概率规划的研究与发展

闫书亚[1] , 殷明浩[1,2] , 谷文祥[1] , 刘小飞[1]
(1. 东北师范大学 计算机学院 ,吉林 长春 130117; 2. 吉林大学 计算机科学技术学院 ,吉林 长春 130012)

摘 要:概率规划是智能规划研究的一个重要方面 , 首先给出概率规划领域定义语言 , 并介绍其语法及语义 , 随后重点介绍了求解概率规划的各种方法 , 如动态规划、启发式动态规划和基于规划图的方法等 , 并分析了各种方法的特点. 最后对国际概率规划比赛进行了介绍.

关键词:智能规划; 概率规划; 动态规划; 概率规划领域定义语言
中图分类号: TP18 **文献标识码**: A **文章编号**:1673-4785(2008)01-0009-14

Research and advances in probabilistic planning

YAN Shu-ya[1] , YIN Ming-hao[1,2] , GU Wen-xiang[1] , LIU Xiao-fei[1]
(1. School of Computer , Northeast Normal University , Changchun 130117 , China; 2. College of Computer Science and Technology , Jilin University , Changchun 130012 , China)

Abstract: Probabilistic planning has an important role in allowing intelligent planning to adapt to uncertainty. This paper introduces a new probabilistic plan domain definition language (PPDDL) , followed by its syntax and semantics. Various methods of probabilistic planning are described , such as dynamic programming algorithms , heuristic dynamic programming algorithms and algorithms based on planning graph. The features of each algorithm are then analyzed. Finally , we give a brief introduction to the international probabilistic planning competition. The conclusions in this paper should be helpful to researchers interested in this field.

Key words: intelligent planning; probabilistic planning; dynamic programming; PPDDL

智能规划是当前人工智能领域中极为活跃的一个研究热点 , 近 10 年来 , 有关智能规划的研究在问题描述和问题求解两方面得到了新的突破. 相对于早期的智能规划系统 , 现代规划系统无论在规划求解效率上还是在规划求解规模上都有指数级的提高[1]. 然而 , 经典智能规划要求知识的完整性 , 即假设 Agent 对于规划世界的知识是完全的 , 规划过程中动作的效果是确定的 , 但是 , 在现实世界中得到的信息往往是不完全、不确定的 , 这就使规划理论的应用范围受到极大的限制.

针对这些问题 , 国内外很多研究人员开始寻找更一般的算法来处理这些不确定性问题. 在文献[2-5]中 , Weld 等人提出了一致性规划问题 , 在这类规划问题中 , Agent 需要考虑到初始状态和动作的不确定性; 感知规划问题需要在规划执行过程中考虑感知动作[6-7]; 时态规划问题和资源规划问题在生成规划解时需要考虑时态约束和资源约束[8-11]; 概率规划问题通过使用概率分布来刻画动作效果的不确定性 ,试图找到完成规划目标的最大概率规划解[12-14]. 事实上 , 非经典规划问题的研究已经成为目前人工智能规划领域研究的主要研究领域之一. 在 2004 年举办的第 4 届国际智能规划竞赛中 ,智能规划研究人员举办了第 1 届概率规划组的比赛; 在 2006 年的第 5 届智能规划竞赛中 , 同样有概率组的比赛. 人工智能研究杂志(journal of artificial intelligence research , JAIR) 组织专刊以发表参加比赛的智能规划系统的研究报告. 人工智能杂志在 2003 年也组织了一期专刊用于介绍和推广

收稿日期:2007-07-19.
基金项目:国家自然科学基金资助项目(60573067 , 60473042) ;东北师范大学青年自然科学基金资助项目(20070601) .
通讯作者:闫书亚. E-mail :yansy276 @nenu. edu. cn.

不确定规划问题.概率规划问题作为不确定规划问题,也得到了研究者的广泛关注,其求解方法也日趋成熟.文中首先对概率规划问题进行介绍,然后着重介绍解决概率规划问题的概率规划算法,并比较分析各种方法,最后对国际概率规划比赛进行了介绍.

1 概率规划表示

1.1 概率规划领域定义语言

1998年,Drew McDermott提出了规划领域定义语言(plan domain definition language, PDDL),随后PDDL逐渐成为规划领域模型表示和交流的通用标准,并成为国际智能规划比赛的标准语言[10,15-16].2004年,H. H. Younes和M. Littman提出了PPDDL1.0以处理动作带有不确定效果的概率规划问题[17],概率部分的比赛使用的语言即是PPDDL1.0[18].

1.2 PPDDL语法

相对于经典规划领域定义语言PDDL,PPDDL1.0增加了新的表达能力,主要包括2个方面,即对概率效果的支持、对马尔可夫决策过程回报值的支持.在概率规划中,动作的效果是随机的,因此PPDDL增加了对概率效果的支持,其语法表示为

$$(\text{probabilistic } p_1\ e_1, \cdots, p_k\ e_k).$$

式中:e_i表示动作的一个可能效果,p_i表示动作效果e_i出现的概率,并且p_i满足如下约束:

$$p_i \geq 0, \sum_i^k p_i = 1.$$

但有时也允许某个空的概率效果出现,即

$$(\text{probabilistic } p_1\ e_1, \cdots, p_l\ e_l).$$

式中:$\sum_i^l p_i \leq 1$.

这种表示和下面的表示是等价的:

$$(\text{probabilistic } p_1\ e_1, \cdots, p_l e_l\ q\ (\text{and})).$$

式中:$q = 1 - \sum_i^l p_i$,(and)表示空效果.

例如,(probabilistic 0.9 (clogged))表示命题clogged以0.9的概率在下一状态中变为真,以0.1的概率保持原有真值.

另一方面,PPDDL使用关键字reward来表示动作或规划的回报值,其基本语法表示为

(<additive-op> <reward fluent> <f-exp>).

式中:<additive-op>为增加回报算子或减少回报算子;<f-exp>是不包括reward的数值表达,动作的前提和效果不涉及变量reward.

图1显示了咖啡传送域[19]的PPDDL语言表示,当"buy-coffee"动作执行时,如果使用者得到了coffee,则获得0.8的回报.当在is-wet为假时,执行"buy-coffee"动作,获得的回报则是0.2.而当user-has-coffee和¬is-wet均为真的时候,得到的回报则可以是1.

```
(define (domain coffee-delivery)
    (:requirements :negative-preconditions
                   :disjunctive-preconditions
                   :conditional-effects :mdp)
    (:predicates (in-office) (raining) (has-umbrella)
                 (is-wet) (has-coffee) (user-has-coffee))
    (:action buy-coffee
             :effect (and (when (not (in-office))
                          (probabilistic 0.8 (has-coffee)))
                          (when (user-has-coffee) (increase
                          (reward) 0.8)) (when (not (is-wet))
                              (increase (reward) 0.2))))
...)
```

图1 咖啡传送域的部分PPDDL编码

Fig. 1 Part of PPDDL encoding of "coffee delivery" domain

如果在规划问题定义中没有明确定义一个规划尺度(planning metric),则求解一个概率规划问题时默认的目标为寻找到一条最大化目标实现概率的规划.

1.3 PPDDL语义

一个基于PPDDL描述的概率规划问题通常可以被映射为一个带有回报的概率转移系统(probabilistic transition system).

设V为一个概率规划问题的状态变量集合,状态s为对状态变量的一次赋值,则变量的所有可能赋值的集合构成了概率规划问题的整个状态空间.$p_0: S \to [0, 1]$, $\sum_{s \in S} p_0(s) = 1$(即$p_0$是状态上的概率分布),$V$上的公式$\Phi$刻画了目标状态$G = \{s | s| = \Phi\}$,动作集合$A$由动作概要实例化得到.

动作$a \in A$包含前提Φ_a和效果e_a.动作a在状态s上可用当且仅当$s| = \Phi_a$,如果$s| \neq \Phi_a$,在s上运用a则是错误的.这与PDDL2.1[10]的语义是一致的.效果循环如下定义:

1) $\top$是空效果,在PPDDL中用(and)表示;

2) 若$b \in V$是一个布尔型状态变量,则b和$\neg b$是其效果;

3) $x \leftarrow f$是一效果,如果$x \in V$是一数值型状

态变量且 f 是数值状态变量上的实值函数;

4) $r \uparrow f$ 是一效果,如果 f 是数值状态变量上的实值函数;

5) $e_1 \wedge \cdots \wedge e_n$ 是效果,如果 e_1, …, e_n 是效果;

6) $c \rhd e$ 是效果,如果 c 是 V 上的一个公式且 e 是一个效果;

7) $p_1 e_1 | \cdots | p_n e_n$ 是效果,如果 e_1, …, e_n 是效果,对于所有 $i \in \{1, \cdots, n\}$, $p_i \geq 0$ 且 $\sum_i^n p_i = 1$.

其中 2)～4) 称为简单效果,$x \leftarrow f$ 意为当前状态中的值 f 在下一状态变为数值状态变量 x 的值,效果 $r \uparrow f$ 用来与转移回报相联系.

动作 $a = \langle \phi_a, e_a \rangle$ 定义了一个转移概率矩阵 P_a 和转移回报矩阵 R_a, p_{ij}^a 表示在状态 i 上运用动作 a,转移到状态 j 时的概率; r_{ij}^a 表示在状态 i 上运用动作 a,转移到状态 j 时得到的回报. 效果 $c \rhd e$ 定义了一组状态转移,其中 e 是简单效果的合取. 计算 P_a 和 R_a 时,可首先将 e_a 转化为 $p_1 e_1 | \cdots | p_n e_n$,其中 e_i 是确定效果. 具体的转化方法见文献[18].

当结果为多个效果 $p_1 e_1 | \cdots | p_n e_n$ 时,具有同样的转移,这时要求一个特定转移的回报与结果是一致的.

2　概率规划求解方法

2.1　动态规划算法

一个概率规划问题通常可以模型化为随机最短路径问题 (stochastic shortest path problems, SSPs)[20]. 随机最短路径问题是一类特殊的马尔可夫决策问题(Markov decision problems, MDPs). SSPs 为一个六元组:$\langle S, A, T, c, G, S_0 \rangle$,其中,$S$ 是一组有限状态的集合; A 是有限动作的集合,对任一 $s \in S$, $A(s)$ 表示在状态 s 上的可用动作的集合; T 是转移模型,即对在每个可能状态中的每种行动的结果概率的详细说明. 对于 $s, s' \in S$, $T(s, a, s')$ 表示在状态 s 上执行动作 a 时到达状态 s' 的概率; c 是代价函数,对任一 $s \in S$, $a \in A(s)$, $c(a, s)$ 表示在状态 s 上执行动作 a 的立即代价; G 是目标集, $G \subseteq S$; $S_0 \in S$ 是初始状态.

这时,求解概率规划问题即转换为求解下述不动点方程[21]:

$$f(s) = \min_{a \in A(s)} [c(a,s) + \sum_{s'} T(s,a,s') f(s')]. \tag{1}$$

显然价值迭代算法和策略迭代算法可以用于求解上述问题.

2.1.1　价值迭代算法

价值迭代算法 (value iteration, VI) 是求解 MDP 问题的一种经典的动态规划算法,它从任意的初始评估值开始,通过迭代来求解方程. 令 $f_i(s)$ 为状态 s 在第 i 次迭代中的评估值,那么价值迭代执行下述更新:

$$f_{i+1}(s) = \min_{a \in A(s)} [c(a,s) + \sum_{s'} T(s,a,s') f(s')]. \tag{2}$$

如果无限地应用贝尔曼更新方程,价值迭代最终会收敛在贝尔曼方程组的惟一解上. 在这种情况下,对应的策略是最优的.

$$\pi(s) = \arg \min_{a \in A(s)} [c(a,s) + \sum_{s'} T(s,a,s') f(s')]. \tag{3}$$

对于 $\forall s \in S$, 当 $|f_i(s) - f_{i-1}(s)| \leq \varepsilon$, 价值迭代算法停止,其中 $\varepsilon > 0$ 是给定的误差界限,即当评估函数 f 足够接近于最优时,价值迭代算法停止. 但是,由于 VI 更新每一个状态的值,它求解的过程是相当缓慢的. 一个优化的措施是当它进行状态更新时,对其进行限制,只更新那些从初始状态可以到达的状态的评估值.

2.1.2　策略迭代算法

价值迭代算法中,在每一时间步,评估函数只是被用来计算动作,而策略则没有被显式地存储或记录. 策略迭代,在求解过程中则在存储更新评估值的同时也存储并更新策略的值.

策略迭代算法从某个初始策略 π_0 开始,交替执行下面的 2 个步骤[22-23]:

1) 策略评价:确定当前策略的评估函数 f,即给定策略 π, 计算 f^π;

2) 策略改进:基于当前评估函数,更新当前策略,即当前策略 π 不是最优策略时,利用更新方程,找到一个新的策略 π', 使得对任一状态 i, $f^{\pi'}(i) \leq f^\pi(i)$, 并且至少存在一个状态 i, 使得 $f^{\pi'}(i) < f^\pi(i)$.

当策略改进步骤没有产生评估值的改变时,算法终止. 对于有限的状态空间而言,策略数是有限的,并且每一次迭代都可以产生更好的策略,所以在有限次迭代后,算法会收敛于一个最优策略,并终止.

因为策略把每个状态中的行动都固定了,在第 i 次迭代中,策略 π_i 指定了状态 s 中的行动 $\pi_i(s)$,

由此，可得到贝尔曼方程的一个简化版本[22]：

$$f_i(s) = c(\pi_i(s), s) + \gamma \sum_{s'} T(s, \pi_i(s), s') f_i(s'). \tag{4}$$

在求解规划的过程中，由于策略迭代要更新每个状态的策略，其求解速度也是相对缓慢的.

2.2 启发式动态规划算法

用动态规划方法求解规划问题起到了良好的效果. 但是，这些算法在更新状态的评估值时，均考虑了所有状态的评估值或策略，对于大的状态空间问题，其复杂度是让人望而却步的. 鉴于此，研究者将启发式技术引入到动态规划算法中，以提高求解规划的效率. 下面重点介绍几种启发式动态规划算法(heuristic search / dynamic programming algorithms, HS/DP).

2.2.1 基于 *RTDP* 的方法

1995年Barto等人提出了实时动态规划算法(real-time dynamic programming, RTDP)[24]，这里的“实时”指的是动作必须在严格的时间约束下执行，其实质是通过对Korf提出的LRTA *[25]算法进行扩展，来解决随机最短路径问题. 在RTDP算法中，并不对所有状态的评估函数进行更新，因而减少了状态空间，在一定的假设前提条件下，RTDP收敛于一种最优策略.

RTDP在一个单一的执行路迹(trace)上模拟贪婪策略，并使用贝尔曼更新方程更新它们访问到的状态的评估值. 这里主要介绍trial-based RTDP算法. 该算法通过将状态上的计算组织为Sequential trials而求解规划问题. Sequential trials是一种一边试验一边修正试验方案的试验方法. 一个RTDP试验是一个路径，每一个试验包含一系列步骤(steps). 在每一步骤中，根据前向一步或多步深度搜索而选择动作，基于动作的结果，选择当前的状态. 包含当前状态的一个状态子集的代价函数被更新，也即控制器总是遵循与最近更新的最优评估函数相对应的贪婪策略，通过这个策略来选择将要执行的动作，基于被选择的这个动作来更新当前状态. 在每一个试验的开始，当前状态被设置为开始状态，当到达目标状态或经过一定的步骤数之后，试验结束. RTDP反复执行这样的试验，直至收敛. RTDP的一个重要特性是，它只更新从初始状态能够到达的那些状态的评估值，可能忽略了状态空间中的许多状态，因而可以相对比较快地产生一个相对比较好的策略. 文献[24]证明在确定性合理条件下，RTDP逐渐收敛于最优解而不用估计整个状态空间.

Korf等人以模拟自动车辆行驶路径的路径追踪问题为例[26]，比较了几种不同算法[27]，得出RTDP能够快速收敛于一种次优策略；其后，其他研究人员也以实验结果证明了较大状态空间问题中该算法的快速收敛性. 因而RTDP的最大优点是可以在较短的时间内得到一个较好的策略. 但由于很多状态被频繁地更新，所以在每个状态上完全收敛的速度是非常慢的，这是其缺点之一. 另外，RTDP算法要求系统的状态是完全可观察的，但在现实中却很难满足. 因此，还需要改进算法，以提高效率或者解决更复杂的问题.

2.2.2 基于 Labeled RTDP 的方法

由于RTDP的收敛速度比较缓慢，Labeled RTDP(LRTDP)算法[13]引入一个标记方案，提高了RTDP的收敛速度，同时保留了RTDP的其他优点.

Labeled RTDP算法的标记功能由一个被称为Checksolved的标记程序完成. 它的主要思想是：当贪婪策略定义在状态 s 上的启发式值和从 s 出发利用贪婪选择策略可达的那些状态上的启发式值都收敛时，标记状态为可解状态(solved). 状态 s 及由 s 出发利用贪婪策略可达的那些状态组成的图称为状态 s 的贪婪图(greedy graph)，贪婪图中的状态构成的集合成为状态 s 的贪婪封装集(greedy envelope). 状态 s 被激活时，Checksolved程序开始搜索以 s 为根结点的贪婪图，寻找是否存在误差大于 ε 的状态 s'，如果不存在误差大于 ε 的状态，则标记 s 为可解状态且Checksolved (s)返回true，即当值函数在状态 s 上收敛时，标记 s 为可解. 如果在状态 s 的贪婪封装集中出现误差大于 ε 的一个状态 s' 时，状态 s 和与其相关的状态都要被更新，Checksolved (s)返回false. Checksolved的代码在文献[13]中的算法3中给出，状态 s 的标记在程序中用哈希表的1个比特位表示，记为 s.solved，一个状态 s 被标记为可解当且仅当 s.solved = true. LRTDP算法通过模拟试验实现，除了终止条件不同，它与RTDP的试验非常相似. 实验从初始状态开始，以贪婪策略 π_i 选择动作，以相应的转移概率选择后继状态，直到到达一个可解状态时实验结束(初始时只有目标状态是可解状态). 每次实验结束，调用标记程序Checksolved，反向核对状态的值函数的值，从最后一个不可解状态(unsolved)开始回溯遍历到 s_0，直到程序在一些状态上返回false. 当初始状态 s_0 被标记为可解时，Labeled

RTDP 算法终止. LRTDP 返回的策略 π_v 是就状态 s 而言的一个偏序最优策略,它只保证了在 s 和 s 可达的那些状态上策略 π_v 最优,与 RTDP 一样不需要考虑整个状态空间. LRTDP 的代码在文献[13]中的算法 4 中给出.

实践证明,LRTDP 大大地加快了 RTDP 的收敛速度. 当启发式值 $h=0$ 时,LRTDP 通常也比 VI 收敛速度快. 这表明,LRTDP 会有一个广泛的应用空间. mGPT 规划器[28]就将 LRTDP 作为主要的算法之一. mGPT 利用启发式搜索求解由高层规划语言 PPDDL 描述的 MDP 模型,它支持多种算法和启发式函数,其中最主要的 2 个算法就是 Labeled RTDP 和 HDP[29],mGPT 在 IPC-4 中表现突出,下面就来介绍基于 HDP 的方法.

2.2.3　基于 HDP 的方法

Bonet 和 Geffner 在文献[29]中提出了新的启发式动态规划算法 HDP(faster heuristic and dynamic programming search algorithms for planning),在 Tarjan 的强连通分支[30]的基础上,通过引入标记机制来对状态进行判断更新.

文献[29]模型化了带有回馈(feedback)的非确定规划问题,具体说是随机最短路径问题. 它首先给出一个一般化的算法框架 Find-and-Revise,并给定 ε-一致/不一致贪婪图(greedy graph)的定义. 一个值函数 v 在状态 s 上是 ε一致/不一致的是指它在 s 上的剩余(residual)不大于/大于 ε; v 本身是 ε一致的当它在所有的状态上是 ε一致的,这里的状态是指在 π_v 下,从 s_0 可以到达的那些状态. 贪婪图 G_v 是从初始状态 s_0 出发,执行贪婪策略 π_v 得到的图,即 s_0 是 G_v 中唯一的根节点,对于 G_v 中的每一非目标状态 s,他们的孩子节点是在 s 上执行动作 $\pi(s)$ 可能引起的那些状态.

Find-and-Revise 通过在贪婪图中搜索不一致状态并更新它们来计算 ε-一致值函数,直到不存在不一致的状态. HDP 是 Find-and-Revise 的一个实例化算法,通过 Find 操作,系统地进行深度优先搜索,并为访问过的状态作标记. 一个状态定义为可解的(solved)是指:在 s 上值函数 v 是 ε-一致的,并且在所有从 s 出发,经贪婪策略 π_v 能到达的状态上也是 ε-一致. 当这些条件满足时,在 s 及从 s 能到达的状态上,不需要进一步进行更新. 当初始状态 s_0 是可解的并且因此一个 ε-一致的值函数找到的时候,算法结束.

由于贪婪图中存在环,HDP 采用 LRTDP[13]中解决循环的标记方法. 在 LRTDP 中,通过对从 s 到达的不一致的状态进行一个统一的搜索,给试验中的最后一个未解决(unsloved)的状态 s 标记. 如果这样一个状态被找到了,更新它,然后新的试验被执行,否则,状态 s 和它所有的未解决的后继被标记为可解的,一个新的 RTDP 循环和标记检查被激活. HDP 采用这种方法,但对其进行了改进,它去掉了标记检查所需要的额外搜索. 标记检查作为 Find 搜索的一部分,它运用 Tarjan 线性算法[30]来检查有向图的强连通分支. 在有向图 G_v 中,s 与 s' 之间的"～"关系为 $s=s'$ 或 s 可从 s' 到达且 s' 可从 s 到达. G_v 中的强连通分支称为等价类,由 s 与 s' 之间的"～"定义,并因此形成一个特定的集合,用 C 表示,并定义当 C 中的所有 s 都满足 ε一致时,Component C 是 ε一致的,当 C 中的每一个 s 都是可解的 Component C 是可解的. 由此定义 G^C_v:当 C' 中某一状态能由 C 中的某一状态到达时,图的顶点是 G_v 中的 Components,边是 $C \to C'$. 显然,G^C_v 是一个非循环的图. 另外:

1) 状态 s 是可解的当且仅当 Component C 是可解的;

2) Component C 是可解的当且仅当 C 是一致的且所有的 C'、$C \to C'$ 是可解的.

从而在循环图 G_v 中标记状态的问题可映射到非循环图 G^C_v 中标记 Components 的问题,而后者则可通过标准的动态规划方法解决.

HDP 算法阐明了存在于 HS/DP 算法背后的基本观点:寻找不一致的状态并更新它们,直到不存在这样的状态. 试验结果显示,与当前的一些算法相比,HDP 还是很有竞争力的.

2.2.4　基于 *LDFS* 的方法

启发式动态规划算法可以有效处理大状态空间问题,但这些问题和它们使用的各种技术之间的共同点往往不是那么清晰. 如何将它们转化为更一般的模型以有效的利用,针对此问题,Bonet 和 Geffner 结合 DP 算法的优点和 HS 算法的有效性,发展了一种通用的算法框架 (learning in depth-first search,LDFS)[31-32],以期达到一般性和有效性.

LDFS 框架清晰化和一般化了贯穿于各种算法模型中的启发式算法族的 2 个关键点:学习(learning)和下界(lower bounds). LDFS 结合更新,下界和初始状态知识来为规划问题计算偏序最优策略(partial optimal policies),它利用这些概念,表示出一个一般性的框架,这个框架能够覆盖一系列模型问题.

LDFS 算法的基本特点是:求解时,结合有界

和深度优先搜索算法进行求解. 在每次迭代中, 如果存在一个不超过最优代价的解, 那么找到解, 否则, 更新评价函数, 求解过程重新开始. LDFS 用统一的符号元素为不同的模型定义, 如确定的、非确定的、MDP、GT 模型等, 并给出针对这些模型的贝尔曼方程. 在此基础上, 给出一个一般化的算法, 即 Find-and-Revise. 对于任何一个可采纳的评估函数, 算法在策略 π 下找到一个从初始状态 s_0 出发能够到达的状态 s, 在保证 $\mathrm{Res}_V(s) > \varepsilon$ 情况下更新状态的评价值, 最后返回评价函数. LDFS 作为这一算法框架的一个实例, 主要通过 2 个循环来实现算法: 一个循环位于状态 s 中的贪婪动作 a 上, $a \in A(s)$; 另一个循环位于 s 的可能后继 s' 上. 搜索中的端(tip)节点是不一致的状态 s、终止状态和被标记为可解(solved)的状态. 一个状态被标记为可解的,即在 s 下的搜索没有找到不一致的状态. 如果 s 是一致的且在对 s 的后继状态 s' 搜索后标记位为真, 则 s 被标记为可解的, 并记录动作 a, s 上的其他动作将不再被搜索. 否则, 对下一贪婪动作进行试验, 一直到没有这样的动作存在时为止, 更新 s 值. 算法结束时返回评估函数和贪婪策略.

试验结果表明, 在某些模型问题中, 如 Max、AND/OR 图中, LDFS 并不比其他的一些算法表现差, 事实上, 在某些方面 LDFS 的性能还略胜一筹.

2.3 *FPG* 方法

分解策略梯度规划器 (factored policy gradient planner, FPG)[33] 是为解决较大规划问题域而设计的概率时序规划器, FPG 与传统规划方法的区别主要有 2 点, 首先, FPG 是有导向的进行规划搜索, 利用在线梯度下降法寻找最优的参数规划. 其次, 策略被分解到每个动作中, 从部分观察值映射到动作可能执行的概率, 反映出每个动作可取的程度. 与其他概率时序规划器不同, FPG 在保证总 makespan 最小的同时使成功到达目标的概率最大.

FPG 的目标是处理现实世界问题域并生成好的策略, 它通过以下几点能实现目标: 1) 使用梯度下降法导向策略搜索; 2) 将策略分解成针对每个动作简单的近似策略; 3) 利用临界观察提取出每个策略; 4) 使用蒙特卡罗(Monte-Carlo)算法的思想, 存储器需求独立于状态空间的大小. 而前面所介绍的规划算法将概率规划问题描述成为一个状态空间, 对长期价值效用和选择每个动作的代价进行评估[34]. 它们的缺点是需要估计大量的状态 - 动作对的值, 即使算法修剪了大部分状态,但当问题域扩大时,相关状态会成指数级增长,使得算法失效. 而 FPG 借鉴了策略梯度增强学习的算法思想, 这类算法不需要估计规划状态 - 动作对的值, 而是估计整个搜索过程长期平均回报的梯度. FPG 使用的是 OL $\mathrm{P}_{\mathrm{OMDP}}$ (在线部分可观察马尔可夫决策过程)策略梯度强化算法[35]. 梯度的计算与控制与在决策点选择动作的参数有关, 这些参数决定了策略、规划及系统. 在梯度的影响下, 调节参数的值使得期望回报增加或初始状态的值增加.

FPG 的分解参数策略通过对每个动作应用参数函数生成, 此函数输入规划状态的描述, 返回每个可取动作的概率分布. FPG 的输入语言是 mdp-sim 可处理的全部语言,即在 PDDL 上进行了微小的扩展[36]. FPG 使用了 mdpsim 的数据结构和函数实现了一个规划域的模拟器. 文中算法 1 完整的描述了 FPG 是怎样利用 OL $\mathrm{P}_{\mathrm{OMDP}}$ 分解动作策略实现规划的. 算法的目的是从状态空间中抽取一条路径: 1) 第 1 个状态作为规划的第 0 时间步的初始状态; 2) 与可取动作相连的感知机输入当前状态 s_t 的观察向量 o_t; 3) 每个感知机的网络计算其相对应动作的可取程度; 4) 选择一个规划动作; 5) 选择一个状态转移; 6) 规划器接收新的状态动作的全局回报同时生成即时梯度 $g_t = r_t e_t$; 7) 所有参数在梯度 g_t 的导向下立即更新.

FPG 规划器能对较大问题域生成好的策略,它的缺点是其局部最优性、参数简化、观察范围减少导致可能生成次优规划. 梯度的变化次数是关于 POMDP 混合时间(mixing time)的函数, 随着状态的增多会成指数级增长, 怎样计算任意 MDP 的 mixing time 仍是一个开放问题.

2.4 基于 LAO* 的方法

经典的启发式搜索算法可找到以简单路径 (A^*)、树或非循环图 (AO^*) 形式的解. Eric A. Hansen 和 Shlomo Zilberstein 提出了一个新的一般化的启发式搜索算法 LAO* (generalization of AO^*)[12,37], 它是目前求解决策理论规划问题最为高效的算法之一. 该算法结合了启发式搜索和动态规划的优点, 可以在不需评估整个状态空间的基础上寻找一条带有循环(loop)的最优解. 在 2004 年举办的第 1 届国际不确定智能规划比赛中, 它取得了"overall, non-blocks/box"组的冠军[14].

AO^* 是 AI 中应用于状态空间搜索的一个著名启发式算法. AO^* 算法找到的解是解图, 形式化为树或更一般化的是一个非循环的图. AO^* 算法通过从初始状态出发, 逐渐构造一个解图的方法来解决状态空间搜索问题. 它可以理解为下列 2 个主要运

算的反复. 首先,一个自上而下的图生长运算,通过跟踪有标记的连接符寻找最好的局部解图. 这些以前计算过的标记指明在搜索图中离开每个节点的当前的最好局部解图(在该算法终止之前,最好的局部解图尚未产生它的全部终叶节点,所以称它为局部的). 对这个最好的局部解图的非终叶节点之一进行扩展,并把某个费用赋给它的后继节点. 算法中的第 2 个主要运算是一个自下而上的费用修正、连接符标记和 solved 标记的过程. 从刚被扩展的节点开始,此过程修正其费用值(利用其后继节点最新计算的费用),并把外向连接符标记到被估计为达到终节点的最好路径上. 仅仅那些经过费用修正的节点,其祖先才有可能拥有它们的修正值. 因此,并非所有的祖先都需要进行费用修正,只有那些具有最好的局部解图而且含有修正费用的后裔的祖先才需要进行费用修正.

LAO* 算法可以看作是 AO* 算法的泛化. 由于在 AO* 算法中假设一个无循环的解,因而它只能解决具有无循环解的问题. 鉴于此,作者认识到 AO* 的费用修正步是一个 DP 算法,适当地一般化这个过程,从而得到 LAO* 算法,使得算法能处理解中包含循环的 MDPs. LAO* 算法包括 2 个阶段:在第一阶段,和 AO* 算法类似,LAO* 算法不断扩展最优部分解图,并使用一个可纳的启发式函数评估新扩展出的边缘节点;在第二阶段,LAO* 算法使用价值迭代或策略迭代等动态规划方法更新状态的评估值. 上述 2 个过程交替执行,直到找到最优解[37]. 同 AO* 一样,LAO* 的费用修正步也只在一定的状态集合上执行,即扩展状态和显式图中那些可从扩展状态到达的状态. LAO* 中,费用修正步可以利用 VI 或 PI. 运用 PI,在有限次迭代后,在对端节点状态进行启发估计的基础上,可以为显式图中的每一个状态计算一个精确的代价,而运用 VI,状态的精确代价是逐渐逼近的,但是在大规模问题上,这一缺点被 VI 的效率所抵消. 不论怎样,通过结合启发式搜索和动态规划的方法,LAO* 算法无需扩展整个状态空间,从而可以快速有效的求解不确定规划问题.

2.5　基于 Graphplan 的方法

图规划方法在智能规划领域取得了巨大的成功[1,38-41]. 由于其良好的性能,许多研究者扩展图规划,使其能解决不确定规划问题,并取得了很多重大成果. 1998 年,David E. Smith 和 Daniel Weld 提出了一致图规划[42]. 同年,D. Weld, C. Anderson 和 D. Smith 提出了扩展图规划以处理不确定动作和感知动作[6]. 1999 年,Blum 和 J. Langford 提出利用规划图结构来进行概率规划问题求解的算法 Pgraphplan[43]. 2003 年,谷文祥、欧华杰和姜贵栋等人开发出一个带有互斥约束的概率规划器 MPGP[44]. 2006 年,I. Little 和 S. Thiébaux 提出了图框架下的并行概率规划方法 Paragraph[45]. 实践证明,图规划框架下的不确定规划方法在实际应用中具有独特优势.

2.5.1　PGraphplan 方法

Blum 和 Langford 等人将图规划扩展到概率规划中,来处理初始状态确定,而动作结果不确定的规划问题,并设计了 PGP 概率规划器[43].

Pgraphplan 算法利用规划图结构求解初始状态已知、动作结果不确定的概率规划问题. 它采用概率分布描述动作结果的不确定性,动作执行后有若干个可能结果,每个结果附有相关的概率值. Pgraphplan 算法在给定时间限制内产生一个最优规划,得到的是从初始状态到目标状态的最优动作序列,其中"最优"是指成功概率最大. Pgraphplan 算法主要分为 2 个阶段:第 1 个阶段是在一个给定的时间步内扩张规划图;第 2 个阶段是对已创建的规划图进行前向搜索,求出成功概率最高的有效规划.

PGP 的求解过程与图规划类似,只有 2 点不同之处:1) 在连接动作结点与其添加结果、删除效果之间的边时一定要标记概率值;2) 规划图一直扩张到限定时间步 T_{max},才检查是否所有目标都出现在命题列,若出现则从初始状态开始用自顶向下的动态规划算法进行解搜索,否则说明在限定时间内找不到规划解.

图扩张完毕,进行解搜索利用. 解搜索过程类似于马尔可夫决策过程且为完全观察的 MDP,PGP 以标准的自顶向下动态规划算法作为起点,利用规划图结构所提供的信息来减少搜索范围. PGP 算法采用前向链搜索,这时图规划中的互斥信息已失效,因此 PGP 提出了几种新的有助于减少搜索空间的信息类型来加速搜索过程. 在 PGP 的搜索过程中,并没有使用互斥关系,它采用的是深度优先的状态空间搜索,所以它的速度稍慢一些.

2.5.2　MPGP 方法

PGP 采用多项式级别的规划图结构来处理初始状态的确定,而动作结果不确定的规划问题,较同类概率规划器而言,具有较快的速度. 但是同时它也存在一些问题:比如它没有利用互斥约束进行优化、不允许动作的并行执行等. 针对这一问题,2003 年,谷文祥、欧华杰和姜贵栋等人在 PGP 的基

础上成功地开发出一个带有互斥约束并可以快速生成有效规划的概率规划器 MPGP[44]. 该规划器引入命题互斥的思想及不确定环境下全新命题和命题互斥的概念,在创建规划图的过程中判定完全实例化的操作的前提是否互斥,在前提条件不互斥的情况下,才在下一动作列中添加新的动作节点,避免了在各时间步加入不会执行的动作节点,减少了规划图扩张过程中创建的节点数,节省了存储空间,提高了效率.

MPGP 算法也主要分为 2 个阶段:第 1 个阶段是在一个给定的时间步内扩张规划图;第 2 个阶段进行解搜索. 所不同的是,在第一阶段,MPGP 需要判断动作的前提条件的互斥关系. MPGP 的图扩张过程和解搜索过程类似 PGP[44].

MPGP 可以解决大部分概率规划问题,实验结果也表明它在很多方面优于 PGP.

2.5.3 Paragraph 方法

并行概率规划要解决两方面的问题:动作带有概率效果和动作在一定条件下可以并行执行[45]. Paragraph 规划器[45]的主要思想就是将整个图规划框架扩展到概率规划问题域中,利用状态信息将由 Graphplan 目标回溯搜索所找出的所有路径合并在一起生成最优并行随机规划. Paragraph 规划器能生成非循环规划和循环规划,选择性地开发利用问题潜在的并行性. Paragraph 规划器为了能包含非循环解和循环解,将规划定义为一个确定的有穷自动机,同时在规划图定义中增加了描述动作多种可能结果的结果结点和结果结点间的互斥关系. 根据对并行程度的限制级别,管理并行性分为 3 种模型:不限制并行模型,限制并行模型和非 - 并行模型. Paragraph 规划器实现了对非 - 并行模型和限制模型的处理.

Paragraph 非循环规划的生成算法:1) 根据问题描述扩展规划图,直到所有目标命题都出现且不互斥或规划图达到稳定状态,假设为前者;2) 从规划图最后一层的目标结点开始进行深度优先搜索,找出从初始状态到目标状态的所有路径;3) 从时间步 0 开始,前向模拟计算每个路径结点可能的世界状态;4) 从目标结点开始后向传递代价值,更新当前结点/状态的代价. 当后向搜索遇到新的结点或前向状态模拟计算出新的状态时,潜在的路径被搜索到并且加入到规划中. 以上 4 个步骤交替进行,后向搜索,前向状态模拟,后向代价传递,前向图扩张,直到遇到终止条件. 终止条件包括:代价为 0 的解被搜索到、超出有限的时间限定和后向搜索前规划图达到稳定状态. 最后,前向遍历搜索空间,利用贪婪选择策略提取最优随机规划.

Paragraph 循环规划的生成算法:首先扩展规划图直到规划图达到稳定状态,然后进行后向搜索,采用深度优先搜索或迭代加深搜索算法. 深度优先适用于搜索整个搜索空间,但路径被发现的顺序是不可预知的;而迭代加深搜索算法会首先找出最短路径,当仅能利用搜索空间的一个子集时这种算法更有效. 前向状态模拟与非循环算法类似,代价值的传递采用贝尔曼方程更新法,对代价函数进行更新直到其收敛在给定阈值内. 当整个搜索空间都被搜索到时,搜索结束. 解的提取与非循环算法相同.

Paragraph 的优点是它可以充分利用规划图的的互斥关系和搜索方法来提高求解效率,但它没有采用启发式的方法. 运用启发式的知识以及将 Paragraph 扩展到概率时序域上,是值得研究的方向.

2.6 基于 FOALP 的方法

Scott Sanner 和 Craig Boutilier 介绍了一种新的解决概率规划问题的方法[46],该方法把 PPDDL 表示的规划问题转化为一阶 MDP(FOMDP),并使用 FOMDP 问题的近似解决技术来得到价值函数以及与之相关的策略,其中价值函数是一阶基础函数(first-order basis functions)的集合. FOMDP 方法线性地表示价值函数,并运用 lifted、一阶近似线性规划(first-order approximate linear programming, FOALP)和近似策略迭代(first-order approximate policy iteration, FOAPI)计算合适的权值.

运用 FOMDP 方法求解概率规划问题,首先需要将 PPDDL 表示的规划转换为 FOMDP 所使用的状态演算表示. 初步的转换是将 PPDDL 的动作概要转化为状态演算中效果公式(effect axioms),然后再将后者转化为后继状态公式(successor-state axioms)[47]. FOMDP 的形式化表示以及算子操作详见文献[48-49]. 符号 $A_i(x)$ 表示 FOMDP 中参数化的动作对每一动作和变量(fluent),假设其后继状态公式已经定义,*r*Case, *v*Case 和 *p*Case 在 FOMDP 中分别表示回报值,值和转移函数.

对于 FOALP[49],作者运用带有一阶约束的线性规划 linear program (LP) 一般化从 MDPs 转到 FOMDPs 的过程:

Variables: $w_i;\ i \leq k$

Minimize: $\sum_{i=1}^{k} w_i \sum_{\langle \Phi_j, t_j \rangle \in b\mathrm{Case}_i} \frac{t_j}{b\mathrm{Case}_i}$

$$\text{Subject to}: 0 \geq B^{A}_{\max}\left(\oplus_{i=1}^{k} w_i \cdot b\text{Case}_i(s)\right) \ominus\left(\oplus_{i=1}^{k} w_i \cdot b\text{Case}_i(s)\right);\ \forall A, s. \quad (5)$$

对于 FOAPI 来说,策略迭代要求一个由值函数 $v\text{Case}(s)$ 得到的合适的一阶策略表示. 假设有 m 个参数化的动作 $\{A_1(x), \cdots, A_m(x)\}$,则策略 $\pi\text{Case}(s)$ 表示为

$$\pi\text{Case}(s) = \max\left(\bigcup_{i=1\cdots m} B^{A_i}(v\text{Case}(s))\right). \quad (6)$$

式中:$B^{A_i}(v\text{Case}(s))$ 表示可由任一实例化的动作 $A_i(x)$ 得到的值,max 操作算子保证了将最大可能值赋给每一个 s.

根据 Guestrin 等[50]提供的 factored MDPs 的近似策略迭代方法,可通过计算权 $w_j^{(i)}$ 的连续迭代来一般化近似策略迭代方法到一阶 MDPs 的过程,其中,$w_j^{(i)}$ 表示在第 i 次迭代中策略 $\pi\text{Case}^{(i)}(s)$ 的不动点值函数的最优近似. 在第 i 次迭代中通过执行下述 2 步完成:1) 从当前值函数获得策略 $\pi\text{Case}(s)$,并用式(6)获得权 $\sum_{j=1}^{k} w_{(i)j} b\text{Case}_j(s)$;2) 求解下述 LP 公式,该公式决定策略 $\pi\text{Case}(s)$ 的最小近似值函数的贝尔曼误差的权:

$$\text{Variables}: w_1^{i+1}, \cdots, w_k^{i+1};$$

$$\text{Minimize}: \Phi^{i+1)}$$

$$\text{Subject to}:$$

$$\Phi^{i+1)} \geq \left| \pi\text{Case}_A(s) \oplus\oplus_{j=1}^{k}\left[w_j^{(i+1)} b\text{Case}_j(s)\right] \ominus\oplus_{j=1}^{k} w_j^{(i+1)}\left(B^{A}_{\max} b\text{Case}_j\right)(s) \right|\ \forall A, s. \quad (7)$$

基于 FOALP 的方法可能不适合所有的域,但在 box-world 逻辑域上它是高效的. 其缺点是在当前的情况下,该方法只能处理带有全程量词回报值的情况.

2.7　基于 sfDP 的方法

Florent 和 Patrick 提出了基于 MDPs 的随机规划器 sfDP (symbolic focused dynamic programming)[51],该规划器将 PPDDL 问题转化为因子 MDPs 问题(factored MDPs),并通过一个修改的 VI(structed modified)算法来求解,其中修改的 VI 算法是基于从初始状态到目标状态的安全随机路径计算(safest stochastic path computation)的. 在进行规划求解时,1)通过使所有的转移确定化的方法计算一个状态子空间;2)在当前策略下,交替执行在当前可达的状态子空间上运用修改的 VI 算法和进行可达状态空间的扩展.

文中的规划器使用基于 ADDs(algebraic decision diagrams)的 MDPs 紧致因子(compact factored)表示法[52],ADDs 一般化了二元决策表(binary decision diagrams, BDDs),文献[51]在文献[53]的基础上使用决策表(decision diagrams)优化 MDPs. 使用该方法,需要把 PPDDL 域及规划定义转化为 ADDs-based MDP 表示. 在比赛中作者使用 CUDD 包[54]来处理 ADDs 和 BDDs.

状态空间分解(factorization)包含有二元状态变量的向量积:$S = \otimes_{i=1}^{n} v_i$,这些变量是每一个常量和目标的 PPDDL 参数化谓词的实例化,这样的变量使得在适当的时候可以处理一个状态集而不是单一的一个状态. 动作由实例化 PPDDL 参数化的动作而得到. 对于每一个动作,转移概率函数可以通过动态贝叶斯网络(dynamic Bayesian network, DBN)[55]表示. 而该 DBNs 可被编码为 ADDs[53].

此外,文中的方法不仅运用了初始状态信息,还运用了目标状态信息,以减少状态空间的搜索. 它使用确定可达性分析(deterministic reachability analysis)来计算可达子空间,通过运用初始状态和目标状态知识,在任一近似或最优动态规划计算之前计算可达子空间. 计算的首要一步是确定化所有的转移,然后高效的扩展从初始状态可达的那些状态直到至少一个目标状态被到达,而不用记忆从初始状态到目标状态的所有动作.

产生初始可达状态子空间 W 后,计算最大化到达目标状态子空间 $G(G \subset W)$ 的概率的策略,这样的策略称为安全随机路径策略(safest stochastic path policy),它最大化到达至少一个目标状态的概率. 前述得到的策略不保证在整个状态空间上是最优的,因为它只是在 $W \subset S$ 中最优. 为提高策略的质量,作者采用交替执行确定性可达分析和安全随机路径策略最优化这 2 个步骤. 当策略在先前(previous)可达状态子空间上收敛时,循环结束.

sfDP 在初始状态和目标状态知识上更加关注策略. 试验结果标明,它在一定程度上还是比较有效的.

2.8　基于 API 的方法

S. W. Yoon, A. Fern 和 R. Givan 提出了新的规划器[56],该规划器系统基于一种变化的近似策略迭代算法(approximate policy iteration),它采用归纳机器学习(inductive machine learning)和模拟进行策略改进. 给定一个规划域,系统迭代改进当前找到的最优策略直到策略不再被改进或超过一定的时间限制.

为处理大的状态空间问题,作者将系统建构在不依赖于状态空间枚举的 API(approximate policy iteration)上. 现有的 API 框架[57]通过价值函数间接的表示策略,并使用机器学习的方法来选择价值

函数的近似. 然而, 在许多域上, 尤其是在关系结构(first-order)上, 间接的表示方法比直接的表示方法更复杂, 基于此, 作者采用一个新的表示 API 表示方法[58], 该 API 直接将策略表示为状态/动作的映射. 该系统的性能主要依赖于以下 2 点: 1) 必须提供策略表示语言和关联学习者(associated learner), 该学习者允许系统找到好的策略的近似; 2) 对于复杂域, 需要提供一种引导机制来引导 API 过程. 更详细的信息请见文献[58-59].

图 2 显示了 API 的核心部件. API 的每一个迭代包含 2 个主要步骤: 策略评价和策略选择. 直观地说, 策略评价使用模拟来产生一个训练集, 该训练集描述一个关于当前策略的改进策略. 策略选择使用机器学习方法找到基于当前训练集的改进策略的一个近似解. 这样, 如果给定一个当前策略并顺序的应用这些步骤(steps), 就可以得到一个(近似)的改进策略, 系统反复这些步骤直到观察到的策略不再被改进.

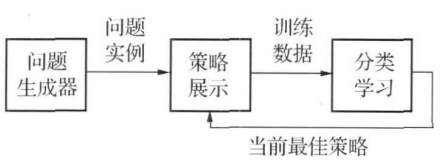


图 2 近似策略迭代

Fig. 2 Block diagram of approximation policy iteration

由于系统的归纳特性, 它不保证选择到的策略的性能. 然而, 试验结果显示了其良好的性能.

2.9 基于随机 SAT 的方法

除了上述方法以外, S. M. Majercik 和 M. L. Littman 将不确定环境下的随机规划作为随机可满足性问题 SSAT(stochastic satisfiability)来处理[60]. 他们提出了一种新的规划技术, 将概率随机规划问题转化为 SSAT 实例进行求解, 首次将可满足性规划问题扩展到随机域, 并在此基础上设计了规划器 Zander.

一个确定性的可满足问题要求一个可满足的赋值, 即指定公式中变量的真值, 使公式的值为真. 令 $x = < x_1, x_2, \cdots, x_n >$ 是 n 个布尔变量的集合, $\phi(x)$ 是这些变量上的布尔公式, 一个赋值 A 是可满足的且 $\phi(x)$ 是可被满足的, 如果在映射 A 下 $\phi(x)$ 的值为 True. 用公式表示为 $\exists x_1, \cdots, \exists x_n(E[\phi(x) \leftrightarrow \text{True}] = 1.0)$. 其中, $\phi(x)$ 是带有 m 个子句的合取范式(CNF), 每个子句是文字的析取, 一个文字是一个变量或变量的否定. ϕ 为 True 当且仅当在每个子句中至少存在一个真值为 True 的文字. 一个赋值是从 x 到集合{True, False}的映射.

随机满足 SSAT 的关键是引入一个随机变量 $\mathcal{R}$. 随机变量将不确定性引入问题中, 而不管问题是否存在一个可满足的赋值. 一个 SSAT 公式由 3 元组 (ϕ, Q, θ) 定义, 其中 ϕ 是变量 $x_1, x_2, \cdots, x_n$ 上的合取范式, Q 是从变量 $(x_1, x_2, \cdots, x_n)$ 到量词(存在量词和随机量词 $\mathcal{R}$)的映射, $\theta \in [0, 1]$ 是可满足阈值. 定义 $\phi\lceil_{xi=b}$ 是 $(n-1)$-变量 CNF 公式, 它由在 n 变量 CNF 公式 ϕ 中为单一变量 x_i 指定布尔值 b 后得到. 在量词顺序 Q 下, ϕ 可满足的最大概率 $\text{val}(\phi, Q)$ 定义为量词上的归纳: 令 x_1 是与最远的量词相联系的变量,

1) 如果 ϕ 包含一个空的子句, 则 $\text{val}(\phi, Q) = 0$;

2) 如果 ϕ 不包含子句, 则 $\text{val}(\phi, Q) = 1$;

3) 如果 $\phi(x_1) = \exists$, 则 $\text{val}(\phi, Q) = \max(\text{val}(\phi\lceil_{x1=0}, Q), \text{val}(\phi\lceil_{x1=1}, Q))$;

4) 如果 $\phi(x_1) = \mathcal{R}$, 则 $\text{val}(\phi, Q) = \max(\text{val}(\phi\lceil_{x1=0}, Q) + \text{val}(\phi\lceil_{x1=1}, Q))/2$.

给定 ϕ、Q、θ、(ϕ, Q, θ) 为 True 当且仅当 $\text{val}(\phi, Q) \geq \theta$.

解决 SSAT 问题的算法是 EVALSSAT. 给定任意 SSAT 的实例 (ϕ, Q, θ), 算法保证返回一个正确的值. EVALSSAT 算法可看作是 DPLL[61]算法的扩展, DPLL 是 SSAT 求解器的基础, 它枚举所有可能的赋值, 并在任意可能的时刻简化公式. 它的简化方法(或者说是剪枝规则)使得解决整个赋值集合不能被完全枚举的问题是可能的. EVALSSAT 算法以 θ、Q、低阈值 θ_l 和高 θ_h 作为输入. 算法返回一个小于 θ_l 的值, 当且仅当 SSAT 公式的值小于 θ_l 时; 算法返回一个大于 θ_h 的值, 当且仅当 SSAT 公式的值大于 θ_h 时; 否则返回 SSAT 公式的值.

随机可满足 SSAT 是概率规划技术的核心, Zander 将随机规划问题转化为 SSAT 的实例, 然后进行求解. 它首先将随机规划问题编码为 SSAT 问题, 转换单元是一个 Java 程序, 它以 PPL(一种高层动作语言, 由动作语言 AR[62]扩展而来)表示的规划问题作为输入, 并将输入转化为 SSAT 公式. 之后, Zander 进行求解. Zander 的求解单元是 C++程序, 以 SSAT 表示的规划问题作为输入, 并找到一棵赋值树. 求解器为所有可能的赋值构造二叉树. 它通过执行 DPLL-based 算法对树进行一个深度优先搜索, 并通过对每一节点计算到目前为止给定的偏序赋值的概率构建一棵子树. 求解器通

过决定具有最大成功概率的子树来找到最优解.

Zander 在不确定环境下,运用随机可满足性来求解随机规划,它接受以命题形式表示的部分可观察 MDP,并将其高度一般化. 它可以处理初始状态不确定,观察不确定和转移不确定的规划问题. 试验结果表明,在许多问题上,Zander 优于其他方法.

3　国际概率规划比赛

3.1　第 4 届国际规划比赛(IPC-4,2004 年)

2004 年 6 月,第 4 届国际规划比赛 IPC-4(international planning competition)[63] 在加拿大的 Whistler 举行. 此次国际规划比赛首次设立了概率规划的比赛,概率问题域描述语言为 PPDDL1.0,有共有 10 个不同的规划器参与了比赛.

在所有域上,规划器 J3 取得冠军,而规划器 P 则取得了亚军的好成绩. 其中,J3 采用的是重规划(FF-replan)方法,它是一个带有启发式方法的算法;而规划器 P 即是 mGPT,它所使用的方法是 2.2.2 节介绍的 LRTDP 算法. 在 Overall,Non-Blocks/Box 域上,规划器 C 取得冠军,它是一种符号启发式搜索,它所使用的算是 LAO*,在 2.4 节已经介绍;而在 Domain-specific 域上,规划器 J1 取得了第一的成绩. 事实上,J1 和 J3 为同一组参赛人员所设计,J1 采用 Classy's 策略语言表示规划,它所用的算法在 2.8 节已经介绍.

各规划器按顺序进行比赛,设定的 6 个类别分别是: Overall、Goal-based、Overall、Non-Blocks/Box、Domain-specific、Conformant. 比赛的结果如下表所示:

表 1　概率规划比赛分类结果

Table 1　Summary of competition results by category

Category	1st	2nd
Overall	J3	P
Goal-based Domains	J3	
Overall, Non-Blocks/Box	C	
Domain-specific, No Tuning	J3	
Domain-specific	J1	

3.2　第 5 届国际规划比赛(IPC-5,2006 年)

继 IPC-4 之后,2006 年在英国的 Cumbria 召开的第 5 届国际规划比赛 IPC-5[64] 也分为 2 个部分,确定性组和概率组. 概率组包含具有完全可观察的概率规划问题,比赛的焦点是能够传达实时决策作为完全策略的规划器,并使用 IPC-4 开发的客户端/服务器结构来评估规划器的质量.

FOALP、sfDP、FPG、Paragraph 这 4 个规划系统参加了概率部分的比赛. 大赛使用了 9 个域,每一个域上设计了 15 个问题实例. 规划器通过每个实例 30 轮共 4 050 轮的比赛来评价其性能. 表 2 显示了每个规划器完成的成功轮数的总数、平均时间和每一轮的平均步数. 最后由来自澳大利亚的 Olivier Buffet 和 Douglas Aberdeen 设计的 FPG,获得了冠军,其他规划器的名次依次如下: FOALP, Paragraph, sfDP. 这几个规划所使用的算法已分别在第 2 节中介绍.

表 2　比赛中规划器完成的轮数和平均时间

Table 2　Average time and turns in the competition by planner

规划器	成功轮数	占总轮数的 %	平均时间/轮
FOALP	1 376	33.90	13 735.70
sfDP	302	7.45	13 719.50
FPG	2 199	54.29	4 232.89
Paragraph	644	15.90	363.12

4　结束语

文中对国内外概率规划研究的方法进行了总结,介绍解决概率规划问题的较为流行的方法,并对国际概率规划比赛进行了介绍. 目前概率规划能处理的问题主要集中在完全可观察环境下能用 PPDDL 语言表示的问题,而对于部分可观察环境下的问题研究不多. 同时,当前的概率规划认为动作的发生是瞬时的,没有考虑动作的持续性,因此在之后的研究工作中,在概率规划中应该考虑加入时态约束. 另外,目前概率规划在衡量规划时只希望找到 1 条最大概率规划而不考虑其他资源,事实上,在实际问题中,还需要考虑资源约束. 随着研究的不断深入,相信概率规划方法会越来越完善.

参考文献:

[1] WELD D. Recent advances in AI planning[R]. UW-CSE-98-10-01, 1999.

[2] CIMATTI A, ROVERI M. Conformant planning via symbolic model checking[J]. Journal of Artificial Intelligence Research, 2000, 13: 305-338.

[3] CIMATTI A, ROVERI M, BERTOLI P. Conformant planning via symbolic model checking and heuristic

search[J]. Artificial Intelligence, 2004, 159(1-2):127-206.

[4] HOFFMAN J, BRAFTMAN R. Conformant planning via heuristic forward search: a new approach[J]. Artificial Intelligence, 2006, 170(6-7): 507-541.

[5] SMITH D, WELD D. Conformant graphplan[C]// Proceedings of 15th National Conference on Artificial Intelligence. Madison, Wisconsin, 1998.

[6] WELD D, ANDERSON C, SMITH D. Extending graphplan to handle uncertainty & sensing actions[C]// Proceedings of 15th National Conference on Artificial Intelligence. Madison, Wisconsin, 1998.

[7] HOFFMANN J, BRAFMAN R I. Contingent planning via heuristic forward search with implicit belief states [C]// Proceedings of Fifteenth International Conference on Automated Planning & Scheduling. [S. l.]: AAAI Press, 2005.

[8] WAH B W, CHEN Y. Constrained partitioning in penalty formulations for solving temporal planning problems [J]. Artificial Intelligence, 2006, 170(3): 187-231.

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

[10] FOX M, LONG D. PDDL2.1: an extension to PDDL for expressing temporal planning domains[J]. Journal of Artificial Intelligence Research, 2003, 20: 61-124.

[11] PHILIPPE L. Algorithms for propagating resource constraints in AI planning and scheduling: existing approaches and new results [J]. Artificial Intelligence, 2003, 143(2):151-188.

[12] HANSEN E, ZILBERSTEIN S. LAO *: a heuristic search algorithm that finds solutions with loops[J]. Artificial Intelligence, 2001, 129(1-2): 35-62.

[13] BONET B, GEFFNER H. Improving the convergence of realtime dynamic programming[C]// Proceedings of 13th International Conference on Automated Planning and Scheduling(ICAPS). Trento, Italy, 2003.

[14] BONET B, GEFFNER H. An algorithm better than AO *? [C]// Proceedings of AAAI-05. Pittsburgh, Pennsylvania: AAAI Press, 2005.

[15] MCDERMOTT D. The 1998 AI planning systems competition[J]. AI Magazine, 2000, 21(2): 35-55.

[16] GHALLAB M, HOWE A E, KNOBLOCK C A, et al. PDDL-the planning domain definition language [R]. CVC TR-98-003/DCS TR-1165, 1998.

[17] YOUNES H, LITTMAN M L, WEISSMAN D, et al. The first probabilistic track of the international planning competition[J]. Journal of Artificial Intelligence Research, 2005, 24: 851-887.

[18] YOUNES H, LITTMAN M. The language for the probabilistic part of IPC-4[C]// Proceedings of International Planning Competition. Whistler, Canada, 2004.

[19] DEARDEN R, BOUTILIER C. Abstraction and approximate decision-theoretic planning[J]. Artificial Intelligence, 1997, 89(1-2):219-283.

[20] BERTSEKAS D P. Dynamic programming and optimal control[M]. Belmont: Athena Scientific, 1995.

[21] BELLMAN R E. Dynamic programming[M]. Princeton: Princeton University Press, 1957.

[22] RUSSELL S, NORVIG P. 人工智能——一种现代方法[M]. 2版. 姜 哲, 金奕江, 译. 北京: 人民邮电出版社, 2004.

[23] BERTSEKAS D P. Dynamic programming: deterministic and stochastic models[M]. New Jersey: Prentice-Hall, 1987.

[24] BARTO A, BRADTKE S, SINGH S. Learning to act using real-time dynamic programming[J]. Artificial Intelligence, 1995, 72(1-2): 81-138.

[25] KORF R. Real-time heuristic search[J]. Artificial Intelligence, 1990, 42(2-3): 189-211.

[26] GARDNER M. Mathematical games[J]. Sci Amer, 1973, 228(1): 108.

[27] WATKINS C J. Learning from delayed rewards[D]. Cambridge: Cambridge University, 1989.

[28] BONET B, GEFFER H. mGPT: a probabilistic planner based on heuristic search[J]. Journal of Artificial Intelligence Research, 2005, 24: 933-944.

[29] BONET B, GEFFNER H. Faster heuristic search algorithms for planning with uncertainty and full feedback [C]// Proceedings of the 18th International Joint Conference on Artificial Intelligence. Acapulco, Mexico, 2003.

[30] TARJAN R E. Depth first search and linear graph algorithms[J]. SIAM Journal on Computing, 1972, 1(2): 146-160.

[31] BONET B, GEFFNER H. Learning in depth-first search: a unified approach to heuristic search in deterministic, non-deterministic, probabilistic, and game tree settings [R]. Caracas: Universidad Simon Bolivar, 2005.

[32] BONET B, GEFFNER H. Learning depth-first search: a unified approach to heuristic search in deterministic and non-deterministic settings, and its application to MDPs[C]// Proceedings of 16th International Conference on Automated Planning and Scheduling. Cumbria,

UK, 2006.

[33]BUFFET O, ABERDEEN D. The factored policy gradient planner[C]// Proceedings of the Fifth International Planning Competition. Cumbria, UK, 2006.

[34]MAUSAM, WELD D. Concurrent probabilistic temporal planning[C]// International Conference on Automated Planning and Scheduling (ICAPS). Monterey, CA, USA, 2005.

[35]BAXTER J, BARTLETT P, WEAVER L. Experiments with infinite-horizon[J]. Journal of Artificial Intelligence Research, 2001, 15: 351-381.

[36]YOUNES H L S, LITTMAN M L. An extension to PDDL for expressing planning domains with probabilistic effects[R]. CMU-CS-04-167, Carnegie Mellon University, 2004.

[37]FENG Z, HANSEN E. Symbolic heuristic search for factored Markov decision processes[C]// Proceedings of the 18th National Conference on Artificial Intelligence (AAAI-02). Edmonton, Alberta, Canada, 2002.

[38]BLUM A, FURST M. Fast planning through planning graph analysis[J]. Artificial Intelligence, 1997, 90(1): 281-300.

[39]YANG Q, WU K H, JIANG Y F. Learning action models from plan examples using weighted MAX-SAT [J]. Artificial Intelligence, 2007, 171(2-3):107-143.

[40]HUANG W, WEN Z H, JIANG Y F, et al. Observation reduction for strong plans[C]// Proceedings of the 20th International Joint Conference on Artificial Intelligence. Hyderabad, India, 2007.

[41]**陈蔼祥**, **姜云飞**, **张学农**, **等**. **GP-基于规划图的遗传规划算法**[J]. **计算机学报**, 2007, 30 (1): 153-160.
CHEN Aixiang JIANG Yunfei, ZHANG Xuenong, et al. GP: genetic planning algorithm based on planning graph[J]. Chinese Journal of Computers, 2007, 30(1): 153-160.

[42]SMITH D E, WELD D. Conformant graphplan[C]// Proceedings of 15th National Conference on Artificial Intelligence. Madison, Wisconsin, USA, 1998.

[43]BLUM A, LANGFORD J. Probabilistic planning in the graphplan framework[C]// Proceedings of the Fifth European Conference on Planning. Durham, UK, 1999.

[44]GU W X, OU H J, LIU R X, et al. An improved probabilistic planning algorithm based on pgraphplan[C]// Proceedings of the Third International Conference on Machine Learning and Cybernetics. Shanghai, 2004.

[45]LITTLE I, BAUX S. Concurrent probabilistic planning in the graphplan framework[C]// The 16th International Conference on Automated Planning and Scheduling (ICAPS). Cumbria, UK, 2006.

[46]SANNER S, BOUTILIER C. Probabilistic planning via linear value-approximation of first-order MDPs[C]// The 16th International Conference on Automated Planning and Scheduling(ICAPS). Cumbria, UK, 2006.

[47]REITER R. Knowledge in action: logical foundations for specifying and implementing dynamical systems [M]. Cambridge: MIT Press, 2001.

[48]BOUTILIER C, REITER R, PRICE B. Symbolic dynamic programming for first-order MDPs[C]// Proceedings of the 17th International Joint Conference on Artificial Intelligence. Seattle, WA, 2001.

[49]SANNER S, BOUTILIER C. Approximate linear programming for first-order MDPs[M]. Arlington, Virginia: AUAI Press, 2005.

[50]GUESTRIN C, KOLLER D, PARR R, et al. Efficient solution methods for factored MDPs[J]. Journal of Artificial Intelligence Research, 2002, 19: 399-468.

[51]KOENIGSBUCH F T, FABIANI P. Symbolic stochastic focused dynamic programming with decision diagrams[C]// Proceedings of International Planning Competition. Cumbria, UK, 2006.

[52]BAHAR R I, FROHM E A, GAONA C M, et al. Algebraic decision diagrams and their applications [C]// IEEE/ACM International Conference on CAD. Santa Clara, USA, 1993.

[53]HOEY J, ST-AUBIN R, HU A, et al. Optimal and approximate stochastic planning using decision diagrams [R]. TR-2000-05, University of British Columbia, 2000.

[54]SOMENZI F. Cudd: cu decision diagram package release[Z]. University of Colorado at Boulder, 1998.

[55]DEAN T, KANAZAWA K. A model for reasoning about persistence and causation[J]. Computational Intelligence, 1989, 5(3): 142-150.

[56]YOON S W, FERN A, GIVAN R. Learning reactive policies for probabilistic planning domains[C]// Proceedings of International Planning Competition. Whistler, CA, 2004.

[57]BERTSEKAS D P, TSITSIKLIS J N. Neuro-dynamic programming[M]. Belmont: Athena Scientific, 1996.

[58]FERN A, YOON S, GIVAN R. Approximate policy iteration with a policy language bias[C]// Proceedings of the Neural Information Processing Conference. Istanbulm, Turkey, 2003

[59]FERN A, YOON S, GIVAN R. Learning domain-specific control knowledge from random walks[C]// Pro-

ceedings of 14th International Conference on Automated Planning and Scheduling. Whistler, British Columbia, Canada, 2004.

[60] MAJERCIK S M, LITTMAN M L. Contingent planning under uncertainty via stochastic satisfiability[J]. Artificial Intelligence, 2003, 147(1): 119-162.

[61] DAVIS M, LOGEMANN G, LOVELAND D. A machine program for theorem proving[J]. Comm ACM 5, 1962, 5(7): 394-397.

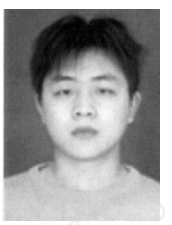
[62] GIUNCHIGLIA E, KARTHA G N, LIFSCHITZ V. Representing action: indeterminacy and ramifications [J]. Artificial Intelligence, 1997, 95(2): 409-438.

[63] YOUNES H S, LITTMAN M L, WEISSMAN D, et al. The first probabilistic track of the international planning competition[J]. Journal of Artificial Intelligence Research, 2005, 24: 851-887.

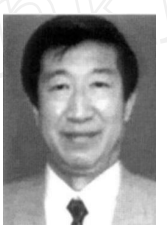
[64] BONET B, GIVAN B. Results of probabilistic track in the 5 th international planning competition[C]// Proceedings of International Planing Competition. Cumbria, UK, 2006.

作者简介:

闫书亚,女,1982年生,硕士研究生,主要研究方向为智能规划和规划识别.



殷明浩,男,1979年生,助教,博士研究生,主要研究方向为自动推理和智能规划.



谷文祥,男,1947年生,教授,博士生导师,主要研究方向为智能规划和规划识别、形式语言与自动机理论、模糊数学及其应用,发表论文百余篇.

第7届基于Web学习国际会议

The 7th International Conference on Web-based Learning (ICWL)

ICWL国际会议是由香港万维网科技协会协同ACM-HK和IEEE-HK组织的每年举办一次的有关基于Web学习的国际会议.自2002年首次在香港举办以来,已分别在澳大利亚(2003)、中国(2004)、中国香港(2005),马来西亚(2006)和英国(2007)成功举办了6届.ICWL2008将是第7次国际会议,这次会议将在中国国家历史文化名城浙江金华举办.本次大会主席由中国著名计算机专家何积丰院士、浙江师范大学数理与信息工程学院院长赵建民教授、香港城市大学李青教授、英国Durham大学刘永雄教授和David Budgen教授共同担任,程序委员会主席由中国台湾淡江大学Timothy Shih教授、英国Durham大学Frederick Li博士、美国Dennis McLeod教授共同担任;中国计算机学会、中国国家自然科学基金委员会领导将专程赴会.

征稿范围(不仅限于此):

- E-Learning Platforms and Tools
- Web-based Learning for Oriental Languages
- Mobile e-learning
- Learning Resource Deployment, Organization and Management
- Design, Model and Framework of E-learning Systems
- E-learning Standards
- Practice and Experience Sharing
- Pedagogical Issues

详见会议网站:http://icwl2008.zjnu.cn.