第12卷 第5期 2012年2月
1671—1815(2012)05-1052-05

科 学 技 术 与 工 程
Science Technology and Engineering

Vol. 12 No. 5 Feb. 2012
© 2012 Sci. Tech. Engrg.

计算机技术

一种改进的基于博弈树模型的五子棋系统

杨云强 吴 姣
(中国航空计算技术研究所,西安 710065)

摘 要 通过对五子棋算法的研究,探讨了知识抽象、知识表示、估价函数、博弈树及搜索策略等人工智能领域的问题,并基于博弈树模型设计了一个智能五子棋系统。该系统采用多链表结构的知识表示方法记录棋局信息,可以全面地描述和分析棋局形势。最后,结合局部性原理,采取增量分析法、α-β剪枝及低层剪枝等手段加速分析和搜索效率,提高了该系统的反应速度和智能化程度。

关键词 五子棋 人工智能 博弈树 估价函数 知识表示
中图法分类号 TP301.6; **文献标志码** A

五子棋是起源于中国古代的传统黑白棋种之一。因为变化多端,不仅能增强思维能力,提高智力,而且非常富有趣味性和消遣性,所以成为一项深受人们喜爱的益智游戏。

在人工智能领域,对于五子棋这种棋子种类较少,但走法又十分富有变化的项目,具有非常强的研究价值和普适性。在推导五子棋博弈算法过程中,对知识抽象、知识表示、估价函数、博弈树和搜索策略进行了一些研究,设计并实现了一个人机对弈的智能五子棋系统。

1 基本问题与解决方案

1.1 基本问题

实现人机对弈,重点需要关注以下几个问题[1]:

(1) 一种描述棋局的表示方法,能够让程序知道博弈的状态;

(2) 一套走棋的规则,能够让程序判定应由谁来走棋,走棋是否合法,以及棋局的输赢;

(3) 一种从棋局提取知识的方法和一种局面优劣的评估方法,使得机器人能够分析各种合理的走法,从而做出智能的选择;

(4) 一个人机交互界面,让这个程序跑起来。

2011年11月21日收到,12月1日修改
第一作者简介:杨云强(1986—),男,江苏宜兴人,硕士研究生,研究方向:计算机科学与技术。

1.2 解决方案

针对上述问题,现提供如下的解决方案:

(1) 构建一个状态机模型,通过状态机中各个状态的迁移变化来反映博弈状态的变化;

(2) 基于面向对象思想,抽象出了人类、机器人和裁判三个对象。走棋合法性和棋局的输赢专门来由裁判这个角色判断执行。走棋的顺序则可以由裁判和状态机配合实现;

(3) 基于人工智能,对棋局进行合理的知识抽象与表示,在博弈树算法基础上加以改进与实现智能选择;

(4) 最后,设计一个 MFC 的游戏界面,如图1所示,那么一个智能五子棋系统就建立了。

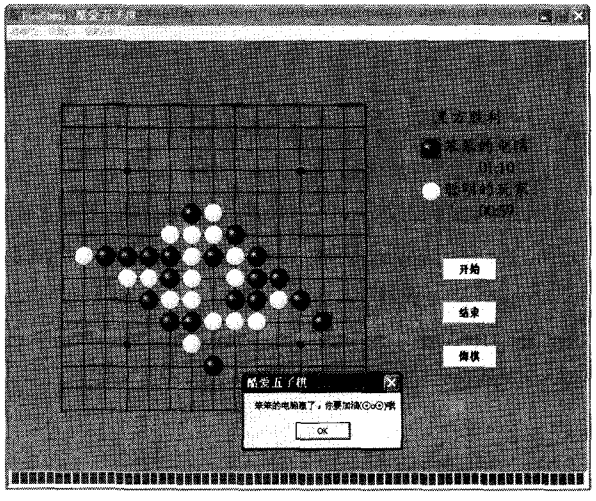


图1 "酷爱五子棋"游戏界面

2　智能五子棋系统的设计

2.1　软件框架和状态机

首先,基于人机对弈的流程,搭建一个软件框架,程序结构如下所示,可以看出难点在于输赢判定和机器人下棋。

① 初始化;
② 如果机器人执黑,先下第一步棋
while(1){
③ 人类下棋
④ 判断是否有输赢。是,break
⑤ 机器人下棋
⑥ 判断是否有输赢。是,break
};
⑦ 显示棋局中止状态;
⑧ 是否开始新一局游戏。是,跳转到①;
⑨ 退出。

其次,根据对该游戏流程的分析,可以抽象出三个对象:人类(theHuman,玩家1)、机器人(theRobot,玩家2)和裁判(theGameCtrl,负责游戏控制和裁判工作)。

第三,不妨定义一些主要的游戏状态:初始化、等待黑方、等待白方、黑方赢、白方赢、和棋,以及错误。由此,通过建立一个状态机基本模型(图2)来控制游戏的正常运行,不仅可以更清晰地描述对弈棋局的状态,而且也便于程序的调试和软件可靠性的验证。

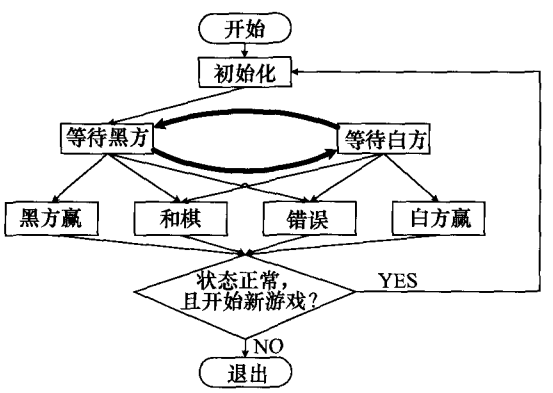


图2　状态机模型

2.2　基础算法模型

2.2.1　棋局形势的描述

对于棋盘(图3),定义了图4所示的坐标轴和方向。方向共分为"横"(HENG)、"竖"(SHU)、"撇"(PIE)、"捺"(NA)4类,同时编有序号。坐标(x,y)处各方向对应序号的定义为:横$[x]$,竖$[y]$,撇$[x+y]$和捺$[14-x+y]$。x和y的值是从0到14,因此,"横"向和"竖"向的编号从0到14,"撇"向和"捺"向的编号从0到28。

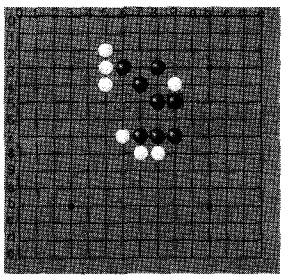
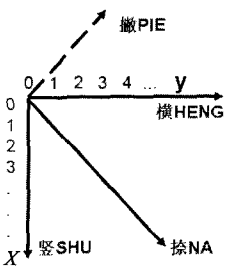


图3　棋局示例



4　棋盘坐标和方向定义

为了描述棋局,本文定义了连串结构体link_t,连串类型表TypeList和连串方向表DrcList,将空白位置所属的连串类型、方向和数量等信息使用结构体链表保存起来。

```
struct link_t {
int x, y;     /* 空白点坐标   */
int player;     /* 1. 黑方或白方 */
int type;     /* 2. 连串方向   */
int drc, drcNo;  /* 3. 连串方向和编号 */
struct link_t * type_next; /* 类型链表指针 */
struct link_t * type_prev;
struct link_t * drc_next;  /* 方向链表指针 */
struct link_t * drc_prev;
...
};
struct link_t * TypeList[3][TYPENUM];  /* 连串类型链表 */
struct link_t * DrcList[4][2 * N - 1];     /* 连串方向链表 */
```

以图3棋局为例,位置(5,8)和(5,9)处黑方已经在横[5]方向上连成了"活二",那么空白位置(5,7)处就是黑方横[5]方向的"活三"点。同理,空白位置(5,6)处,既是黑方横[5]和撇[11]方向的"双活三"点,也是白方捺[15]方向的"活二"点。图3棋局将形成图5的结构体链表,这种知识表示方法可以有效地反映棋局形势,便于后续的棋局评估。

2.2.2　单点估价函数

估算某个空白位置对于下棋方的重要程度时,

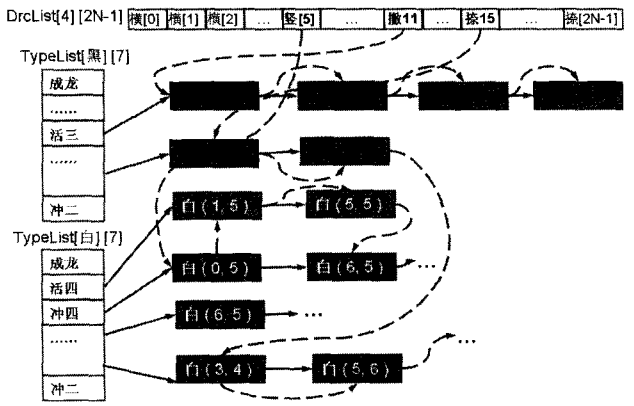


图5 结构体链表

既需要考虑该处对于己方的价值，也要考虑到对于对方的价值[2]。以黑方下棋为例，利用式(1)，即可求得(i,j)处对于黑方的单点估价值$H(i,j)$black。

$$H(i,j)\text{black} = \mu\ \text{BlackValue}(i,j) + \text{WhiteValue}(i,j) \tag{1}$$

$$H(i,j)\text{white} = \mu\ \text{WhiteValue}(i,j) + \text{BlackValue}(i,j) \tag{2}$$

其中，(i,j)代表空白位置的坐标。BlackValue(i,j)代表该点对于黑方的价值。WhiteValue(i,j)代表该点对于白方的价值。假如在该处放下黑子，计算出由该黑子连成的各方向上“活二”、“冲四”、“双活三”、“活三冲四”等组合的个数，并将个数乘以相应的估价值再求和，即可得出 BlackValue(i,j)，如式(3)所示。进攻系数$\mu=1.1$，考虑到下棋时，采取进攻要优于防守的策略，特意引入了这个系数。

$$\text{BlackValue}(i,j) = \sum (\ \text{BlackCount}[t](i,j)\ \text{TypeValue}[t]\ ) \tag{3}$$

$$\text{WhiteValue}(i,j) = \sum (\ \text{WhiteCount}[t](i,j)\ \text{TypeValue}[t]\ ) \tag{4}$$

(t=“成龙”,“活四”,“冲四”,……)

假设机器人执黑棋，那么在其下棋时，利用单点估价函数，选择所有空白位置中选择$H(i,j)$black值最大者落子即可。这种基本的智能选择方法，实际上就是一种贪心算法。该算法比较简单，具有了较强的智能性，但是没有考虑到整个棋局的形势和未来的发展趋势，所以需要设计一种更加全面综合的评估方法。

2.3 博弈树模型算法

2.3.1 全局估价函数

单点估价函数只能估算单个空白位置的价值，但全局估价函数可以估算所有的空白位置的总价值，进而评估整个棋局。对于当前棋局，将所有空白位置对下棋方的有利和不利程度总和相减即可得出全局估价值。以黑方为例，利用式(5)，即可求得黑方在当前棋局下的全局估价值 F_black。

$$\text{F\_black} = \sum \text{BlackValue}(i,j) - \sum \text{WhiteValue}(i,j) \tag{5}$$

$$\text{F\_white} = \sum \text{WhiteValue}(i,j) - \sum \text{BlackValue}(i,j) \tag{6}$$

2.3.2 极大极小算法

甲乙双方对弈，假定现在先由甲走棋，甲有若干种走法，而对于甲的任一走法，乙也有对应的多种不同走法，然后再次轮到甲走棋，甲同样又有若干种方法应对……，如此反复。将上述棋局记录下来，即可构成了一棵树，如图7所示，这就是博弈树模型[3]。

假定博弈双方都很聪明，足够智能。那么在甲下棋时，甲肯定选择对甲最有利的走法；而轮到乙下棋时，乙必然会选择对甲最不利的走法。以图6情况为例，在A棋局情况下，如果甲选择走B棋局，那么乙将选择走对甲不利的E棋局；如果甲选择走C棋局，乙将选择走G棋局；如果甲选择走D棋局，那么乙将选择走K棋局……所以，综合来看，甲应该选择走B棋局，这样对自己最有利，这种算法就是极大极小算法[4]。

图6中，叶子节点棋局的估价值是根据全局估价函数(F_black 或 F_white)直接计算得到的，其它非叶子节点棋局的估价值则是利用上述逻辑关系逐层倒推出来的。

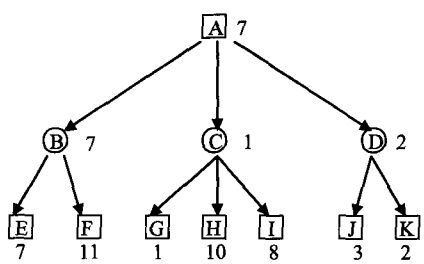


图6 博弈树模型

采用博弈树模型算法,可以有效地预判棋局未来的发展趋势,估算新下棋子对于整个棋局的影响力,使得机器人在智能性方面有明显的提高。树的层次越深,表明机器人考虑的走棋步骤越多。每个节点向下扩展的子节点越多,代表考虑得越全面。但是,博弈树越庞大,算法的复杂度就越高,效率也就越低,必须对其进行优化设计,使得机器人能够在棋手能忍受的有限时间内,做出快速又明智的选择。

2.4 算法优化与实现

2.4.1 增量分析法

因为博弈树叶子节点棋局的估价值是直接由全局估价函数计算得来,所以如何对棋局进行快速分析将直接影响到算法的运行效率。

整个棋局的信息由"横[0]、横[1]……横[14]、竖[0]……竖[14]、撇[0]……撇[28]、捺[0]……捺[28]"共88个不同方向上的不同类型连串所组成。但如果针对棋局的每一次变化,都把棋盘的88个不同方向扫描一遍,那将非常耗时,严重影响棋局分析的效率。通过观察可以发现:每新下一颗棋子,它对于棋盘上连串的影响仅仅是在该坐标(x,y)所处的"横竖撇捺"4个方向上。所以,只需要重新扫描对应的这4个方向,而不必88个方向全都扫一遍。采用这种增加分析法,在棋局分析阶段的效率将直接提高22倍。

2.4.2 局部性原理

如何控制博弈树的层次和每层扩展分支的数量,对机器人的智能程度和反应速度非常关键。对于甲的任一走法,乙却有着数十种数百种走法与之对应。逐一搜索估算,显然不合理。观察棋局,可以发现:首先,单点估价值高的点,其对应棋局往往是比较优的解,需要优先考虑。其次,新下棋子对于整个棋盘的影响,与其距离成正相关。距离远,影响小;距离越近,影响越大。因此,基于这个局部性原理,还可以选择相近位置对应的棋局作为候选棋局,重点估算。这样,控制每层扩展分支数量,只选取"有潜力"的棋局作为候选棋局,显著缩小了搜索范围,大大提高了横向的搜索效率。

2.4.3 α-β 剪枝

在搜索阶段,很多时候,遍历博弈树所有节点是没有必要的。以图6为例,在A棋局下轮到甲走棋,甲首先搜索叶子节点$E=7$,$F=11$,计算得到B棋局的倒推值为7;然后,当甲继续搜索到节点$G=1$时,发现$C(C\leqslant 1)$必定小于B。也就是说,走第二个子节点C棋局肯定不如走第一个子节点B棋局有利,那么子节点C棋局往下的其它节点棋局(H、I)就不用再搜索了,此过程就称为剪枝。如果甲所在的层是选最大值的层(如当前的A节点),则称此剪枝为α剪枝,否则成为β剪枝。实践表明,在智能搜索过程中,采用α-β剪枝可以有效地提高效率。局势不明朗时,机器人"思考"的时间会稍长(剪掉的枝杈比较少);但当机器人处于优势时,下棋速度很快。优势越明显,下棋速度越快。这一点与人的思考过程也是比较吻合的。

表1　五子棋系统的测试结果

采用算法及优化手段	博弈树每层分支数	博弈树层次	下棋速度	智力水平
基本的单点估价算法	几十或几百	1层	很慢	一般
基本的博弈树算法	几十或几百	至多3层	极慢	略微提高
加入优化1,2	7	增至5层	很快	约3成胜算
加入优化3	7	增至8层	快	约5成胜算
加入优化4	7或3	增至11层	正常	8成多胜算

注:"智力水平"指与普通棋手下棋时,机器人取胜的可能性。

2.4.4 低层剪枝

人在思考下棋的时候有这样的习惯:考虑当前一两步时,会比较仔细的推敲;考虑到三步四步或者更多步时,因为可能形成的棋局太多,一般会忽略他认为无关紧要的棋局,而只关注他所关心的几个典型棋局。借鉴这种"集中精力法",提出低层剪枝的搜索策略。假定博弈树总共10层,机器人在搜索时,会自动对最低的3层进行适当剪枝,每层只选定最重要的3个节点作为候选棋局。这样,在保证机器人反应速度的同时,可以加深博弈树层次,提高机器人判断能力。采取以上优化手段后,形成表1的测试结果。

3 总结

在推导五子棋博弈算法过程中,对知识抽象、知识表示、估价函数、博弈树及搜索策略进行了一

(下转第1060页)

191—204
8 王 磊,陈曙晖,苏金树,等.深度报文检测中基于GPU的正则表达式匹配引擎;计算机应用研究,2010;27(11):4324—4327
9 NVIDIA. CUDA Programming Guide. http://developer.download.nvidia.com/compute/DevZone/docs/html/C/doc/CUDA_C_Programming_Guide.pdf
10 Thompson K. Programming techniques: Regular expression search algorithm. Communications of the ACM,1968;11(6):419—422

A Signature Matching Mechanism Based on the Cooperation of CPU and GPU

YANG Ke, GU Jian-hua, ZHANG Chun-yong
(School of Compute Science and Engineering, Northwestern Polytechnical University, Xi'an 710072, P. R. China)

[**Abstract**] In network security system, a signature matching mechanism based on the cooperation of CPU and GPU is proposed. Firstly, a small DFA(deterministic finite automaton) is constructed by using the first n prefix of each rule expression. Secondly, the non-exact matching is done on the GPU to pick out the dubious packets. Lastly, the exact matching is done by CPU. Experimental results show that the matching throughput reaches 19 Gbits/s.

[**Key words**] regular expression signature matching GPU DFA

(上接第1055页)

些研究,介绍了一个实现人机对弈的智能五子棋系统。该系统采用改进的博弈树算法,依据单点估价函数和全局估价函数,辅助一定的启发信息,进行智能选择。同时,结合局部性原理,增量分析法、α-β剪枝及低层剪枝,控制博弈树各层的搜索范围,显著地提高了分析与搜索的效率。

设计的五子棋系统可以做出快速而明智的选择,考虑整个棋局的形势,预判未来的发展趋势,因而,具备一定的"全局意识"和"发展观"。除了本文采用的方法外,还可以通过增加机器学习的功能来提高系统的判断能力[5]。让机器人对棋局进行记忆、总结和学习,可以进一步提高系统的智能化程度。

参 考 文 献

1 李勤丰. 智能五子棋中的算法研究. 广西轻工业,2007;(11):69—70
2 朱全民,陈松乔. 五子棋算法的研究与思考. 计算技术与自动化,2006;25(2):72—74
3 Russell S, Norvig P. 人工智能———一种现代方法(第二版). 姜 哲,金奕江,张 敏,等,译. 北京:人民邮电出版社,2010
4 蔡自兴 徐光佑. 人工智能及其应用. 北京:清华大学出版社,1999
5 蒋加伏,陈蔼祥,唐贤英. 基于知识推理的博弈树搜索算法. 计算机工程与应用,2004;(1):74—76

An Improved Gobang System Based on Game-Playing Tree

YANG Yun-qiang, WU Jiao
(Aeronautical Computing Technique Research Institute, Xi'an 710065, P. R. China)

[**Abstract**] By analyzing knowledge abstraction, knowledge representation, evaluation function, game-playing tree and search strategy in the field of artificial intelligence, an improved gobang system based on game-playing tree is designed. Using multi-list structure to record the chess game information, the gobang system can fully describe and analyze chess game situation. Finally, with the locality principle, the incremental analysis, α-β pruning and lower level pruning are used to improve the system's response speed and intelligence.

[**Key words**] gobang algorithm artificial intelligence game-playing tree evaluation function knowledge representation