分 类 号：TP301　　密　级：公开

论文编号：　　学　号：30708120306

重庆理工大学硕士学位论文

六子棋计算机博弈系统的研究与实现

研　究　生：　李 翠 珠

指 导 教 师：　韩 逢 庆　教授

学 科 专 业：　计 算 机 应 用 技 术

研 究 方 向：　计 算 智 能 与 智 能 软 件

培 养 单 位：　计 算 机 科 学 与 工 程 学 院

论文完成时间：　2010 年 4 月 20 日

论文答辩日期：　2010 年 6 月 1 日

万方数据

Category Number：TP301 **Level of Secrecy：Public**

Serial Number ： **Student Number：30708120306**

Master's Dissertation of Chongqing

University of Technology

Research and Implementation of the System of Connect6 Game

Postgraduate: **Li Cuizhu**

Supervisor： **Prof. Han Fengqing**

Specialty: **Computer Applied Technology**

Research Direction: **Computational Intelligence and Intelligent Software**

Training Unit: **The Computer Science and Engineering School**

Thesis Deadline: **On April 20, 2010**

Oral Defense Date: **On June 1, 2010**

万方数据

重庆理工大学
学位论文原创性声明

本人郑重声明：所呈交的学位论文是本人在导师的指导下，独立进行研究所取得的成果。除文中特别加以标注引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写的成果、作品。对本文的研究做出重要贡献的集体和个人，均已在文中以明确方式标明。

本人承担本声明的法律后果。

作者签名：　　日期：　　年　月　日

学位论文使用授权声明

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权重庆理工大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于（请在以下相应方框内打“√”）：

1.保密□，在___年解密后适用本授权书。

2.不保密□。

作者签名：　　日期：　　年　月　日

导师签名：　　日期：　　年　月　日

万方数据

摘 要

计算机博弈是人工智能的重要研究内容，人工智能的发展在很大程度上得益于博弈研究的发展。棋类游戏作为博弈研究的主要内容之一，在国际象棋、中国象棋、五子棋等方面都有了较好的解决方法。六子棋是台湾吴毅成教授发明的，它改良自五子棋，同时兼具规则简单、变化复杂、游戏公平的特性。

六子棋发明后，引起了研究人员广泛的关注，六子棋是检验人工智能发展水平的良好环境，如何提高程序的棋力是人工智能领域的一大难题。同时，开发出与人类棋手水平相当的六子棋程序，有助于对人类认知能力进行理解。所以，六子棋计算机博弈研究具有重要的理论意义和实用价值。

本文对计算机博弈的研究现状、六子棋的研究现状进行了简单介绍，对评估函数的分析及其参数的改进方面进行了详细介绍。在评估函数的改进方面主要涉及的方法有空位估值方法，棋形估值方法，空位估值方法是对当前棋面未落子位置进行评估的方法，棋形估值方法是利用棋形向量进行评估的方法。在评估函数的参数改进方面主要应用的是粒子群优化算法，采用该方法对系统中的棋型参数进行改进。最后，介绍了六子棋计算机博弈系统，Connect6。

具体说来，本文主要完成的工作与创新点包括以下几个方面：

一、选用二次估值方法进行棋局评估，该方法引入了局势因子，随着局势因子的变化，可以体现出对棋局局势的不同需求，指导落子。

二、模拟人类思维模式提出空位估值方法。人类棋手对弈过程中，思考未落子位置对当前局势可能产生的影响。在抑制对方局势的前提下，如果某空位对我方有利，棋手会选择最为有利的位置落子。本文提出的空位估值方法实现了这一思想，通过多位棋类高手的经验，设定了五种空位置的估值。实验过程中，系统使用该方法得到了比较理想的效果。

三、采用粒子群优化算法对五种空位的估值进行参数优化，将优化后的估值应用与系统中，系统性能得到较大提高。

四、一般使用的估值函数都是针对孤立棋子进行评估，棋子与棋子之间由于位置关系也会产生不同的影响，所以本文提出新的估值方法——棋形估值方法。该方法利用棋子与棋子之间的位置关系，即棋型，进行估值。提出棋形向量的概念，即在已选棋型基础上，对当前棋面中的棋型进行量化得到的向量。将棋形向量与空位估值方法相结合，运用到系统中。

关键词：六子棋，二次估值，空位估值方法，棋形向量，粒子群优化算法

万方数据

ABSTRACT

Computer game is an important research topic in Artificial Intelligence, it plays a significant role in the development of Artificial Intelligence. Chess Computer Game, chess, Chinese chess, backgammon have better solutions. Connect6 invented by Professor Wu in Taiwan and improved from the backgammon, are simple game. There are two plays, we called Black and White. Black plays first and puts only one black stone on one unoccupied intersection. Subsequently, Black and White alternately put two of their own stones on two unoccupied grids. Complexity has been assessed as second only to Go and Shogi, is much higher than backgammon and chess, and chess comparable or slightly higher. Fairness, because the parties after each hand is finished, one more disk than the other , therefore, the game can reach a balanced state of nature, which makes equity greatly improve.

Connect6 appears, caused widespread concern of researchers, is a good test environment for the development of Artificial Intelligence, how to improve the process of thinking depth is a major problem in Artificial Intelligence. At the same time, developed a considerable level of human players in Connect6 will also help the understanding of human cognitive abilities. Therefore, a computer chess game six sub-study has important theoretical and practical value.

This study has a brief introduction to current research on the status of the computer game and Connect6, describes the improvement of evaluation function and parameters in detail. The main method of improvements in evaluation function is space valuation method and chess-shaped method of valuation. Space valuation method is assessed using the current vacancy, chess-shaped method of valuation is assessed using chess shaped vector method. In assessing the function of the parameters of the major improvements is the application of particle swarm optimization algorithm, using the method of the system to improve chess-type parameters, and with the improvement of the system before the game.

Specifically, we have completed the major work and innovation, including the following:

First, use the revaluation method in the system, which introduces the situation factor, as the situation factor changes, can reflect the different needs of the situation on the chessboard.

Second, use particle swarm optimization algorithm to optimize parameter.

Third, simulation of human thinking proposed method of space valuation. Human

万方数据

chess players in games, will think every empty position on the possible impact of the current situation. Inhibition of the other premise situation, the player will choose the most advantageous position expanded store-opening. In this paper, the method of valuation is to achieve this idea, through the experience of a number of chess masters, set the valuation of five vacancies, the method comparison experiment has been the ideal result.

Fourth, the valuation functions are generally used on isolated pieces of assessment, but the pieces and pieces in different locations will be different impacts, the proposed method of valuation chess form. The method used the chess-type and the positional relationship between the pieces. Chess shape and space vector combination of valuation methods, applied to the system.

Keywords: Connect6, revaluation, space valuation method, Chess Shape Vector, Particle Swarm Optimization

万方数据

目 录

万方数据

万方数据

万方数据

万方数据

1 绪论

1.1 计算机博弈的研究意义

人工智能是在多学科相互渗透的基础上，发展起来的综合性很强的一门新兴边缘学科。它的中心任务是研究如何用机器来模拟和实现人类的智能行为。经过许多专家、学者的不懈努力，人工智能的应用在不少领域得到发展，在我们的日常生活和学习当中也有很多地方得到应用，例如专家系统，机器翻译等。还有结合人体自身生理特征研发出的指纹识别、虹膜识别等唯一性身份识别系统，给人们的生活带来方便的同时，解决了人类有效身份识别的难题。

目前，各国都把人工智能作为重点列入本国的高科技发展计划。人们常常把计算机博弈描述为人工智能的“果蝇”，即人类在计算机博弈的研究中衍生了大量的研究成果，这些成果对更广泛的领域（政治、经济、军事和生物竞争领域）产生了重要影响，也广泛应用于军事指挥和经济决策中。

果蝇，由于它喂养容易，生长周期短、繁殖快、性状明显，被遗传学家视为最理想的遗传学研究载体。摩尔根和他的学生们[1]发现了伴性遗传、基因重组等现象，并且通过多次实验确定了基因位于染色体上，根据果蝇的性状特征，他们绘制出第 1 张染色体图，从而使得经典遗传学进入顶峰时期。计算机博弈之所以被称为人工智能的“果蝇”，主要因为它的简单又变化无穷的逻辑思维模式，并且进行机器对弈的时间周期短，每完成一次比赛，就可以对计算机的“智能”进行一次测试，可以根据比赛过程中的实际情况，进行系统的调整，使得系统更健壮。而且在系统构建时，对其设定了悔棋、复盘等操作，利用这样的操作更能发现电脑与人脑的功能差距，从而不断地积累经验提高电脑的智力水平。在机器博弈的基础上，我们可以直接开展如下的人工智能领域的研究工作[2]：

① 人人对弈时会有残局产生，如何将大量的残局棋谱，经过提炼和抽象融入到博弈程序当中，使程序能利用这些棋谱，这属于知识工程的典型问题；

② 不同的棋手会有不同的下棋风格，所谓知己知彼，百战不殆。如何通过某位棋类高手的大量棋谱，系统自动归纳出他的棋风、走步特点及其优缺点。这属于知识挖掘的典型问题；

③ 人类的对弈过程实际上是个自学习的过程，可以发现自己的缺点加以改正，同时还能吸收对手的优点。如何剖析一盘棋的胜败原因，进而自动修改博弈程序，这属于机器学习的典型问题；

万方数据

④ 六子棋中含有大量的定石，围棋中有各种定式，如何将这些进行表示和分辨，这属于模式识别的典型问题；

⑤ 经过对弈后，在系统的修改过程中，会涉及到优化计算和进化过程，这就应用到了各种智能算法；

提到人工智能的研究载体，人们首先想到的是机器人。基于当今的科学技术，构造一个机器人，让它完成一次试验，是相当复杂的事情，需要消耗大量的时间和金钱。但是在机器智能领域，如果采用机器博弈作为研究载体的话，只要建立了性能优良的博弈系统，作为实验平台，就能成为人工智能研究的“果蝇”。

人工智能中大多以下棋（如国际象棋、中国象棋、围棋、五子棋、西洋跳棋等）为例研究博弈规律，博弈成为了一个典型的问题。早在上世纪五十年代，就有人设想利用机器智能来实现机器与人的对弈。国外许多知名学者和知名研究机构都曾经涉足这方面的研究，历经半个多世纪，到目前为止已经取得了许多惊人的成就。

在计算机日益普及和大众化的现代社会，高水平的博弈系统很容易获得可观的商业价值。目前，世界领先的计算机国际象棋程序基本上都是商业产品。事实上，个人计算机软件市场的大约 80%销售份额是来自游戏软件，其中有传统的博弈游戏，而非博弈游戏中也不可缺少人工智能与博弈的成分。因此，对游戏开发过程中的人工智能技术的研究自然也就成了业界的一个热门研究方向。

1.2 计算机博弈的关键技术

机器博弈被认为是人工智能领域最具挑战性的研究方向之一。国际象棋的计算机博弈已经有了很长的历史，并且经历了一场波澜壮阔的“搏杀”，“深蓝”计算机的胜利也给人类留下了难以忘怀的记忆。在国际象棋成熟技术的基础上，结合中国象棋和五子棋计算机博弈已有的研究成果，总结出完全知识博弈计算机系统的关键技术为：棋局表示、着法生成、棋局评估、博弈树搜索。

1.3 计算机博弈的研究方法

充分阅读计算机博弈相关文献、六子棋类书籍和棋评，体会棋手的思维过程，并与其它计算机六子棋研究者密切交流。在掌握经典算法的基础上努力创新，设计有效且实用的搜索算法、优化算法和学习算法，并采用易于维护的面向对象语言予以实现，同时设计实验，测试算法有效性，并比较各算法的时间和空间性能。

万方数据

1.4 六子棋计算机博弈

六子棋[3]英文名称为 Connect6，是由两位棋手，一方执黑子，一方执白子在 19 路棋盘上进行著手的竞技。著手是指轮到下棋的棋手在棋盘的空点上落子，或者放弃在棋盘上落子，这种称为虚手（PASS）。它的棋路开阔，对攻性强，可变性大，且又免去了五子棋繁杂的禁手，因此极受大众欢迎。

六子棋中涉及到的术语有线、连线、连六、长连。线是指由同一色棋子在直向、横向或斜向所形成的一种组合，其中各方向由盘端、另一色棋子、与空点所限制，且组合中间没有对手棋子。连线是指线中没有空点。连六是指由六个棋子所成的连线。长连是指由七个以上的棋子所成的连线。

六子棋游戏规则：

① 因为六子棋游戏具有公平性，棋盘可以无限大。在实际对弈中采用19路棋盘。

② 空秤开局。除了第一次黑方下一颗子外，之后黑白双方轮流每次各下两子，（即黑 1，白 2、白 3，黑 4、黑 5，白 6、白 7……如此类推）棋盘中先完成于横、竖、斜方向上连六或连六以上者获胜。

③ 当所有棋盘交点全部下满而无人连六以上，或对局双方皆同意和局即为和局。

六子棋它改良自五子棋，同时兼具规则简单、变化复杂、游戏公平的特性。规则简单，除了黑的第一手下一子外，黑白双方轮流各下两子，最后，连成六子者胜；变化复杂，由于一次下两子，组合非常多，可以说是千变万化。复杂度已被评估为仅次于围棋及日本将棋，远高于五子棋及西洋棋，与象棋相当或略高；游戏公平，由于各方每次下完一手后，盘面都比对方多一子，因此，赛局可自然达成平衡的状态，这使得公平性大为提升。不若许多棋种如五子棋、象棋、西洋棋，先手具有一些优势。

2005 年 9 月，六子棋发明者吴毅成教授于第十一届国际电脑竞赛发展研讨会（Advances in Computer Games）上正式发表全世界第一篇六子棋的国际论文后，获得相当多的回应。六子棋已经成为奥林匹亚电脑竞赛项目之一，中国机器博弈锦标赛项目之一，已登入国际线上维基百科全书，并有许多诘棋和定石提出。随着专家学者对六子棋的深入研究，六子棋规则也在不断调整。

台湾六子棋协会举办的六子棋公开赛的规则也在不断调整，比赛中设立了段位赛，即六子棋选手的升段比赛。根据比赛过程中积累的经验添加了 PASS 著手，即在段位组比赛中，可选择两子皆不落，当全部下满且无人连六以上时，以先下出 PASS 著手者获胜。

万方数据

1.5 论文结构

本章为绪论。

第二章介绍了计算机博弈的研究现状。本章中，首先对当前机器博弈的关键技术进行了介绍，然后重点介绍了六子棋计算机博弈系统开发中涉及到的几个概念，明确指出了研究中解决的实际问题。

第三章阐述了对评估函数的分析，分析了静态估值函数的不足，基于静态估值函数不能满足局势跌宕起伏的变化这一缺点，提出了采用二次估值及时对评估函数进行修改的方法。在研究过程中，提出了一种新的模拟人类思维模式的空位估值方法，在本章对该方法进行了详细的介绍。

第四章介绍了评估函数参数优化。有些学者将遗传算法引入到评估函数中，对自定义棋型对棋局产生的影响评估值进行优化。后面介绍了粒子群优化算法的原理，算法步骤，并利用该算法对空位估值方法中的参数进行优化。

第五章阐明棋形估值方法。该方法是通过对棋局进行棋型向量化指导走步的方法。主要介绍了棋形向量的概念，棋形估值方法的应用，以及该方法的优点和不足。

第六章介绍了六子棋计算机博弈系统，Connect6。对系统的各个模块进行了详细的介绍。

第七章为总结与展望。总结了本课题的主要工作和创新点。对该系统进行了客观的评价，指出其中的不足，并进一步制订了研究计划。

万方数据

2 研究现状

2.1 计算机博弈的研究现状

人类棋手进行对弈时，首先要明确比赛规则是什么，然后需要观察棋子位置，局势变化等信息，综合评定后，依据个人实战经验进行走步。如果让计算机进行比赛，那么在遵守棋类游戏规则的前提下，计算机首先必须能够识别棋盘，知道落子位置，了解当前局势。然后根据对当前棋局的评估进行落子位置选择，并且进行走步。为了能让计算机完成这一系列的“动作”，许多专家学者对其进行了深入的研究，总结出计算机博弈的关键技术为棋局表示，着法生成，棋局评估，搜索算法。将中国象棋和五子棋作为实例，分别对各关键技术进行介绍。

2.1.1 棋局表示

要让计算机进行对弈，需要解决的首要问题是棋盘和棋局的计算机状态表示。在中国象棋计算机博弈系统[4]中，由于黑红双方各有 16 颗棋子，7 个兵种，需要分别对棋盘状态、兵种、棋子信息进行计算机表示。在中国象棋的对弈过程中，有时只需要了解棋子的分布，不用知道棋子类别，这时选用比特棋盘。

中国象棋棋盘有 9 路 10 行，共有 90 个交叉点，棋盘可以用 10×9 的数偶矩阵 $M_{10\times9}$ 来表示，该矩阵可以描述棋盘交叉点与坐标的对应关系。M^B 矩阵就是对棋盘的数偶表示。

$$M^B=\begin{bmatrix}1,1 & 1,2 & 1,3 & 1,4 & 1,5 & 1,6 & 1,7 & 1,8 & 1,9\\ 2,1 & 2,2 & 2,3 & 2,4 & 2,5 & 2,6 & 2,7 & 2,8 & 2,9\\ 3,1 & 3,2 & 3,3 & 3,4 & 3,5 & 3,6 & 3,7 & 3,8 & 3,9\\ 4,1 & 4,2 & 4,3 & 4,4 & 4,5 & 4,6 & 4,7 & 4,8 & 4,9\\ 5,1 & 5,2 & 5,3 & 5,4 & 5,5 & 5,6 & 5,7 & 5,8 & 5,9\\ 6,1 & 6,2 & 6,3 & 6,4 & 6,5 & 6,6 & 6,7 & 6,8 & 6,9\\ 7,1 & 7,2 & 7,3 & 7,4 & 7,5 & 7,6 & 7,7 & 7,8 & 7,9\\ 8,1 & 8,2 & 8,3 & 8,4 & 8,5 & 8,6 & 8,7 & 8,8 & 8,9\\ 9,1 & 9,2 & 9,3 & 9,4 & 9,5 & 9,6 & 9,7 & 9,8 & 9,9\\ 10,1 & 10,2 & 10,3 & 10,4 & 10,5 & 10,6 & 10,7 & 10,8 & 10,9\end{bmatrix}$$

要表示棋局状态首先要给双方的 7 个兵种进行编码。如表 2.1 是象棋兵种的中英文对照和棋天大圣、象棋明星的兵种编码。

万方数据

表 2.1 象棋兵种的中英文对照和棋天大圣、象棋明星的兵种编码

国际象棋	King	Rook	Knight	Cannon	Queen	Bishop	Pawn	
中国象棋	King	Rook	Horse	Cannon	Guard	Elephant	Pawn	
红子	帅	车	马	炮	仕	相	兵	Null
字母代号	k	r	h	c	b	e	p	
棋天大圣兵种编码	1	2	3	4	5	6	7	0
象棋明星兵种编码	02	04	08	06	0c	0a	0e	
黑子	将	车(砗)	马(码)	炮(砲)	士	象	卒	
字母代号	K	R	H	C	B	E	P	
棋天大圣兵种编码	−1	−2	−3	−4	−5	−6	−7	
象棋明星兵种编码	12	14	18	16	1c	1a	1e	

对兵种编码后，我们还需要对棋子进行编码。这种编码方法是任意的，满足棋子在计算机内的表示唯一、可区分即可。如表 2.2 是棋天大圣的棋子编码。

表 2.2 棋子编码（号）表

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
黑方	将	黑车		黑马		黑炮		黑士		黑象		黑卒				
j	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
红方	帅	红车		红马		红炮		红仕		红相		红兵				

对棋子，兵种分别进行编码后，可以得到棋盘状态矩阵。

比特棋盘是一个 N 位长度的变量，用来记录棋盘上某些位置的布尔值，实际上是棋盘状态矩阵的布尔表示。比特棋盘的定义为：

$$B=[b_{i,j}]_{10\times 9}, b_{i,j}=1,\ s_{i,j}\neq 0;\quad b_{i,j}=0, s_{i,j}=0 \qquad \text{式(2-1)}$$

其中 $s_{i,j}$ 表示当前棋盘状态矩阵中某一位置的数值，$s_{i,j}\neq 0$ 表示对应位置有棋子，这时 $b_{i,j}$ 取 1；$s_{i,j}=0$ 表示对应位置为空，即无子，$b_{i,j}$ 为 0。比特棋盘矩阵常用于着法生成过程中。

五子棋[5]起源于古代中国，在近代得到了快速发展和完善。五子棋采用 15 路线的棋盘，黑白双方轮流落子，当某方在横向、竖向或斜向上连成五子时，该方为获胜方。五子棋能让计算机识别，同样需要对棋盘，棋子进行表示。五子棋的棋盘可以用一个 15×15 的二维数组表示；因为没有兵种的区别，它的棋子表示相对简单。一般情况下，用“0”表示黑子，用“1”表示白子，用“-1”表示空位置。

万方数据

与中国象棋不同的一点是五子棋开局时棋盘为空，对弈过程中进行落子，所以五子棋的落子顺序也需要表示出来。五子棋中采用栈结构存储棋子及其坐标，依照落子顺序依次入栈，通过访问栈，可以得到落子顺序。

2.1.2 着法生成

着法生成是指根据当前局势产生所有有效的着法。着法是博弈树展开的依据。经常采用的着法生成方法包括：棋盘扫描法、模板匹配法、预置表法，有时还结合使用。

① 棋盘扫描法

棋盘扫描方法需根据不同棋类游戏的规则，定义可行区域才能使用。该着法的生成过程是反复在棋盘上扫描有效区域、制约条件和落子状况，确定有效的落子位置。不同的棋类游戏有不同的规则，比如五子棋，棋盘有效区域内的所有空白的交叉点都是可行落子点；这样在五子棋的走法产生模块[5]里，只要扫描棋盘，寻找到所有的空白，就可以罗列出所有符合规则的下一步。中国象棋根据游戏规则的不同，对落子位置有较多的限制，不适合用该方法。

在着法的表达上，棋盘扫描法最为直观，在五子棋、围棋等棋类设计中经常使用，但时间开销巨大。

② 模板匹配法

目前，该方法主要在中国象棋机器博弈系统中使用，比较适合使用模板匹配方法的棋子是马、相（象）。

以中国象棋为例，当动子（某一回合中计算机方要移动的棋子）确定之后，其落址（根据中国象棋中兵种的走步规则，找到的可能的落子位置）与提址（该动子所在的棋盘位置）的相对关系便被固定下来。所以可以为某些动子设计“走子匹配模板”，只要与提址匹配，就能快速找到落址。如图 2.1 中国象棋的走马匹配模板。当马位于中心提址，〇代表符合“马走日”这一规则的落址，×代表蹩马腿的制约条件，根据×的具体分布，很容易判断可能的落址。再通过单项比特矩阵比对，实现“本方子则止，对方子则吃”，完成“提-动-落-吃”内容的确定。

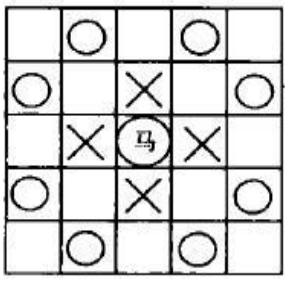

图 2.1 中国象棋的走马匹配模板

万方数据

③ 预置表法

预置表法是最为常用的着法生成方法，该方法的本质是用空间换时间。针对当前棋局，给出动子可能的吃子走法和非吃子走法，以空间换取对弈过程中的生成着法的扫描时间。在中国象棋计算机博弈中，对于炮、车等可以利用预置表法进行定义。如下图 2.2 所示是炮的一个行着法预置表。

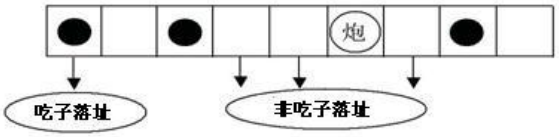

图 2.2 炮的一个行着法预置表

炮的提址为该行的第 6 列，采用布尔向量表示该行棋子的分布为[101000010]。根据炮的着法规则，此时炮的非吃子着法的落址用布尔向量表示为[000110100]，而可能的吃子着法的落址为[100000000]。将炮的列号、该行棋子的布尔分布作为输入，吃子落址、非吃子落址的布尔表示关系作为输出，写入预置表中。当轮到该动子走步时，通过查询预置表，可以很快得到可行的着法。同时还能确定是吃子着法或非吃子着法，这样有利于搜索路径的选择。

2.1.3 评估函数

前面介绍的棋局表示和着法生成主要是使计算机能够认识棋盘，识别棋子信息。现在要介绍的评估函数则是用来进行落子指导。人类对弈时，根据自我的比赛经验和习惯来选取落子位置。那么，计算机也需要根据游戏规则和以往的经验来进行落子，它的经验主要来自于评估函数。评估函数一般来说必须包括 5 个方面的要素，分别是固定子力值、棋子位置值、棋子灵活度值、威胁与保护值、动态调整值，每一方面的值又是由许多参数值构成的。对于中国象棋来说，它的评估函数就有 20 多个参数，将这些值线性地组合在一起得到最终的评估值[6]。

下面介绍几种方法可以获取评估函数中的数值：

① 爬山法(Hill-Climbing)

爬山算法[7]是对深度优先搜索算法的一种改进，利用反馈信息得到解，是一种局部择优方法。该算法的思想是从当前节点开始，和周围的邻居节点的值进行比较。如果当前节点是最大的，那么返回当前节点，作为最大值（既山峰最高点）；反之就用最高的邻居节点来替换当前节点，从而实现向山峰的高处攀爬的目的。如此循环直到达到最高点。

爬山算法的优点是避免遍历，通过启发选择部分节点，可以提高算法效率；缺点是该算法是局部择优方法，得到的结果可能不是最优。

万方数据

一般存在以下问题：第一，局部最大，某个节点比周围任何一个邻居都高，但它却不是问题的最优解；第二，高地，也称为平顶，搜索一旦到达高地，就无法确定搜索最佳方向，会产生随机走动，使得搜索效率降低；第三，山脊，搜索可能会在山脊的两面来回震荡，前进步伐很小。

② 模拟退火法(Simulated Annealing)

模拟退火算法来源于固体退火原理，将固体加温到充分高，再让其徐徐冷却。加温时，固体内部粒子随温度升高变为无序状态，内能增大，而慢慢冷却时粒子渐趋有序，在每个温度都达到平衡态，最后在常温时达到基态，内能减小为最小。根据Metropolis准则，粒子在温度T时趋于平衡的概率为$e^{-\Delta E/(kT)}$，其中E为温度T时的内能，ΔE为其改变量，k为玻尔兹曼常量（Boltzmann constant）。用固体退火模拟组合优化问题，将内能E模拟为目标函数值f，温度T演化成控制参数t，即得到解组合优化问题的模拟退火算法：由初始解i和控制参数初值t开始，对当前解重复产生新解、计算目标函数差、接受或舍弃的迭代，并逐步衰减t值，算法终止时的当前解即为所得近似最优解，这是基于蒙特卡罗迭代求解法的一种启发式随机搜索过程。

利用该方法对局面进行评价时，类似于爬山法，通过改变权重来寻找最优解。如果权重的改变没有影响当前解，可以试着采用某一概率随机地进行改变。模拟退火法比爬山法更慢，但是最终可能得到比较好的值。

③ 神经网络(Neural Networks)

思维学普遍认为，人类大脑的思维分为抽象（逻辑）思维、形象（直观）思维和灵感（顿悟）思维三种基本方式。神经网络就是模拟人类思维的第二种方式，形象思维。这是一个非线性动力学系统，其特色在于信息的分布式存储和并行协同处理。虽然单个神经元的结构极其简单，功能有限，但大量神经元构成的网络系统所能实现的行为却是极其丰富多彩的。神经网络方法实际上是评估函数的一种类型，并不是用来获取数值的。神经网络的优点在于它并不需要很多人类棋手的相关经验或智慧，就可以得到比较好的评估函数。但是，有些研究人员证实，利用神经网络不如人类棋手速度快，效果好。

④ 遗传算法(Genetic Algorithms)

利用爬山法和模拟退火法可以得到一组好的权重，它们是逐渐变化的。而遗传算法[8]可以得到几组不同的好的权重，通过选择，交叉，变异操作来控制种群的数量。

有些研究人员将遗传算法引入到评估函数中，遗传算法的评估值在很大程度上也依赖于研究人员的经验。另外，通过与高水平名局棋谱或其他博弈程序对弈，利用获胜概率来确定某一组参数，经过多次实验一般就可以得到较好的静态估值。

万方数据

2.1.4 搜索算法

搜索算法是利用计算机的高性能来有目的的穷举一个问题的部分或所有的可能情况，从而求出问题解的一种方法。搜索过程实际上是根据初始条件和扩展规则构造一棵解答树，并寻找符合目标状态节点的过程。所有棋类的计算机博弈都涉及到搜索算法，即根据当前的棋局状态以及规定的搜索深度和宽度，在博弈树中找到一条最佳路径。博弈树与一般的搜索树不同,它是由对弈双方共同产生的一种“变性”搜索树。

搜索算法是博弈树[9]求解的灵魂，只有利用有效的搜索算法才能在有限时间内从众多可供选择的解中找到正确解。在博弈问题中，每走一步可供选择的方案都有很多，因此会生成十分庞大的博弈树，如果试图通过直到终局的博弈树进行搜索是不可能的。所以，搜索目标便成为如何在有限深度的博弈树中找到评估值最高而又不剧烈波动的最佳状态。最常使用的搜索算法有极大极小搜索，$\alpha-\beta$剪枝搜索、$\alpha-\beta$窗口搜索、迭代深化搜索、启发式搜索、负极大值搜索等方法。

① 极大极小搜索

我们假设博弈的双方中一方为 MAX，另一方为 MIN。MAX 先走，之后两人交替走步直到游戏结束。由于不可能对完整解图进行搜索，所以我们定义一个静态评估函数 f，以便对游戏状态的当前势态进行估值[10]。一般规定有利于 MAX 的状态取 $f(p)>0$，有利于 MIN 的状态取 $f(p)<0$。极大极小搜索的基本思想[11]是：

1）生成整个博弈树，即扩展树的每个节点。

2）用静态估值函数 f 对每个叶节点进行估值，得出每个终节点的评价值。

3）用终节点的估值来得到其搜索树上一层节点的估值。

4）重复 c 过程在 MAX 层取其分支的最大值，MIN 层取其分支的最小值，一直回溯到根结点。

5）最终，根结点选择分支值最大的走步走[12]。

这就是极大极小搜索[13]过程(MINIMAX Decision)。

② $\alpha-\beta$剪枝搜索

在极小极大搜索方法[14]中，由于先要按照指定深度生成所有节点，博弈树展开的节点数将随着搜索深度的增加而呈现指数增长，这样，会导致算法效率比较低。能否寻找一种方法，使搜索深度不变，但是能减少节点生成数量呢？于是，利用已有搜索信息，在极大极小搜索算法的基础上提出了$\alpha-\beta$剪枝搜索方法。

$\alpha-\beta$剪枝搜索[15]：是一种利用已有搜索信息，去掉不影响最终结果的分支，返回与极大极小搜索相同走步的搜索算法，是一种基于$\alpha-\beta$剪枝[16]的深度优先搜索(Depth-first search)。

α剪枝：在对博弈树进行深度优先搜索时，从左路分枝的叶节点开始，依照极大极小算法，从下往上进行倒推，得到某一层 MAX 节点的值，该值表示的是到目前为

万方数据

止得以“落实”的着法的最佳值，记为α。显然此α值可以作为MAX方着法指标的下界[17]。在搜索此MAX节点的其它子节点时，如果发现一个回合，也就是说2步棋之后，评估值变差，即孙节点评估值低于下界α值，则可以剪掉此枝（以该子节点为根的子树），即不再考虑此“软着”的延伸。此类剪枝称为α剪枝。图2.3给出了α剪枝的搜索过程[18]，粗箭头所示的是选择的最佳路径。

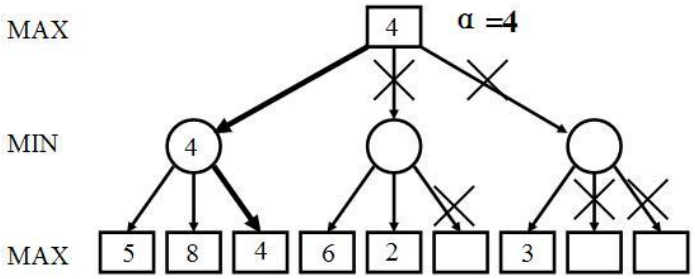

图2.3 α剪枝

β剪枝：同理，由左路分枝的叶节点开始，依照极大极小算法，从下往上进行倒推，得到某一层MIN节点的值，该值表示的是到目前为止对方着法的钳制值[18]，记为β。显然此β值可作为MAX方可能实现着法指标的上界。在搜索该MIN节点的其它子节点，如果发现一个回合，也就是说2步棋之后，钳制局面减弱，即孙节点评估值高于上界β值，则便可以剪掉此枝，即不再考虑此“软着”的延伸[19]。此类剪枝称为β剪枝。图2.4给出了β剪枝的搜索过程，粗箭头所示的是选择的最佳路径。

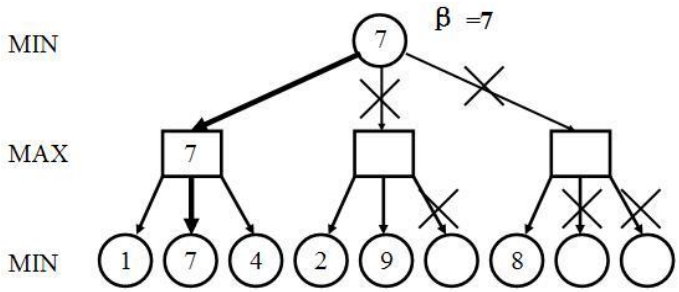

图 2.4 β剪枝

1975年Knuth和Moore给出了$\alpha-\beta$算法正确性的数学证明[20]。$\alpha-\beta$剪枝算法的效率与子节点展开的先后顺序相关。有研究人员将$\alpha-\beta$剪枝算法应用到多个回合之后，于是又出现了深层$\alpha-\beta$剪枝(Deep $\alpha-\beta$ cut-off) 算法[21]，该算法也取得了很好效果。

③ $\alpha-\beta$窗口搜索

从$\alpha-\beta$剪枝原理中得知，α值可作为MAX方可实现着法指标的下界，而β值便

万方数据

成为 MAX 方可实现着法指标的上界，于是由α和β可以形成一个 MAX 方候选着法的窗口。围绕如何能够快速得到一个尽可能小而又尽可能准确的窗口，出现了各种各样的$\alpha-\beta$窗口搜索算法[22]。如 Fail-Soft Alpha-Beta、Aspiration Search（渴望搜索）、Minimal Window Search（最小窗口搜索）、Principal Variable Search（PVS 搜索）/ Negescout 搜索、宽容搜寻(Tolerance Search)等[23]。

④ 启发式搜索

具体问题的领域决定了初始状态、算符和目标状态，进而决定了搜索空间。启发式搜索[28]就是在搜索空间中对每一个搜索位置进行评估，得到估值好的位置，再从这个位置出发进行搜索直到目标。这样可以省略繁杂的搜索路径，提到搜索效率。在启发式搜索中，对位置的估值是十分重要的。采用了不同的估值可以有不同的效果。我们先看看估值是如何表示的，启发式搜索中的估值是用评估函数表示的，如式（2-2）所示：

$$f(n)=g(n)+h(n) \qquad \text{式(2-2)}$$

其中$f(n)$是节点n的评估函数，$g(n)$是在状态空间中从初始节点到n节点的实际代价，$h(n)$是从n到目标节点最佳路径的估值代价。$h(n)$主要体现了搜索的启发信息，

⑤ 负极大值算法

前面我们介绍了博弈树的搜索是一种“变性”搜索。

假设计算机方的估值总和记为$Cvalue$，对方估值总和记为$Ovalue$，那么一个局面的估值$Value$就是计算机方和对手方的估值差。表达形式如下：

$$Value=Cvalue-Ovalue \qquad \text{式（2-3）}$$

假如计算机方落子后形成的局面是子节点，则估值是$Cvalue-Ovalue$；如果该节点有父节点，则父节点必然是对手方走棋的局面，它是取极大值节点，我们希望取$Ovalue-Cvalue$中的最大的子节点。如果对手方走棋所形成的局面是叶节点，则估值是$Ovalue-Cvalue$，如果该节点有父节点，则必然是计算机方走棋的局面，它是取极小值节点，我们希望取$Cvalue-Ovalue$中最大的子节点。在偶数层进行极大值搜索，而在奇数层进行极小值搜索[29]。这样的搜索策略给算法的实现带来了一些麻烦。

Knuth 和 Moore 于 1975 年提出了具有重大意义的负极大值算法，该算法充分考虑了“变性”搜索的内在规律[30]。它的思想是：父节点v的值是各子节点$(v_1,v_2,v_3,...,v_n)$值的负数的极大值，从而避免偶数层取极大而奇数层取极小的局面。

$F(v)=\max\{-F(v_1),-F(v_2),...,-F(v_n)\}$，其中$v_1,v_2,...,v_n$为$v$的子节点。

从以上的介绍可以看出，博弈树的搜索算法多种多样，改革与创新的余地很大，一定会成为计算机博弈研究的重点。

万方数据

2.2 六子棋计算机博弈的研究现状

2.2.1 定石

定石（Joseki）是指经过高段棋手审慎并完整的研究，所分析出较为固定的开局方式，若不依照这些下法，通常会导致不利的局面，英文通常称之为 Joseki（来自日语）。由于六子棋的定石，通常发生在开局，也叫做 Opening。

目前，六子棋爱好者提出了多种开局命名规则，包括开局命名法，山水开局命名法，DIF 开局命名法。现在通常采用 DIF 开局命名法，即利用坐标进行命名。统一坐标为：横坐标用 A-S 表示，纵坐标由下到上用 1-19 表示。黑方第一手棋子下于 J10，即围棋中所谓的天元。白方第一手两子需要重新定位，将依照以下规则做旋转对称。

① 决定第一子。在所有八种旋转对称的棋型中，先寻找所有南南西方（约棋盘 1/8 区域）出现的棋型。在这 1/8 区域的棋型中，找寻距离天元最近者（先看纵轴，再看横轴）为第一子，另一子为第二子。

② 去除重复对称盘面。以第二子在右方或下方为先。当在上项中找到第一子后，若仍有对称致使无法区分时，则看第二子，以在右方或下方为先。

③ 重定位后，进行白第一手的命名。坐标命名：直接以第一手坐标及第二子坐标命名。

可以按照下面的规则将命名简易化。第一子依据其距离天元情形，改用更简单的命名方式如下:第一子依照其坐标，如 J9，称之为 J9-珠型。简称的规则为：T，代表往下一格；X，代表往左下一格。

DIF 开局命名法，以原五子棋常用之 D/I(Direct/Indirect)为主，这里我们采用 T/X 是因为 T 与 X 可避免与坐标 A-S 冲突，且可有 TT, TX, XX 等变化。

下面是 26 种较常用的定石：

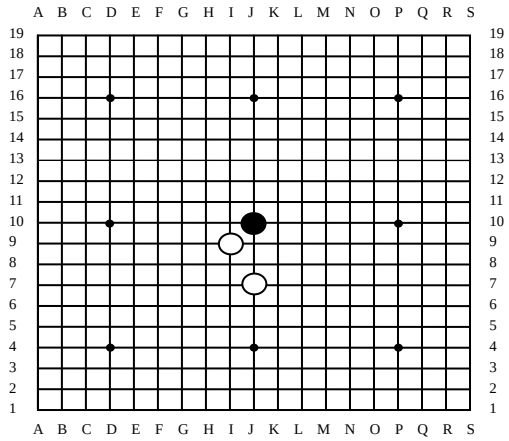

图 2.5 X-J8 定石

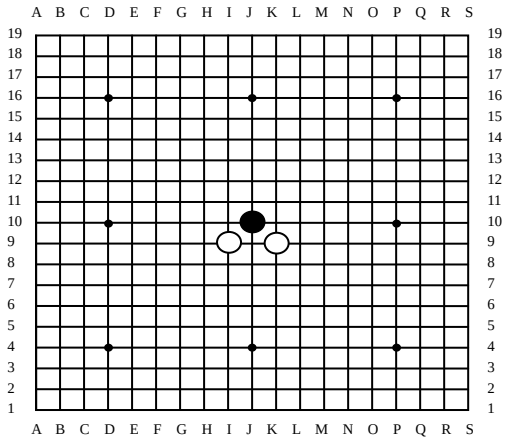

图 2.6 X-K9 定石

万方数据

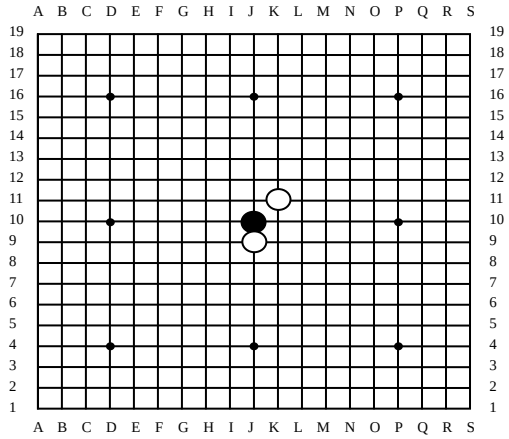

图 2.7 T-K11 定石

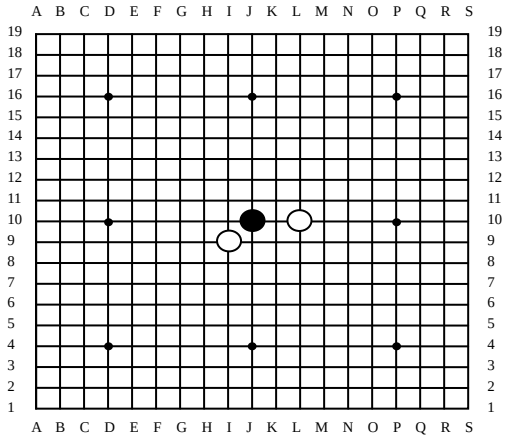

图 2.8 X-L10 定石

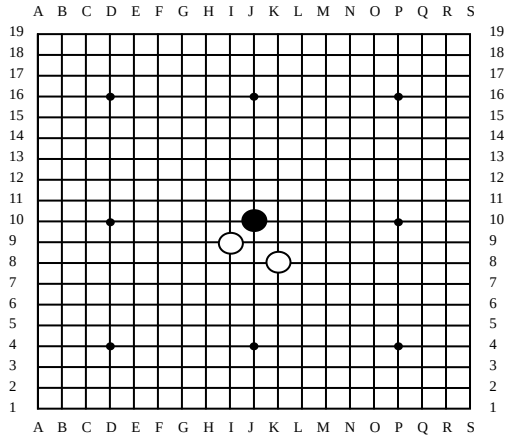

图 2.9 X-K8 定石

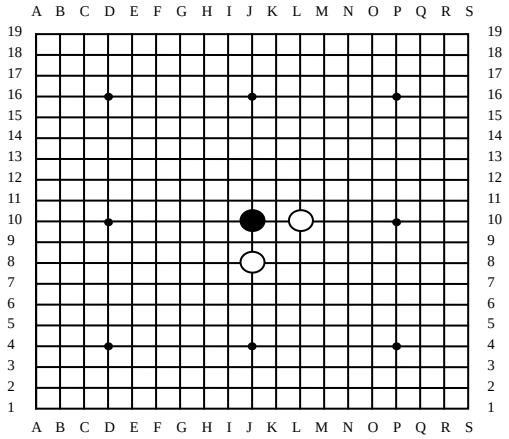

图 2.10 TT-L10 定石

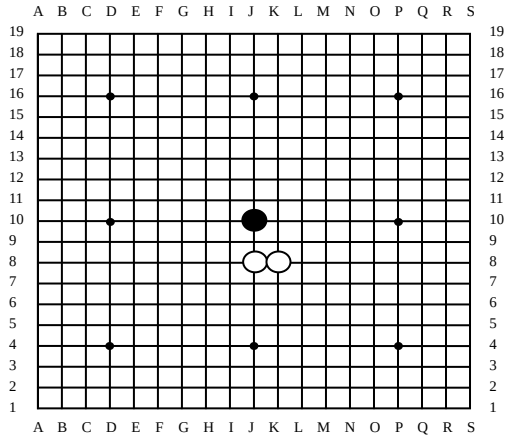

图 2.11 TT-K8 定石

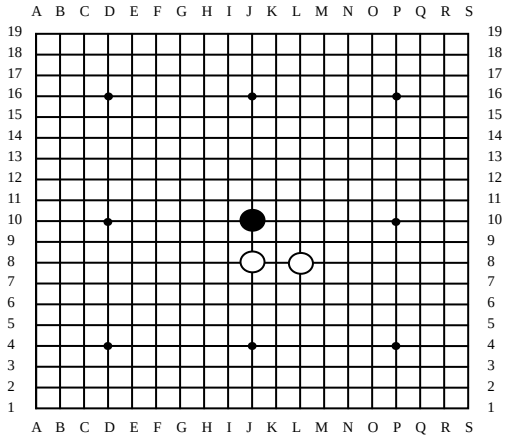

图 2.12 TT-L8 定石

万方数据

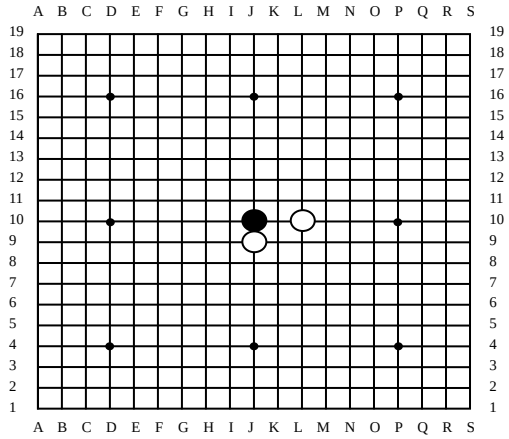

图 2.13 T-J12 定石

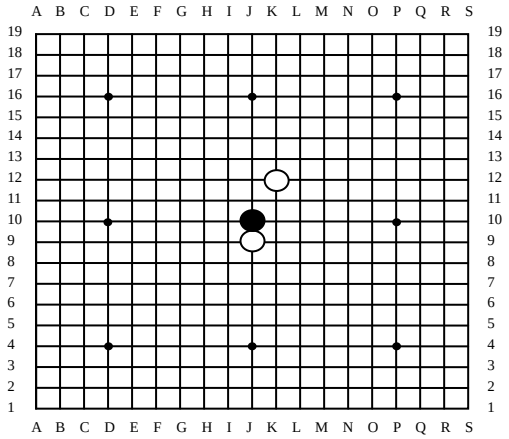

图 2.14 T-K12 定石

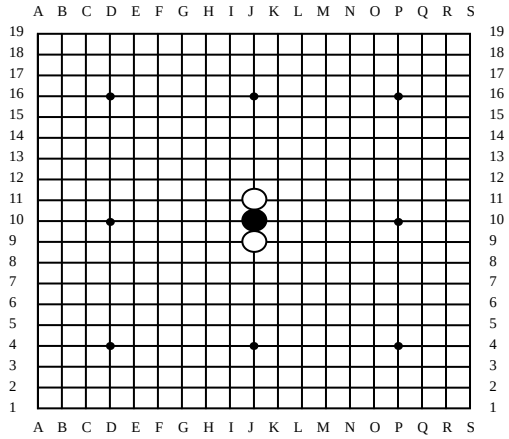

图 2.15 T-J11 定石

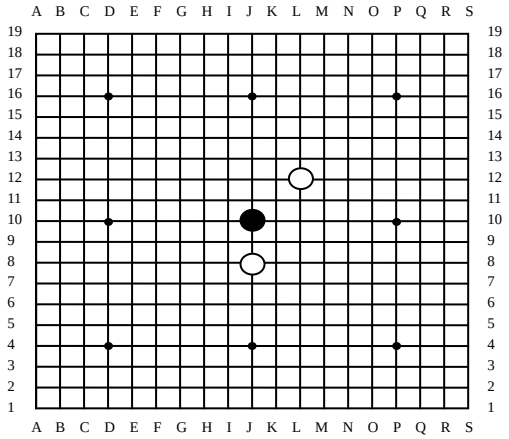

图 2.16 TT-L12 定石

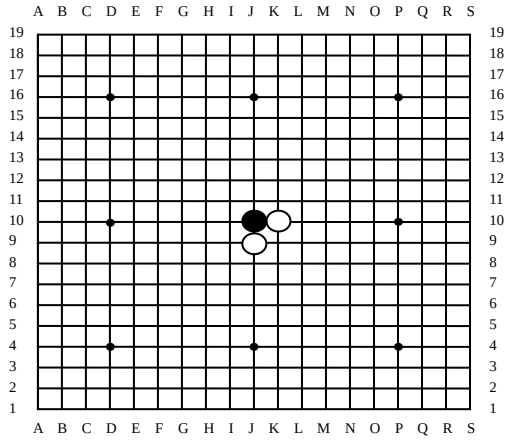

图 2.17 T-K10 定石

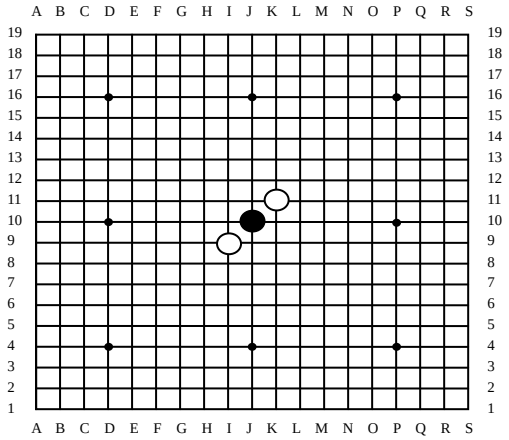

图 2.18 X-K11 定石

万方数据

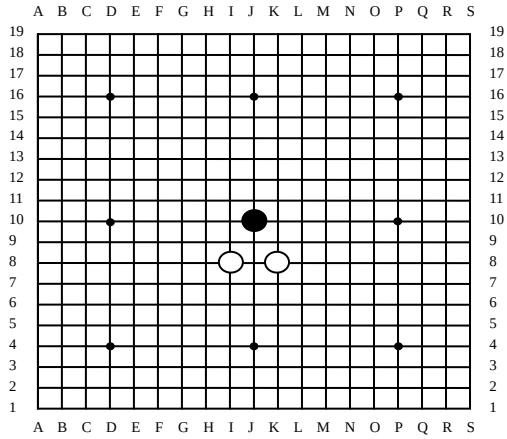

图 2.19 TX-K8 定石

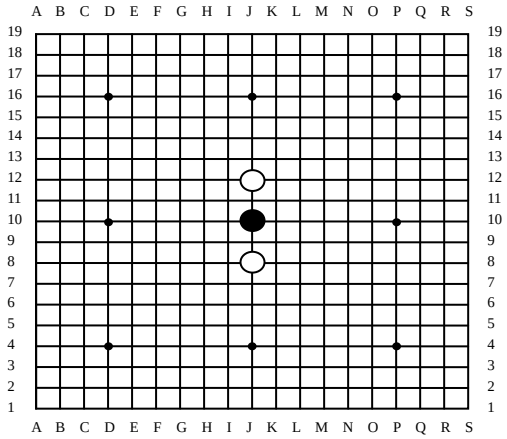

图 2.20 TT-J12 定石

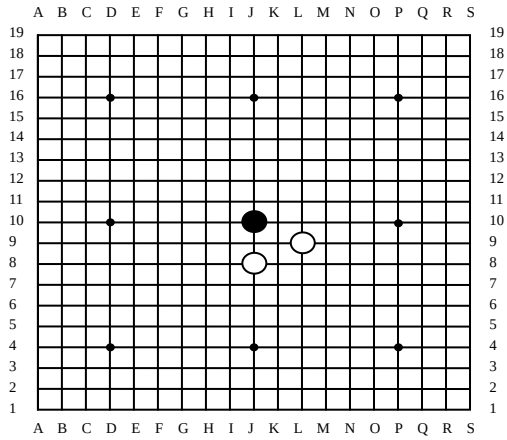

图 2.21 TT-L8 定石

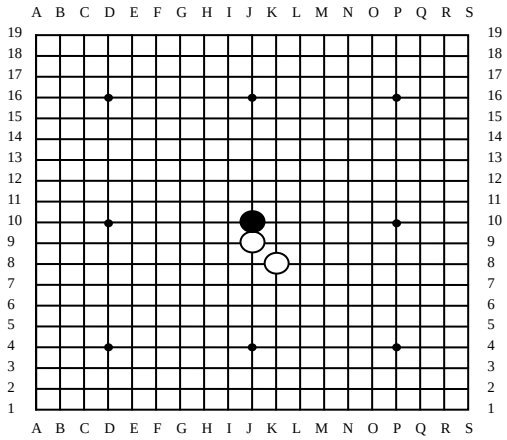

图 2.22 T-K8 定石

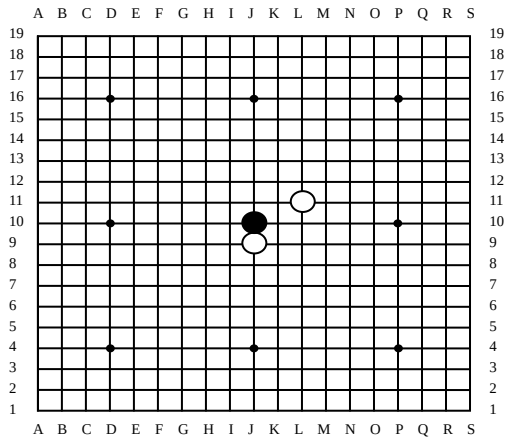

图 2.23 T-L11 定石

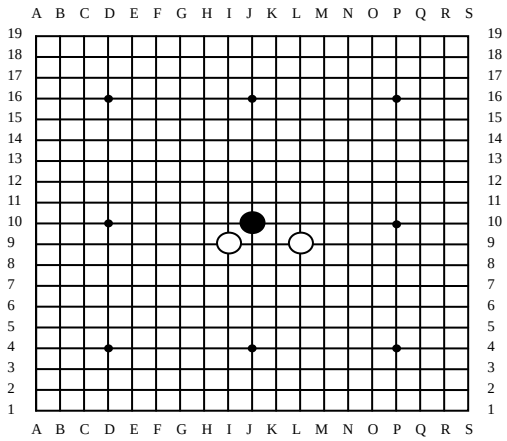

图 2.24 X-L9 定石

万方数据

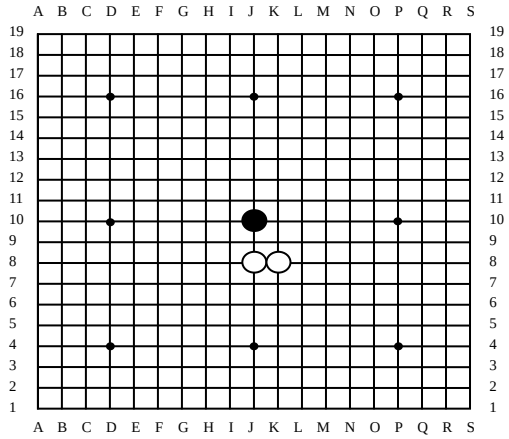

图 2.25 TT-K8 定石

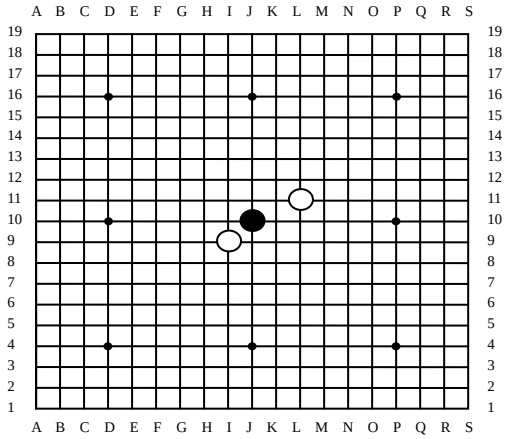

图 2.26 X-L11 定石

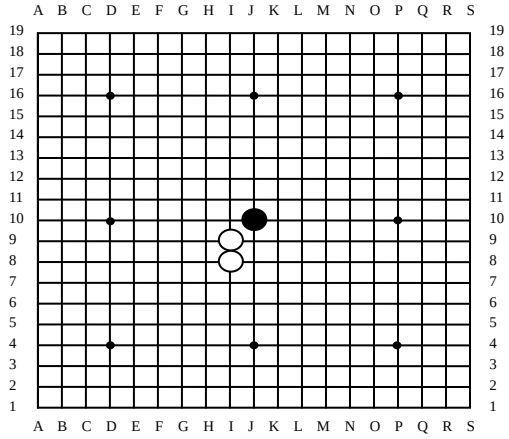

图 2.27 X-I8 定石

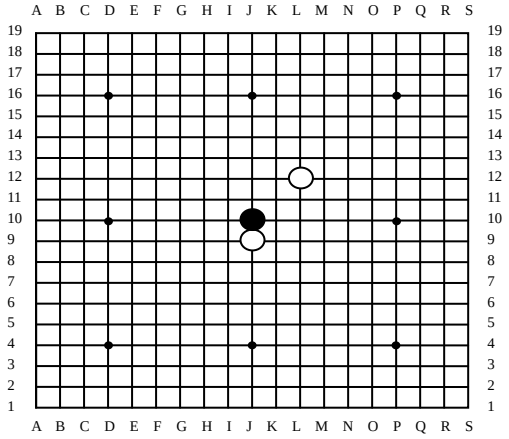

图 2.28 T-L12 定石

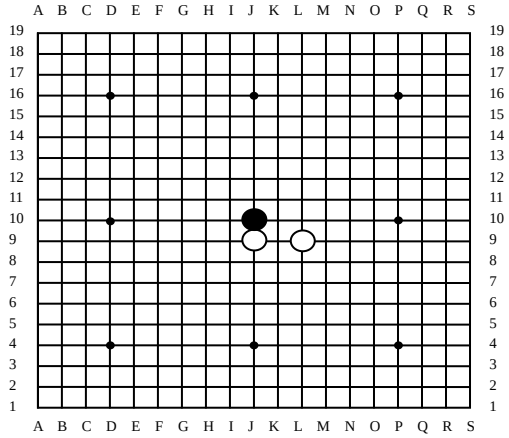

图 2.29 T-L9 定石

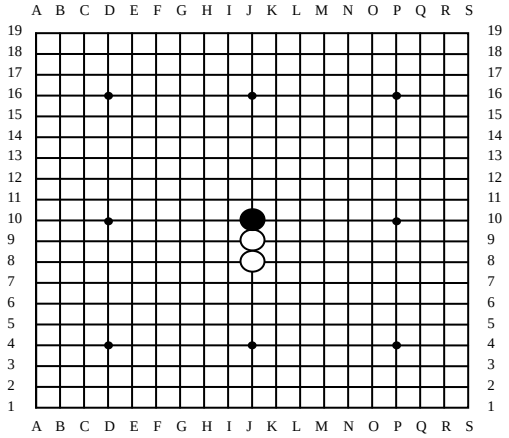

图 2.30 T-J8 定石

万方数据

2.2.2 诘棋

诘棋是指巧妙设计的盘面，解题者必须很有技巧地找出问题所要求达到的目的，例如必胜下法、阻挡下法。围棋、五子棋都称为诘棋，象棋多称为解残局。当然，六子棋也有，并沿用围棋、五子棋的说法，称之为诘棋。但有些六子棋的诘棋并非只有唯一的一解，这样的情形，答案尽量选择较简易的解法。

如图 2.31 和图 2.32 是两种诘棋，六子棋爱好者可以试着解答一下：

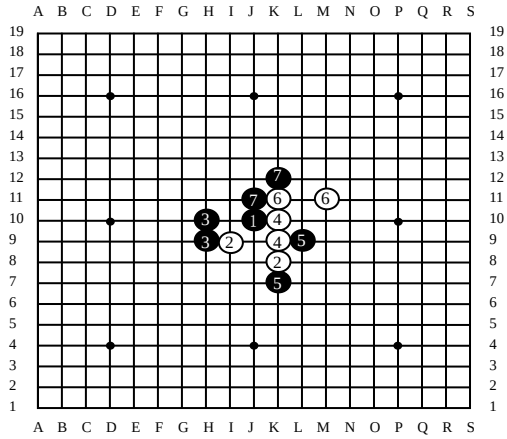

图 2.31 白先胜（追四胜）

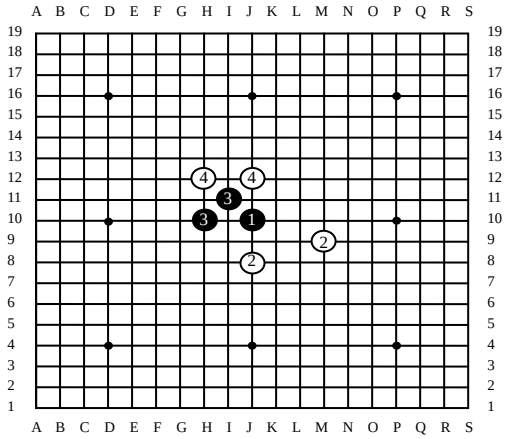

图 2.32 黑先胜（追四胜）

2.2.3 棋型

六子棋中的棋型如下：

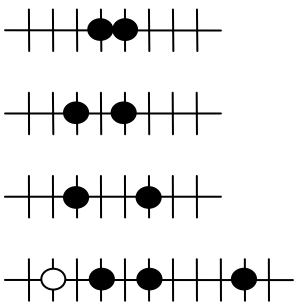

图 2.33 活二棋型

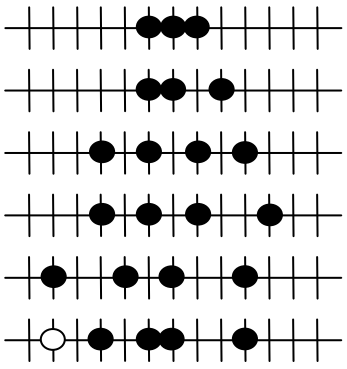

图 2.34 活三棋型

万方数据

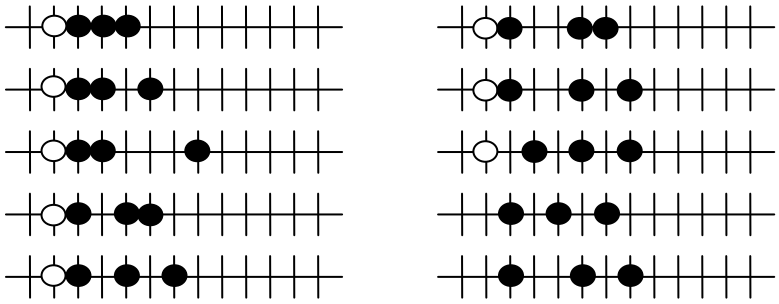

图 2.35 死三棋型

2.3 部分六子棋计算机博弈程序

2.3.1 NCTU6 计算机博弈程序

NCTU6（交大六号）是由台湾交通大学吴毅成教授及其带领的六子棋研究小组设计开发的一款六子棋计算机博弈软件。NCTU6 在多次机器博弈锦标赛中都取得了很好的成绩。该系统采用搜索算法是$\alpha-\beta$剪枝搜索，搜索深度为 3 层，并且结合了基于双迫着的迫着空间（threat-space）搜索[31]。

迫着(Threats)的定义如下：若一方需要下 t 颗子来避免另一方连成 k 颗子，则称该方有 t 个迫着。以六子棋为例，如图 2.36 所示，(a)单迫着(one threat)，(b)双迫着(two threats)，(c)三迫着(three threats)。如果一方某次下子后产生一个迫着，则称该落子为单迫着走步（single-threat move)。产生两个迫着，则称该落子为双迫着走步（double-threats move)。如果是三迫着的情况，则该方赢得比赛。

对六子棋来说，赢的策略就是阻挡所有对方的迫着，并同时产生三个以上的迫着。

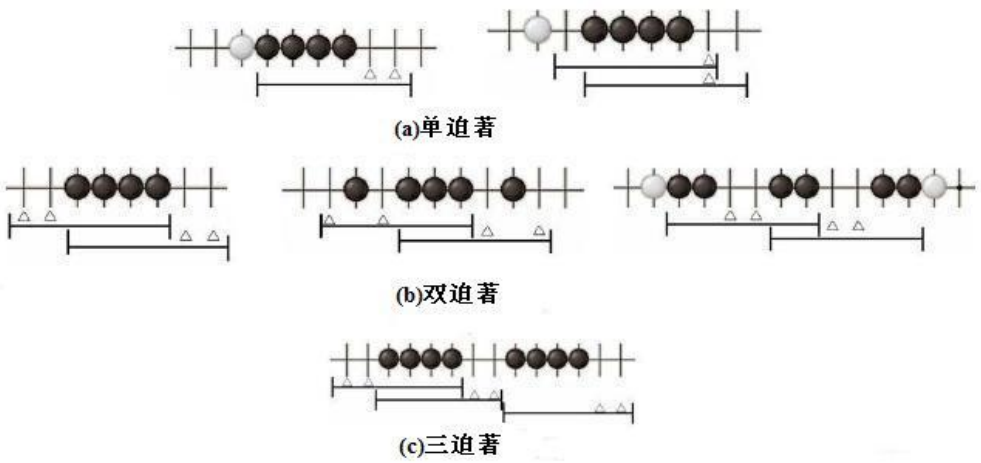

图 2.36 六子棋的迫著示意图

万方数据

六子棋中每下一颗子，对每一行最多可产生两个迫着，这一理论已得到证明。由这一理论衍生出活三、死三、活二、死二定义：若仅再下(4-t)颗子，就可以产生一个迫着，则为死-t 迫著；若仅再下(4-t)颗子，就可以产生两个迫着，则为活-t 迫著。如图 2.37 所示。

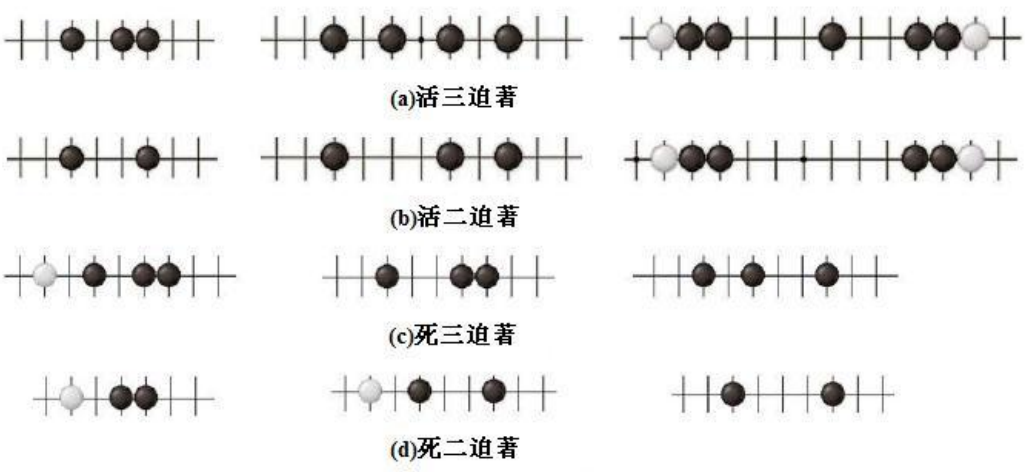

图 2.37 六子棋的-t 迫著示意图

对六子棋而言，活三、死三、活二、死二都很重要，这是因为每次可下两颗子，所以下一手棋就有机会形成真正的迫着。在所有的四种迫着中，活三、活二、死三迫著也被称为高度潜在迫着（HP-threats），因为一个子可以创造至少一个迫着或者两个子可以创造至少两个迫着。在所有高度潜在迫着中，活三和死三也被称为 HP3-threats，该迫着只需要下一个子就可以有至少一个迫着。玩家在进攻时通常想结合真正的迫着和高度潜在迫着。这种策略在六子棋中十分有效。

2.3.2 其他六子棋软件

花溪棋友是一款六子棋计算机博弈软件，曾经在第二次机器博弈锦标赛中代表重庆工学院进行比赛，取得了不错的成绩。该系统采用的是空位估值方法，搜索深度为一层。

在第 13 届国际机器博弈锦标赛中，NCTU6-Lite（台湾交大）获得了第一名的成绩，BitStronger（北京理工大学）、Neuconn6（东北大学）分别获得第二和第三名的好成绩。

万方数据

2.4 需要解决的主要问题

2.4.1 评估函数的分析

六子棋计算机博弈系统中评估函数模块的任务是：确定评估函数的参数类型以及参数个数，研究和解决由“两步”到“一步”的综合评估问题以及对评估函数的优化问题。

一般情况下，六子棋的评估函数设定为各方棋子在横、竖、斜方向上组成的棋型评估值之和。根据棋型特征对棋局的影响大小，来设定估值的高低。

本系统中拟选用新的评估函数进行估值，首先采用了基于静态估值方法的二次估值方法，后续又提出了空位估值方法，这两种方法在第三章中会有详细的介绍。

2.4.2 粒子群优化算法的应用

在对评估函数参数进行改进时，有些学者提出利用遗传算法对棋型估值进行优化，以达到更好的估值效果。标准遗传算法的实现过程中，影响遗传算法行为和性能的关键是交叉率 p_c 和变异率 p_m 的选取，这两个参数会直接影响算法的收敛性。但是，目前还没有通用的能够一次性确定 p_c 和 p_m 的方法[32]。针对不同的优化问题，需要反复通过实验来确定 p_c 和 p_m，这是一件非常繁琐的工作。

在空位估值方法中，我们设定的五种空位的估值主要来自于人类棋手的经验，估值更多的体现了人类棋手的对弈习惯。我们选用粒子群优化算法进行空位估值方法中参数的改进。根据粒子群优化算法的特点，能在更短时间内达到参数寻优。

2.4.3 棋形估值方法

针对六子棋的多种棋型，本文提出棋形向量这一概念。棋形向量是指对计算机方当前棋面棋型进行向量化得到的向量。该方法选取几种棋型作为量化基础，对棋形向量进行字典排序后，应用与走步指导。

2.5 小结

本章主要介绍了计算机博弈的研究现状，结合中国象棋和五子棋较成熟的计算机博弈技术，介绍了棋局表示、着法生成、评估函数、搜索算法等关键技术。介绍了六子棋的相关概念：定石、诘棋、棋型，并对部分六子棋程序进行了分析。最后指出了本文需要解决的关键问题。

万方数据

万方数据

3 评估函数的分析

3.1 静态评估函数分析

计算机博弈系统中，一般采用静态估值方法进行棋局评估，这种方法存在以下不足[33]：

① 对弈过程中，双方局势不会一成不变。如果对棋局评估时，博弈者对所有局势均使用同一评估函数，不能适应局势的变化。在实战中，由于棋局的不断变化会产生不同的棋面局势，即使是相同的棋面，对其进行评估的结果也有可能不同。局势变化越多，对其进行评估时产生的结果偏差越大，这种误差沿着回溯路径向上传播，将严重影响博弈者的决策。

② 博弈树搜索算法中用双方棋力差值作为评判标准。但是在实战中，双方棋力的变化是以各自在不同棋面下占有总棋力百分比为标志的。静态估值仅反映双方棋力的强弱，没有反映出差值量的权重。

③ 静态估值方法所针对的对象是一定深度处节点，通常是搜索到的最底层节点，再回溯确定博弈走法，仅是一种“由下而上”的方法。而在实战中，对于不同的局势每个棋手都会施加主观战略意图，伴随着一种“自上而下”的方法。思维方式的“剪刀差”使原有的静态估值方法无法体现博弈者的战略意图。

如果要掌握博弈的主动权，就需要在博弈过程中适时根据双方局势进行估值函数的调整，使得己方在对局势的估值方面占优势，或者尽可能与对方的估值方法一致。采用何种评估函数会直接决定博弈水平的等级[34]。

3.2 二次估值方法

计算机博弈中，如果双方采用相同的棋局评估函数，那么比赛只会出现两种结果，一种结果是先手赢，还有可能是平局。但在实战中，博弈双方的评估函数往往因人而异，对某一棋面状态的认识不同就会设定不同的估值，而且在棋面局势变化时，难免存在偏差。如果能够找到一种方法，可以不断的根据局势变化调整估值函数，使得己方的估值函数可以优于对方的估值函数，就可以掌握博弈的主动权。

基于这一点，我们需要在对弈过程中，充分考虑局势变化和己方的博弈特点，根据局势的变化进行评估函数的调整，从而提出了基于静态估值函数的二次估值方法。

万方数据

3.2.1 二次估值方法的概念

二次估值方法[35]是指在原有静态估值基础上，根据不同局势，通过引入一个合适的“局势因子”，指导双方局势走向，提高估值实用性的一种方法。数学表达式如下：

$$E = a\sum black + (1-a)\sum white \qquad \text{式(3-1)}$$

其中 E 代表棋面上棋子的总评估值，$\sum black$ 、$\sum white$ 分别代表黑方、白方所有棋子估值的累加之和。a 、$(1-a)$ 分别代表黑方和白方的局势因子，即黑白双方分别占总评估值 E 的比例。局势因子 a 存在以下约束条件：

$$0 < a < 1 \qquad \text{式(3-2)}$$

3.2.2 局势因子对路径选择的影响

局势因子的大小是对弈双方战略决策的反映，选择不同的局势因子会使局势产生不同的变化。在六子棋中，当己方意图进攻时，己方的局势因子的值应当大于对方的局势因子，即设定局势因子大于0.5；当我方想要进行防守时，则我方的局势因子应当小于对方的局势因子，即设定局势因子小于0.5。

3.2.3 二次估值方法的步骤

以六子棋中黑方为例，如图3.1二次估值方法的步骤：

① 根据静态估值方法，进行棋面评估，确定黑方的攻守形式。

② 六子棋开局时，设定局势因子为0.5。对弈过程中，首先利用传统搜索算法进行搜索，记录下博弈路径中己方所占棋力的百分比，确定博弈走法。

③ 在下一回合中，先使用原有的局势因子进行棋面评估，将该回合中己方棋力所占百分比与上一回合中的棋力百分比进行比较。

④ 如果这两个差值大于设定的局势变化阈值，则寻找其他走步位置。继续第(2)步。

⑤ 若差值小于设定的局势变化阈值，则记下该博弈路径上黑方棋力所占的百分比，给出博弈走法。

⑥ 以此类推，直至终局。

万方数据

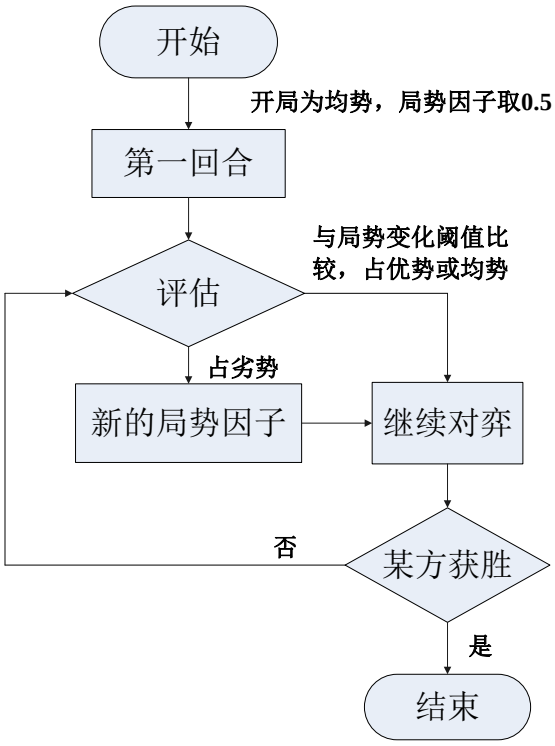

图3.1　二次估值方法的步骤

3.2.4 二次估值的应用

研究人员曾将二次估值方法应用于中国象棋计算机博弈系统中，针对红方进行实验，证实红方所选择的走步，是所有可能的走步中棋力百分比最高的。

我们将二次估值应用在六子棋计算机博弈中，想通过该方法改进六子棋计算机博弈系统。棋面总棋力的数学表达式为：

$$F = \sum black + \sum white \qquad \text{式(3-3)}$$

黑方的棋力百分比的数学表达式为：

$$p_b = \sum black / F \qquad \text{式(3-4)}$$

每次落子时，都要对棋面总棋力和黑方的棋力百分比进行统计，根据每次走步时，黑方的棋力百分比 p_b 的差值，与设定局势变化阈值进行比较，若大于局势变化阈值，则修改局势因子，重新进行搜索，确定新的博弈路径；若小于局势变化阈值，则继续走步。

六子棋的对弈过程可以看成是由具有时序关系的多个局势和棋面组成的。由棋面总棋力，双方棋力所占的比值组成了不同的局势，如攻势，守势，均势。在六子棋计算机博弈过程中，每个棋面是由多种棋型组成，对系统进行落子时，主要是通过对不

万方数据

同棋型进行估值，判断局势，进行落子。对棋型的赋值主要是对棋型可能对局势产生的影响进行量化。对棋型的估值规则如下：

① ○○●○○○○⇒ *value* =2

② ○○●●○○○⇒ *value* =40

③ ○○●●●○○⇒ *value* =200

④ ▲●●●●○○⇒ *value* =350

⑤ ○●●●●○○⇒ *value* =600

⑥ ▲●●●●●○⇒ *value* =1500

⑦ ○●●●●●○⇒ *value* =2000

⑧ ●●●●●●⇒ *value* =5000

其中，○表示该位置为空，▲表示该位置为白方棋子，●表示该位置为黑方棋子。根据对称原则，对这样两个棋型○○●○○○○和○○○○●○○，设定相同的估值。同理，对其他具有该对称规则的棋型其估值都统一为相同的。我们选用的棋型是对一般下棋规律的总结，在实际应用中，可以添加其他的规则，并且根据个人经验对评分准则加以修改。

运用二次估值方法后，六子棋计算机博弈系统的搜索路径可以根据局势因子的取值不同，发生相应的改变。局势因子的使用可以使博弈者按照自己对棋局的理解进行攻防之间的转变，从而能掌握博弈的主动权。

3.3 空位估值方法

3.3.1 模拟人的思维模式提出空位估值方法

计算机博弈中经常采用搜索过程中结合局面评价函数的方法进行棋局评估，这是一种数据驱动的算法，在国际象棋，中国象棋等棋类游戏中有广泛的应用，并且取得了很好的成绩。采用该方法进行走步指导是在一定搜索深度和经验的基础上得出落子位置，有些时候可能很不合理。不过人类棋手能很容易避免不合理走步，在满足某一目标的前提下，寻找一种当前局势下的最优走步，这是一种以目标为导向的驱动算法。

搜索算法在设计时必定要考虑空间和时间复杂度，穷举式搜索的方式行不通。在设计搜索和估值方法时，我们最好的参照物是人类棋手，人类棋手可以体验并表达对弈过程中的思维方式，这种思想体现在对弈过程中，体现在棋谱中。现如今，编程思想越来越普及，它强调以人类的思维模型为基础，设计相应的数据结构和算法。

六子棋对弈过程中，黑白双方对弈者针对当前棋面局势进行落子操作。双方思考棋盘中未落子位置对本方局势产生的影响，根据个人实战经验，对空位进行分析后，棋手总是寻找能抑制对方局势，同时使本方更快达到获胜局面的空位落子。

万方数据

我们模拟人类棋手的这一思维模式提出空位估值方法，即轮到计算机方进行落子操作时，系统对当前未落子位置（称为空位）进行评估，寻找对计算机方最为有利的空位进行落子操作。

3.3.2 空位估值方法的参数设定

对弈过程中，当某一方的六颗或六颗以上的棋子连成一条直线时（横向、竖向或斜向），该方即获胜。空秤开局时，系统先设定棋面的获胜组合，即棋面中所有可能六子连成一线的组合。在 19 路线的棋盘中共存在 924 种获胜组合，用 w 表示获胜组合（$0 \le w < 924$）。

棋面的某一空位可能存在于多个获胜组合中，根据每个获胜组合中已落子的棋子个数不同，分别赋予该空位不同的分值。对各个获胜组合中空位的分值进行累加，得到系统对该空位的估值。C_{iw} 表示计算机方（黑方）某一空位在 w 获胜组合中已落 i 颗子时的估值；P_{iw} 表示玩家（白方）某一空位在 w 获胜组合中已落 i 颗子时的估值。

按照棋手积累的经验进行棋局量化估值后，得到对空位的估值参数如下：

$$C_{1w}=5, C_{2w}=50, C_{3w}=100, C_{4w}=200, C_{5w}=400;$$

$$P_{1w}=5, P_{2w}=50, P_{3w}=100, P_{4w}=600, P_{5w}=800;$$

在设定量化估值时，对计算机方设定相对小的估值，更有利于计算机方在均势情况中进行“攻击”。

利用空位估值方法进行分析，针对黑方，当前棋盘空位包含在某几个获胜组合中，利用获胜组合中已有棋子个数的统计，进行空位估值；若包含该空位的任一获胜组合中，已经存在的棋子数为 1，系统对该空位的量化估值进行加 C_{1w} 操作，若该空位所在的获胜组合中，已落子数为 2，则对该空位的量化估值进行加 C_{2w} 操作，若该空位所在的获胜组合中，已落子数为 3，则对该空位的量化估值进行加 C_{3w} 操作，以此类推。用 Sc_{ij} 记录计算机方（黑方）某一空位置的估值，i 用棋盘的横向坐标表示，j 用棋盘的纵向坐标表示。对于白方采取类似的方法进行空位估值，白方某一空位置的估值记为 Sp_{ij}。

如图 3.2 所示，是某一对弈棋局。我们采用棋盘坐标表示空位置，I10 是棋盘的一个空位，该空位包含在多个获胜组合中，根据给定空位的估值参数，计算该空位的估值方法如下。首先要确定空位 I10 存在几个获胜组合中。获胜组合是指在棋盘横向、纵向和斜向方向上所有可能六子连成一线的组合，采用起点位置和终点位置的坐标来表示某一获胜组合。根据 I10 的位置，可以知道在横向方向上，I10 存在于六组获胜组合中，分别是（D10, I10）（E10, J10）（F10, K10）（G10, L10）（H10, M10）（I10, N10）。在纵向方向上，I10 存在于六组获胜组合中，分别是（I5, I10）（I6, I11）（I7, I12）（I8, I13）（I9, I14）（I10, I15）。在左斜向上，I10 存在于六组获胜组合中，分别是（I10, D15）

万方数据

（J9, E14）（K8, F13）（L7, G12）（M6, H11）（N5, I10）。在右斜向上，I10 存在于六组获胜组合中，（D5, I10）（E6, J11）（F7, K12）（G8, L13）（H9, M14）（I10, N15）。

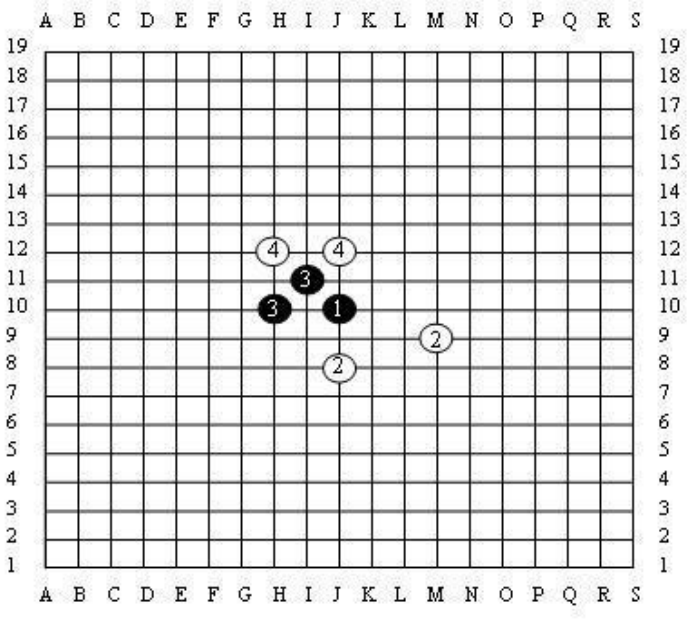

图 3.2 某对弈棋局

为了方便表示，简化空位估值的参数表示形式，对于计算机方，根据获胜组合中的落子个数 i $(i=1,2,3,4,5)$，将估值参数用 C_i 表示。

$C_1=5, C_2=50, C_3=100, C_4=200, C_5=400;$

根据上述分析，我们知道空位 I10 存在于 24 组获胜组合中，对 24 组获胜组合中已经落子的个数进行统计。落子个数为 1 个的获胜组合有 7 组，落子个数为 2 个的获胜组合有 4 组，其余获胜组合中都没有落子。根据空位估值方法，得到对空位 I10 的估值 Sc_{I10} 为 235，如式（3-5）所示。

$$Sc_{I10}=7\times C_1+4\times C_2+0\times C_3+0\times C_4+0\times C_5=235 \quad \text{式(3-5)}$$

当棋面中所有的空位分别估值后，对黑白双方具有最大估值的空位（黑方空位的最大估值记为 C_{grade}，白方空位的最大估值记为 P_{grade}）进行比较，若 C_{grade} 大于等于 P_{grade}，则选择该黑方空位落子，即黑方攻击白方，否则选择白方空位落子，即黑方防守。

3.3.3 空位估值方法的应用

利用空位估值方法构建六子棋计算机博弈系统，采用二维数组 board 来记录当前棋盘的状态，利用二维数组 win 来记录黑白双方在各获胜组合中所填入的组合数，二维数组 cgrades 和 pgrades 分别记录计算机和玩家在棋盘空格子上的分数，依照 3.3.2 节中的参数使用空位估值方法进行走步指导，搜索深度为一层。每次落子后检测是否

万方数据

某方已获胜，若获胜，则比赛结束，否则继续进行走步。算法 3.1 是空位估值算法。

算法 3.1 空位估值算法

```
void Connect6Form::ComTurn()//进行空位估值计算
{
    for(i=0;i<=18;i++)      //计算玩家在空格子上的获胜分数
        for(j=0;j<=18;j++)
        {
            pgrades[i][j]=0;
            if(board[i][j] == 2)
                for(k=0;k<924;k++)
                    if(ptable[i][j][k])
                    {
                        switch(win[0][k])
                        {
                        case 1:
                            pgrades[i][j]+=5;
                            break;
                        case 2:
                            pgrades[i][j]+=50;
                            break;
                        case 3:
                            pgrades[i][j]+=100;
                            break;
                        case 4:
                            pgrades[i][j]+=600;
                                break;
                        case 5:
                            pgrades[i][j]+=800;
                            break;
                        }
                    }
        }

    for(i=0;i<=18;i++)      //计算计算机在空格子上的获胜分数
        for(j=0;j<=18;j++)
        {
            cgrades[i][j]=0;
            if(board[i][j] == 2)
                    if(ctable[i][j][k])
                    {
                        switch(win[1][k])
                        {
                            case 1:
                                cgrades[i][j]+=5;
                             break;
```


万方数据

```
                        case 2:
                            cgrades[i][j]+=50;
                            break;
                        case 3:
                            cgrades[i][j]+=100;
                            break;
                        case 4:
                            cgrades[i][j]+=200;
                            break;
                        case 5:
                            cgrades[i][j]+=400;
                            break;
                        }
                    }
            }
        if((computer == 1&&!start)||computer == 2)      //取出 cgrades 与 pgrades 数组
中的最高分数
        {
            for(i=0;i<19;i++)
                for(j=0;j<19;j++)
                    if(board[i][j] == 2)
                    {
                        if(cgrades[i][j]>=cgrade)
                        {
                            cgrade = cgrades[i][j];
                            mat = i;
                            nat = j;
                        }
                        if(pgrades[i][j]>=pgrade)
                        {
                            pgrade = pgrades[i][j];
                            mde = i;
                        }
                    }
            if(cgrade>=pgrade)      //cgrade>=pgrade 计算机攻击玩家
            {
                m = mat;             //落子在对计算机方有利的位置
                n = nat;
            }
            else                    //计算机防守
            {
                m = mde;            //落子在对玩家有利的位置，进行防御
                n = nde;
            }
        }
}
```


万方数据

3.3.4 空位估值方法的优缺点

空位估值方法是通过模拟人类棋手的思维模式而设计的一种估值方法，该方法充分考虑了人类的思维模式，通过棋手在对弈过程中积累的经验设定空位估值参数，依据此参数进行棋局评估，以及落子指导。

空位估值方法还存在以下不足，参数是根据棋手经验设定的，由于水平有限，不能精确表示各空位对局势的影响。

3.4 小结

本章首先对静态评估函数进行了分析，静态评估函数不能适应局势变化，并且在应用过程中，存在思维方式的“剪刀差”。基于静态评估函数的不足，我们提出了结合局势变化和己方博弈特点的新的评估方法——二次估值方法，该方法使计算机方的棋力得到了较大的增长。在实战中，人类棋手关注的是当前棋面局势和棋盘空位置可能对本方产生的影响，模拟人的思维模式提出空位估值方法，该方法充分考虑了人类思维模式，依据人类棋手经验进行落子操作。

万方数据

万方数据

4 评估函数参数优化

计算机博弈系统中，评估函数的选取至关重要，评估函数对落子位置，局势走向都有直接的影响。评估函数中的参数设置更是一直以来的难点问题，大多评估函数是根据人类棋手的经验设定参数值，并没有通过数学方法进行验证。随着各种优化算法的出现，已经有许多学者将优化算法引入到评估函数[36]中。

4.1 遗传算法对评估函数的参数优化

如果对棋局评估时，采用静态评估函数，函数的估值设定需要设计人员有足够的经验，对下棋的方法有深入的了解，并且能充分判断局面中的某一特征在局势变化中所起的重要作用，即相应的分值，并给整个局面比较准确的评分，这一点非常困难。近年来，有些学者为了解决这一难题，将遗传算法引入到六子棋计算机博弈系统的评估函数中，对评估函数进行参数优化，取得了较好的效果。

4.1.1 遗传算法

遗传算法是模拟生物在自然环境中的遗传和进化过程而形成的一种自适应全局优化概率搜索算法。它最早由美国密执安大学 Jone Holland 教授提出，起源于 60 年代对自然和人工自适应系统的研究[37]。70 年代 DeJong 基于遗传算法的思想在计算机上进行了大量的纯数值函数优化计算实验[38]。在一系列研究工作的基础上，80 年代 Goldberg 进行归纳总结，形成了遗传算法的基本框架[39]。

遗传算法的基本原理[40]：

① 初始化群体，设置进化代数计数器t，设置最大进化代数T，随机生成M个个体作为初始群体$P(0)$。

② 个体评价，计算群体$P(t)$中各个个体的适应度。

③ 选择运算，将选择算子作用于群体。

④ 交叉运算，将交叉算子作用于群体，按概率p_c进行交叉操作。

⑤ 变异运算，将变异算子作用于群体，群体按概率p_m进行变异操作。群体$P(t)$经选择、交叉、变异运算之后得到下一代群体$P(t+1)$。

⑥ 终止条件判断，若$t \le T$，则$t \leftarrow t+1$，转到步骤二；若$t > T$，则进化过程中所得到的具有最大适应度的个体作为最优解输出，终止计算。

4.1.2 遗传算法的应用

由于六子棋只有黑白双方，没有棋子与棋子之间的差别，它的评估函数只跟当前棋子的落子位置和周边棋子的位置状态有关系。通过这样的分析，研究人员把当前落

万方数据

子位置作为中心，向横向、竖向、左斜向、右斜向进行定长为 13 的扫描，生成状态数组。状态数组是用来描述当前棋面的落子情况，主要由四个整型数值表示，-1, 0, 1, 2，分别表示棋盘外，黑子，白子，无子的情况。根据人类棋手经验对状态数组表示的棋型给予不同的估值。人类棋手是根据自己的经验和习惯给出的估值，可能会出现偏差，研究人员采用遗传算法对估值进行优化[41]。

初始化时，根据给定的估值，采用直接编码方法，即直接将估值转化为二进制，编码精度为 16 位，占 2 个字节，高位用“0”填满。将其作为输入，通过锦标赛选择、均匀交叉、变异等遗传操作过程，最终得到下一代种群，重复上述遗传操作过程，最终得到理想的一个个体或一组个体。

4.2 粒子群优化算法

通过模拟生物群体的行为来解决计算问题已经成为新的研究热点，形成了以群体智能(Swarm Intelligence)为核心的理论体系，并已在一些实际应用领域取得突破性进展。作为群体智能的典型实现模式，模拟蚂蚁群落食物采集过程的蚁群优化算法(Ant Colony Optimization)和模拟鸟群运动模式的粒子群优化算法[42](Particle Swarm Optimization)受到学术界的广泛关注。粒子群优化算法是由 Eberhart 和 Kennedy 等于 1995 年开发的一种进化计算技术[43]。

4.2.1 算法原理

粒子群优化算法是基于鸟群觅食行为的一种演化算法，具有进化计算和群智能的特点。设想这样一个场景：某一区域内有一块食物，一群鸟在随机搜寻食物，所以的鸟都不知道食物在哪里，但它们知道当前位置与食物之间的距离。在这种情况下，想要找到食物的唯一方法就是在距离食物最近的鸟的周围区域进行搜索。在 PSO 中，每个潜在的可能解都是搜索空间中的一只“鸟”，称之为“粒子”。所有的粒子都有一个被目标函数决定的适应值（fitness value），在每次迭代中，粒子通过跟踪两个极值来更新自己[44]：第一个就是粒子本身所找到的最优解，叫做个体极值 *pBest*；另一个极值是整个种群目前找到的最优解，叫做全局极值 *gBest*。粒子群优化算法实质上也是通过个体间的协作与竞争，实现复杂空间中的最优解的搜索。

通过数学描述为：设在 N 维搜索空间中，由 M 个粒子组成的种群。其中粒子 i 的位置为 $X_i=(x_{i1},x_{i2},...,x_{iN})^T$，速度为 $V_i=(v_{i1},v_{i2},...,v_{iN})^T$，该粒子的个体极值表示为 $P_i=(p_{i1},p_{i2},...,p_{iN})^T$，该极值可以认为是粒子自己的飞行经验。全局极值表示为 $P_g=(p_{g1},p_{g2},...,p_{gN})^T$，该极值可以认为是粒子的群体经验。粒子就是通过自己的飞行经验和群体经验来选择下一个移动位置。对于第 $k+1$ 次迭代，每一个粒子是按照下式进行变化的：

万方数据

$$v_{id}^{k+1}=v_{id}^{k}+c_1\times rand()\times(p_{id}-x_{id}^{k})+c_2\times rand()\times(p_{gd}-x_{id}^{k}) \quad \text{式(4-1)}$$

$$x_{id}^{k+1}=x_{id}^{k}+v_{id}^{k+1} \quad \text{式(4-2)}$$

式中，$i=1,2,...,M,M$ 是种群规模。$d=1,2,...,N,N$ 为解空间的维数，即自变量的个数。加速因子 c_1、c_2 分别调节向 *pBest* 和 *gBest* 方向飞行的最大步长，合适的 c_1、c_2 可以加快收敛且不易陷入局部最优。一般情况下，我们取 c_1,c_2 分别为 2，可以得到较好的搜索效率。为防止粒子飞出可行区域，可以设定最大速度 $V_{\max}$，粒子的每一维速度 v_{id} 都会被限制在 $[-v_{d\max},+v_{d\max}]$ 之间，假设搜索空间的第 d 维定义为区间 $[-x_{d\max},+x_{d\max}]$，则通常 $v_{d\max}=k\bullet x_{d\max},0.1\le k\le 0.2$，每一维都用相同的设置方法[45]。

公式(4-1)主要通过三部分来计算粒子 i 更新的速度：第 1 部分是动量部分，是粒子对自身运动行为的信任，为自身提供了继续运动的惯性，即指粒子 i 前一时刻的速度 v_{id}^{k}；第 2 部分是“认知（cognition）”部分，体现粒子自身的搜索结果，即指粒子 i 当前位置与自己历史最好位置之间的距离 $(p_{id}-x_{id}^{k})$；第 3 部分是“社会（social）”部分，体现了粒子间的信息共享与合作，即指粒子 i 当前位置与群体最好位置之间的距离 $(p_{gd}-x_{id}^{k})$。粒子 $p_{gd}-x_{id}^{k}$ 通过公式（4-2）计算新位置的坐标。粒子移动原理如图 4.1 所示：

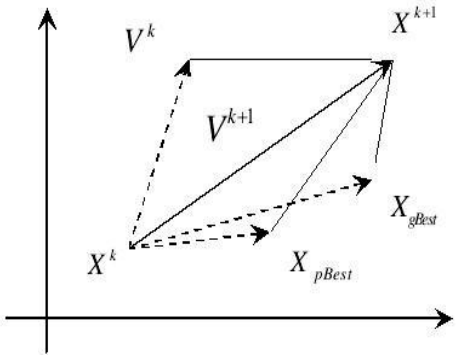

图 4.1 粒子移动原理图

公式(4-1)的第一项对应多样化(Diversification)的特点，第二项、第三项对应于搜索过程的集中化(Intensification)特点，因此这 3 项之间的相互平衡和制约决定了算法的主要性能。

4.2.2 算法步骤

① 初始化粒子群：给定群体规模 M，解空间维数 N，随机产生每个粒子的位置 X_i、速度 V_i。

② 计算每个粒子的当前适应值。

③ 更新个体极值。

万方数据

④ 更新全局极值。

⑤ 更新速度和位置：通过公式（4-1）、(4-2)来更新每个粒子的速度V_i和位置X_i。

⑥ 检查是否满足终止条件，若满足，则退出；否则，转至步骤(2）。

4.2.3 惯性权重（inertia weight）的引入

如果没有公式（4-1）的第一部分，粒子群算法会在迭代过程中使搜索空间逐渐收缩，体现出一种局部搜索能力；反之，如果增加第一部分的权重，粒子就有能力扩展搜索空间，展现出一种全局搜索能力；搜索过程中的两种能力的平衡对于寻优有着重要的作用。基于以上的考虑，Shi 等人[46]在公式(4-1)中引入了惯性权重w，引入w后公式（4-1）变为：

$$v_{id}^{k+1}=w\times v_{id}^{k}+c_1\times rand()\times(p_{id}-x_{id}^{k})+c_2\times rand()\times(p_{gd}-x_{id}^{k}) \qquad \text{式(4-3)}$$

算法改进初期，惯性权重w选用固定常数，但是经过试验发现，动态惯性常数比固定常数具有更好的寻优能力，所以现在一般都采用 Shi 提出的线性递减权值（linearly decreasing weight）策略，其惯性权重计算公式如下：

$$w=w_{\max}-\frac{w_{\max}-w_{\min}}{iter_{\max}}\times iter \qquad \text{式(4-4)}$$

$w_{\max},w_{\min}$分别是w的最大值和最小值；$iter,iter_{\max}$分别是当前迭代次数和最大迭代次数。一般情况下，$w_{\max}$的取值为 0.9，$w_{\min}$的取值为 0.4，可以使粒子群算法在开始时搜索较大区域，迅速定位最优解的大致位置，随着w逐渐减小，粒子慢慢减速，开始在局部进行搜索。该方法使 PSO 更好的控制局部搜索和全局搜索能力，加快了收敛速度，又称之为 LDW(Linearing Decreasing Inertia Weight)。

4.2.4 收缩因子（constriction factor）的引入

粒子群优化算法起源于模拟鸟群觅食活动，算法本身缺乏坚实的数学基础。1999 年 Clerk 对算法的数学研究证明[47]，采用收缩因子能够确保算法的收敛。收缩因子模型如下所示：

$$v_{id}^{k+1}=k\times[v_{id}^{k}+c_1\times rand()\times(p_{id}-x_{id}^{k})+c_2\times rand()\times(p_{gd}-x_{id}^{k})] \qquad \text{式(4-5)}$$

$$k=\frac{2}{\left|2-\phi-\sqrt{\phi^2-4\phi}\right|} \qquad \text{其中}\ \phi=c_1+c_2,\phi>4 \qquad \text{式(4-6)}$$

$$x_{id}^{k+1}=x_{id}^{k}+v_{id}^{k+1} \qquad \text{式(4-7)}$$

通常ϕ值取 4.1（$c_1=c_2=2.05$）。收缩因子能控制系统的最终收敛，该方法简记为 CFM（Constriction Factor Model）。

万方数据

4.2.5 仿真实例

为了验证粒子群优化算法的寻优能力，以函数 $f(x,y)$ 为例进行验证。

$$f(x,y)=x\sin(4\pi x)-y\sin(4\pi y+\pi+1) \qquad \text{式(4-8)}$$

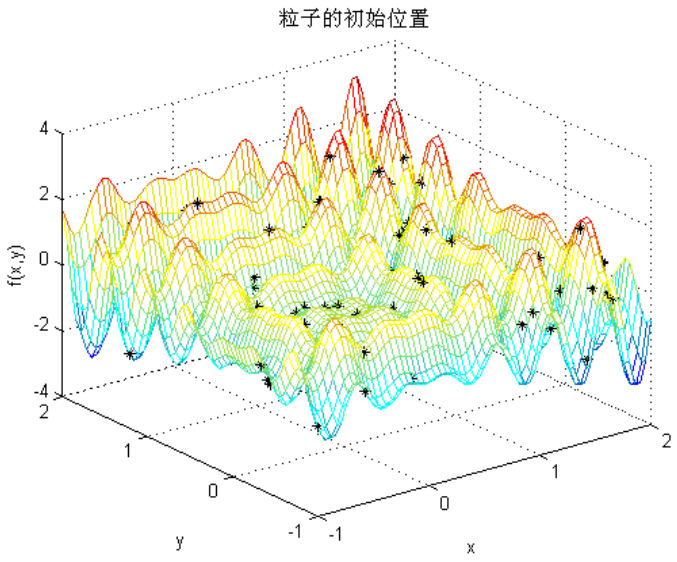

图 4.2 函数（4-8）粒子的初始位置

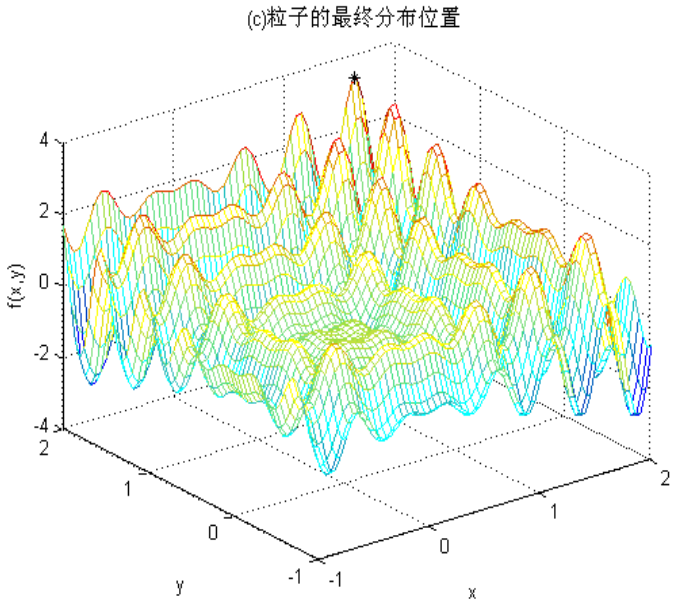

图 4.3 粒子的最终分布位置

万方数据

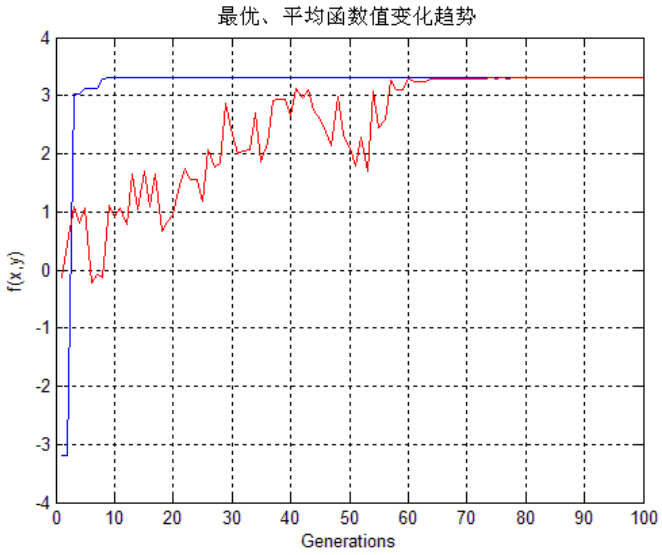

图 4.4 最优、平均函数值变化趋势

4.3 粒子群优化算法对评估函数的参数优化

计算机博弈系统中直接影响对弈效果的两个关键技术，一个是评估函数，另一个是博弈树搜索，如果能选取精确表示棋面局势的评估函数，就能在对弈过程中发挥本方的优势，达到获胜局面；如果博弈树的搜索深度无限制，那么在开局的时候就可以找到胜局的走法，这两方面是我们希望能达到的。但是基于硬件环境的限制，博弈树的展开不可能无限大，只能在固定深度中进行搜索。那么，评估函数需要寻找最优估值来进行棋局评估，达到完美效果。

在六子棋计算机博弈系统中，主要利用当前双方棋面的估值进行落子指导。评估函数包括对棋型、棋型位置和空位置的估值。我们采用引入了惯性权重和收缩因子的粒子群优化算法对空位置的估值进行了参数优化。

空位估值方法是指轮到计算机方进行落子操作时，系统对当前未落子位置（称为空位）进行评估，寻找对计算机方最为有利的空位进行落子操作。根据人类棋手经验，我们设定的计算机方（C_{iw}）和玩家（P_{iw}）的空位置的估值为：

$$C_{1w}=5, C_{2w}=50, C_{3w}=100, C_{4w}=200, C_{5w}=400;$$

$$P_{1w}=5, P_{2w}=50, P_{3w}=100, P_{4w}=600, P_{5w}=800;$$

利用粒子群优化算法对估值进行优化，首先需要解决的问题是将评估函数中需要优化的估值在粒子群优化算法中找到合适的表示方法。我们设定群体规模为 20，解空间维数为 10，随机产生每个粒子的初始位置 $X_i=(x_{i1}, x_{i2}, \ldots x_{i10}{}^T)$和初始速度 $V_i=(v_{i1}, v_{i2}, \ldots y_{i10}{}^T)$, 初始位置和初始速度的维数为 10，第 1 维到第 5 维依次表示的是计算机方某一获胜组合中有一颗，两颗，三颗，四颗和五颗棋子时的估值，第 6 维到

万方数据

第 10 维依次表示的是玩家某一获胜组合中有一颗，两颗，三颗，四颗和五颗棋子时的估值。

粒子群优化算法中，一般将目标函数作为适应度函数。六子棋的空位估值判定没有确定的函数可以描述，所以，我们采用专门的棋类游戏适应度函数计算方法--锦标赛算法。我们将系统进行改进，让其具有两个评估函数，分别读取不同的参数组来进行计算机之间的对弈，记录比赛结果。初始状态时，我们设定双方具有相同的适应度常数。根据比赛结果，对适应度常数进行调整，胜一局加 1 分，负一局减 1 分，平局不进行操作。这样，可以根据适应度函数值来确定个体极值和全局极值，通过公式（4-1）和（4-2）来更新每个粒子的速度和位置，算法 4.1 是粒子飞行中的寻优过程。当满足终止条件时停止迭代，最终找到合适的评估值。进行多次实验后我们得出利用粒子群优化算法进行改进的空位置的估值为：

$$C_{1w}=7, C_{2w}=65, C_{3w}=122, C_{4w}=298, C_{5w}=525;$$
$$P_{1w}=9, P_{2w}=60, P_{3w}=150, P_{4w}=330, P_{5w}=903;$$

怎样证明参数改进后系统具有更好的效果？我们将使用原参数和改进后参数的两个系统进行比赛，依据比赛获胜概率来进行评定。使用原参数的系统称为 S1，使用改进后参数的系统称为 S2，双方共进行 15 场比赛，比赛结果如表 4.1 所示。根据比赛结果可以看出，15 场对局中，S2 赢得 10 场比赛，平两局，负三局，参数改进后的 S2 具有绝对的优势。从而证明了粒子群优化算法对空位估值中参数的改进是有效果的，可以将改进后的参数应用于系统，系统具有更好的性能。

表 4.1 S1 与 S2 比赛记录表

获胜结果 / 局数 / 对弈双方	胜	平	负	总计
S1	3	2	10	15
S2	10	2	3	15

万方数据

算法 4.1 粒子群优化算法的寻优过程

```
void ParticleFly()//粒子飞行寻优
{       srand((unsigned)time(NULL));
        static int k=10;
        w=Wmax-k*(Wmax-Wmin)/Kmax;      //惯性权重采用线性递减方法
        k++;
        for (int i=0;i<Pnum;i++)                    //Pnum 代表种群规模
         {       for (int j=0;j<Dim;j++)            //Dim 代表粒子维数
            {
             Parr.V[j]=(int)(w*Parr.V[j])+(int)(c1*rand()/(double)RAND_MAX*
                    (Parr.XBest[j]-Parr.X[j])+c2*rand()/(double)RAND_MAX*
                    (Parr[GBIndex].XBest[j]-Parr.X[j]));
                    }
                for (int j=0;j<Dim;j++)
                {
                    Parr.X[j]+=Parr.V[j];
                }
         }
        CalculateFit();
        for (int i=0;i<Pnum;i++)
        {
            if (Parr.Fit>=Parr.FitBest)
            {   Parr.FitBest=Parr.Fit;
                for (int j=0;j<Dim;j++)
             {
                Parr.XBest[j]=Parr.X[j];
             }
        }
   }
   GBIndex=0;       //GBIndex 最优粒子索引
   for (int i=0;i<Pnum;i++)
    {
        if (Parr.FitBest>Parr[GBIndex].FitBest&&i!=GBIndex)
     {
      GBIndex=i;
     }}}
```


万方数据

4.4 小结

本章介绍了如何采用遗传算法和粒子群优化算法对评估函数进行参数优化。采用遗传算法主要是针对状态特征对棋局影响的估值进行优化。我们采用粒子群算法对空位置的估值进行了优化，将优化结果应用到系统中，通过与应用原估值的系统进行比赛，证实了优化后系统的性能更好。

万方数据

万方数据

5 棋形估值方法

5.1 棋形估值方法的提出

使用粒子群优化算法对评估函数进行参数改进时，只对单个棋子对局势产生的影响进行了估值，并没有考虑棋子与棋子之间的联系。任何事物都不是孤立存在的，是否能通过寻找棋子与棋子之间的联系，使棋局评估有更好的效果？

棋型是某方棋子在棋盘上落子后表现出来的棋子间位置关系的总称，某一棋型（参见 2.2.3 节）一般由多颗棋子组成。如果对其进行整体考虑，用来进行棋局评估是否能达到更好的效果？

基于这一方面的考虑，提出了棋形估值方法（Chess-shaped Method of Valuation）。

5.2 棋形向量

棋形向量（Chess Shape Vector）是指在已选棋型基础上，对计算机方当前棋面棋型在横、竖、斜三个方向进行量化后得到的向量。假如选取 n 种棋型，那么生成的棋形向量为 $(s_1, s_2, s_3 ... s_n)$。根据棋型对局势的影响程度，我们对其进行优先级排序，对局势影响程度越大，其优先级越高，进行量化后将其放在棋型向量的第 i 维，则 i 值越小。空秤开局时，棋形向量为 $(0,0,0,...,0)_n$。

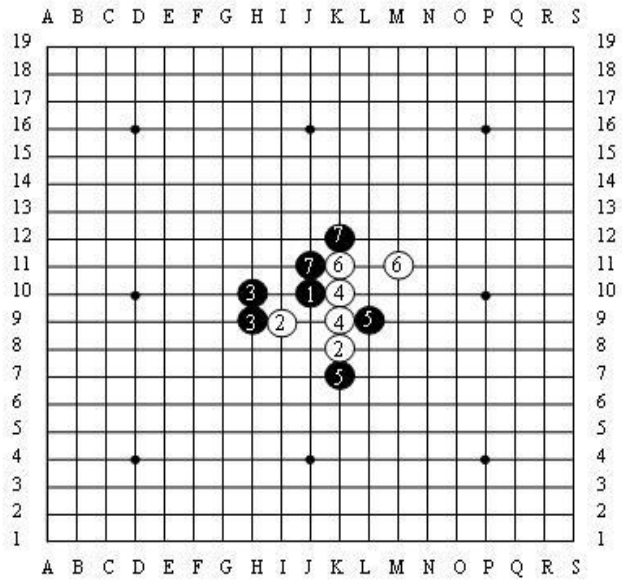

图 5.1 某对弈棋局

万方数据

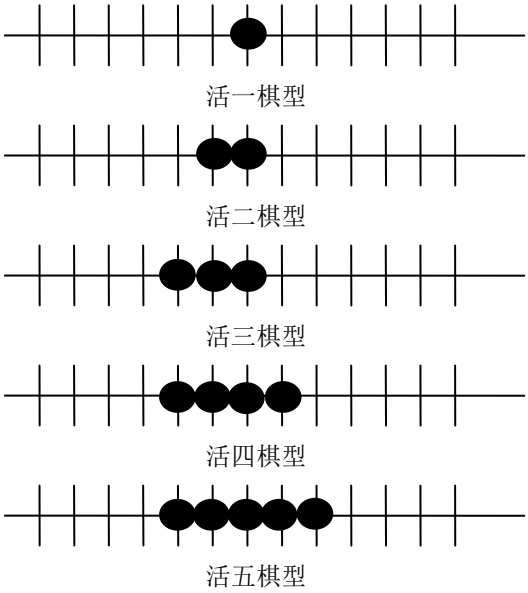

图 5.2 选用棋型

如图 5.2 是选用棋型，依据选用棋型，根据其对局势的影响，很容易得到棋型的优先级关系，由高到低依次是活五棋型、活四棋型、活三棋型、活二棋型、活一棋型。根据得出的优先级，可生成一个 6 维的棋形向量，该向量的第一维是获胜棋型（即六子连珠）的个数。

图 5.1 为某对弈棋局，黑方进行量化后得到棋形向量为$(0,0,0,0,3,16)$，白方进行量化后得到的棋形向量为$(0,0,0,0,0,36)$。注意，每个棋子在横、竖、斜三个方向都要进行统计；同时可以看到白方棋子 2、4、4、6 在一条直线上，但由于两端都有黑方棋子牵制，所以在进行向量统计时不予计数。

5.3 棋形估值方法的应用

棋形估值方法主要应用于博弈树搜索过程中，对叶节点和终节点进行评估，指导走步。在极大极小搜索算法中，首先要根据落子方，进行博弈树的展开，按照搜索深度，展开整棵博弈树。利用评估函数对每个节点进行估值。用终节点的估值来得到其搜索树上一层节点的估值。在 MAX 层取分支的最大值，MIN 层取分支的最小值，一直回溯到根结点。最终，根结点选择分支值最大的走步走。

六子棋中，当前状态下，所有的空位置都是可行走步，如果按这种规则进行博弈树的展开，那么这棵博弈树将非常庞大。所以，在对其进行展开时，我们应该对展开节点进行限制，将博弈树控制在一定范围内。搜索过程中节点的展开主要依据对空位置的估值。将算法 3.1 进行修改，分别对黑白双方进行空位估值，并对得到的估值进行排序，估值越大，其排序越靠前，这里，我们选取前三个位置的空位作为博弈树展

万方数据

开的节点，即设定博弈树每层的节点数为 3 个。

将黑方记为 MAX, 白方记为 MIN。如果轮到黑方落子，根据对黑方空位置的估值，选取估值的前三个，进行博弈树的展开。对落子后的棋面利用棋形估值方法得到棋形向量，将棋形向量作为每个叶节点的估值。白方也进行相同的操作。如图 5.3 极大极小搜索算法（搜索深度为一层）。

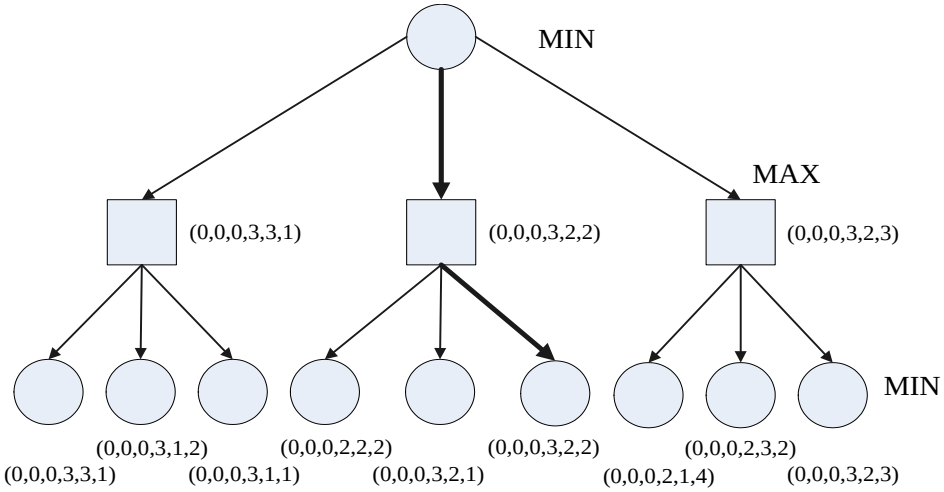

图 5.3 极大极小搜索算法

对每个叶节点和终节点进行评价后，按照极大极小搜索算法的规则，从终节点选取棋形向量最小的节点（对棋形向量利用字典排序比较大小），通过该节点搜索其上一层节点，得到对应的落子位置。如图 5.3 所示粗线所指示的节点。

5.4 棋形估值方法的不足

棋形估值方法的优点：对棋型向量进行分析时，我们将棋型作为一个整体，大大减少了静态估值时对当前棋面每个棋子进行估值的繁琐操作，节省了扫描时间。在走步指导过程中涉及到的给定值只有五个对空位置的估值，估值量较小，减少了人为因素对棋局走势产生的影响，能更好的通过棋型确定落子位置。

棋形估值方法的缺点：棋形向量是在选定棋型基础上生成的，选定棋型数量少，不全面就不能完全体现棋局。我们在设定系统时，需要通过实验确定有效棋型。

另外，可以考虑增加棋型位置之间的评估，因为有些棋型在一定的棋盘距离内会产生影响。

万方数据

5.5 小结

本章主要介绍了棋形估值方法，该方法利用棋形向量进行当前局面棋型估值，指导走步。将该方法与空位估值方法相结合，应用到计算机博弈系统中，取得了很好的效果。

万方数据

6 六子棋计算机博弈系统

通过前面各章节的研究工作，我们构建了六子棋计算机博弈平台，Connect6。

6.1 Connect6 系统功能

6.1.1 Connect6 系统介绍

Connect6 系统构建的关键技术包括：棋局表示、着法生成、评估函数、搜索算法。该系统采用搜索算法是极大极小搜索，搜索深度为两层，并结合了基于空位估值和棋形估值的搜索方法。

计算机能进行对弈，首先需要识别棋盘信息。六子棋采用 19 路线的棋盘，盘面上共有 361 个交叉点，采用 19×19 的数偶矩阵表示棋盘中的落子位置。在计算机中采用二维数组 board 来描述棋盘的状态，其中“0”表示黑子，“1”表示白子，“2”表示空位置。

计算机可以识别棋盘信息后，下一步需要完成的工作就是落子。根据游戏规则，选择正确的位置进行落子操作，完成这一系列“动作”，应用到的技术有着法生成、评估函数和搜索算法。六子棋中，当前棋盘状态下，所有的空位置都是可行落子位置，采用棋盘扫描法确定所有可行的落子位置，利用空位估值方法计算得到各个空位置的估值。空位估值方法中的参数设定是利用 4.2.5 节中的估值参数 C_{iw} 、 P_{iw} 。

$C_{1w}=7, C_{2w}=65, C_{3w}=122, C_{4w}=298, C_{5w}=525;$

$P_{1w}=9, P_{2w}=60, P_{3w}=150, P_{4w}=330, P_{5w}=903;$

Connect6 的评估函数采用棋形估值方法，棋形估值模块中选用的棋型如图 5.2 所示。由于系统采用极大极小搜索算法进行两层搜索，搜索过程中节点主要是依据对空位置的估值进行展开。棋形估值方法主要应用于博弈树的各个节点的估值，棋形向量进行比较后，选取最优路径，进行落子操作。

6.1.2 Connect6 系统功能

该系统的开发过程包含下面这些内容：

① 棋盘棋子信息显示。我们选用二维数组表示棋盘信息，二维数组中的存储数值有三种类型：0，1，2。0 代表该位置为黑子，1 代表该位置为白子，2 代表该位置为空。

② 获胜局面设置。在 19 路棋盘中，在横、竖、斜三个方向上所有六子连成一线

万方数据

的可能为 924 种，在初始化时，先得到所有的获胜组合。

③ 先手控制，走子提示。

④ 悔棋操作，在对弈过程中，可以进行悔棋操作。

⑤ 运用空位估值方法进行当前棋面空位置的估值计算。

⑥ 结合空位估值方法，进行极大极小搜索，将利用棋形估值方法生成的棋形向量作为节点的评估值，搜索落子位置。

⑦ 判定是否获胜。每次落子后都对当期棋面进行获胜判定，若某方获胜，则在界面下方提示某方获胜。

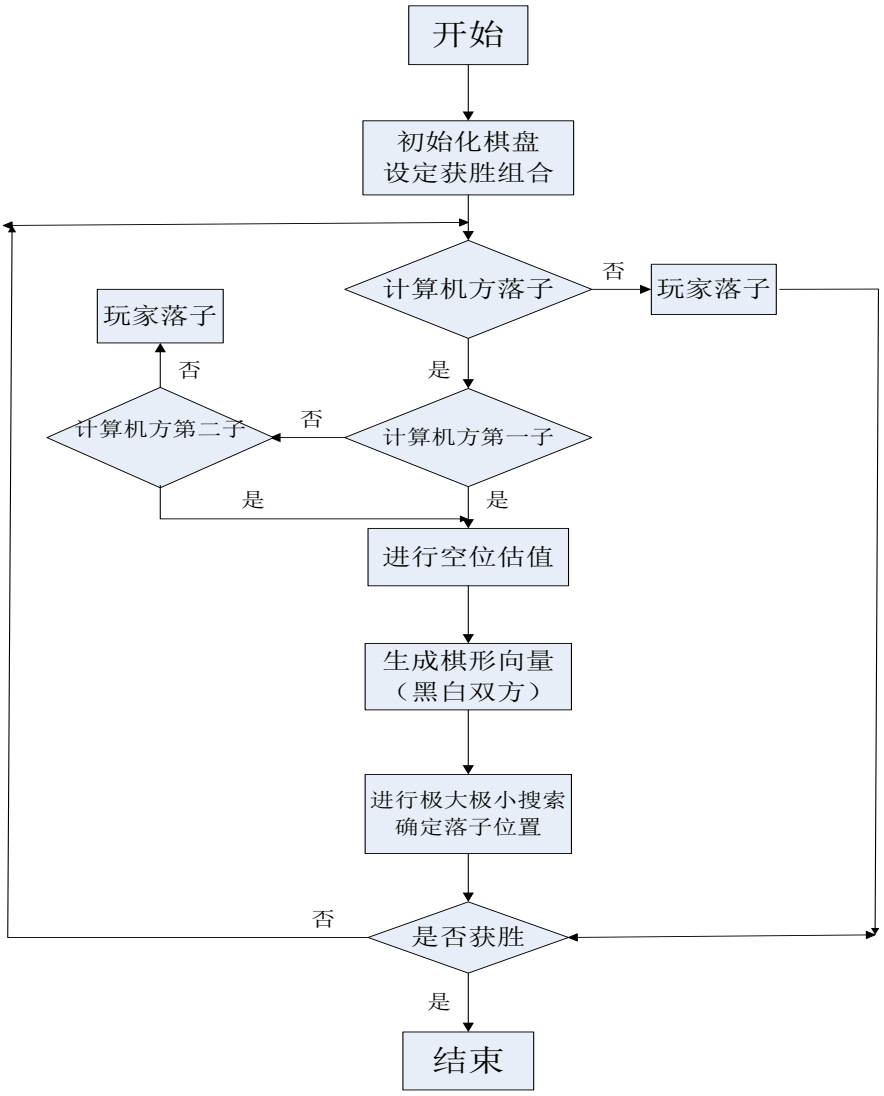

图 6.1 系统流程图

万方数据

6.2 Connect6 界面介绍

Connect6 的界面如下图 6.1 所示：

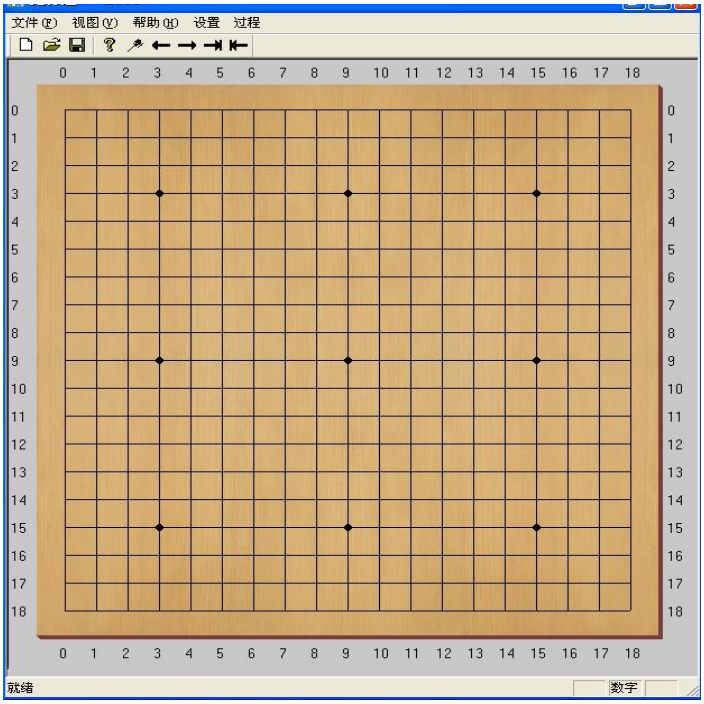

图 6.2 Connect6 界面

界面菜单栏中包括文件、视图、帮助、设置、过程菜单。该系统可以进行悔棋操作，灵活控制对弈过程中的各种情况。

6.3 实验方法

开发的六子棋博弈系统是不是性能更好，这一点需要通过实验来进行证明。我们设计了两套实验方案。第一套实验方案，人与机器对弈。曾经的“深蓝”因为打败了卡斯帕罗夫而名噪一时，也证明了机器终于可以和人进行智力的比拼。我们邀请六子棋爱好者，让其与 Connect6 进行对弈，记录比赛成绩，通过获胜概率来分析 Connect6 的性能。第二套实验方案，机器与机器对弈。将 Connect6 与花溪棋友进行对弈，采用人工方式进行落子位置传递。

6.4 实验结果

采用第一套实验方案时，我们邀请了两位六子棋爱好者，采用五局三胜制，分别与 Connect6 进行对弈。结果是 Connect6 大获全胜。对此次实验结果进行分析，一方面，我们邀请的是业余爱好者，可能在实战中缺少足够的经验；另一方面，说明 Connect6 在对弈过程中能进行正确的选择，才能在对弈过程中赢得全胜。此次实验证

万方数据

明：Connect6 的基本性能达到要求。

第二套实验方案是，Connect6 与花溪棋友进行对弈。采用两个评价指标来衡量系统的整体性能，一方面是系统在比赛中进行棋盘扫描，评估，搜索时所用的时间；另一方面是比赛过程中的获胜局数。图 6.2、6.3 是实验过程中 Connect6 和花溪棋友的界面截图。

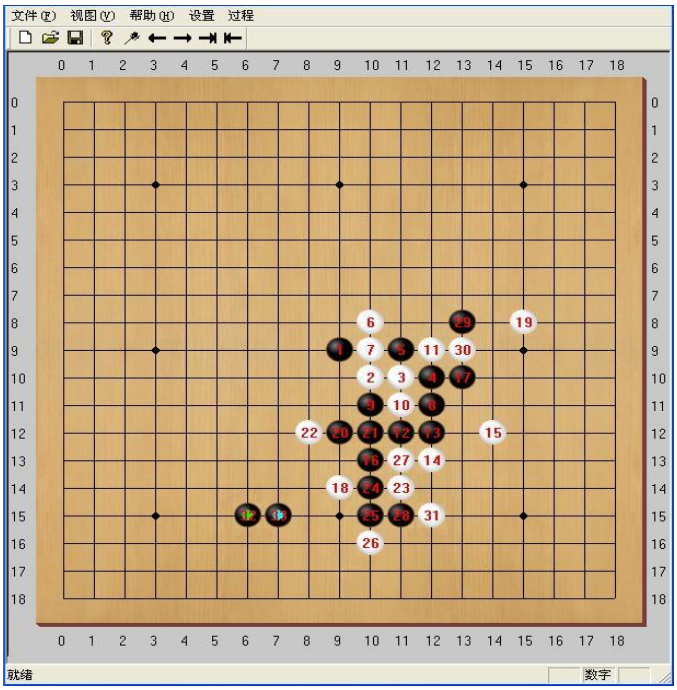

图 6.3 实验过程中 Connect6 的截图

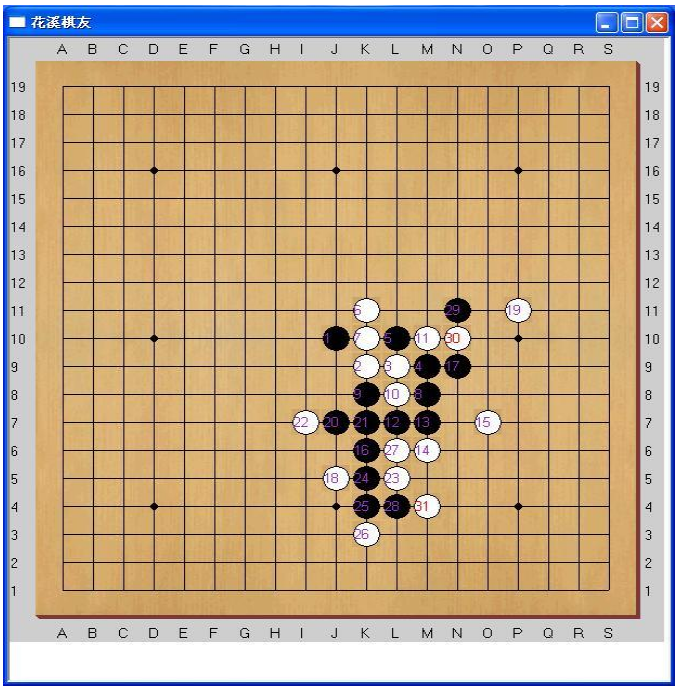

图 6.4 实验过程中花溪棋友的截图

万方数据

Connect6 与花溪棋友六子棋程序进行多次对弈后，对弈结果如表 6.1、6.2 所示，表 6.1 记录了每局对弈中所用的总时间，表 6.2 显示的是 Connect6 与花溪棋友比赛的成绩记录。

表 6.1 每局对弈中所用总时间

对弈双方 / 时间 / 对弈局数	Connect6	花溪棋友
对局 1（Connect6 为先手）	9 分 15 秒	8 分 27 秒
对局 2（花溪棋友为先手）	7 分 19 秒	8 分 30 秒
对局 3（Connect6 为先手）	6 分 12 秒	6 分 45 秒
对局 4（花溪棋友为先手）	7 分 20 秒	6 分 26 秒
对局 5（Connect6 为先手）	7 分 19 秒	8 分 32 秒
对局 6（花溪棋友为先手）	5 分 45 秒	6 分 56 秒
对局 7（Connect6 为先手）	6 分 34 秒	7 分 7 秒
对局 8（花溪棋友为先手）	7 分 17 秒	7 分 15 秒
对局 9（Connect6 为先手）	5 分 56 秒	6 分 19 秒
对局 10（花溪棋友为先手）	6 分 10 秒	7 分 12 秒
对局 11（Connect6 为先手）	5 分 19 秒	6 分 17 秒
对局 12（花溪棋友为先手）	8 分 30 秒	7 分 34 秒
对局 13（Connect6 为先手）	7 分 32 秒	8 分 17 秒
对局 14（花溪棋友为先手）	7 分 33 秒	7 分 14 秒
对局 15（Connect6 为先手）	8 分 18 秒	8 分 29 秒

万方数据

表 6.2 Connect6 与花溪棋友比赛记录表

获胜结果 / 局数 / 对弈双方	胜	平	负
Connect6	9	3	3
花溪棋友	3	3	9

对对弈结果进行分析，Connect6 与花溪棋友共进行 15 次比赛，Connect6 九胜三平三负，在对弈时间记录中，Connect6 有九局比赛所用时间短于花溪棋友。从整体数据来看，Connect6 略胜一筹，棋形估值方法结合空位估值方法的应用得到了很好的证实。

6.5 小结

本章主要介绍了六子棋计算机博弈系统，Connect6。Connect6 是综合前面几章介绍的方法开发出的六子棋计算机博弈系统，在与人类棋手进行对弈中，证实了系统的基本性能。在与花溪棋友进行比赛，证实了其具有在不同局势下的应变能力，能作出明智的决策，是一款性价比较高的六子棋软件。

万方数据

7 总结与展望

7.1 研究工作总结

六子棋是台湾吴毅成教授发明的，由于其出现时间较短，相关领域的研究工作都比较少。本文进行的研究工作如下：

① 计算机博弈系统性能与评估函数的选取有很大关系。一般博弈系统采用静态估值函数，该估值函数不能依据局势变化做出适时调整；在博弈树搜索过程中，将双方估值差值作为评判标准，而不是双方棋力所占百分百；与人类棋手思维模式存在“剪刀差”。所以选用二次估值方法进行棋局评估，该方法引入了局势因子，随着局势因子的变化，可以体现出棋局局势的不同需求，指导落子。

② 采用粒子群优化算法对选取的静态估值进行参数优化。粒子群优化算法与遗传算法有很多共同之处，初始化时两者都是随机生成初始种群，使用适应度函数来评价系统，并进行一定的随机搜索。粒子群优化算法是根据自己的速度来决定搜索[44]，没有遗传算法中的交叉和变异过程。在对空位估值中设定的参数进行优化后，证实该方法可以应用于棋类游戏的评估函数优化中。

③ 模拟人类思维模式提出空位估值方法。人类棋手对弈过程中，会思考空位置对当前局势可能产生的影响。在抑制对方局势的前提下，如果某空位置对我方有利，棋手会选择最为有利的位置落子。本文提出的空位估值方法就是实现了这一思想，通过多位棋类高手的经验，设定了五种空位的估值，该方法实验过程中得到了比较理想的效果。

④ 一般使用的估值函数都是对孤立棋子的评估，但是棋子与棋子之间由于位置的不同也会产生不同的影响，所以提出棋形估值方法。该方法利用的是棋子与棋子之间的位置关系，即棋型，进行估值。提出棋形向量的概念，即在已选棋型基础上，对当前棋面中的棋型进行量化得到的向量。将棋形向量与空位估值方法相结合，运用到系统中。

7.2 本系统目前存在的问题和不足

本系统与其他六子棋程序进行多次对弈，实验结果较理想。但目前仍存在以下不足：

① 利用棋形估值方法时，选用棋型个数有限，应当适当增加棋型。

② 评估函数由空位置的估值，棋形估值两方面组成，可以适当增加棋型距离之间的估值，使评估函数性能更好。

万方数据

7.3 后续工作

在已有的工作基础上，我将继续六子棋计算机博弈系统的研究工作。

① 将棋形估值方法做进一步的改进，添加棋型库，使系统能自动更新棋型。

② 增加定石库，初始状态根据定石库进行落子，使局势更有利于我方。

③ 六子棋规则是除了第一次落一子外，黑白双方轮流下两子。改进系统中的博弈树，使“两步”合并为“一步”。

7.4 展望

机器博弈被称为是人工智能的“果蝇”，其对人工智能的发展有很大的推动作用。机器博弈各项技术的研究成果有些也适用于其他方面。随着我国各高校、研究所以及业余棋类爱好者对机器博弈的重视，国内出现了机器博弈研究的热潮。六子棋由于其简单的规则，更高的公平性，在发明之初就受到大家的喜爱，相关领域的研究人员也在逐渐增加。借助对机器博弈的大力研究，人工智能也将得到前所未有的发展。

万方数据

致 谢

我的研究生生活即将结束，在论文完成之际，我要向所有关心、支持和帮助过我的老师、同学表示最诚挚的谢意。

衷心感谢我的导师韩逢庆教授，感谢您几年来对我的培育和教导，您渊博的专业知识，严谨的治学作风和宽厚正直的品格使我在为学为人之道上受益终身。韩老师治学严谨、待人真诚，他高屋建瓴的学术眼光、对事业孜孜不倦的追求和勤奋不辍的精神使我受益匪浅，是我终生学习的榜样，在此向恩师致以最诚挚的谢意和最美好的祝愿。

衷心感谢读研期间的各位任课老师给予的关心与帮助，我有幸能够聆听王越老师、李祖枢老师、杨武老师、张建勋老师、陈媛老师、王培荣老师等的授课，他们渊博的知识极大的拓展了我的知识视野和专业水平，在此向各位老师致以最诚挚的谢意！

衷心感谢我的师弟师妹，文成，赵文娟，龚钰梁。他们在本课题的研究工作方面，给予我多方面的帮助和协作，也正是由于他们的参与，和我们相互之间的讨论，才有了论文中新思想的提出。

在多年的学习生活中，除了家人的鼎力支持外，同学、朋友的鼓励和关怀也是我解决难题的动力，在此，向所有关心和帮助过我的领导、老师、同学、朋友表示由衷的谢意。

衷心感谢在百忙之中评阅论文和参加答辩的各位专家、教授。

李翠珠

二零一零年四月 于重庆理工大学杨家坪校区

万方数据

万方数据

参考文献

[1] 摩 尔 根 与 果 蝇 [EB/OL].[2008-01-06] http://basic.shsmuedu.Cn/jpkc/Marx_philosophy/yxyzx/12.ppt.

[2] 徐心和，邓志立，王骄等. 机器博弈研究面临的各种挑战[J]. 智能系统学报，2008，3(4)：287-293.

[3] 六子棋主页. http://www.connect6.org/.

[4] 徐心和，王骄. 中国象棋计算机博弈关键技术分析[J]. 小型微型计算机系统，2006，27 (6)：961-969.

[5] 董红安. 计算机五子棋博弈系统的研究与实现[D]. 山东师范大学. 2006，8.

[6] Yen S J, Chen J C, Yang T N. Computer Chinese chess[J]. ICGA Journal, 2004, (3)：3-18.

[7] Russell, Stuart J, Norvig Peter (2003),Artificial Intelligence：A Modern Approach (2nd ed.)[J], Upper Saddle River, NJ：Prentice Hall, pp.111-114.

[8] L.Ingber. Simulated annealing: Practice versus theory[J]. Mathl. Comput. Modelling, Vol.18 No.11, pp.29-57, 1993.

[9] Marsland T A. Computer chess and search[D].Edmonton: University of Alberta,1991.

[10] J. Schaeffer. Distributed Game-tree Searching[J].Journal of Parallel and Distributed Computing, Vol. 6,1989

[11] A.Plaat, J.Schaeffer, W.Pijls, A.de Bruin, Best-first Fixed-depth Minimax Algorithms[J]. Aritificial Intelligence,1996.

[12] David Kruglinski, Scot Wingo, George Shepherd. Programming MicrosoftVisual C++[M]. Sth Edition, Microsoft Press, 1999.

[13] Rivest, R.L.Intelligence, Game Tree Searching by MinMax Approximation[J]. Artificial 1998, Vol.34, No.1

[14] G.C.Stockman, A Minimax Algorithm Better than Alphabeta[J].Aritificial Intelligence, Vol.12, No.2, 1979.

[15] H.J.Berliner, C.McConnell, B*Probability Based Search[J], Artificial Intelligence, Vol.86, No.1, 1996.

[16] L.Victor Allis:"Searching for Solutions in Games and Artificial intelligence " Part I.PhD Thesis, September 1994, ISBN 90-9007488-0.

[17] L.Victor Allis:" Searching for Solutions in Games and Artificial intelligence" Part II.PhD Thesis, September 1994, ISBN 90-9007488-0.

[18] Martin Schmidt: "Temporal Difference Learning and Chess" Part I. Aarhus University.

[19] 朱全民, 陈松乔. 五子棋算法的研究与思考[J]. 计算技术与自动化, 2006, 25 卷(2期):71-74.

[20] 林尧瑞, 马少平. 人工智能导论[M]. 北京:清华大学出版社. 2002, (9).

[21] T.Anthony Marsland. A review of game-tree pruning[J]. ICCA Journal, March1986, Vol 9, No1:3-19.

[22] Hall M.R. and Loeb D.E.(1992).Thoughts on Programming a Diplomat.Heuristic Programming in Artificial Intelligence 3:thethird computer olympiad (eds.H.J.Van denHerik and L.V. Allis), pp.123-145. Ellis Horwood Ltd, Chichester.(6)

万方数据

［23］Owner’s Handbook. Chess Program Design[M].Chap.6,Turbo Pascal GameWorks,Ver.4.0, Borland, 1987.

［24］Knuth D E, Moore R N. An analyze of alpha-beta pruning[J].Artificial Intelligence, 1975, 6: 293–326.

［25］Jonathan Schaeffer.The history heuristic and alpha-beta search enhancements in practice. IEEE Transactions on Pattern Analysis and Machine Intelligence,November 1989, pami-11(1): 1203-1212.

［26］David J Slate, Laweence R Atkin.Chess4.5-the northwestern university chess program, chess skill in man and machine[M].New York,Springer-Verlag,1997:82-118.

［27］Hermann Kaindl, Reza Shams, Helmut Horacek. Minimax search algorithms with and without aspiration windows[J].IEEE Transactions Pattern Analysis and Machine Intelligence, December 1991,13 (12):1225-1235.

［28］张玉志. 计算机围棋博弈系统[D]. 北京:中国科学院计算技术研究所 1991.

［29］黄晨. 棋类游戏中的先行权[J]. 智能系统学报, 2007, 2 卷(3 期):91-94.

［30］Knuth, D.E.and Moore, R.W.(1975). An Analysis of Alpha-Beta Pruning[J]. Artificial Intelligence, Vol.6,No.4,293-326.

［31］I-Chen Wu, Dei-Yen Huang, and Hsiu-Chen Chang.CONNECT6. Hsinchu,Taiwan.2005,12.

［32］王骄, 王涛, 罗艳红, 徐心和. 中国象棋计算机博弈系统评估函数的自适应遗传算法实现[J]. 东北大学学报, 2005, 26卷(10期):949－952.

［33］陆汝钤. 人工智能[M]. 北京:科学出版社, 1995.

［34］Law A M, Kelton W D. Simulation Modeling and Analysis, Third Edition [M]. McGraw-Hill Press, 2000: 3-13.

［35］张赜. 计算机中国象棋博弈中的二次估值方法及其优化的研究[D]. 东北大学, 2006, 2.

［36］Lorenz D , Markovitch S. Derivative evaluation function learning using genetic operators [A] . Proceedings of the AAA I Fall Symposium on Games : Planning and Learning [C]. New Carolina, 1993. 106 - 114.

［37］Holland, Adaptation in Nature and Artificial Systems. MIT Press, 1992.

［38］De Jong K A, An Analysis of Behavior of a Class of Genetic Adaptive Systems. Ph.D Dissertation, University of Michigan, No.76-9381, 1975.

［39］Godberg D E, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, 1989.

［40］J.Kennedy, R.C.Eberhart. Particle swarm optimization [A].Proceedings of the IEEE International Conference on Neural Networks[C].1995:1942-1948.

［41］李果. 六子棋计算机博弈及其系统的研究与实现[D]. 重庆大学，2007, 4.

［42］S.S.Rao.Optimization Theory and Application.New Delhi: Wiley Eastern Limited, 1984.

［43］张荣沂. 一种新的集群优化算法:粒子群优化算法[J]. 黑龙江工程学院学报(自然科学版)． 2004, 18(4):131-135.

［44］R.C.Eberhart, Y.H.Shi. Particle Swarm Optimization: Developments, Applications and Resources [A]. Proc. Congress on Evolutionary Computation 2001. Piscataway, NJ: IEEE Press, 2001: 81-86.

［45］Y. H. Shi, R. C. Eberhart. Empirical Study of Particle Swarm Optimization [A].Proceeding of Congress on Evolutionary Computation [C]. Piscataway, NJ:IEEE Service Center, 1999:1945-1949.

万方数据

［46］M. Clerc. The Swarm and Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization [R]. Proceedings of Congress on Evolutionary Computation, Washinton D C, 1999:1951-1957.

［47］Angeline, P.Evolutionary optimization versus Particle swarm Philosophy and Performance Differences[C].In:Evolutionary Programming,1998:601-610.

［48］Angeline P. Using selection to improve particle swarm optimization [A]. Proceedings of IJCNN 99 [C].Washington, USA, 1999. 84~89.

万方数据

万方数据

个人简历、在学期间发表的学术论文及取得的研究成果

姓名	李翠珠	出生日期	1985 年 9 月 12 日	籍贯	山东省聊城市
获得学士学位时间、学校		2007 年 7 月 海军航空工程学院			
现所学学科、专业		工学　计算机应用技术		入学时间	2007 年 9 月

学习（大学以上）及工作经历

年 月— 年 月	就学的学校、专业/工作单位、职务
2007 年 9 月-2010 年 7 月	重庆理工大学　计算机应用技术专业
2003 年 9 月-2007 年 7 月	海军航空工程学院　计算机科学与技术专业

在学期间发表的学术论文及取得的研究成果（包括鉴定项目、获奖、专利）

序号	论文或成果、专利名称	全体作者 （按顺序排列）	发表刊物或鉴定单位 或获奖名称、等级或专利类别	时间
1	六子棋博弈的二次估值	韩逢庆，李翠珠，李为	重庆工学院学报（自然科学版）	2009 年 11 月

在学期间尚未发表但已被录用的学术论文

序号	论文名称	全体作者 （按顺序排列）	拟发表刊物	预计发表时间

万方数据

否

万方数据