图规划框架下的启发式搜索的研究与发展

谷文祥[1] 王改革[1] 殷明浩[1] 孙 焱[2,3]
(东北师范大学计算机学院 长春 130117)[1] (中国科学院研究生院 北京 100049)[2]
(中国科学院长春光学精密机械与物理研究所 长春 130033)[3]

摘 要 随着智能规划研究的深入,以往的规划器已不能满足实际应用的需要。为了提高规划器求解实际问题的能力,启发式搜索产生了。对近10年来各种启发式搜索方法进行了分析,指出了它们的优缺点,并进行了比较。同时对智能规划及其启发式搜索的未来发展方向进行了分析与预测,旨在让研究和关心该领域的学者较为全面地了解这一领域。

关键词 人工智能,智能规划,启发式搜索,图规划
中图法分类号 TP18 **文献标识码** A

Research and Development of Plangragph Based on Heuristic State Search

GU Wen-xiang[1] WANG Gai-ge[1] YIN Ming-hao[1] SUN Yan[2,3]
(School of Computer Science, Northeast Normal University, Changchun 130117, China)[1]
(Graduate School of Chinese Academy of Sciences, Beijing 100049, China)[2]
(Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China)[3]

Abstract Traditionally, planning problems are cast in terms of imperative constraints that are either wholly satisfied or wholly violated. Heuristic search is generated and improves the performance of planner. A variety of heuristic searches were argued in the near decade, then the pros and cons of it were showed. At once, the future of heuristic search was also introduced.

Keywords AI, Intelligent planning, State heuristic search, GRAPHPLAN

1 引言

智能规划自20世纪60年代产生以来,就备受人们关注。特别是近十几年,规划系统的求解效率有了很大提高。这得益于以下3种方法在规划中的应用:(1)图规划(GRAPHPLAN)[1],它使得规划有了革命性的发展,是规划发展史上的里程碑。(2)把规划问题转化为命题SAT[2](比如规划器SATPLAN[3]和BLACKBOX[4])。(3)HSP(heuristic search planning)[5]。由于HSP在AIPS98中表现非常出色,在以后的历届AIPS中,大多数规划器都采用了启发式搜索技术。本文对10年来国内外出现的采用启发式搜索技术的规划器进行了深入的研究,并指出了它们的优缺点,进而给出了改进方法。研究启发式搜索技术的发展过程与方向,对于今后设计解决实际问题的启发式函数很有指导意义。

通常,求解规划问题是困难的。在纯STRIPS规划中,如果忽略删除效果,规划的求解难度会从PSPACE降低到P[6]。限制长度、忽略删除效果的放松规划问题仍然是NP-Hard的,不限制长度时,是PSPACE-Complete的。而对于非经典规划问题,求解就更为困难。例如不确定性规划问题是EXP-Compolete到2EXP-Complete的,概率规划是从NPPP-Complete到不可判定的(Undecidable)。因此,怎样提高规划器的性能,成为非常值得研究的问题。在这方面,主要有问题公式化、搜索空间、启发式搜索等方法。其中以启发式搜索效果最好,并且可以结合其他方法。求解启发式的方法有很多,主要有取层号法、最大启发值法、取和法、放松规划(RP, Relaxed Plans)等。此外,还可以利用互斥对启发式进行提高。启发式搜索可以用来指导前向、后向和POCL(partial order causal link)搜索等。

2 图规划框架下的启发式搜索方法的最新进展

受20世纪60年代中期Drew McDermott的在状态空间启发式规划的启发,许多研究者转向了这个方向。在过去的几年,许多研究者在域独立、状态空间、启发式规划这些领域进行了深入的研究,并且开发出一些性能很高的规划器。这些规划器通常接受STRIPS[7]公式。在STRIPS中,动作A由3部分构成:$pre(a)$,$add(a)$和$del(a)$,它们分别表示动作A的前提、添加效果和删除效果。它们都是命题的集合。一个规划任务P是一个三元组$\langle A,I,G\rangle$,这里I是初始状态,A

到稿日期:2008-12-15 返修日期:2009-03-09 本文受国家自然科学基金项目(编号:60473042,60573067和60803102)资助。
谷文祥(1947—),男,教授,博士生导师,主要研究方向为智能规划与规划识别、形式语言与自动机理论、模糊数学及其应用,E-mail:gwx@nenu.edu.cn;王改革(1984—),男,硕士生,主要研究方向为智能规划与规划识别;殷明浩(1980—),男,讲师,主要研究方向为智能规划与规划识别、自动推理;孙 焱(1979—)男,博士生,主要研究方向为图像处理。

是可用动作集，G 是目标集。状态被表示成一些命题的集合。基于规划任务 $P=(A,I,G)$ 的放松规划任务 P'[8,9] 定义为

$P'=(A',I,G)$

$A'=\{(pre(a),add(a),\phi)|(pre(a),add(a),del(a))\in A\}$

放松规划是不可纳的(admissible)，但是效率非常高。寻找一个最优的放松规划解是 NP-hard 的。

为了叙述方便，我们再定义一些概念。

状态空间(state space)：是一个四元组 (S,A,f,c)，其中 S 为有限状态集合，f 为状态转换函数，c 为成本函数 $c(a,S)>0$(它表示在状态 S 上执行动作 a 的代价)。如果一个状态空间加上一个给定的初始状态 I 和目标集 G，我们称其为一个状态模型，即一个状态模型是一个六元组 (S,I,G,A,f,c)，其中 S 是有限非空状态集；$I\in S$ 是初始状态；G；$G\supset S$ 是非空目标集；$A(S)$：$\supset A$ 是在状态 S 上能够执行的动作；$f(a,S)$ 是对于所有 S，$a\in A(S)$；$c(a,S)$ 是在状态 S 上执行动作 a 的代价。

2.1 UNPOP：第一个启发式搜索规划器(1996)

在智能规划发展的早期，很多人对偏序规划(POP，partial order planning)进行了研究，并开发出了许多规划器[10-15]，但它们的性能不是很理想。其中，比较著名的一个规划器是 UCPOP[16]，它能够处理 ADL 动作描述、条件效果[17] 和全称量化的前提和效果。UCPOP 没有使用任何启发式，在此不做具体介绍(有关偏序规划的基础知识见文献[11])。下面重点介绍它的扩展——UNPOP。

在状态空间启发式规划中，UNPOP 是第一个规划器。UNPOP 通过建立规划图来扩展著名的 Means-ends 分析——GRMG(greedy regression-match graph)，它包括次目标和能够达到次目标的动作。一个规划问题的次目标是问题的目标和能够得到这些目标动作的前提。GRMG 的创建从问题目标开始，直到所有的次目标到达问题的最后一层，即初始状态。从图中得到的信息可以用于搜索阶段，这有两个目的：a)估计给定目标 S' 和初始状态之间的距离；b)对图中没有出现的动作进行剪枝。搜索从目标开始，后向推进，在每个状态都重建 GRMS。

2.2 HSP：第一个性能卓著的启发式搜索规划器(1998)

GRAPHPLAN 的出现是规划发展史上的一个里程碑。GRAPHPLAN 以其卓越的性能赢得了人们的重视。但在 AIPS98 中，HSP 超越了 GRAPHPLAN 和 SATPLAN，一举赢得了冠军。HSP[18,19] 的确非常好，与 IPP[20]，STAN[21]，BLACKBOX 比较，HSP 能求解更多的问题。

在 HSP 中，它定义了一个估价函数：

$$g_s(p)=\begin{cases}0, & \text{如果 } p\in s\\ \min op\in O(p)[1+g_s(\text{Prec}(op))], & \text{其他}\end{cases}$$

其中，$g_s(p)$ 代表在状态 S 上得到命题 P 的代价。采用的启发式函数是 $h(s)=g_s(p)$。HSP 采用的是累加启发值(h_{add})。

$$g_s^+(c)=\sum_{r\in C}g_s(r)$$

HSP 用 h_{add} 来指导爬山算法。爬山算法非常简单，每步都选择最好的后继去扩展、循环，直到实现目标。因为累加启发值 h_{add} 会高估代价，因此它是不可纳的。事实上，参加 AIPS98 的是 HSP1.2，用 C 语言书写。

2.3 HSPr：后向搜索规划器(1999)

在 HSP 中，主要瓶颈是对于每个新状态都要重新计算启发值。在 HSPr 中，一个重要的改进就是改变搜索方向来避免这个问题。

HSPr 从目标开始后向搜索，而不是从初始状态开始的前向搜索，即回溯搜索。在 HSPr 中，最重要的一点就是 $g(p)$ 是从初始状态 I 开始到达命题 P 的估计代价，这个值不用每次计算。换句话说，在 HSPr 中，估计代价 $g(p)$ 只从 I 计算一次，就可以用于定义任何状态的启发值。这是 HSP 和 HSPr 最大的不同点。在 HSPr 中，采用的启发式函数是 $h(s)=\sum_{p\in s}g(p)$

在 HSP 中，计算 $h(s)$ 时，目标固定。随着状态 s 的改变，每次进入一个新的状态，必须重新进行计算；而 HSPr 在计算 $h(s)$ 时，固定初始状态 I，计算到达每个状态的值。在解的提取阶段，HSPr 利用启发值指导进行后向搜索，这与图规划类似。此外，HSPr 采用了类似于图规划的互斥来进行剪枝。

最后，对 HSP 和 HSPr 使用的启发式函数总结如下：

$h(s)=\sum_{p\in G}g(p,s)\quad hr(s)=\sum_{p\in s}g(p,I)$

HSP 中使用的非可纳的启发值 h_{add} 来源于一个放松问题的最优代价函数的一个估计值。在这里，删除效果被忽略了。这种忽略明显存在两个问题：(1)这个估计值并不好，因为它忽略了次目标之间的促进作用(一个目标的实现可以使另一个子目标变得简单)；(2)放松是不好的，它忽略了次目标之间的负相互作用。这需要一个更好的启发值来指导搜索。尽管 HSPr 在某些领域比 HSP 性能优越，但是 HSPr 只能找到合理的规划解，并不难找到最优解。此外，HSPr 占用内存比较大。

2.4 GRT：使用表结构的规划器(1999)

GRT(Greedy Regression Tables)[22-24] 参加了 AIPS00 比赛，但是没有拿到名次，这给未来的工作提供了一个好的方向。这里只简单介绍它的工作原理。GRT 是 HSP 的扩展，能够处理 STRIPS 世界的域独立规划器。它使用的启发式是估计每一个当前状态和目标的距离，可以用于指导任何状态空间规划器的搜索过程。在问题求解过程的开始，首先建立一个表，表里的记录是域中的命题以及从目标开始估计的距离。另外，记录还包括当同时得到不同命题时递归所需要的信息。在解搜索阶段，使用表的启发值，能提取出非常精确的从当前状态到目标的距离。我们把这个启发式算法叫做 GRT。一个使用了启发式的简单的最好优先搜索(BFS，best-first search)规划器已经用 C++ 实现了，并用 AIPS98 中的实例进行了测试。结果表明，GRT 在所有实例上速度都较快，并在大多数实例上得到了较短的规划解。

在 GRT 中，使用的启发式来源于 ASP[25]。在 ASP 中，首先定义

$$g(p,s)=\begin{cases}0, & \text{如果 } p\in s\\ i+1, & \text{对于 } C\rightarrow p,\sum_{r\in C}g(r,s)=i\\ \infty, & \text{如果从 } s \text{ 不能达到 } p\end{cases}$$

对于命题集 S，定义

$g(S,s)\overset{def}{=}\sum_{q\in P}g(q,s)$

ASP 中使用的启发式

$h_{ASP}(s)\overset{def}{=}g(G,s)$

其中，G 为问题的目标。在对基本化的动作与命题进行计算时，$H_{ASP}(s)$ 的复杂度都是线性的。这个函数总是过高估计到

达目标的代价，因此 ASP 所有的启发式是不可纳的。

$h_{ASP}(s)$有两个明显的缺点：1)对于每一个状态都要重新进行计算；2)没有考虑次目标之间的相互作用。在 GRT 中，用 h_{GRT}来表示新的启发值。h_{GRT}采用从目标到当前状态的后向估计距离。在求解过程的开始，从目标开始估计所有命题的距离。在求解过程中，用上面的估计值计算从目标到当前状态的距离。通常情况下，计算两个状态之间的距离，使用前向和后向方法往往是不同的(虽然它们都是不精确的)。然而，我们没有理由选择一个而不选择另一个。它通过翻转动作来求得这个启发值。具体参见文献[22]。

此外，在 GRT 的基础上，Ioannis Refanidis，Ioannis Vlahavas 于 2003 年开发出了能够处理多目标的规划器，即 MO-GRT(Multiobjective GRT)[26]。

2.5 FF：前向搜索规划器(2000)

FF[27]是一款性能非常优异的规划器，它在 IPC-2 中获得了第一名的好成绩。FF 中使用的启发式如下：

$$h(s):=\sum_{i=0,\cdots,n-1}|a_i|$$

其中，a_i 是时间步 i 所选的并行动作的集合，n 是表示包含了所有目标的第一个命题层的层数。实验表明，这种方式所获得的估计值通常要比 HSP 估计值小，因为在求解一个规划时要考虑命题之间的积极作用。同时，FF 采用了两个启发式优化技术——NOOP 优先启发式(NOOP-FIRST)和难易程度启发式。

在 GRAPHPLAN 中，使用了默认的 NOOP 优先启发式。也就是，若存在一个 NOOP 支持命题 p，那么在规划器尝试选择其它实动作支持 p 前，优先考虑该 NOOP。对于放松任务，NOOP 优先启发式确保了所返回的规划是一个最小度量。使用这个策略，使得 GRAPHPLAN 所返回的规划将包含每个动作至多一次。如果能通过 NOOP 到达一个命题，那就这样做。问题是，当 NOOP 不可用时，应该选择哪个支持动作？一个好的想法就是选择支持动作的前提看起来“简单”。从规划图的创建阶段来看，能得到一个简单的度量——对一个动作的前提的难易程度的度量，如下式所示：

$$\text{difficulty}(o):=\sum_{p\in pre(o)}\min\{i \mid p \text{ 为时间步 } i \text{ 命题层中的成员}\}$$

当每个动作被首次插入图中时，它的 difficulty 就被设置完成。在规划提取过程中，当遇到一个命题的 NOOP 不可用时，就简单地选择一个带有最小 difficulty 的支持动作。这个启发式在有多个动作支持同一个命题时运行得很好。但是，一些动作的 difficulty 必须少于其他动作的 difficulty。

FF 中使用的搜索算法是爬山算法的一个改进算法——加强爬山算法。

在 AIPS98 竞赛中使用的 HSP1(实际参加比赛的是 HSP1.2)就是 HSP 的第一个版本。它使用的是一个改进的爬山算法搜索策略，此策略通常选择当前状态的一个最优后继。选择使用局部搜索，是因为状态估计的代价是昂贵的，希望使用尽可能少的状态估计来到达目标状态。

FF 中采用了一个不同的搜索算法，即一个爬山的加强形式，它结合了局部搜索和全局搜索(具体算法，参见文献[27])。与爬山算法相似，加强爬山算法也是从初始状态出发的。那么，对于中间搜索状态 s，从 s 出发，使用完整的广度优先搜索。这个算法或者找到最近的较优后继(而不是最优后继)，即带有严格较优估计的最近的状态 s'，或者失败。在后种情况下，整个算法失败，在前种情况下，把从 s 到 s'的路径添加到当前规划中，并且迭代执行该算法，直到到达目标状态——估计值为零的状态，搜索结束。

对从 s 出发的广度优先算法的执行进行标准化，所产生的状态保存在队列中。搜索反复删除队列的第一个状态 s'，并且通过运行 GRAPHPLAN 对其进行估计。如果这个估计值优于 s 的估计值，搜索成功。否则，把 s'的后继放到队列尾部(避免平原和局部极小)。为了避免重复访问同一状态，把访问过的状态放在内存中的一个哈希表中。如果没有新的状态出现，广度优先搜索失败。此外，FF 中另一个重要的创新点是帮助动作的使用。状态 s 帮助动作集 $H(S)$定义为：

$$H(s):=\{O(\text{pre}(o)\subseteq S,\text{add}(o)\cap G_1(S)\neq\emptyset\}$$

其中，$G_1(S)$表示当(A',S,G)任务开始时在第一时间步放松的 GRAPHPLAN 所创建的目标集合。总之，把这些在第一时间步可应用的并且能至少添加一个目标(命题)的动作视为帮助动作。

为了给读者一个 FF 系统体系结构的概览，图 1 显示了 FF 是如何安排它的最基本的技巧。

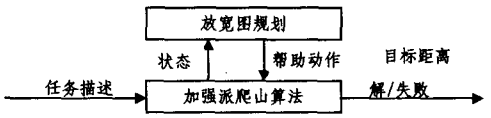

图 1 FF 系统基本体系结构

2.6 MIPS：使用了 BDD 的规划器(2000)

MIPS (intelligent model checking and planning system)[28,29]也是 STRIPS 规划器，它使用二元决策表 BDD(binary decision diagram)来指导求解规划。在 MIPS 中，BDD 作为一个整体来操作一个布尔函数。前面讲过，Kautz 和 Selman 在 1996 年设计出了 SATPLAN，即把规划转化为 SAT 问题。但是这有一个明显的缺点：即使是一个非常小的问题，转化之后 SAT 公式的变量数也是非常多的。因此，许多类似的压缩数据结构出现了，其中之一就是 BDD。在规划过程中，为了提高规划器的性能，MIPS 将改变 BDD 表示。在 MIPS 的最新版本 1.5 中，使用的启发式是改进的 FF 启发式。在预编译阶段，给定的一个命题层被解释成命题的位向量。

2.7 HSP2.0：HSP 的改进(2000)

如前所述，HSP 可以和当时最好的规划器相媲美。然而，HSP 不是一个最优的规划器。更糟糕的是，HSP 采用的算法是不完备的。HSP2.0[30]使用最好优先算法(Best-First Search)来克服这个问题。因此，把用递增启发值 h_{add}来指导 BFS 算法的规划器叫做 HSP2.0。BFS 像 A* 算法一样，需要构建节点的 open 表和 closed 表。但是节点的权值是通过估价函数 $f(n)=g(n)+w*h(n)$来给定的，这里 $g(n)$是一个累积代价，$h(n)$是到目标的估计代价，$w\geqslant 1$ 为常量；$w=1$ 时为 A* 算法。$w\neq 1$ 就是 WA* 算法。HSP2.0 使用了非可纳的启发值 h_{add}来指导 WA* 算法，h_{add}在产生每个新状态时都从头开始计算。参数 w 的值一般固定为 5。w 的值即使在[2,10]之间，也不会产生大的影响。实验结果表明，HSP2.0在整个 HSP 家族中，整体性能是最优异的。

2.8 ALTALT：一个混合式启发式状态规划器(2000)

ALTALT[31]是用 C 语言实现的规划器，建立在规划器

STAN 和 HSPr 基础上。STAN 是一个用来生成规划图的 GRAPHPLAN 模式的规划器。前面讲过，HSPr 是一种提供优化的状态搜索机制的启发式规划器，ALTALT 利用基于图规划的启发函数来指导其搜索过程。在 ALTALT 中默认使用的启发函数是 H_{AdjSum2M}，启发函数 H_{AdjSum2M} 的基本思想是从考虑次目标之间的促进作用及反作用来调整和代价法。$H_{\text{AdjSum2M}}=\text{cost}_p(S)+\Delta_{\max}(S)$，其中 $\text{cost}_p(S)$ 利用公式 $\text{cost}_p(S)=1+\text{cost}_{tp}(S+\text{pre}(a)-\text{add}(a))$ 来计算，$\Delta(S)$ 用 $\Delta_{\max}(S)=\max\delta(p,q)$ 来计算。$\max\delta(p,q)=\text{lev}(p,q)-\max(\text{lev}(p),\text{lev}(q))$，$p,q\in S$。$\text{lev}(S)$ 表示命题集合 S 中的所有命题以非互斥关系出现在串行规划器中的最小命题列。ALTALT 的性能在 AIPS00 的规划器大赛上的测试问题都是非常健壮的。对 ALTALT 的主要评价为：1）ALTALT 的性能优于 STAN 与 HSPr，充分显示了 GRAPHPLAN 及其启发式状态空间搜索的互补作用的力量。2）ALTALT 能够降低启发式计算，虽然所生成的解的质量有点影响。

2.9 REPOP：POP 中的佼佼者（2001）

REPOP[32] 仍然是 UCPOP 的扩展版本，它使用了几个新的启发式控制技术。实验表明，它的整体性能要比图规划好很多。在 REPOP 中，采用了新的方法来调整基于距离的启发式可行性分析，析取约束处理技术。这些方法大幅度提高了 POP 算法的性能，使它们能与状态空间规划器相媲美，同时保持了灵活性。

POP 算法的性能主要取决于从搜索队列中选择偏序规划的方式及用于选择和解决 FLAW（在 POP 中，把开放条件和不安全的联接叫做 FLAW，参见文献[11]）的策略。在 REPOP 中，处理 FLAW 的方式与标准的 POP 算法有很大的不同，它只解决开放条件 FLAW。因此，在搜索队列中，一种规划排序方法是估计解决所有开放条件 FLAW 所需要的最小新动作数。给定一个偏序规划 P，定义 $h^*(P)$ 来表示达到一个解规划需要添加到 P 中的新动作的最小数目。估计 h^* 一个比较著名的启发式是开放条件启发式[15]，即 $h_{oc}(P)=|OC|$，OC 是一个开放条件集。这个启发式在许多领域是不可纳的，因为它对每个开放状态平等对待。如果考虑次目标之间的相互作用[33]，在早期，有很多启发值不但不考虑负作用，而且假设次目标之间是相互独立的[34]。最近，有些研究考虑了次目标之间的正作用（当然还是忽略负作用）[35,36]。我们用 $\text{lev}(P)$ 表示命题 P 在规划图第一次出现的层，$\text{lev}(S)$ 表示所有命题在 S 中首次出现的层。假设 pS 是 S 中的命题，则 $\text{lev}(pS)=\max_{p_i\in S}\text{lev}(p_i)$。$pS$ 可能是在执行过程中最后得到的命题。假设在规划图中，aS 是在 $\text{lev}(pS)$ 中得到 pS 的动作。通过添加 aS 到规划图中，就可以得到 pS。aS 的引入改变了目标集，可以得到状态 $S'=S+\text{pre}(aS)-\text{add}(aS)$。可以用 aS 和 S' 的代价来表示 S 的代价：

$$\text{cost}(S)=\text{cost}(aS)+\text{cost}(S+\text{pre}(aS)-\text{add}(aS))\quad(*)$$

其中，如果 $aS\notin A$，则 $\text{cost}(aS)=1$，否则为 0。既然 $\text{lev}(\text{pre}(aS))$ 严格小于 $\text{lev}(pS)$，递归应用等式（*），就可以最终得到用 $\text{cost}(I)$ 表示的 $\text{cost}(S)$ 和动作 aS 的代价。这个过程的效率非常高，因为数字的应用限制在 $\text{lev}(S)$ 中。下面看一个放松的启发式 $h_{\text{relax}}(P)=\text{cost}(S)$，这里，$S=\{p\mid(p,a)\in OC\}$，$\text{cost}(S)$ 用等式（*）来计算。给定这样一个启发式估计值，就可以用下式来对搜索队列进行排序：

$$f(P)=|A|+w*h(P)$$

用参数 w 来增加启发式搜索的贪婪度，默认设置为 5。

REPOP 的性能比以前所有的 POP 规划器都要好，可以与 GRAPHPLAN 和 ALTALT 相媲美。

2.10 SAPA：能处理时间和资源的规划器（2001）

SAPA（Scheduling And Planning Agent）[37] 是一个前向状态空间系统，用 Java 实现。它采用基于图规划的启发式，生成代价敏感时序规划（Cost-sensitive Temporal Planning）。SAPA 满足规划质量的多目标和执行时的代价。动作具有持续性，它们的前提条件可以是即时的（Instantaneous）或是持续的（Durative），并且动作效果可在执行期间的任何时间点产生。每一个动作 A 和一个执行代价 $\text{Cexec}(A)$ 相关。从规划图中产生启发式，对每一个动作引入代价函数 $C(a,t)$，表示估计的代价使动作 a 在时间 t 执行。类似地，对每一个命题 p，代价函数 $C(p,t)$ 说明在时间 t 到达 p 的代价。SAPA 用与 TGP[38] 不同的方法通过规划图传播代价信息：$C(a,t)=\max\{C(p,t):f\in\text{pre}(a)\}$（最大化传播）；$C(a,t)=\sum\{C(p,t):p\in\text{pre}(a)\}$（和传播）；$C(A,t)=0.5\max\{C(p,t):p\in\text{pres}(a)\}+0.5\sum\{C(p,t):p\in\text{pre}(a)\}$（结合传播）。通常在与固定点相连时才传播代价信息，并在整个过程中忽略互斥信息。这样 SAPA 基于传播的代价函数产生几个启发式信息，指导规划进行前向搜索。对于每一个状态 S，$h(S)$ 通过 $f(C(Ps),T(Ps))=\alpha C(Ps)+(1-\alpha)T(Ps)$[39]，这里 $T(Ps)$ 是具有限定代价（从 S 开始的规划到达到目标所估计的最小量）和所有目标都达到的最早时间，$C(Ps)$ 是每一个单独目标在它们各自的最终期限达到的代价总和，$0<\alpha<1$[80]。

2.11 Metric-FF：数值型规划器（2001）

许多年来，具有数值型状态变量（numeric state variables）的规划一直是一个挑战，它作为 IPC-3 的一部分出现了。在 STRIPS 规划领域，比较成功的算法是用一个启发式函数来指导搜索。在这里，这个启发式函数是基于放松规划任务，这个放松的规划任务忽略动作的删除效果，即 FF。我们把忽略删除效果扩展到数值型状态变量，它保持了 STRIPS 的理论特性，因为数值型任务是单调的（monotonic）。在这里，介绍 IPC-3 使用的一种数值型语言——线性任务（linear task）。在线性规划中，单调性可以通过预处理来得到。基于这个理论，把启发式规划系统 FF 扩展到线性任务，这个系统就叫 Metric-FF。根据 IPC-3 的结果，它是当时最好的两个数值型规划器之一（另外还有 LPG，限于篇幅，不做具体介绍，具体见参考文献[40]）。

在 2002 年召开的 IPC-3 中，首次引入了时序和数值型规划器。在 IPC-3 中，采用了 Fox 和 Long 定义的 PDDL2.1[41] 作为输入语言。Metric-FF 是 FF 的扩展，能处理 ADL 语言[42]。在数值型规划任务中，有数值约束（在动作前提和目标中）和数值效果（在动作效果中）。约束和效果可以是不用的类型。例如，一个约束可以要求给定变量的值或者至少大于（或者至多不大于）给定的常量。在语义角度，一个数值效果增加或者减少了受影响变量的值。在这里，通过放松规划任务来忽略动作的删除效果。这种思想的难点是忽略删除效果有时并不能够简化问题。例如，当目标要求 $x<0$，而 x 的初值为 0 时，删除效果是必需的，因此放松任务没有解。放松只有在具有更大值变量的任务中才是有效的。我们把这种任

务叫做单调任务。它只是 IPC-3 任务的一个子集，线性任务可以转化为范式，因此是单调的。基于这个理论，只把 FF 扩展到线性任务。

在 FF 中，搜索从初始状态开始，并且使用一个启发式函数来指导。在每个状态 s，启发式函数的值就是在状态 s 达到目标所需要的动作数(假定删除效果为空)。我们选择启发值小的状态。把 FF 扩展到数值型状态变量主要的障碍就是扩展这种机制，使其在每个状态都能够处理放松的规划任务。一旦这个机制被定义，后面的系统就可以利用了。

给定一个线性任务，Metric-FF 把它转化为 LNF(linear normal form)。在 LNF 任务中，表达式用变量的和来衡量，其值都大于 0。

除了 Metric-FF 外，MIPS 和 GRT(如 GRT-R[43])也有数值型的版本，可以查阅相关文献。

2.12 TP4：最优时态规划器(2001)

TP4[44]参加了 2002 年规划大赛，成绩不理想，而在 2004 年的规划大赛上，取得了比较理想的成绩。虽然 TP4 的语义与 PDDL2.1 的标准略有不同，但是 TP4 仍然是最优的时态规划器。TP4 使用回溯，自动提取可纳启发值来指导搜索。TP4 为带有持续动作的 STRIPS 问题寻找时态规划，这个规划是最优的，即规划总执行时间最少，并且能够保证不与某些资源约束冲突。

TP4 虽然使用了 PDDL2.1 的语法，但它不能直接接受 PDDL2.1 输入，持续动作和变量都是在 maner 中被解释，这一点与 PDDL2.1 不同[45]。TP4 中使用的持续动作的语义来自 Smith 和 Weld 的 TGP 规划器。对于动作 a，定义 $\mathrm{dur}(a)>0$ 为动作 a 持续的时间，$\mathrm{per}(a)=\mathrm{pre}(a)-\mathrm{del}(a)$。如果动作 a 在 $[t,t+\mathrm{dur}(a)]$ 内执行，则 $\mathrm{pre}(a)$ 在 t 时刻必须为真，$\mathrm{per}(a)$ 在整个时间段内都为真。此外，TP4 还要表示资源。例如，一个动作具有效果(at start (decrease f m))和(at end (increase f m))，条件(over all (>= f 0))，它使用变量 f 作为可重用的资源。

TP4 使用 IDA*，还有一些改进，如循环检测、受限转换表、来自特定问题的启发值。规划器更详细的描述见文献[46]。TP4 使用的启发式函数是

$$h^*(s)=\begin{cases}0, & \text{如果 } s \text{ 是在最终状态}\\ \min s'\in R(s)c(s,s')+h^*(s') & \text{其他}\end{cases}$$

其中，$R(S)$ 代表 S 的回溯集，即从状态 S 通过分枝规则建立的状态集。

在规划大赛上，TP4 进行了优化。如，无关检测(检测并删除无关的命题和动作，从而提高规划器的速度)和改进的 IDA* 算法(每次递归时代价倾向于增加目标的最大公约数)。

2.13 VHPOP：完全实例化的 POP 规划器(2002)

在上个世纪 80 年代末，许多域独立规划的研究集中在 POCL(partial order causal link)规划器上。其中最著名的两个 POCL 规划器是 SNLP[48]和 UCPOP。其他大量的工作都是对这两个规划器的改进。2001 年，Nguyen and Kambhampati[31]证明了用基于距离的启发式可行性分析等可以提高域独立规划器的整体性能，也能够提高 POCL 规划器的性能。因此，这促使人们改进这些方法，来处理域独立规划问题。通过以前的经验和 FLAW 选择策略，开发了 VHPOP (Versatile Heuristic Partial Order Planner)[47]。在 IPC-3 中，VHPOP 表现出了优异的性能。

虽然在 VHPOP 中的启发式可以完全实例化，也可以部分实例化动作，但在 IPC-3 中，VHPOP 选择的是完全实例化。VHPOP 可以高效执行所有的 FLAW 选择策略，例如 Dsep[15]，LCFR[49]和 ZLIFO[50]等。在 VHPOP 中，ZLIFO 和 LCFR 的组合得到了非常有效的 FLAW 选择策略。理想地，应找到一个最优的 FLAW 选择策略。我们发现了这样一个完全的策略，这个主意就是并行运行很多的规划器。在 VHPOP 中，在所有实例上，使用同一个相同的基本的 POCL 规划算法，但是并行地使用不同的 FLAW 选择策略。同时，VHPOP 通过支持带有持续动作的规划扩展了经典 POCL 规划器的能力，这可以通过添加一个 STN(simple temporal network)[51]到 POCL 规划器的正规规划描述中来实现。另外，这种方法还可以处理时序 POCL 规划。

VHPOP 使用 A* 算法[52]来搜索整个规划空间。A* 算法需要一个搜索节点估计函数 $f(n)=g(n)+h(n)$，这里 $g(n)$是从起始节点到 n 的代价，$h(n)$是达到目标节点的剩余代价的估计。我们想要得到一个包含动作尽量少的规划。对于一个规划 $P=\langle A,L,O,B\rangle$，有 $g(P)=|A|$。在 IPC-3 中，VHPOP 使用的启发式代价函数是 h_{add} 的改进，并且结合了 tie-breaking rank[53]。h_{add} 最重要的假设就是次目标是独立的。下面给出 POCL 规划 h_{add} 的递归定义。

给定一个文字 q，假设 $GA(q)$是完全实例化动作的集合，文字 q 的代价定义如下：

$$h_{\mathrm{add}}(q)=\begin{cases}0, & \text{如果 } q \text{ 在初始状态为真}\\ \min a\in GA(q)h_{\mathrm{add}}(a), & \text{如果 } GA(q)\neq\emptyset\\ \infty, & \text{其他}\end{cases}$$

如果 q 是初始状态的一部分，则为真，否则为假。动作 a 的代价是

$$h_{\mathrm{add}}(a)=1+h_{\mathrm{add}}(\mathrm{pre}(a))$$

其中，$\mathrm{pre}(a)$是一个表示动作 a 前提的 NNF(negation normal form)命题范式。任何命题范式都可以转换为 NNF，当解析域描述文件时，VHPOP 完成这个动作前提的转换。在动作前提中，存在量词变量可以看作动作的附加变量。一个存在量词前提的代价可以简单定义如下：

$$h_{\mathrm{add}}(\exists x.\Phi)=h_{\mathrm{add}}(\Phi)$$

同理，可以通过在预处理阶段完全实例化来处理全称量词前提。下面给出定义合取的代价：

$$h_{\mathrm{add}}(\exists x\bigcap_i\Phi_i)=\sum_i h_{\mathrm{add}}(\Phi_i)$$

以上定义是基于动作之间是独立的，实际上极有可能会高估合取目标的实际代价，即启发值是不可纳的。析取代价定义如下：

$$h_{\mathrm{add}}(\bigcup_i\Phi_i)=\min_i h_{\mathrm{add}}(\Phi_i)$$

对于 POCL 规划，h_{add} 代价函数定义如下：

$$h_{\mathrm{add}}(\pi)=\sum_{\xrightarrow{q}_{a_i}\in OC(\pi)}h_{\mathrm{add}}(q)$$

H_{add} 启发式没有考虑动作重用，因此它经常过高地估计完成规划所需的动作数。在 IPC-3 中，VHPOP 做了一个微小的改变来解决动作的重用问题：

$$h^r_{\mathrm{add}}=\sum_{\xrightarrow{q}_{a_i}\in OC(\pi)}\begin{cases}0, & \text{如果 } \exists a_j\in A\\ h_{\mathrm{add}}(q), & \text{其他}\end{cases}$$

这个改进的启发式仍然是不可纳的。换句话说，我们仅

仅考虑了存在动作的重用，而不是潜在的动作。

2.14 BP：第一个真正意义上的双向搜索规划器(2001)

以上分析表明，虽然这些规划器的性能在很大程度上取决于使用的启发式函数的精确性，但实验表明，它们搜索空间的顺序也对其有很大的影响。像 UNPOP，GRT 和 HSP/ASP，它们从初始状态开始搜索，直到达到目标；相反，HSPr 和 ALTALT 从问题的目标开始搜索，直到达到问题的初始状态。在 HSP2.0 中，用户可以自己决定搜索的方向，实验表明这存在一个明显的问题：有些人偏爱这个或另一个方向，而没有证据表明这两个方向应该被选择。下面介绍一种双向策略——BP(Bi-Directional Heuristic Planner)[54]。

双向搜索是一些有关人工智能的教材中经常提到的一种比较著名的搜索策略[55]。但是，作为一种搜索策略，双向搜索并没有被广泛采纳。特别是在规划领域，仅仅有一些系统能够执行两个方向的混合搜索。据作者所知，仅有的双向搜索规划器有 PRODIGY[56,57]，FLECS[58] 等。所有这些规划器都是由卡耐基梅隆大学的 PRODIGY 项目(有关 RODIGY 项目的更多信息，参看 http://www.cs.cmu.edu/~prodigy/)开发的。但是实际上，这些规划器都是前向规划器，后向搜索仅作为一种选择动作的机制。BP 是第一个真正意义上的双向搜索规划器。

BP 是一种域独立的混合搜索策略，组合了前向和后向搜索策略。搜索从初始状态开始，用一个加权的 A* 算法向前搜索，直到没有更优的状态为止；在这一点，算法开始改变方向，并且从目标开始搜索，直到到达前面一步的最佳状态。在找到一个解之前，搜索方向可能改变很多次。

BP 使用了两个域独立的启发函数：一个是基于 ASP/HSP 的目标排序(Goal Ordering)技术，另一个是双向规划系统(bi-directional planning system)。对 AIPS 一些规划实例进行了验证，得到了很好的效果。在前向搜索阶段，BP 使用的是一个爬山算法；在后向阶段，BP 处理的是状态集，而不是像很多规划器那样处理的是状态。它包括回溯测试和状态回溯两个方面，可以用下式表示：

$$h(S)=w_1\times L(S)+w_2\times h(S)$$

其中，$L(S)$是从初始状态得到到达状态 S 的时间步数，$h(S)$是回溯阶段返回的启发值，w_1，w_2 是用户自己定义的变量(在 A* 算法中的权值)。

随着智能规划的发展，近年来 PSP(Partial Satisfaction Planning，亦称过描述 over-description Planning)在许多应用领域兴起，它主要应用在实体要达到比现有资源要多的目标。PSP 规划的目标是寻找一个规划，使得它有最高的净效益，它只能满足目标的部分要求。在 PSP 问题中，最著名的是 NET BENEFIT 问题。早期比较著名的规划器是 OP(Orienteering graph)[59]。迄今为止，性能比较突出的具有启发式搜索的 PSP 规划器有 SapaPS，ALTWLT[60]，PrePlan[61,62]，Optiplan[63] 等。

2.15 ALTWLT：具有启发式搜索的 PSP 规划器(A Little of This and a Whole Lot of That)(2005)

ALTWLT 是一款具有启发式搜索的 PSP 规划器。这个规划器之所以取名为 ALTWLT，是因为它以 ALTALTps 为基础。下面以 NET BENEFIT 问题为例来介绍 ALTWLT 的工作原理。在 NET BENEFIT 问题中，每个合取目标都有一个相关的固定 Utility，每个实例化动作都有一个与之相关的代价。我们的目标就是找到一个最优的 NET BENEFIT，即总 Utility 与总代价之差。

我们定义一个规划问题 $P=(A,I,G)$(有关 PSP 的更详细知识见文献[64])，图 2 是许多 PSP 问题的层次结构。

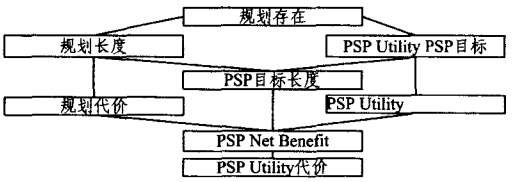

图 2 PSP 层次结构图

ALTALT^{PS}使用一个贪心方法来到达前面的次目标。次目标可能是指数级的。一旦目标被选择，ALTALT^{PS}就找到了一个规划解。但是 ALTALTPS 使用的启发式对 Utility 和代价是敏感的。要计算启发值，必须首先计算代价传播。此外，ALTWLT 在求解时，没有考虑互斥情况；如果放松规划中有互斥情况，则对目标进行惩罚(penalty)——使用相互作用因素来调整代价，类似地来调整和启发式 $\max_{g_1,g_2\in G}\{\text{lev}(g_1,g_2)-\max(\text{lev}(g_1),\text{lev}(g_2))\}$，最终为每个目标寻找最好的规划。ALTWLT 比较突出的一点是考虑了残余代价(residual cost)。

2.16 Optiplan：第一个参加 IPC 整数编程的规划器(2005)

Optiplan 是第一个参加 IPC(具体指 2004 年的 IPC-4)的、使用整数编程 IP(integer linear programming)的求解 STRIPS 问题的规划器，在 IPC 中获得了第二名的优异成绩。Optiplan 的整数编程机制扩展了 Vossen[65] 的状态转换模型。它的层次结构与 BLOCKBOX 和 GP-CSP[66] 非常相似，但是没有使用 SAT 或 CSP，Optiplan 使用了 IP 方法。如同 BLOCKBOX 和 GP-CSP 等规划器，Optiplan 也是分两个阶段：第一阶段是规划图的构建并将其转换为 IP 公式，第二阶段用商用求解器 ILOG CPLEX[67] 求解 IP 公式。Optiplan 与以前的状态转换模型主要不同的是，前者是把动作和命题先实例化为规划图，然后作为输入，而后者是直接实例化动作和命题。

把 LP(linear programming)和 IP 应用于智能规划中的研究还不是很深入。1997 年，Bylander 提出了一种用 IP 公式解决经典规划的方法[68]。近年来，LP 和 IP 方法已经被用于解决非经典规划问题。例如，Dimopoulos 和 Gerevini 提出了一种用 IP 解决时序规划的方法[69]；Wolfman 和 Weld 把 LP 和 SAT 结合，用来解决带有资源的规划问题[70]。另外，Kautz 和 Walser 也用 IP 解决了带有资源、动作代价和多目标的规划[71]。而 Optiplan 是第一个技术比较成熟的、参加 IPC 竞赛的、使用 IP 方法的规划器。

2.17 CFF：第一款性能突出的一致性规划器(2006)

一致性规划是一种不确定性规划。不确定规划的来源有两个，分别为初始状态不确定和动作效果不确定。一致性规划是处理初始状态不确定的规划方法。同时，一致性规划在规划执行过程中不具有感知能力[73]。无论从哪个可能初始世界开始，我们都能找到规划解。一致性规划可以转化为在信念空间(即它的元素是可能世界状态的集合)中的搜索问题。Bonet 和 Geffner[74] 引入了一种在信念状态中利用启发式来指导前向搜索、解决规划问题的方法。但是在信念状态

中，可能世界的个数非常大——甚至在一些比较简单的例子中，在其系统 GPT 中，都不可解。其他的一些求解器，像 Bertoli，Cimatti 和 Roveri [75,76] 用 BDD 表示信念状态来处理这个问题，通常效率是比较高的，但是 BDD 的构建仍然是个瓶颈。在 CFF[72] 中，提出了第三种处理信念状态的方法。在这种方法中，目标和动作的前提被限制在命题的析取中（如同在 STRIPS 中一样）。这样，不管实体在哪一个状态，只要考虑所有世界的交集就足够了。一个动作序列是一致性规划，当且仅当它产生的所有世界的交集组成的信念状态包含所有的目标。对于每一个命题 p，如果包含在信念状态 s 中，即在执行完一个动作 a 后为真，我们就称这样的命题 p 在信念状态 s 是可知的（known）。但是这个测试是困难的，通常是 co-NP 完全问题。

这个系统之所以叫 CFF（Conformant-FF），是因为它以 FF 代码为基础，如同 FF，CFF 也使用了全局搜索技术。CFF 与其他一些一致性规划器（如 KACMBP[76] 和 POND[77]）进行比较，结果显示，CFF 的整体性能要优于 KACMBP 和 POND。

在经典规划中，基于放松规划任务的启发式在指导搜索时取得了巨大成功。CFF 把这个思想扩展到了不确定性规划——一致性放松（Conformant Relaxation）中。在考虑启发式函数的选取时，通常要考虑两个因素：一是可纳性，FF 系统为了换取运行时间牺牲了可纳性；二是信息性（informativity，即一个非正式概念，它指一个启发式能够提供的用来指导搜索的信息质量）和计算效率之间的权衡。通常情况下，计算启发式函数用的时间越多，得到的启发值携带的信息也越多。

虽然求解忽略删除效果的放松规划问题是 NP-hard 的，但是可以利用这个放松作为启发式函数的基础，通过在多项式时间计算放松规划的长度来作为启发值。同时，在搜索阶段，对于每个搜索状态，调用 SAT 方法（在 FF 中，使用的是 Chaff [78]），但是最坏情况下，其复杂度是指数级的。为了避免打破动作效果之间的交集带来的计算复杂度，CFF 在其一致性放松中仍然忽略了动作的删除效果。在进行搜索之前，CFF 需要建立一致性放松规划图 CRPG（conformant relaxed planning graph）。建立一致性放松规划图 CRPG 算法是完备的，之后就可以进行解的提取 extract-CRPlan。如果 CRPG 返回失败，CFF 将启发值 h 设为 ∞，这个状态也将被证明为一个死胡同（dead end，从目标开始搜索不可能达到这个状态），搜索空间将不包括这个状态。如果 CRPG 成功到达状态，启发值 h 为通过 extract-CRPlan 选择的动作数（即在放松规划中到达信念状态的动作数）。通常，启发值 $h(s)=0$ 代表已经到达目标状态，1 到 m（m 为 CRPG 层数）代表执行动作 a 之后目标状态是不可知的。启发值 $h(s)=\infty$ 表示状态 s 是不可解的（unsolvable），CFF 不考虑这个状态。此外，CFF 对搜索进行了一些改进，如为了避免重复搜索状态，利用了哈希表技术。

结束语 随着智能规划研究的深入，特别是启发式搜索的应用，智能规划已经能够解决许多实际问题。2006 年召开的 IPC-5 中，已经开发出了很多混合模型（Hybrid Models）规划器，例如能处理数值-时序和带有资源问题的 SAPA、能处理时序和不确定性规划问题的 Prottle、能处理 PSP 和资源问题的 $SAPA^{MPS}$、能处理基于代价的条件规划的 CLUG 等。这些规划器的开发，使得智能规划处理现实问题的能力大大提高。但是这些规划器的求解能力是有限的，提高它们处理问题的范围和速度将是今后努力的主要方向。今后智能规划的发展应该克服以下问题：

1）设计更精确的启发式函数

混合模型规划器大大提高了智能规划求解现实问题的能力，但是仍不能满足需要。启发式搜索在指导规划器搜索方面表现出了卓越的性能。如果能找到更好、更精确的启发式函数，无疑会大大提高规划器的性能。可以考虑：a）设计出一个更精确的通用启发式函数，使规划器求解问题的范围更广；b）开发出求解特定问题的启发式函数，使得规划器解决特定问题能力增强。另外，可以对现有启发式函数进行改进，如可以对 BP 使用的启发式调节 w_1，w_2 两个因子，使其启发式函数 $h(s)=w_1\times h_1(s)+w_2\times h_2(s)$（$h_1(s)$ 和 $h_2(s)$ 分别代表前向和后向启发式）适应不同的具体问题。

2）突破封闭世界假设

现在的规划器都是基于封闭世界假设的，这使得研究问题简单，易于处理。但是现实世界是复杂多变的，规划器在执行过程中，极有可能受到各种各样的干扰。规划器若能打破封闭世界假设，使其具有抗干扰能力，将会解决大量的实际问题，如网络攻防、战争指挥等。

3）如何将智能规划与规划识别统一起来

规划问题与规划识别问题密切相关，若能将其结合起来，构建规划与识别的统一体，将会对实体协作、网络安全、战争指挥等方面有重要的作用。在该统一体中，规划器不但能够建立规划库，还应该加入规划库没有的动作；而规划识别器能够辅助规划器确定下一个目标。如果能够将二者统一起来，将能够解决大量的实际问题。但目前国内外还很少有这方面的研究。

参 考 文 献

[1] Blum A L, Furst M L. Fast planning through planning graph analysis[J]. Artificial Intelligence, 1997, 90: 281-300

[2] Kautz H, Selman B. Unifying SAT-based and graph-based planning[C]// Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI'99). Stockholm, Sweden; Morgan Kaufmann, 1999: 318-325

[3] Kautz H A, Selman B. Pushing the envelope: Planning, propositional logic, and stochastic[C]// Proceedings of the 13th National Conference of the American Association for Artificial Intelligence. MIT Press, 1996: 1194-1201

[4] Kautz H, Selman B. Blackbox: Unifying sat-based and graph-based planning[C]// Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-99). 1999: 318-325

[5] Bonet B, Geffner H. HSP: Heuristic Search Planner[J]. Entry at AIPS-98 Planning Competition. AI Magazine, 2000, 21

[6] Bylander T. The computational complexity of propositional STRIPS planning[J]. Artificial Intelligence, 1994, 69 (1/2): 165-204

[7] Fikes R E, Nilsson N J. Strips: A new approach to the application of theorem proving to problem solving[J]. Artificial Intelligence, 1971 (2): 189-208

[8] McDermott D. A heuristic estimator for means-ends analysis in planning[C]// Proceedings of the 3rd International Conference

on Artificial Intelligence Planning Systems (AIPS'96). Menlo Park: AAAI Press, 1996: 142-149

[9] Bonet B, Loerincs G, Geffner H. A robust and fast action selection mechanism for planning[C]// Proceedings of the 14rd National Conference of the Americanfor Artificial Intelligence (AAAI'97). AAAI Press, 1997: 714-719

[10] McAllester D, Rosenblitt D. Systematic nonlinear planning[C]// Proc. AAAI'91

[11] Weld D. An introduction to least commitment planning[J]. AI magazine, 1994

[12] Joslin D, Pollack M. Least-cost ? aw repair: A plan refinement strategy for partial-order planning[C]// Proc. AAAI'94

[13] Kambhampati S, Knoblock C, Yang Q. Planning as Refinement Search: A unified framework for evaluating design tradeoffs in partial-order planning[J]. Artificial Intelligence, 1995

[14] Gerevini A, Schubert L. Accelerating partial - order planners : Some techniques for effective search control and pruning[J]. JAIR, 1996(5): 95-137

[15] Peot M A, Smith D E. Threat-removal strategies for partial-order planning[C]// Proceedings of the Eleventh National Conference on Refinement Intelligence. Washington, DC: AAAI Press, 1993: 492-499

[16] Penberthy J S, Weld D S. UCPOP: A Sound, Complete, Partial Order Planner for ADL[C]// Proceedings of the Third International Conference on Principles of Knowledge Represen and Reasoning. Boston, MA, 1992: 103-114

[17] Anderson C R, Smith D E, Weld D S. Conditional Effects in GRAPHPLAN[C]// AIPS'98

[18] Bonet B, Geffner H. Planning as Heuristic Search. Artificial Intelligence[J]. Special issue on Heuristic Search, 2001, 129 (1/2)

[19] Bonet B, Geffner H. Planning as heuristic search: New results [C]// Recent Advances in AI Planning: Proceedings of the Fifth European Conference on Planning. Springer, Lecture Notes in AI 1809, 1999: 359-371

[20] Koehler J, Nebel B, Hoffman J, et al. Extending planning graphs to an ADL subset[C] // Proceedings 4th European Conf. on Planning. LNAI 1248, Springer, 1997

[21] Long D, Fox M. The efficient implementation of the plangraph [J]. JAIR, 1999(10): 85-115

[22] Refanidis I V. GRT: A Domain Independent Heuristic for STRIPS Worlds based on Greedy Regression Tables[C]// 5th European Conference on Planning (ECP'99). Durham, UK: Springer-Verlag, September 8-10, 1999: 346-358

[23] Ioannis Refanidis I V. The GRT Planner [J]. AI Magazine , 2001, 22 (3)

[24] Refanidis I V. The GRT planning system: Backward heuristic construction in forward state-space planning[J]. Journal of Artificial Intelligence Research, 2001 (15): 115-161

[25] Bonet B, Loerincs G, Geffner H. A robust and fast action selection mechanism for planning[C]// 14th International Conference of the American Association of Artificial Intelligence (AAAI' 97). Providence, Rhode Island, 1997: 714-719

[26] Refanidis I V. Multiobjective heuristic state-space planning[J]. Artificial Intelligence, 2003 (145): 1-32

[27] Hoffmann J, Nebel B. The FF planning system: Fast plan generation through heuristic search[J]. Journal of Artificial Intelligence Res, 2001 (14): 253-302

[28] Edelkamp S, Helmert M. On the Implementation of MIPS [C]// Proceedings of AIPS-2000

[29] Edelkamp S. Taming numbers and durations in the model checking integrated planning system[J]. Journal of Artificial Intelligence Research, 2003(20): 195-238

[30] Bonet B, Geffner H. Heuristic Search Planner 2. 0. Decription HSP Planner in AIPS-2000 Competition[J]. AI Magazine, 2001, 22 (3): 77-80

[31] Nguyen X, Kambhampati S, Sanchez R. Planning graph as the basis for deriving heuristics for plan synthesis by state space and CSP search[J]. Artificial Intelligence, 2002, 135(1/2): 73-123

[32] Nguyen X, Kambhampati S. Reviving partial order[C]// International Joint Conference on Artificial Intelligence(IJCAI'01). Seattle, Washington, USA: Morgan Kaufmann, 2001: 459-464

[33] Nguyen X, Kambhampati S. Extracting effective and admissible state space heuristics from the planning graph [C] // Proc. AAAI'00

[34] McDermott D. Using regression graphs to control search in planning[J]. Artificial Intelligence, 1999, 109(1/2): 111-160

[35] Nguyen X, Kambhampati S. Extracting effective and admissible state space heuristics from the planning graph [C] // Proc. AAAI'00

[36] Fukunagapanner M B, Kambhampati S. Sapa : A domain - independent heuristic metric temporal planner[C]// Proc. 6th European Conference on Planning. 2001

[37] Minh B Do, Kambhampati S. Sapa: a multi-objective metric temporal planner[J]. JAIR, 2003(20): 155-194

[38] Smith D, Weld D. Temporal Planning with Mutual Exclusion Reasoning[C]// Proc IJCAI'99. 1999

[39] Minh B Do, Kambhampati S. Planning Graph - based Heuristics for Cost-sensitive Temporal Planning[C]// Proc. AIPS'02

[40] Geerevini A, Saetti A S. Planning through stochastic local search and temporal action graphs[J]. Journanl of Artificial Intelligence Research, 2003

[41] Fox M, Long D. The third international planning competition : Temporal and metric planning[C] // Ghallab M, Hertzberg J, Traverso P, eds. Proceedings of the 6th International Conference on Artificial Intelligence Planning and Scheduling (AIPS' 02). Toulouse, France: Morgan Kaufmann, 2002: 333-335

[42] Pednault E P. ADL : exploring the middle ground between STRIPS and the situation calculus[C]// Brachman R, Levsque H J, Reiter R, eds. Principles of Knowledge Representation and Reasonning: Proceedings of the First International Conference (KB'89). Tronto ON: Morgan Kaufmann, 1989: 324-331

[43] Refanidis I, Vlahavas I. Heuristic planning with resources[C]// Proceedings of the 14th European Conference on Artificial Intelligence (ECAI'00). Berlin, Germany: Wiley. 2000: 521-525

[44] Haslum P. TP'o4 and HSPa* [M]. International Planning Competition-4 Publication, 2004

[45] Fox M, Liong D. PDDL2. 1: An extension to PDDL for expressing tempal planning domains[J]. Journal of AI Research, 2003 (20): 61-124

[46] Smith D, Weld D. Temporal planning with mutual exclusion reasoning[C] // Proceeedings 16th International Joint Conference on Artificial Intelligence. 199: 326-333

[47] Younes H L S, Simmons R G. VHPOP: Versatile Heuristic Partial Order Planner [J]. Journal of Artificial Intelligence Research, 2003: 405-430

[48] McAllester D A, Rosenblitt D. Systematic nonlinear planning [C]// Proceedings of the Ninth National Conference on Refinement Intelligence. Anaheim, CA: AAAI Press, 1991: 634-639

[49] Joslin D, Pollack M E. Least-cost ? aw repair: A plan refinement strategy for partial-order planning [C] // Proceedings of the Twelfth National Conference on Artificial Intelligence. Seattle, WA: AAAI Press, 1994: 1004-1009

[50] Schubert L, Gerevini A. Accelerating partial order planners by improving plan and goal choices [C] // Proceedings of the Seventh International Conference on Tools with Artificial Intelligence. Herndon, VA: IEEE Computer Society Press, 1995: 442-450

[51] Dechter R, Meiri I, Pearl J. Temporal constraint networks [J]. Artificial Intelligence, 1991, 49 (1-3): 61-95

[52] Hart P E, Nilsson N J, Raphael B. A formal basis for the heuristic determination of minimum cost paths [J]. IEEE Transactions on Systems Science and Cybernetics (SSC), 1968, 4 (2): 100-107

[53] Younes H L S, Simmons R G. On the role of ground actions in refinement planning [C] // Ghallab M, Hertzberg J, Traverso P, eds. Proceedings of the Sixth International Conference on Artificial Intelligence Planning and Scheduling Systems. Toulouse, France. AAAI Press, 2002: 54-61

[54] Vrakas D, Vlahavas I. Bi-Directional Heuristic Planning in State-Spaces [C] // Proc. 8th Panhellenic Conference on Informatics. 2001 (1): 511-520

[55] Russell S, Novig P. Artificial Intelligence: A Modern Approach [M]. NJ: Prentice-Hall Englewood Cliffs, 1995

[56] Carbonell J, Etzioni O, Gil Y, et al. PRODIGY: an integrated architecture for planning and learning [J]. ACM SIGART Bulletin Veloso, 1991, 2(4): 51- 55

[57] Veloso M, Carbonell J, Prez A, et al. Integrating planning and learning: the PRODIGY architecture [J]. Journal of Experimental & Theoretical Artificial Intelligence, 1995, 7(1): 81-120

[58] Veloso M, Stone P. FLECS : Planning with a Flexible Commitment Strategy [J]. Journal of Artificial Intelligence, 1995(3): 25-52

[59] Smith D. Choosing objectives in over-subscription planning [C] // Proceedings of ICAPS'04. 2004

[60] Nigenda R S, Kambhampati S. Planning graph heuristics for selecting objectives in over-subscription planning problems [C] // ICAPS. 2005

[61] Brafman R I, Chernyavsky Y. Planning with goal preferences and constraints [C] // Proceedings 15th Internatianal Conference AI Planning and Scheduling (ICAPS). 2005

[62] Golden K, Brafman R, Pang Wanlin. Preferences in Data Production Planning. Multidisciplinary. IJCAI-05

[63] van den Briel M, Kambhampati S. Optiplan: Unifying IP-based and Graph-based Planning [J]. Journal of Artificial Intelligence Research, 2005

[64] van den Briel M, Sanchez R, Do M, et al. Effective approaches for partial satisfation (over-subscription) planning [C] // Proceedings of AAAI'04. 2004

[65] Vossen T, Ball M, Lotem A, et al. On the use of integer programming models in AI planning [C] // Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'99). 1999: 304-309

[66] Do M, Kambhampati S. Planning as constraint satisfaction: Solving the planning graph by compiling it into CSP [J]. Artificial Intelligence, 2001, 132 (2): 151-182

[67] ILOG Inc. ILOG CPLEX 8. 0 user's manual [M]. Mountain View, CA, 2002

[68] Bylander T. A linear programming heuristic for optimal planning [C] // AAAI'97/IAAI'97 Proceedings. 1997: 694-699

[70] Wolfman S, Weld D. The LPSAT engine and its application to resource planning [C] // Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'99). 1999: 310-317

[71] Kautz H, Walser J. State-space planning by integer optimization [C] // AAAI'99/IAAI'99 Proceedings. 1999: 526-533

[72] Hoffmann J, Brafman R I. Conformant planning via heuristic forward search: A new approach [J]. Artificial Intelligence, 2006 (170): 507-541

[73] Smith D E, Anderson C R, Weld D S. Extending GRAPHPLAN to handle uncertainty and sensing actions [C] // AAAI-98. July 1998

[74] Bonet B, Geffner H. Planning with incomplete information as heuristic search in belief space [C] // Chien S, Kambhampati R, Knoblock C, eds. Proceedings of the 5th International Conference on Artificial Intelligence Planning Systems (AIPS'00). Menlo Park, CA: AAAI Press, 2000: 52-61

[75] Cimatti A, Roveri M. Conformant planning via symbolic model checking [J]. JAIR, 2000 (13): 305-338

[76] Bertoli P, Cimatti A. Improving heuristics for planning as search in belief space [J]. Artificial Intelligence Planning Systems, 2002: 143-152

[77] Bryce D, Kambhampati S, Smith D E. Planning in belief space with a labelled uncertainty graph [C] // Proc. AAAI'04 Workshop on Learning and Planning in Markov Decision Processes. 2004

[78] Moskewicz M W, Madigan C F, Zhao Y, et al. Chaff : Engineering an efficient SAT solver [C, EB/OL]. Proceedings of the 38th Design Automation Conference (DAC'01). http: // www. ee. princeton. edu/-chaff/DAC2001v56. pdf, 2001

[79] Dimopoulos Y, Gerevini A. Temporal planning through mixed integer programming [C] // Proceeding of the AIPS

[80] 张友红，谷文祥，刘日仙. 非确定规划及带有时间和资源的规划的研究 [J]. 计算机应用研究，2005