第5卷第5期
1994年5月

软 件 学 报
JOURNAL OF SOFTWARE

Vol. 5, No. 5
May 1994

利用算符分析的问题求解*

何钦铭　　王申康　　俞瑞钊

(浙江大学人工智能研究所,杭州 310027)

摘要　将状态空间的问题求解过程变换为逐步缩小与目标状态的差异过程是一种问题的分解方式.求解差异的顺序可通过分析算符对状态的影响而作出规划,规划的原则是最大限度地在不改变最近已实现子目标的条件下实现下一子目标.为此,在问题分解时各层子目标选择的依据是让各算符有最大的可利用率,即以状态对算符最小约束传播的原则选择各层子目标;最后生成一个子目标规划层次集.问题求解过程就表现为从初始状态开始实现层次集中的某一子目标序列,其间可能涉及子目标回溯.

关键词　问题分解,搜索,问题求解.

在无任何启发信息的情况下,问题求解可以逐步缩小与目标状态差异的方式进行.然而,首先解决什么差异,下一步缩小哪个差异是一个需要规划的问题.一般我们总希望已无差异的部分在将来解决其它差异时不再变动,即线性规划的思想.这样,子问题求解的工作量将会大为减少.下面我们将讨论通过对算符的分析,建立一个子目标规划层次集(而不是单一的规划路径),最大限度地在不改变最近已实现子目标的条件下实现下一子目标.问题求解表现为从初始状态出发,沿规划层次集中的某一子目标序列,顺序求解各个子问题,逐步缩小现有子目标状态与目标状态的差异.

1　算符分析与问题求解

1.1　问题表示

我们以状态的向量表示讨论基于算符分析的问题分解,即状态空间中每一状态均用一向量$(x_1, x_2, \cdots, x_n)$表示,其中x_i为第i个分量值.不失一般性,假定初始状态S_0和目标状态G都是单一状态,而不是状态集.算符集记为Γ,每一算符均用产生式规则表示,条件部分涉及对状态分量的测试,动作部分涉及对有关分量的修改.

问题(S_0, G, Γ)的搜索目标为:从S_0和Γ决定的隐含图中寻找一条从S_0到目标G的路

* 本文1991－06－29收到,1992－01－21定稿

本文是国家863高技术资助项目.**何钦铭**,29岁,讲师,主要研究领域为人工智能,机器学习,专家系统,搜索.**王申康**,49岁,副教授,主要研究领域为人工智能,机器学习,专家系统及其应用,智能计算机辅助教学.**俞瑞钊**,57岁,教授,主要研究领域为人工智能.

本文通讯联系人:何钦铭,杭州 310027,浙江大学人工智能研究所

径,不要求是最佳路径.

1.2 问题分解策略

我们的求解路径是逐步缩小与目标 G 的差异.为此要作一个解决什么差异及其顺序的规划,并尽可能使得求解过程尽量不破坏最近已实现的子目标.

逐步缩小与目标 G 差异的求解方式,从子目标的角度上看表现为顺序实现一子目标序列 $G_1,G_2,\cdots,G_n,G$,其中每一子目标 G_i 都是总目标 G 的一部分,且与总目标的差异数是严格递减的,亦即匹配数(子目标与总目标的相同分量数)是严格递增的.

定义. 设 D_i 是满足 G_i 的状态集合,令 $D_i(f)=\{S|S\in D_i, f\in\Gamma, f(S)$有定义$\}$.设 $f\in\Gamma$,若 $D_i(f)\neq\Phi$,且$\forall\ S\in D_i(f)$有 $f(S)\in D_i$,则称算符 f 对子目标 G_i 是稳定的.

若能定出一个子目标序列 $G_i,i=1,2,\cdots n$,也就是定出了一个差异求解顺序.在求解每次差异时,我们并不要求保持所有已实现的差异不动,而只要求不改变最近实现的子目标(保持相应状态分量不变).亦即将整个搜索问题化解为以下 n+1 个子问题:

$(S_0,G_1,\Gamma),(S_1,G_2,\Gamma_1),\cdots,(S_{n-1},G_n,\Gamma_{n-1}),(S_n,G,\Gamma_n)$

其中 $\Gamma_i=\{f|f\in\Gamma$ 且 f 对 G_i 是稳定的$\}$, S_i 为上个子问题的搜索结束点(从而 S_i 满足 G_i, $i=1,2,\cdots,n$).

这种分解方式,由于限制了允许改变的状态分量数和可用算符个数,使得扩展结点数大为减少,然而,也正是由于这种限制有可能使得子问题无解.因而,我们进行问题分解时,在相同匹配数下,应选择那些对 Γ 中算符有最小限制的子目标,即在该子目标下,算符集 Γ 的利用率最大.记在子目标 G_i 下,Γ 的利用率为 $U(\Gamma|G_i)$,一种简单的计算方法为:

$$U(\Gamma|G_i)=\sum_{f\in\Gamma}U(\{f\}|G_i)$$

$$U(\{f\}|G_i)=\begin{cases}1 & f\text{ 对 }G_i\text{ 是稳定的}\\ 0 & \text{其它}\end{cases}$$

f 对 Gi 的稳定性可通过分析 f 的前提和结论与状态分量的关系而作出判断.

由于在相同匹配数下,具有算符利用率最大的子目标可能不止一个,此时,我们保留所有这些子目标.因而,最后形成的规划不一定是一条路径,可能是一个层次集.该层次集中每一结点是一子目标.同一层子目标具有相同的匹配数.从最低层结点到顶点的每一子目标序列代表一种规划,至于采用哪种规划在问题求解时动态确定.

我们可以从顶向下,通过计算不同子目标下算符的利用率,生成子目标规划层次集:顶层顶点为总目标 G,以后以增加一个差异(即减少一个匹配数)为步进单位,计算下层子目标.下层子目标为在相同匹配数下具有非零最大算符可利用率的子目标.直到最底层子目标的匹配数为 1(如果相应最大算符可利用率非零).

利用算符分析的问题分解方法,实际上体现了最小约束传播原理,[2]这里的约束是指已实现子目标对算符集可应用性的约束.

1.3 算符分析后的搜索

设通过算符分析后得到的规划层次集为$\{G_{ij}, i=1,2,\cdots,n, j=1,2,\cdots,m_i\}$, 第 i 层子

目标为 G_{ij}, $j=1,2,\cdots,m_i$. 层次集中每一条从底层到顶层的子目标序列都是潜在的规划路径. 真正求解经过的路径将在实际搜索中确定. 在求解各个子问题时,由于无进一步的启发信息,只能用一般的弱方法,如宽度优先搜索法.

在$\{G_{ij}\}$上新增加一个只含一个子目标的最顶层$\{G_{(N+1)1}=G\}$. 设$\bigcup_{i\geqslant 1}G_{ij}$表示第1层以上(包括第1层)的所有子目标集合;Stack 为一个子问题堆栈,每一项元素为初态、目标和操作集三元组.

问题搜索过程为:

(1) 将栈 Stack 初始化为$(S_0,\bigcup_{i\geqslant 1}G_{ij},\Gamma)$.

(2) While 栈不空 DO:

a. 设$(S_h,\bigcup_{i\geqslant 1}G_{ij},\Gamma_{lk})=$POP(Stack), 求解该子问题.

b. 若以上子题有解转 d,否则:

c. 通过回溯扩大操作集 Γ_{lk}. 扩大后的操作集 Γ_{xy}必须满足:存在一子目标 G_{xy}, S_h 满足 G_{xy},且 $x<1$, $\Gamma_{xy}=\{f|f\in\Gamma$ 且 f 对 G_{xy}是稳定的$\}$. 若再次无解,可再扩大 Γ_{xy}集或就此执行下了循环.

d. 若子问题有解,设搜索结束点为 $S_{h'}$, 实现子目标为 $G_{l'k'}$. 判别 $S_{h'}$是否满足 G,满足则求解成功,结束;否则,将 Stack 的栈顶元素中的目标 $\bigcup_{i\geqslant 1}G_{ij}$改为 $\bigcup_{i\geqslant 1}G_{ij}-G_{l'k'}$(从集合中删除元素 $G_{l'k'}$), 将$(S_{h'},\bigcup_{i\geqslant 1}G_{ij},\Gamma_{l'k'})$压入栈顶,其中 $\Gamma_{l'k'}$为$\{f|f\in\Gamma$ 且对 $G_{l'k'}$是稳定的$\}$.

(3) 如果栈空,问题无解,退出.

从以上算法可以看出,由于$G\in\bigcup_{i\geqslant 1}G_{ij}$并且一个目标从目标集 UG_{ij}中删除当且仅当此目标最近被实现过,所以栈底子问题$(S_0,\bigcup_{i\geqslant n}G_{ij},\Gamma)$在搜索过程中始终包含问题$(S_0, G,\Gamma)$,即$(S_0,G,\Gamma)$是整个搜索过程的岗哨. 上述算法退出时,要么有解(结束于②的 d 步)或者(S_0,G,Γ)无解(结束于③步).

算法②中的 C 步是对操作集回溯,即扩大操作可用集 Γ_{lk}. ②中的 d 步将已实现的子目标从原来子问题中删除,用于将来可能的回溯,使得回溯时重新求解原问题,但却结束于另一个目标.

2 实验结果

目前,我们已经按上述方法实现了一个问题求解系统,并在该系统上求解河内塔问题、猴子与香蕉问题,八数码,Think－A－Dot[1]和 2×2×2 魔方等人工智能经典问题.

对于八数码、Think－A－Dot、河内塔、猴子与香蕉问题无须任何回溯就可得到解. 而 2×2×2 魔方问题有 30%的例子需回溯,但这些需回溯的实例,其被扩展结点数也大大小于宽度搜索.

以下是算符分析方法与一般宽度搜索法的实验结果比较:

方法 / 问题 / 结果	利用算符分析的搜索		宽度优先搜索	
	平均生成结点数	平均解路径长度	平均生成结点数	平均解路径长度
八数码	712	29	>20000	20
2×2×2 魔方	256	3.83	2278	3.67
Think－A－Dot	20	7.3	96	6.3

从以上结果看，利用算符分析的问题求解方法所得的结果虽不是最优的，但也与宽度搜索所得的最优结果相近，而且算符分析生成的结点数大大少于宽度优先搜索所生成的结点数.

在八数码问题上，我们将算符分析法与 A* 法做了比较. A* 法相应启发函数分别取为 W(n)(错位棋子数)和 P(n)(离家距离之和). 算符分析法生成结点数略多于 h(n)为 P(n)的 A* 方法，而少于 h(n)为 W(n)的 A* 方法. 由于算符分析法是先分解问题再搜索的，所以它的实际搜索时间比 h(n)为 P(n)的 A* 还少. 以下是实验结果：

结果 / 搜索方法	平均生成结点数	平均解路径长度
算符分析法	712	29
A* (h(n)=p(n))	540	20
A* (h(n)=w(n))	≈7000	20

我们的问题求解系统所接受的问题是一般的问题描述，不要求任何启发信息(与 A* 法不同)，它所进行的有效搜索完全得益于算符分析，而这一过程是系统自动进行的.

3　适用性分析

利用算符分析的搜索代价包括以下两部分：产生子目标层次集的代价和子问题求解代价.

设状态空间维数为 N.

按我们目前分析算符的方法，产生子目标层次集的代价是 $O(C_N^{[N/2]})$(即要尝试所有可能的分量组合)，这是 N 的指数函数. 但由于一般问题的维数不大，而且对同一目标状态只要作一次分析即可，因而它不是问题求解的主要代价.

要一般地分析求解每一子问题的代价是很困难的. 这里，我们考虑一种很富有说服力的情况. 即，假定各算符均无前提条件(如 Rubik 魔方中的算符)；最佳解路径的长度是初态与目标态差异数的单调递增函数 g. 此时，搜索的分枝系数 B 是可用算符的个数. 因而，若初态 S_0 与目标态的差异数是 N，按宽度搜索求解原始问题 (S_0, G, Γ) 的复杂度是 $B^{g(N)}$，其中 B 为

Γ的算符个数.对于各子问题(S_h, $\bigcup_{i\geqslant 1} G_{ij}$, Γ_{lk}),如果求解成功后较上一个子问题缩小了v个差异,那么求解该子问题的复杂度为$B_i^{g(v)}$,其中B_i为Γ_{lk}的算符个数.由于B_i小于B,g(v)小于g(N),因而子问题都比原问题简单.整个问题求解的复杂度等于各子问题求解复杂度之和.

算符分析的搜索在不需回溯的情况下,通过逐层实现一些子目标,差异的缩小是稳步的,使得搜索性能良好.当某些子问题求解不成功时,就要进行回溯.回溯不仅使搜索作了些无用的扩展,而且回溯后,面临的子目标集变小(见求解过程的②d步),从而增加子问题的难度.所以,算符分析法的搜索性能好坏,取决于回溯的多少,过多的回溯有可能使搜索复杂度接近于一般的宽度搜索(目前尚未发现这种情况).

在算符分析法中,为了避免过多回溯,在构造子目标层次集时,总选择具有让算符最大利用率的子目标.D. Joslin 和 J. Roach 的研究[3]显示,状态空间法中的子目标对应于状态图中的一个子图.一个子目标序列能否线性实现的必要和充分条件与子目标(或它们的联合)所对应的子图间的连通性有关.连通性越好,线性实现的可能性越大.一般来说,算符可应数越多,连通的可能性也越大.算符分析法不要求保持所有已实现子目标不变,而只要求保持最近实现的子目标不变,因而回溯的可能性更小.对于各子目标G_{ij}对应的子图为强连通的问题来说,根本不需要回溯.

总之,算符分析法在以下这些类问题中能表现出很好的搜索性能:①状态空间的维数小于算符数.②算符操作影响的状态分量数少.因为在这种情况下,保持某些分量不变,对算符集影响小,从而使子问题求解成功的可能性大.③缺乏良好的启发函数.算符分析法有自动分析算符,提出问题求解可能规划的功能,从而较盲目搜索有很好效率.在这一点上,算符分析法较 R. E. Korf 的 MPS[1]有独到之处,因为 MPS 方法要事先人为确定一个子目标求解次序.

参考文献

1 Korf R E. Learning to solve problems by searching for macro-operators. Pitman Publishing INC., 1985.

2 Stefik M. Planning with constraints (MOLTEN:part1). A. I., 1981;16.

3 Joslin D, Roach J. A theoretical analysis of conjunctive goals problems. A. I., 1989;41(1).

PROBLEM SOLVING THROUGH OPERATOR ANALYSIS

He Qinming, Wang Shenkang and Yu Ruizhao

(*Artificial Intelligence Institute, Zhejiang University, Hangzhou* 310027)

Abstract Solving Problems through serially removing differences is a kind of problem decomposition. The order of differences to be removed can be planned based on the analysis of operators. The main idea of this plan is to attain next subgoal without undoing the last attained subgoal. Thus, during problem decomposition, those subgoals which keep the operators most usable are chosen, some what like the least constraint propagation.

When a set of hierarchic subgoals is generated, the search of initial problem is to attain a serial of subgoals in the set, during which backtracking may happen.

Key words Problem decomposition, search, problem solving.