，$\langle\sigma Pc_v, \sigma A_v\rangle$ 中所有动作都不能执行，即 Pd_u 不能用知识子树 $\sigma\langle Pc_v, A_v\rangle$ 中的动作来实现。

(2)在任意后继状态 S' 下

由“σPc_v 是静态谓词”可知：在规划求解过程中，谓词 σPc_v 的真值保持不变；

由已知条件“$\sigma Pc_v\notin S$”和“σPc_v 的真值保持不变”可知：$\sigma Pc_v\notin S'$；

由“$\sigma Pc_v\notin S'$”和情况(1)可知：状态 S' 下，$\langle\sigma Pc_v, \sigma A_v\rangle$ 中所有动作都不能执行，即 Pd_u 不能用知识子树 $\sigma\langle Pc_v, A_v\rangle$ 中的动作实现。

因此，在状态 S 及任意后继状态 S' 下，谓词 Pd_u 不能用静态知识子树 $\sigma\langle Pc_v, A_v\rangle$ 中的动作实现。

引理 1 说明：在具体规划状态下，利用静态知识子树生成谓词规划树时，只需考虑选择条件为“真”的静态知识子树，而无需考虑选择条件为“假”的静态知识子树。

定理 1　假设谓词 Pd 知识树可得 k 个静态知识子树 $\langle Pc_j, A_j\rangle$，且 $Pc_j\neq$TRUE，$j=1,\cdots,k$。在规划状态 S 下有待实现的谓词为 Pd_u，且存在置换 σ，使得：$Pd_u=\sigma Pd$。若 $\sigma Pc_j\notin S, j=1,\cdots,k$，则在状态 S 及任意后继状态 S' 下，谓词 Pd_u 一定是不可实现的。

证明：

由“$Pd_u=\sigma Pd$”可知：谓词 Pd_u 只能由谓词 σPd 的知识树中的动作来实现。

由已知条件可知：谓词 σPd 的知识树由 k 个静态知识子树 $\sigma\langle Pc_j, A_j\rangle$ 所构成，$j=1,\cdots,k$。

所以，谓词 Pd_u 只能由谓词 σPd 的 k 个静态知识子树 $\sigma\langle Pc_j, A_j\rangle$ 中的动作来实现。

用置换 σ 对谓词 Pd 的所有知识子树进行置换，得到谓词 σPd 的所有知识子树：$\langle\sigma Pc_j, \sigma A_j\rangle, j=1,\cdots,k$。

由已知条件“$\sigma Pc_j\notin S, j=1,\cdots,k$”和引理 1 可知：在状态 S 及任意后继状态 S' 下，谓词 Pd_u 都无法用静态知识子树 $\sigma\langle Pc_j, A_j\rangle$ 中的动作实现，$j=1,\cdots,k$。

所以，在状态 S 及任意后继状态 S' 下，谓词 Pd_u 都一定是不可实现的。

定理 2　假设谓词 Pd 知识树可得 k 个静态知识子树 $\langle Pc_j, A_j\rangle$，且 $Pc_j\neq$TRUE，$j=1,\cdots,k$。若 $\exists Pd_u\in S_G-S_0$，$Pd_u=\sigma Pd$，且 $\sigma Pc_j\notin S_0, j=1,\cdots,k$，则目标状态 S_G 一定是不可实现的。

证明：

由“$Pd_u\in S_G-S_0$”可知：$Pd_u\in S_G$，且 $Pd_u\notin S_0$。

所以，在规划求解过程中，谓词 Pd_u 需要用动作来实现。

由已知条件“$Pd_u=\sigma Pd$”可知：谓词 Pd_u 只能由谓词 σPd 的知识子树中的动作来实现。

由已知条件“$\sigma Pc_j\notin S_0, j=1,\cdots,k$”和定理 1 可知：在状态 S_0 及任意后继状态 S' 下，谓词 Pd_u 一定是不可实现的。

所以，目标状态 S_G 中的谓词 Pd_u 是不可实现的。

因此，目标状态 S_G 一定是不可实现的。

定理 1 说明：在规划状态 S 下，若谓词 Pd_u 的所有静态知识子树的选择条件都不成立，则在状态 S 及任意后继状态 S' 下，谓词 Pd_u 一定是不可实现的。

定理 2 说明：在初始状态 S_0 下，若谓词 Pd_u 的所有静态知识子树的选择条件都不成立，则含有谓词 Pd_u 的目标状态一定是不可实现的。

4　谓词知识树分解策略

第 2 节介绍了利用静态相关性对谓词知识树进行分解的原理，下面研究谓词知识树分解策略。

为便于叙述，现给出谓词知识树分解过程中所用符号及其含义。假设领域谓词 Pd 的知识树 KT 中有 $k(k>1)$ 个动作 $Act_1, Act_2,\cdots, Act_k$。

(1)用符号 $R(x)$ 表示知识树 KT 中所有与变量 x 静态相关的动作集合。

$$R(x)=\{Act_i \mid PC(Act_i, x)\neq\emptyset, i=1,\cdots,k\} \tag{1}$$

(2)用符号 $PCs(x)$ 表示动作集 $R(x)$ 中所有动作的与变量 x 相关的静态前提集。

$$PCs(x)=\bigcup_{Acti\in R(x)} PC(Act_i, x) \tag{2}$$

(3)二维表 $T(x)=(t_{ij})_{p\times q}$，其中 t_{ij} 表示动作 $Act_i\in R(x)$ 和静态谓词 $Pc_j\in PCs(x)$ 之间的关系，$p=|R(x)|$，$q=|PCs(x)|$。

(4) 用 $S(j)$ 表示二维表 $T(x)$ 第 j 列中所有取值为“1”的动作集，$j\in\{1,2,\cdots,q\}$。

$$S(j)=\{Act_i \mid t_{ij}=1, i=1,\cdots,p\} \tag{4}$$

假设有领域谓词 $Pd, V(Para(Pd))=\{x_1, x_2,\cdots, x_n\}$，其知识树 KT 动作集 $Act(Pd)=\{Act_1, Act_2,\cdots, Act_k\}$，如图 7 所示。

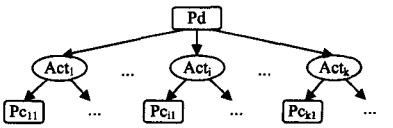


图 7　谓词 Pd 的知识树示意图

根据定义 9 计算谓词 Pd 的静态前提集 $PC(Act_i, x_j)$，$i=1,\cdots,k, j=1,\cdots,n$，其计算结果如表 2 所列。

表 2 动作 $Act \in Act(Pd)$ 与谓词 Pd 参数变量 x 相关的静态前提集

Act \ x	x_1	x_2	…	x_n
Act_1	$PC(Act_1, x_1)$	$PC(Act_1, x_2)$	…	$PC(Act_1, x_n)$
Act_2	$PC(Act_2, x_1)$	$PC(Act_2, x_2)$	…	$PC(Act_2, x_n)$
…	…	…	…	…
Act_k	$PC(Act_k, x_1)$	$PC(Act_k, x_2)$	…	$PC(Act_k, x_n)$

下面选用表 2 中第 i 列变量 x_i 来分析谓词知识树分解策略，对其他变量的分析是一致的。

对于第 i 列变量 x_i，由表 2 可计算其动作集 $R(x_i)$ 和静态前提集 $PCs(x_i)$。不妨假设：$R(x_i)=\{Act_{u1}, Act_{u2}, \cdots, Act_{up}\}$，$PCs(x_i)=\{Pc_1, Pc_2, \cdots, Pc_q\}$。

根据式(3)可得关系表 $T(x_i)$，如表 3 所列。

表 3 $R(x_i)$ 和 $PCs(x_i)$ 之间的关系表 $T(x_i)$

Act′ \ Pc	Pc_1	Pc_2	…	Pc_q
Act_{u_1}	t_{11}	t_{12}	…	t_{1q}
Act_{u_2}	t_{21}	t_{22}	…	t_{2q}
…	…	…	…	…
Act_{u_p}	t_{p1}	t_{p2}	…	t_{pq}

由表 3 中的信息可计算 $S(j), j\in\{1,2,\cdots,q\}$。根据定义 10 可知：$S(j)$ 为动作集 $Act(Pd)$ 关于变量 x_i 的一个划分块，其静态前提为 Pc_j。

例如在 Logistics 领域中，谓词(at x y)表示物体“x”在“y”处，根据定义 1，可构造其知识树，如图 8 所示。其中，该知识树的动作集 $Act((\text{at } x\ y))=\{$(Drive x z y w)，(Fly x z y)，(Unload-truck x z y)，(Unload-airplane x z y)$\}$。

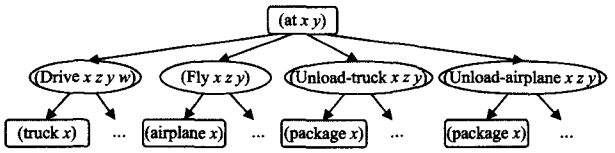


图 8 Logistics 领域中谓词(at x y)的知识树示意图

由 $V(Para((\text{at } x\ y)))=\{?x, ?y\}$，根据定义 9 可计算 $PC(Act_i, x_j)$，其中 $Act_i \in Act((\text{at } x\ y))$，$x_j \in V(Para((\text{at } x\ y)))$，结果如表 4 所列。

表 4 动作 $Act_i \in Act((\text{at } x\ y))$ 与谓词(at x y)参数变量 x_j 相关的静态前提集

Act_i \ x_j	x	y
(Drive x z y w)	{(truck x)}	…
(Fly x z y)	{(airplane x)}	…
(Unload-truck x z y)	{(package x)}	…
(Unload-airplane x z y)	{(package x)}	…

谓词(at x y)的参数变量“x”可以取值包裹、卡车和飞机等，由表 4 可得下列信息：

$R(x)=\{$(Drive x z y w)，(Fly x z y)，(Unload-truck x z y)，(Unload-airplane x z y)$\}$

$PCs(x)=\{$(truck x)，(airplane x)，(package x)$\}$

其中，$p=|R(x)|=4$，$q=|PCs(x)|=3$。

根据式(3)可得二维表 $T(x)$，如表 5 所列。

表 5 $R(x)$ 和 $PCs(x)$ 之间的关系表 $T(x)$

Act′ \ Pc	(truck x)	(airplane x)	(package x)
(Drive x z y w)	1	0	0
(Fly x z y)	0	1	0
(Unload-truck x z y)	0	0	1
(Unload-airplane x z y)	0	0	1

由表 5 中的信息可得每一列的动作集：

$S(1)=\{$(Drive x z y w)$\}$

$S(2)=\{$(Fly x z y)$\}$

$S(3)=\{$(Unload-truck x z y)，(Unload-airplane x z y)$\}$

显然，$S(1)-S(3)$ 都是 $Act((\text{at } x\ y))$ 关于参数变量“x”的划分块，其静态前提分别为(truck x)，(airplane x)和(package x)。

下面介绍利用关系表 $T(x_i)$ 分解谓词知识树的策略。

(1)将谓词知识树分解成一个静态无关的知识子树和一个静态相关的谓词知识树。

对于谓词知识树 KT 中的所有动作，利用它们与变量 x_i 的静态相关性可将其分成二部分：与变量 x_i 静态相关的部分(见图 9(a))、与变量 x_i 静态无关的部分(见图 9(b))。

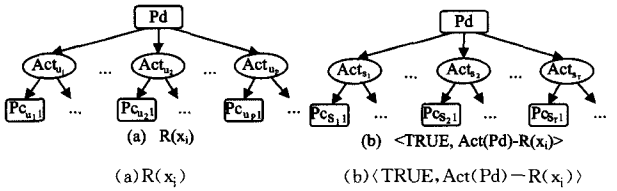


图 9 用动作与变量 x_i 的静态相关性分解谓词 Pd 知识树的示意图

若 $R(x_i)=\emptyset$，则谓词知识树 KT 的所有动作都与变量 x_i 静态无关，即图 9(a)为“空树”，图 9(b)与原知识树 KT(见图 7)是一样的。因此，知识树中的所有动作都不能利用与变量 x_i 的静态相关性来划分。

(2)对静态相关的谓词知识树进行分解。

根据式(3)可构造动作集 $R(x_i)$ 和静态前提集 $PCs(x_i)$ 之间的关系表 $T(x_i)$，如表 3 所列。

根据表 3 可计算 $|S(j)|, j=1,\cdots,q$，不妨假设 $|S(m)|=\min(|S(1)|, |S(2)|, \cdots, |S(q)|)$，$m\in\{1,2,\cdots,q\}$，$Pc_m$ 是其静态前提。

用静态前提 Pc_m 可将动作集 $R(x_i)-S(m)$ 中的动作和 $S(m)$ 中的动作区分开来，即将 $R(x_i)$ 所对应的知识树(见图 9(a))分出一个静态知识子树(见图 10(a))和剩余的谓词知识树(见图 10(b))。

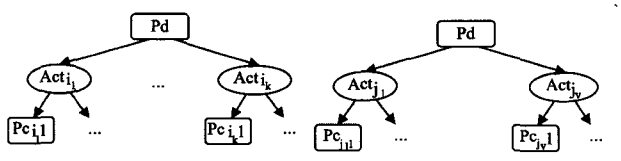


图 10 用静态前提 Pc_m 分解后的知识子树示意图

由于 $|S(m)|=\min(|S(1)|, |S(2)|, \cdots, |S(q)|)$，因此 $S(m)$ 是关于变量 x_i 的规模最小的划分块，即知识子树〈Pc_m，$S(m)$〉所含的动作个数最小。这样分解所得的知识子树对具体的规划状态将更有针对性。

重复情况(2)，继续对剩余的动作集 $R(x_i)-S(m)$ 所对应的知识树(见图 10(b))进行分解，直至树空为止。

例如，在 Logistics 领域中，谓词(at x y)的知识树如图 8

所示，其与参数变量“x”静态相关的动作集合为：

$R(x)=\{$(Drive x z y w)，(Fly x z y)，(Unload-truck x z y)，(Unload-airplane x z y)$\}$。

由表 5 可得关于参数变量“x”的 3 个划分块：

$S(1)=\{$(Drive x z y w)$\}$

$S(2)=\{$(Fly x z y)$\}$

$S(3)=\{$(Unload-truck x z y)，(Unload-airplane x z y)$\}$

根据谓词知识树的分解方法可知：

(1)用静态前提(truck x)将动作集 $A_1=S(1)$从 $R(x)$中分解出来，所得知识子树如图 11(a)所示；

(2)用静态前提(airplane x)将动作集 $A_2=S(2)$从$R(x)-S(1)$中分解出来，所得知识子树如图 11(b)所示；

(3)用静态前提(package x)将动作集 $A_3=S(3)$从 $R(x)-S(1)-S(2)$中分解出来，所得知识子树如图 11(c)所示。

对于谓词(at x y)知识树，按上述知识树的分解策略可得到下面 3 个静态知识子树，如图 11 所示。

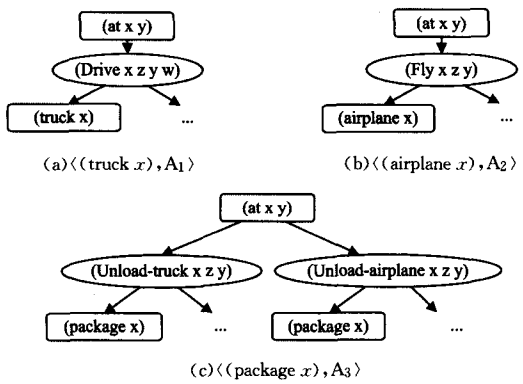


图 11　分解谓词(at x y)知识树所得的知识子树示意图

同理，谓词(in x y)的知识树(见图 1)可分解成⟨(truck y)，{(Load-truck x y z)}⟩和⟨(airplane y)，{(Load-airplane x y z)}⟩两个静态知识子树。

假设在图 3 所示的规划状态 S_i 下生成谓词(in P T)的规划树。谓词(in x y)有两个静态知识子树⟨(truck y)，{(Load-truck x y z)}⟩和⟨(airplane y)，{(Load-airplane x y z)}⟩。

存在 $\sigma=\{P/x, T/y, L/z\}$，使得(in P T)$=\sigma$(in x y)，且有：σ(truck y)=(truck T)$\in S_i$，σ(airplane y)=(airplane T)$\notin S_i$。

由引理 1 可知：在状态 S_i 下生成(in P T)的规划树时，只需考虑知识子树 σ⟨(truck y)，{(Load-truck x y z)}⟩。

同理，谓词(at x y)有 3 个静态知识子树(见图 11)，在状态 S_i 下只需考虑其知识子树(见图 11(a))。

因此，在状态 S_i 下谓词(in P T)规划树如图 12 所示。

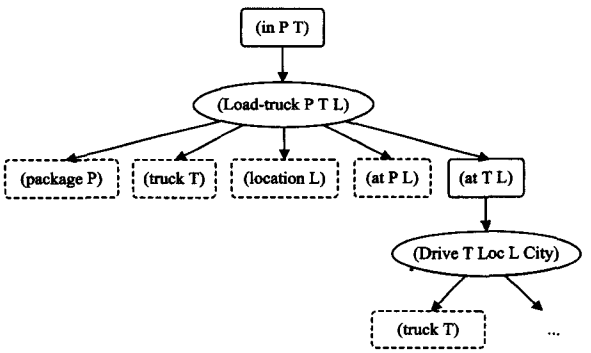


图 12　运用知识子树的谓词(in P T)规划树示意图

对比图 4 和图 12 可见：在生成谓词规划树时，根据静态知识子树“选择条件”的真值来选择相应的知识子树，可有效地缩小搜索空间，提高谓词规划树的生成效率。

5　谓词知识树分解算法

由第 2 节的谓词知识树分解原理可知：静态谓词及静态相关性是从规划领域描述中自动分析出来的，而非人为添加的，所以基于静态前提的知识树分解策略是“与规划领域无关”的。

根据第 4 节谓词知识树分解策略的思想，具体知识树分解算法如下。

算法 1　谓词知识树分解算法 KTD(Knowledge Tree Decomposition)

输入：谓词 Pd 的知识树 KT，其含有的 k 个动作 Act_1，Act_2，…，Act_k

输出：谓词 Pd 的静态知识子树集

```
KTD(KT)
{
1. CSP←Ø,NCSP ←Ø;
2. A←Act(Pd);
3. for each x∈V(Para(Pd)) do {
4.   P←GetPartition(A,x);
// 生成关于变量 x 的静态划分
5.   if (P is complete) {
6.   if (|CSP|<|P|) CSP←P;//划分块最多的完全静态划分
7.   }
8.   else if (|NCSP|<|P|) NCSP←P;//划分块最多的非完全静态划分
9. }
10. if (CSP≠Ø) return GetSKT(CSP);
   // 优先选择完全静态划分生成静态知识子树集
   // 函数 GetSKT 根据定义 13 构造静态知识子树集
11. return GetSKT(NCSP);
}
```

算法 2　静态划分生成算法 GetPartition

输入：谓词 Pd 知识树的动作集 A，参数变量 x

输出：关于变量 x 的静态划分 P

```
GetPartition(A,x)
{
1. B←R(x);
2. if (B==Ø) then return {A};
3. P=Ø;
4. if (A!=B)P←{A−B};
// 动作集合 A 是非完全静态划分
5. while (B!=Ø){
6.   m←GetMIN(B);//计算|S(m)|=min(|S(1)|,|S(2)|,…,|S(q)|)
7.   P←P∪{S(m)};
8.   B←B−S(m);
9. }
10. return P;
}
```

假设规划领域中含有 v 个静态谓词 Pd_1，Pd_2，…，Pd_v，t 为该领域所有静态谓词参数列表中变量个数的最大值，s 为规划树 KT 所有动作前提条件中谓词个数的最大值，即：

$$t=\max(|V(Para(Pd_1))|,|V(Para(Pd_2))|,\cdots,|V(Para(Pd_v))|)$$

$s=\max(|PC(Act_1)|,|PC(Act_2)|,\cdots,|PC(Act_k)|)$

n 为规划树 KT 所有动作前提条件中静态谓词个数的最大值，显然有 $n<s$。

在算法 GetPartition 中，语句(1)的时间复杂度为 $O(ks+knt)$，函数 GetMIN 的时间复杂度为 $O(k^2n)$，循环(5—9)的时间复杂度为 $O(k^3n)$，其他语句均可在常数时间内完成，因此算法 GetPartition 的时间复杂度为 $O(ks+knt+k^3n)$。

在算法 KTD 中，语句(4)的时间复杂度为 $O(ks+knt+k^3n)$，语句(5—8)的时间复杂度为 $O(k)$，循环(3—9)的时间复杂度为 $O((ks+knt+k^3n+k)\times r)$，其中 $r=|V(Para(Pd))|$，语句(10—11)的时间复杂度为 $O(k)$，其他语句均可在常数时间内完成，因此算法 KTD 的时间复杂度为 $O((s+nt+k^2n)\times k\times r)$。

由上述分析可知，算法 KTD 的时间复杂度为多项式级。

为验证算法 KTD 的正确性和有效性，在 Linux 环境下用 C 语言实现了该算法，并把它嵌入到规划器 StepByStep 中。选取 IPCs 中 STRIPS 领域的基准问题进行了相关的对比实验，实验环境：Pentium(R) 3.0GHz，1024M，VMware Workstation，Red Hat Linux。部分实验结果如表 6 所列。

表 6 StepByStep^{KTD} 与 StepByStep 的求解效率比较

Domain	Problem	StepByStep^{KTD}		StepByStep		Time reduced (%)
		Time (s)	Steps	Time (s)	Steps	
Logistics	problogistics-11	0.30	69	0.39	69	23.08
	problogistics-13	2.74	67	3.85	67	28.83
	problogistics-15	5.65	79	7.99	79	29.29
	problogistics-17	16.38	104	22.14	104	26.02
	problogistics-19	19.87	97	27.82	97	28.58
Elevator	s16-0	53.57	58	67.62	58	20.78
	s17-0	94.36	62	124.57	62	24.25
	s18-0	64.16	65	75.82	65	15.38
	s19-0	136.55	71	177.19	71	22.94
	S20-0	135	75	153.70	75	12.17

由表 6 不难看出：

(1)在规划解的质量方面。本文所讨论的内容是谓词知识树的分解策略，它以“减少搜索空间，提高谓词规划树生成效率”为目的。在谓词规划树生成后，规划器 StepByStep^{KTD} 和 StepByStep 的规划求解策略是一致的，所以这两个规划器所得规划解的质量是相同的，即所得规划解的步数是相同的。

(2)在规划求解效率方面。对于不同的规划领域，具有静态前提的动作个数是不同的，含有静态前提的动作个数越多，分解所获得的静态知识子树个数也就可能越多，生成相关谓词规划树的效率也就越高，对提高规划求解效率的作用也就越大。比如：对于 Logistics 规划领域，StepByStep^{KTD} 的求解速率要比 StepByStep 平均提高 27%；对于 Elevator 规划领域，StepBy-Step^{KTD} 的求解速率要比 StepByStep 平均提高近 20%。

对于不含静态谓词的规划领域，算法 KTD 不能分解任何谓词知识树，它可在常数时间内完成。因此，算法 KTD 对该类规划领域求解效率的影响是微乎其微的。

结束语 本文以谓词知识树为研究对象，分析了领域动作与参数变量之间的静态相关性，并基于静态相关性提出了谓词知识树分解策略。文中的谓词知识树分解算法 KTD 已应用于规划器 StepByStep 中，并对 STRIPS 领域的一些基准问题进行了实验。实验结果表明，对谓词知识树进行分解可有效缩小搜索空间。在生成谓词规划树时，只需选用选择条件为“真”的静态知识子树，这样可提高规划树的生成效率，从而达到提高规划求解效率的目的。

本文的谓词知识树分解策略仅与规划领域中的动作定义有关，与具体规划问题无关，因此该方法对于所有规划领域都有效，同时也可应用于基于反向搜索的规划系统的动作选择问题。进一步地，可以通过静态知识子树提取领域知识，将其应用于规划子目标排序问题等。

参 考 文 献

[1] KUTLUHAN E, DANA S N, SUBRAHMANIAN V S. Complexity, Decidability and Undecidability Results for Domain-Independent Planning[J]. Artificial Intelligence, 1995, 76(1/2): 75-88.

[2] ROBERTS M, HOWE A. Learning from planner performance [J]. Artificial Intelligence, 2009, 173(5/6): 536-561.

[3] HELMERT M, RÖGER G. How good is almost perfect[C]// Proceedings of the 23rd National Conference on Artificial Intelligence, 2008. Menlo Park: AAAI Press, 2008: 944-949.

[4] HOFFMANN J, NEBEL B. The FF planning system: Fast plan generation through heuristic search[J]. Journal of Artificial Intelligence Research, 2001, 14: 253-302.

[5] CAI Dun-bo, YIN Ming-hao, GU Wen-xiang, et al. Fast forward planning system based on delayed partly reasoning[J]. Chinese Journal of Computers, 2008, 31(5): 793-802. (in Chinese)
蔡敦波，殷明浩，谷文祥，等. 基于延迟部分推理的快速前向规划系统[J]. 计算机学报，2008，31(5)：793-802.

[6] LIANG Rui-shi, JIANG Yun-fei, BIAN Rui, et al. A High-Quality Domain- Independent Pruning Strategy for Forward-Chaining Planning [J]. Chinese Journal of Computers, 2012, 35(8): 1620-1633. (in Chinese)
梁瑞仕，姜云飞，边芮，等. 一种高质量的领域无关前向规划剪枝策略[J]. 计算机学报，2012，35(8)：1620-1633.

[7] WEI Wei, OUYANG Dan-tong, LV Shuai. Conformant planning based on reducing belief states[J]. Journal of Software, 2013, 24(7): 1557-1570. (in Chinese)
魏唯，欧阳丹彤，吕帅. 基于缩减信念状态的 Conformant 规划方法[J]. 软件学报，2013，24(7)：1557-1570.

[8] KOEHLER J, HOFFMANN J. On reasonable and forced goal orderings and their use in an agenda-driven planning algorithm [J]. Journal of Artificial Intelligence Research, 2000, 12: 338-386.

[9] RICHTER S, WESTPHAL M. The LAMA planner: Guiding cost-based anytime planning with landmarks[J]. Journal of Artificial Intelligence Research, 2010, 39: 127-177.

[10] LIANG Rui-shi, JIANG Yun-fei, BIAN Rui, et al. Admissible subgoal ordering for automated planning[J]. Journal of Software, 2011, 22(5): 914-928. (in Chinese)
梁瑞仕，姜云飞，边芮，等. 智能规划中的可纳子目标排序[J]. 软件学报，2011，22(5)：914-928.

[11] KNOBLOCK C. Automatically generating abstractions for planning[J]. Artificial Intelligence, 1994, 68(2): 243-302.

[12] LANSKY A, GETOOR L. Scope and abstraction: two criteria for localized planning[C]// Proceedings of the 14th International Joint Conference on Artificial Intelligence, 1995. San Francisco: Morgan Kaufmann, 1995: 1612-1619.

(下转第 270 页)

[J]. 计算机科学,2014,41(2):114-118.

[3] ISHIBUCHI H,TSUKAMOTO N,HITOTSUYANAGI Y,et al. Effectiveness of scalability improvement attempts on the performance of NSGA-II for many-objective problems[C]// Congerence on Genetic and Evolutionary Computation. 2008:649-656.

[4] KOPPEN M,YOSHIDA K. Substitute distance assignments in NSGA-II for handling Many-objective optimization problems[C]// Evolutionary Multi-Criterion Optimization. 2007:727-741.

[5] CORNE D,KNOWLES J. Techniques for highly multiobjective optimization:Some nondominated points are better than others [C]// Conference on Genetic and Evolutionary Computation. 2007:773-780.

[6] KUKKONEN S,LAMPINEN J. Ranking-dominance and many-objective optimization[C]// IEEE Congress on Evolutionary Computation. 2007:3983-3990.

[7] SATO H,AGUIRRE H,TANAKA K. Controlling dominance area of solutions and its impact on the performance of MOEAs [C]// Evolutionary Multi-Criterion Optimization. 2007:5-20.

[8] ZITZLER E,KÜNZLI S. Indicator-based selection in multiobjective search[C]// Parallel Problem Solving from Nature-PPSN VIII. Springer Berlin Heidelberg,2004:832-842.

[9] BADER J,ZITZLER E. HypE:An algorithm for fast hypervolume-based many-objective optimization[J]. Evolutionary computation,2011,19(1):45-76.

[10] ZHANG Q,LI H. MOEA/D:A multiobjective evolutionary algorithm based on decomposition[J]. IEEE Transactions on Evolutionary Computation,2007,11(6):712-731.

[11] DEB K,JAIN H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach,part I:Solving problems with box constraints[J]. IEEE Transactions on Evolutionary Computation,2014,18(4):577-601.

[12] IKEDA K,KITA H,KOBAYASHI S. Failure of Pareto-based MOEAs:does non-dominated really mean near to optimal?[C]// Proceedings of the 2001 Congress on Evolutionary Computation,2001. IEEE,2001,2:957-962.

[13] 谭艳艳. 几种改进的分解类多目标进化算法及其应用[D]. 西安:西安电子科技大学,2013.

[14] ZITZLER E,THIELE L,LAUMANNS M,et al. Performance assessment of multiobjective optimizers:an analysis and review [J]. IEEE Transactions on Evolutionary Computation,2003,7(2):117-132.

[15] WANG Z,ZHANG Q,ZHOU A,et al. Adaptive Replacement Strategies for MOEA/D[J]. IEEE Transactions on Cybernetics,2015(1):1-13.

[16] GONG Dun-wei,LIU Yi-ping,SUN Xiao-yan,et al. Parallel Many-objective Evolutionary Optimization Using Objectives Decomposition[J]. Acta Automatica Sinica,2015,41(8):1438-1451. (in Chinese)
巩敦卫,刘益萍,孙晓燕,等. 基于目标分解的高维多目标并行进化优化方法[J]. 自动化学报,2015,41(8):1438-1451.

[17] ZHANG Yi,WAN Xing-yu,ZHENG Xiao-dong,et al. Cellular Genetic Algorithm for Multiobjective Optimization Based on Orthogonal Design[J]. Acta Electronica Sinica,2016,44(1):87-94. (in Chinese)
张屹,万兴余,郑小东,等. 基于正交设计的元胞多目标遗传算法[J]. 电子学报,2016,44(1):87-94.

(上接第 242 页)

[13] AMIR E,ENGELHARDT B. Factored planning [C]// Proceedings of the 18th International Joint Conference on Artificial Intelligence,2003. San Francisco:Morgan Kaufmann,2003:929-935.

[14] KELAREVA E,BUFFET O,HUANG J B,et al. Factored planning using decomposition trees[C]// Proceedings of the 20th International Joint Conference on Artificial Intelligence,2007. Amsterdam:Elsevier Science,2007:1942-1947.

[15] CHEN Yi-xin,YAO Guo-hui. Completeness and Optimality Preserving Reduction for Planning[C]// Proceedings of the 21st International Joint Conference on Artificial Intelligence,2009. Amsterdam:Elsevier Science,2009:1659-1664.

[16] CHEN Yi-xin,XU You,YAO Guo-hui. Stratified Planning [C]// Proceedings of the 21st International Joint Conference on Artificial Intelligence,2009. Amsterdam:Elsevier Science,2009:1665-1670.

[17] CHEN Y X,WAH B W,HSU C W. Temporal planning using subgoal partitioning and resolution in SGPlan[J]. Journal of Artificial Intelligence Research,2006,26:323-369.

[18] WAH B W,CHEN Y X. Constraint Partitioning in Penalty Formulations for Solving Temporal Planning Problems[J]. Artificial Intelligence,2006,170(3):187-231.

[19] CROSBY M,ROVATSOS M,PETRICK R P A. Automated Agent Decomposition for Classical Planning[C]// The Proceedings of the 23rd International Conference on Autonomous Planning and Scheduling,2013. Menlo Park,Calif. AAAI Press 2013:46-54.

[20] ASAI M,FUKUNAGA A. Solving Large-Scale Planning Problems by Decomposition and Macro Generation[C]// The proceedings of the 25th International Conference on Autonomous Planning and Scheduling,2015. Menlo Park,Calif. AAAI Press 2015:16-24.

[21] WU Xiang-jun,JIANG Yun-fei,LING Ying-biao. Research and development of StepByStep planner[J]. Journal of Software,2008,19(9):2243-2264. (in Chinese)
吴向军,姜云飞,凌应标. 智能规划器 StepByStep 的研究和开发[J]. 软件学报,2008,19(9):2243-2264.

[22] WU Xiang-jun,BIAN Rui,LING Ying-biao,et al. Research on Decomposition Strategy for Knowledge Tree of Characteristic Predicate[J]. Journal of Computer Research and Development,2011,48(2):186-194. (in Chinese)
吴向军,边芮,凌应标,等. 特征谓词知识树分解策略的研究[J]. 计算机研究与发展,2011,48(2):186-194.