空间有限条件下博弈搜索算法的优化

张 怀 许林英
(天津大学计算机学院 天津 300072)

摘 要：alpha-beta剪枝算法是一种传统的搜索算法，它大大减少了相同搜索深度下的计算量，但其仍然不能满足有限时间内进行搜索的需求。为此，有很多针对该算法的优化方法，但这些优化方法大都是以消耗更多空间为代价的。本文从博弈程序的全局考虑，提出几种优化策略，在有限的空间条件下，以较少的计算量，获得较高智能性。经过实验测试，在PC机中对相同的搜索层次、使用相同空间的算法所消耗的时间进行对比，发现优化方法的算法可以大幅度降低消耗的时间，最多可以节省10%的时间。

关键词：博弈搜索算法；人工智能；搜索技术；空间有限
中图分类号：TP312 **文献标识码**：A

Approach of optimizing adversarial search in a limited space

Zhang Huai Xu Linying
(Tianjin University, Tianjin 300072)

Abstract: alpha-beta algorithm is a kind of typical method for optimizing adversarial search, which reduces the computation amount obviously in the same search depth, but it still does not meet the requirement of searching in a limited time. So, there are many enhancements on its optimization, but most of those enhancements consume more space. This paper presents some strategies of enhancement from a microscopically angle, and those methods could use less computation amount and get more intelligence at the same time. The experiment shows that the optimized algorithm could reduce almost 10 percent time compared to the non-optimized algorithm at the same condition with equal depth and space.

Keywords: conventional search algorithm; artificial intelligence; searching technique; limited space

0 引 言

目前，已经有很多对alpha-beta算法的优化，提高了搜索的性能，其中有一些已经被广泛证实是有效的算法，它们主要包括以下几种：置换表(transposition tables)、驳斥表(refutation tables)、窄窗搜索(aspiration search)和最小窗口搜索(minimal window)、启发式搜索(heuristic)，它们的主要思想在这里就不再赘述。本文提出了新的优化算法，并与其他的算法作了详细的对比。

1 博弈程序各部分性能分析

本文讨论的博弈程序主要是指确定的、二人、零和、完备信息的博弈搜索。也就是说，没有随机因素的博弈在两个人之间进行，在任何一个时刻，一方失去的利益即为另一方得到的利益，不会出现“双赢”的局面，而且在任何时刻，博弈的双方都明确的知道每一个棋子是否存在和存在于哪里。

博弈程序就是产生一个对自己最有利的走步，主要产生过程就是，博弈程序根据当前棋盘上的局面，获得所有自己能够行进的所有走步，然后对每一种走步进行试探。所谓试探就是如果博弈程序采用该走步，那么对手有可能会产生哪个走步，此时博弈程序会获得对方所有可能会出现的走步，再对对方的每一种可能的走步进行试探，如此反复，当达到指定的层次时，停止试探，并对局面进行评估，博弈程序选择对自己最有利或者选择对自己损害最小的那个走步，于是产生一个试探的路径，这个路径的第一步就是当前局面下最有利的一步。

从前面的博弈程序搜索过程可以看出，程序可以分成4个模块：

(1)当前局面下，所有可走步的产生(Generate)；

(2)对当前局面进行评估(Evaluate)；

(3)产生新的局面(Forward)；

(4)恢复前一个局面(Backward)；

以跳棋游戏的博弈程序为例，搜索深度从2层到5层，各模块消耗的时间如表1所示，此时博弈程序会由表中的数据得出结论：随着搜索深度的变化各模块消耗的时间显

万方数据

著增加,另外,博弈程序各部分时间耗费所占的比例也显著变化,不能保持一种稳定的比例。

表1 中国跳棋博弈程序各模块消耗时间对照表 ms

	2层	3层	4层	5层
Generate	3.2	42.2	1881	14 445
Evaluate	1.0	51.6	574	20 476
Forward	1.3	70.1	794	22 270
Backward	1.4	56.9	796	20 960
Total	6.9	220.8	4 045	78 151

2 新的优化算法

博弈程序消耗的时间与搜索的层数紧密相关,更准确地说,时间是与搜索的节点(局面)数紧密相关。如图1所示,一种简单的游戏树,可以看出每搜索更深一层节点数将呈几何级数增长,每层更多的节点(局面)意味着将要在更多的局面下调用可走步产生模块;将要对更多的局面进行局面评估;将要调用更多的局面产生模块和回复局面模块。因此降低博弈程序的时间就是尽可能地缩减搜索的节点数,而不能单一地减少某个模块消耗的时间,各模块耗费所占比例变化显著的事实就说明了这一点,各模块的时间消耗都是依赖节点数的,即单一模块的性能提高与节点数的减少相比是微不足道的。那么提高博弈程序性能的关键就是减少搜索的节点数。很多算法的优化方法是以空间为代价缩减搜索的节点数。如果空间有限,如何控制节点数呢。

一种可行的方法就是对当前层的所有局面进行评估,而不是等到递归到最深层时才进行局面评估,然后删去对自己最不利的 N 个走步。这种思路基于如下的经验判断:智能者不会选择当前局面较差的走步,因为如果当前局面选择较差的走步,那么在更深一层能够进行弥补的概率几乎为零。如果 N 取1,那么可以想象没有哪个智能者会选择当前局面下最差的一步,这是合理的,也由此可以适当扩大 N 的取值,减少搜索的节点数。

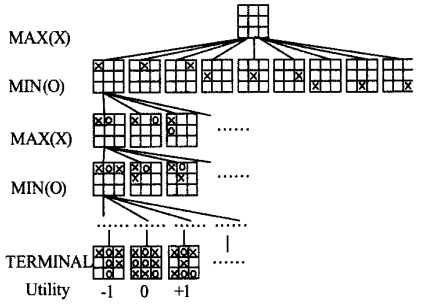

图1 搜索层数与搜索节点数

上面的方法基于经验的成分大一些,理论性不十分可靠,下一种方法将会解决这个问题。

依靠估值函数对局面进行评估,确实存在不精确的问题,而且博弈游戏的规则千差万别很难找到一种很通用的估值方法,另外对局面的评估在一定程度上受博弈程序设计者主观的影响。尽管如此,还是可以在局面评估之前,也就是在产生当前局面所有可走步的时候,理性选择或区分好的走步和差的走步,这与上一种方法的估值不同,它是在可走步产生模块加入一些智能的分析,从而获得较少的节点,达到较少搜索节点的目的。

3 实验设计

以双人中国跳棋为博弈程序的实例,测试和比较两种优化方法。选择中国跳棋作为实例,主要基于以下几个原因:

(1)中国跳棋的规则简单,使得估值函数受主观的影响较小,便于更客观地比较各种优化算法的性能,排除了一些主观因素。

(2)中国跳棋的棋子走法简单,便于测试和比较对可走步产生模块的优化方法。

(3)中国跳棋的获胜规则简单,便于比较在保持相同智能性前提下,各种优化算法的性能

(4)中国跳棋的棋盘和棋子简单,便于比较在空间有限条件下,各种优化算法的空间消耗情况。

中国跳棋的棋盘共有121个点,棋子双方各有10粒,假设每个点用1个字节表示,那么表示中国跳棋的局面就需要121个字节,另外跳棋有移动和跳步两种走法,这需要频繁检测棋子邻接点的状态,所以设计一个邻接表会显著减少程序运行时间,棋盘上每个点都含有6个邻接点(棋盘边界点除外),于是中国跳棋的基本空间需求是847个字节。

通过经验判断,中国跳棋的每个局面可以产生60至100个新局面,即有60到100个走步,假设平均有80个走步,每个走步用2个字节表示,那么中国跳棋博弈程序需要的空间如表2所示。

表2 中国跳棋博弈程序需要的空间 B

	2层	3层	4层	5层	6层	7层
基本空间	847	847	847	847	847	847
走步空间	160	320	480	640	800	960
总合	1 007	1 167	1 327	1 487	1 647	1 807

本实验所运行的硬件环境:Pentium® 4 CPU 2.60 GHz,512MB RAM。软件环境:Microsoft® Windows XP sp1,Visual C++ 6.0。

实验程序是用C语言编写的中国跳棋程序,只包含上述4个模块和一些必要的输入输出,使用2个函数统计时间。

万方数据

两种优化算法中的第一种算法直接删除最差的 N 个走步，需要对当前局面进行排序；第二种算法采用了与第一种算法不同的可走步生成函数，主要不同点是前者优化了可走步产生模块，因为后者的可走步产生模块以棋子为中心，对每一个棋子采用简单的迭代方法产生中国跳棋中的跳步，较费时，而前者以棋盘上空闲的位置为中心，用聚类的办法取代了迭代，可以得到两点好处：一是省时；二是可以方便地加入一些智能约束条件，进一步缩减搜索的节点数，不需要排序。把第一种算法命名为标准排序化算法，把第二种算法命名为智能产生算法。

标准排序化算法涉及到 N 值的选择；智能产生算法涉及到智能条件的选择。

N 值的选择：测试 $N=0$ 和 N 从 15 到 45 的变化，对博弈程序性能的影响。

智能条件的选择（中国跳棋）：测试有、无智能条件对博弈程序性能的影响，其中智能条件为：连续向后跳 2 步以上的可走步不产生；跳回起始位置的可走步不产生。

4 实验结果

alpha-beta 算法与标准排序化算法的比较如表 3、表 4 所示。

表 3 标准排序化算法（N 取 0,15,30,40）时间

时间(ms)		2层	3层	4层	5层	6层
alpha-beta 算法		4.391	71.088	2 384	26 036	654 000
标准排序化算法	$N=0$	10.352	36.387	813.9	2 323	45 785
	$N=15$	8.51	28.6	538.7	1 533	25 048
	$N=30$	6.64	20.85	329	910	11 442
	$N=40$	5.483	18.86	224	595	6 078

表 4 标准排序化算法（N 取 0,15,30,40）实际估值次数

估值次数(次)		2层	3层	4层	5层	6层
alpha-beta 算法		852	44 786	494 089	17 054 342	138 226 276
标准排序化算法	$N=0$	5 974	23 859	480 112	1 717 549	27 190 061
	$N=15$	4 892	18 570	326 654	1 110 086	14 839 327
	$N=30$	3 795	13 570	200 396	637 809	6 770 506
	$N=40$	3 071	10 462	133 332	400 705	3 556 382

alpha-beta 算法与智能产生算法的比较如表 5，表 6 所示。

表 5 智能产生算法（有无排序、有无智能条件）时间

时间(ms)		2层	3层	4层	5层	6层
alpha-beta 算法		4.391	71.088	2 384	26 036	654 000
智能产生算法	无智能条件	5.508	44.933	1 212	8 103	126 068
	无智能条件(排序)	12.801	40.113	1 102	3 004	69 241
	有智能条件	4.243	28.078	537	3 080	42 127
	无智能条件(排序)	7.531	25.03	551.1	1 512	27 959

表 6 智能产生算法（有无排序、有无智能条件）实际估值次数

时间(ms)		2层	3层	4层	5层	6层
alpha-beta 算法		852	44 786	494 089	17 054 342	138 226 276
智能产生算法	无智能条件	732	15 643	110 326	1 991 842	12 004 549
	无智能条件(排序)	6 340	25 680	537 080	1 956 523	34 000 153
	有智能条件	514	8 071	38 778	635 014	3 199 395
	无智能条件(排序)	3 805	15 604	268 075	962 315	13 752 634

万方数据

分析上述数据可以得出:

(1)改善性能程度最大的就是标准排序化算法($N=40$),它在搜索6层时,使时间降低到原来的0.9%;然后依次是标准排序化算法($N=30$)时间降低到原来的1.75%;标准排序化算法($N=15$)时间降低到原来的3.8%;智能产生算法(智能条件,排序)时间降低到原来的4.3%;智能产生算法(智能条件,不排序)时间降低到原来的6.4%;标准排序化算法($N=0$)时间降低到原来的7%;智能产生算法(无智能条件,排序)时间降低到原来的10.6%;智能产生算法(无智能条件,不排序)时间降低到原来的19.3%。表面上看,N取较大值的标准化排序算法可以获得最大的改善,但是如前所述,这种算法存在一种N值选择的问题,因为每一层局面数都是不同的,如果对这种动态变化的局面数,采用固定的N值,有可能出现下面的问题:当某一层的局面数较少,不适当的N值会将有价值的可走步忽略。于是N值的选择很关键,不能过大也不能过小。而智能产生算法不会出现这样的问题。

(2)算法的选择:因为两种算法的很多不同,所以博弈程序选择算法时就要具体分析,下面给出一些参考:如果估值函数能够较精确地对局面进行估值,可以选取较大N值的标准排序化算法,N值最大不应超过每层平均局面数的一半;如果游戏规则可以较大限度地判断走步的合理性,建议选取智能产生算法。

5 结 论

两种新的博弈程序算法是对标准alpha-beta算法的优化,它们的最大特点就是不占用额外的空间而获得良好的优化效果。

从优化效果看,标准排序化算法有更大程度的优化;而智能产生算法具有更好的通用性,实际上它是一种基于知识的方法,算法中的智能条件就是一种知识的显式表示。

参 考 文 献

[1] SCHAEFFER J. The History Heuristic and Alpha-Beta Search Enhancements in Practice [J]. IEEE Transactions on pattern analysis and machine intelligence,11(11).

[2] GILLOGLY J. Performance analysis of the technology chess program [D]. Pittsburgh: Carnegie-Mellon Univ.,1978.

[3] PLAAT R R. Search & Re-Search [D]. Rotterdam: Erasmus University,1996.

[4] BRUIN A, PIJLS W. Trends in Game tree Search[J]. Lecture Notes In Computer Science, 1996, 1175: 255-274.

[5] PLAAT A,SCHAEFFER J,PIJLS W,et al. SSS*=α-β+TT[R]. Canada: Technical Report TR-CS-94-17, Department of Computer Science, University of Alberta,Edmonton,AB,1994.

[6] SCHAEFFER J. The History Heutistic and Alpha-Beta Enhancements in Practice[J]. IEEE Tranactions On Pattern Analysis and Machine Intelligence, 1989,11.

[7] 王小春. PC游戏编程(人机博弈)[M]. 重庆:重庆大学出版社,2002.

[8] 肖其英,王正志. 博弈树搜索与静态估值函数[J]. 计算机应用研究. 1997,14(4):74-76.

作 者 简 介

张怀,男,1981年出生,硕士研究生,主要研究方向为信息获取技术,工作流管理技术。

E-mail:coolbez321@yahoo.com.cn

许林英,男,1963年出生,副教授,硕士生导师,主要研究方向为计算机网络,数据库技术,软件工程。

(上接第10页)

[4] KARAN B, VUKOBRATOVIC M. Calibration and Accuracy of Manipulation Robot Models-An Overview [J]. Mech. Mach. Theory,1994,29(3):479-500.

[5] DENAVIT J,HARTENBERG R. A kinematic notation for lower pair mechanism based on matrices[J]. ASME Journal of Applied Mechanics,1955,22(6):215-221.

[6] 苏步青. 应用几何教程[M]. 上海:复旦大学出版社,1990.

[7] 王学影,刘书桂,王斌,等. 关节臂式柔性三坐标测量系统的数学模型及误差分析[J]. 纳米技术与精密工程,2005,3(4):262-267.

[8] 叶东,车仁生. 仿人多关节坐标测量机的数学建模及其误差分析[J]. 哈尔滨工业大学学报,1999,31(2):28-32.

作 者 简 介

王斌,男,1981年10月出生,天津大学精密仪器与光电子工程学院测试计量技术及仪器专业在读硕士研究生,主要研究方向为关节臂式柔性三坐标测量系统。

E-mail: bwang116@gmail.com

万方数据