

一种任一时间联盟结构生成算法

胡山立¹, 石纯一²

¹(福州大学 计算机科学与技术系, 福建 福州 350002);

²(清华大学 计算机科学与技术系, 北京 100084)

E-mail: husl@fzu.edu.cn

http://www.fzu.edu.cn

摘要: 联盟形成是多 Agent 系统中的一个关键问题, 人们寻求能极大化联盟值的总和的联盟结构, 但通常情况下可能的联盟结构的数目太大, 以致不允许进行穷尽搜索而找出最优解. 给出了一个算法, 可在最小搜索量内保证找到一个与最优解相距在一个限界内的联盟结构. 然后, 这个任一时间算法进一步搜索, 渐进地给出越来越低的限界, 并急剧地降低这个限界, 在这一阶段, 此算法明显地优于由 Sandholm 等人给出的算法.

关键词: 联盟; 联盟结构; 算法; 多 Agent 系统

中图法分类号: TP18 **文献标识码:** A

带有自利 Agent 的多 Agent 系统正在变得越来越重要, 而联盟形成是多 Agent 系统中的一个关键问题. 人们希望寻求能极大化联盟值总和的联盟结构, 但通常所有可能的联盟结构的数目太大, 以至于不能进行穷尽搜索. 那么能否通过部分搜索找到一个能保证其联盟结构值与最优的联盟结构值相距在一个有限的限界内的联盟结构? 如果能够找到, 那么这个限界有保证的最小搜索量是多少? 这个问题已由 Sandholm 等人解决^[1]. 他们证明, 为了建立这样的限界, 搜索联盟结构图的最底二层(即含有一个或两个联盟的所有联盟结构)是必要且充分的. 然而含有一个或两个联盟的联盟结构数在所有可能的联盟结构数中只占有很小的比例, 两者之比当 n 增大时趋于 0, 而由最小搜索量建立的限界 $k(n)=n$, 却是随 n 的增大而增大的. 这个结果向人们提出第 2 个重要问题: 在最小搜索之后(此时限界为 n), 在没有启发信息的条件下, 应怎样进行进一步的部分搜索, 以尽快地降低这个限界呢? 这个问题不仅在理论上有意义, 而且随着 Agent 的计算能力的不断提高, 对那些 Agent 数目不大的系统也越来越具有实用价值.

1 问题描述

设 $A = \{A_1, A_2, \dots, A_n\}$ 是 Agent 集, 且 $n = |A|$, 我们定义一个联盟 S 是 A 的一个非空子集, 一个联盟结构 CS 是 Agent 集的一个划分, 从而同一个联盟结构中的联盟是不相交的, 称所有联盟结构的集合为 M , 定义联盟结构的值是该结构中所有联盟值的总和, 即 $V(cs) = \sum_{s \in cs} v(s)$. 我们作以下假设: (1) 和惯例一样^[2-7], 在特征函数对策(CFGs)中研究联盟形成. 在这样的对策中, 每个联盟 s 的值用一个特征函数 $v(s)$ 给出. 我们假定 $v(s) \geq 0$, 每个联盟的值与非该联盟成员的活动无关. (2) 在非超加对策中研究联盟形成^[7], 即联盟不具有超加性, 或超加性事先不知道(超加性是指对

* 收稿日期: 1999-07-13; 修改日期: 1999-10-28

基金项目: 国家自然科学基金资助项目(69733020; 69973023); 福建省自然科学基金资助项目(F00012)

作者简介: 胡山立(1944-), 男, 福建福州人, 教授, 主要研究领域为人工智能应用基础, 多 Agent 系统; 石纯一(1935-), 男, 河北山海关人, 教授, 博士生导师, 主要研究领域为人工智能应用基础.

任意不相交的联盟 $s_1, s_2 \subseteq A$, 有 $V(s_1 \cup s_2) \geq V(s_1) + V(s_2)$, 在这样的条件下, 联盟结构的生成是平凡的).

联盟结构生成过程可被看作是在联盟结构图中搜索, 如图 1 所示 (当 $n=4$ 时), 图中结点代表联盟结构, 弧线向下时代表两个联盟的合并, 向上时代表一个联盟分裂为两个联盟.

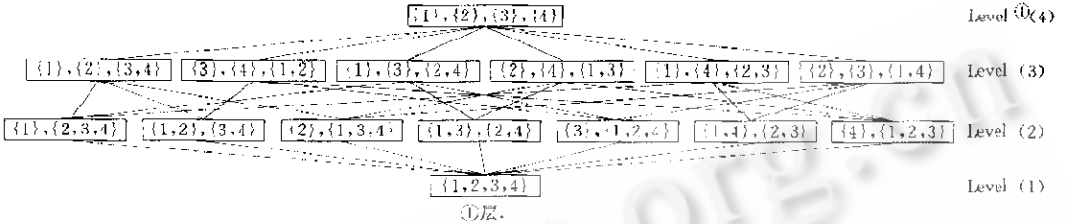


Fig. 1 Coalition structure graph for 4-Agent game
图 1 4-Agent对象的联盟结构图

联盟结构图由所有可能的联盟结构 (作为结点) 组成, 层次由底向上为 $1 \sim n$, 层 l 由所有恰好有 l 个联盟的联盟结构组成. 设 N 是 M 的已搜索的子集, 搜索以层为单位. 在赋值 (特征函数) $V_v(s)$ 下, 用 CS^* 记 M 中值最大的一个联盟结构, CS_N^* 记 N 中值最大的一个联盟结构, 如果对任意的赋值 V_v , 存在与 V_v 无关的 $K(n)$, 则满足 $\frac{V_v(CS^*)}{V_v(CS_N^*)} \leq K(n)$. 称 $K(n)$ 是搜索的解的限界, 它表示最坏情况下已搜索的局部最优解和全局最优解的距离, 也表示已得到的局部最优解的质量. 通常, $K(n)$ 除了与 n (从而与 M) 有关以外, 还与已搜索的局部 N 有关, 但通常情况下 N 从上下文可知, 或不需精确指明, 这时用 $K(n)$ 表示限界. 如果 $K(n)$ 是限界, 并且存在一个赋值 V_v , 使得 $\frac{V_v(CS^*)}{V_v(CS_N^*)} \leq K(n)$ 中的等式成立, 则称限界 $K(n)$ 是紧的.

文献[1]证明, 通过搜索联盟结构图的最底二层 (共有 2^{n-1} 个结点), 便能保证有限界 $K(n) = n$, 且这个限界是紧的, 并且这是能建立起限界的最小搜索. 现在的问题是, 由于最底二层结点数只占整个图的极小部分, 且限界为 n , 如果允许进行进一步的部分搜索, 那又应怎样选择搜索的顺序, 以便能快速地降低这个限界, 提高解的质量, 并在任一时间中断都能得到渐进的较优解呢?

2 算法

当 Agent 的个数 n 很小时, 联盟结构图的结点数不多, 可穷尽搜索. 下面只考虑 n 较大的情况 (例如, $n \geq 8$). 本文讨论的算法在联盟结构图上搜索, 搜索以层为单位, 不考虑同层中结点的搜索顺序.

2.1 算法 1

步骤 1. 搜索联盟结构图的层 1 和层 2 (即最底 2 层).

步骤 2. 只要时间等条件允许, 继续从该图的顶层开始, 向下隔层进行广度优先搜索 (即对 $i = 0, 1, 2, \dots, \lfloor \frac{n}{2} \rfloor - 2$, 搜索层 $n - 2i$), 直到搜索完层 4 或层 5 为止. 否则随时中断, 转步骤 4.

步骤 3. 只要时间等条件允许, 继续从该图的底部层 3 开始, 向上广度优先搜索那些还未搜索过的层, 直到搜索完整个图. 否则随时中断, 转步骤 4.

步骤 4. 返回至今为止所搜索到的联盟结构中, 具有最高联盟值的联盟结构.

2.2 联盟结构图的层次与限界的关系

我们称几个联盟一起被搜索过或一起出现过,是指它们一起被包含在某个已被搜索过的联盟结构中,称几个联盟在层 l 一起出现,是指它们一起被包含在层 l 的某个联盟结构中. 在计算或估计最坏情况下的限界时,经常将联盟结构图按层分为几个部分,对 CS^* 分别属于各部分的情况进行分析,如各种情况下都有限界 $K(n) \leq a (a > 1)$,则可得结论:限界 $K(n) \leq a$. 在这一过程中,一般不必考虑 CS^* 属于已搜索过的层的情况,因为这时 $K(n) = 1$.

定理 1. 为了建立限界 $K(n)$,搜索联盟结构图的最底二层是必要且充分的,此时限界 $K(n) = n$,且是紧的.

证明:参见文献[1].由定理 1 得出结论,无论给定的限界是多少,层 1 和层 2 都是要搜索的.

定理 2. 如果联盟结构图的最底二层已被搜索且层 l 及其以上各层均已被搜索 ($l > 3$),当 $n \equiv h-1 \pmod{h}$ 且 $n \equiv l \pmod{2}$ 时, $K(n) = \left\lceil \frac{n}{h} \right\rceil$, 且是紧的,否则 $K(n) = \left\lfloor \frac{n}{h} \right\rfloor$, 且是紧的,其中 $h = \left\lfloor \frac{n-l}{2} \right\rfloor + 2$.

证明:参见文献[1].

引理 3. 在限界 $K(n)$ 的计算中,对所有联盟 $S \in CS^*$,假设 $V(S) = 0$,不改变 $K(n)$.

证明:设 V_α 是联盟值的一种赋值,它给出最坏情况.显然,对联盟值的任意其他赋值 V_β ,不等式 $K(n) = \frac{V_\alpha(CS^*)}{V_\alpha(CS_N^*)} \geq \frac{V_\beta(CS^*)}{V_\beta(CS_N^*)}$ 成立.所以,对所有联盟 $S \in CS^*$,假设 $v(s) = 0$,不改变 $\frac{V_\alpha(CS^*)}{V_\alpha(CS_N^*)}$,从而不改变 $K(n)$. □

引理 4. 设 $l_1 < l_2 < l_3$,对层 l_1 中的任一联盟结构,如果有 m 个联盟在层 l_3 一起出现,则它们一定在层 l_2 一起出现.

证明:对层 l_1 中的任一联盟结构 CS_1 ,有 m 个联盟在层 l_3 一起出现,则可以对 CS_1 中另外 $l_1 - m$ 个联盟进行 $l_3 - l_1$ 次分裂,得到 l_3 的一个联盟结构 CS_3 ,途中生成层 l_2 中的一个联盟结构 CS_2 ,它含有 CS_1 中的那 m 个联盟,所以这 m 个联盟在层 l_2 一起出现. □

定理 5. 设 $l_1 < l_2$,如果 CS^* 在层 l_1 且层 l_2 已搜索,则层 l_2 以上各层是否搜索过不影响限界 $K(n)$.

证明:设 V_α 是给出最坏情况的一种赋值,由引理 3 不妨假设对所有 $S \in CS^*$, $V(S) = 0$,根据引理 4, $V_\alpha(CS_N^*)$ 的值当 N 为层 l_2 和 N 为层 l_2 及以上各层是一样的.所以,层 l_2 以上的各层是否搜索过不影响限界 $K(n)$. □

定理 6. 若层 $l (l \geq 2)$ 已被搜索且 CS^* 在上一层(即层 $l+1$),而该层未被搜索过,则限界 $K(n) \leq \frac{l+1}{l-1}$.

证明:由假设 CS^* 在层 $l+1$,从而有 $l+1$ 个联盟,其中任意两个联盟合并,其余 $l-1$ 个联盟不变,均生成层 l 的一个联盟结构,它含有 CS^* 中的 $l-1$ 个联盟,这样, CS^* 的任意 $l-1$ 个联盟都一起被搜索过(在 l 层).当只搜索过层 l 时,令 CS^* 中各联盟的值均为 1,不属于 CS^* 的联盟的值为 0, CS^* 是层 l 中值最大的联盟结构,则 $K(n) = \frac{V(CS^*)}{V(CS_l^*)} = \frac{l+1}{l-1}$, 且是紧的,这是因为上述赋值显然是产生最坏情况的一种赋值.考虑到可能还有其他已搜索过的层,所以限界 $K(n) \leq \frac{l+1}{l-1}$. □

定理 7. 只要层 3 还未被搜索,则限界 $K(n) \geq 2$.

证明:层3中的每个联盟结构由3个联盟组成,若 CS^* 在层3且其余各层都已被搜索,构造最坏情况: $CS^* = \{\{A_1\}, \{A_2\}, \{A_3, \dots, A_n\}\}$, $V(\{A_1\}) = V(\{A_2\}) = 1$, $V(\{A_3, \dots, A_n\}) = 2$; 对其余的联盟 S , $V(S) = 0$. 容易证明, 限界 $K(n) = \frac{V(CS^*)}{V(CS_n^*)} = 2$, 且是紧的. 考虑到如果有其他层还未被搜索, CS^* 还可能在其他层, 故 $K(n) \geq 2$. \square

定理 8. 如果整个联盟结构图虽然只有一个结点未被搜索, 限界 $K(n)$ 仍大于 1, 而这个结点若在层 3, 则限界 $K(n)$ 为 2, 且是紧的.

证明: 由于联盟结构图中没有两个结点是 一样的, 因此只要整个结构图未搜索完, 限界 > 1 . 层 3 中的结点(联盟结构)由 3 个联盟组成, 这剩下的惟一未被搜索的结点也不例外. 这 3 个联盟不可能同时出现过, 故这 3 个联盟中或者有一个只单独出现过, 例如 $\{\{A_1\}, \{A_2\}, \{A_3, \dots, A_n\}\}$ 中的 $\{A_3, \dots, A_n\}$, 或者没有一个只单独出现过, 例如 $\{\{A_1, A_2\}, \{A_3, A_4\}, \{A_5, \dots, A_n\}\}$. 容易证明, 前一种情况限界为 2, 后一种情况限界为 $\frac{3}{2}$, 但算法不能保证这个结点一定是后一种情况, 而 $K(n)$ 是最坏情况下的限界, 故限界为 2, 且是紧的. \square

2.3 算法 1 的几个性质

定理 9. 算法 1 在步骤 2, 层 $l (l > 3)$ 搜索结束时, 若 $n \equiv h-1 \pmod{h}$, 限界 $K(n) = \left\lceil \frac{n}{h} \right\rceil$, 且是紧的; 否则, 限界 $K(n) = \left\lfloor \frac{n}{h} \right\rfloor$, 且是紧的, 其中 $h = \frac{n-1}{2} \mid 2$.

证明: 因为层 l 刚被搜索, 而 $l > 3$, 故层 3 还未被搜索, 根据定理 6, 在最坏情况下 CS^* 不可能在层 l 之上, 否则 $K(n) < 2$, 与定理 7 矛盾. 设 CS^* 在层 l 以下, 根据定理 2 和定理 5, 且算法 1 在步骤 2, $n \equiv l \pmod{2}$, 定理 9 得证. \square

推论. 算法 1 在步骤 2, 当完成层 4 (n 为偶数) 或层 5 (n 为奇数) 的搜索时, 限界 $K(n) = 2$, 且是紧的.

证明: n 为偶数, 层 4 已被搜索, $h = \frac{n-4}{2} + 2 = \frac{n}{2}$, 根据定理 9, 限界 $K(n) = \left\lfloor \frac{n}{h} \right\rfloor = 2$, 且是紧的. n 为奇数, 层 5 已被搜索, $h = \frac{n-5}{2} + 2 = \frac{n-1}{2}$, 根据定理 9, 限界 $K(n) = \left\lfloor \frac{n}{h} \right\rfloor = 2$, 且是紧的. \square

定理 10. 算法 1 在步骤 3, 若层 l 及以下各层均已搜索, 而层 $l+1$ 未搜索, $3 \leq l \leq n-2$, 则限界 $K(n) \leq \frac{l+1}{l-1}$.

证明: 由定理 6 可知, 若 CS^* 在层 $l+1$, 则 $K(n) \leq \frac{l+1}{l-1}$. 由于 l 层以上是隔层已被搜索, 且 $\frac{l+1}{l-1}$ 的值随 l 的增大而减小, 故限界 $K(n) \leq \frac{l+1}{l-1}$. \square

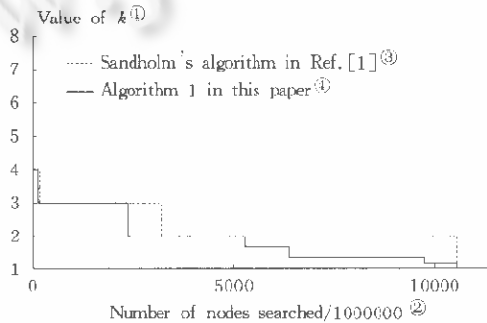
总之, 定理 9 表明本文提出的算法在步骤 2 阶段, 渐进且快速地把限界 $K(n)$ 从 n 降到 2, 在步骤 3 阶段, 根据定理 10, 限界 $K(n) \leq \frac{l+1}{l-1}$, 随 l 的增大而减小, 渐进地把限界从 2 降到 1, 从而是个任 一时间的算法. 另外, 当 $n \equiv h-1 \pmod{h}$ 时, 由定理 9, 限界 $K(n) = \left\lceil \frac{n}{h} \right\rceil$, 而不是 $K(n) = \left\lfloor \frac{n}{h} \right\rfloor$, 两者相差 1, 如果不愿失去这一渐进的小阶段, 可在算法 1 的步骤 2 中, 当 $n \equiv h-1 \pmod{h}$ 时, 增加搜索下一层. 具体讨论从略.

3 与相关算法的比较

以前的特征函数对联盟结构生成算法, 有的不是通用的, 即它们对要考虑的联盟结构或特征

函数作了某些限制,例如文献[8].而以前所有通用的特征函数对策联盟结构生成算法,正如文献[1]所说,就所知道的,例如,文献[4,6]不能建立最坏情况的限界,因为它们的搜索少于 2^{n-1} 个联盟结构.

下面将本文提出的算法1与文献[1]的算法进行比较.(1)在最小搜索之后,由顶向下搜索时,我们用隔层搜索取代了文献[1]的逐层搜索.定理9表明,在把限界从 n 降到2这个阶段中的每个小阶段减少搜索层数的一半.(2)定理7表明,文献[1]的算法只剩下层3还未搜索时,限界还是2.即使只剩下一个结点还未搜索,根据文献[1]的算法,这一结点在层3.定理8表明,文献[1]的算法得到的限界还是2.而本文提出的算法1,在步骤3中,根据定理10,限界 $K(n) \leq \frac{l+1}{l-1}$,随 l 的增大而减小,渐进地把限界从2降到1.因此在这一阶段,文献[1]中的算法不是任一时间的算法,而本文给出的算法是任一时间的算法.例如,如果问题要求限界 $K(n) < \frac{3}{2}$,文献[1]的算法要搜索完整个联盟结构图,而本文提出的算法在步骤3中搜索完层5或层6就可以了,即对层6以上一半的层可以不必搜索.搜索结点数的比较如图2所示,当 $k \geq \lceil \frac{n}{2} \rceil$ 时,两种算法相同,故省去.



① k 值,②已搜索的结点数/1000000,③文献[1]中Sandholm的算法,④本文的算法1.

Fig. 2 Ratio bound k ($k \leq 8$) as a function of search size, when $n=16$

图2 $n=16$ 时,限界 k ($k \leq 8$)作为搜索大小的函数($k \leq 8$ 时)

4 结束语

通用的特征函数对策联盟结构生成算法以往研究较少,在文献[1]以前,这方面已有的算法不是最坏情况下有保证的,即它们不能建立最坏情况下的任何限界.文献[1]给出的算法是目前已知的在这方面的最好结果.本文给出的算法在最小搜索阶段(它们只占联盟结构图的极小部分)和文献[1]一样,在最小搜索之后,明显地优于文献[1]中给出的算法.它不仅在把限界从 n 降到2这一阶段中的每个小阶段把搜索量下降大约一半,而且在把限界从2降到1的过程中是渐进的,没有出现文献[1]的算法中的停滞现象.因此在这一阶段,文献[1]中的算法不是任一时间的算法,而本文给出的算法是任一时间的算法.

本文详细讨论了各层搜索对限界的影响,所提出的算法对于完成最小搜索后有进行进一步部分搜索的系统,具有实用价值.

另外,本文给出的算法,也可像文献[1]那样进行分布计算,讨论从略.

References:

- [1] Sandholm, T. W., Larson, K., Andersson, M. R., et al. Anytime coalition structure generation with worst case guarantees. In: Proceedings of the 15th National Conference on Artificial Intelligence. Menlo Park, CA: AAAI Press, 1998. 46~

- 54.
- [2] Kahan, J. P., Rapoport, A. *Theories of Coalition Formation*. Hillsdale NJ: Lawrence Erlbaum Associates Publishers, 1984.
- [3] Shehory, O., Kraus, S. Task allocation via coalition formation among autonomous agents. In: Mellish, C.S. ed. *Proceedings of the 14th International Joint Conference on Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1995. 655~661.
- [4] Shehory, O., Kraus, S. A kernel-oriented model for coalition formation in general environments: implementation and results. In: *Proceedings of the 13th National Conference on Artificial Intelligence*. Menlo Park, CA: AAAI Press, 1996. 134~140.
- [5] Zlotkin, G., Rosenschein, J.S. Coalition, cryptography and stability: mechanisms for coalition formation in task oriented domains. In: *Proceedings of the 12th National Conference on Artificial Intelligence*. Menlo Park, CA: AAAI Press, 1994. 432~437.
- [6] Ketchpel, S. Forming coalitions in the face of uncertain rewards. In: *Proceedings of the 12th National Conference on Artificial Intelligence*. Menlo Park, CA: AAAI Press, 1994. 414~419.
- [7] Sandholm, T.W., Lesser, V.R. Coalitions among computationally bounded agents. *Artificial Intelligence*, 1997, 94(1):99~137.
- [8] Shehory, O., Kraus, S. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 1998, 101(1-2): 165~200.

An Anytime Coalition Structure Generation Algorithm*

HU Shan-li¹, SHI Chun-yi²

¹(Department of Computer Science and Technology, Fuzhou University, Fuzhou 350002, China);

²(Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

E-mail: husl@fzu.edu.cn

http://www.fzu.edu.cn

Abstract: Coalition formation is a key topic in multi agent systems. People try to search for coalition structure that maximizes the sum of the values of the coalitions, but in most cases the number of coalition structures is too large to search for the optimal one exhaustively. In this paper, an algorithm is presented that within the minimal amount of search can guarantee to find a coalition structure which is within a bound from optimum. Then, the anytime algorithm searches further, and establishes a progressively lower bound, and lowers the bound rapidly. In this stage, it evidently outperforms the algorithm presented by Sandholm etc, which is the new-made better result in this area.

Key words: coalition; coalition structure; algorithm; multi-agent system

* Received July 13, 1999; accepted October 28, 1999

Supported by the National Natural Science Foundation of China under Grant Nos. 69733020, 69973023; the Natural Science Foundation of Fujian Province of China under Grant No. F00012