六子棋博弈系统中机器学习算法设计与研究

于江明,袁斌杰,何灿辉,段琢华
(韶关学院 计算机科学学院, 广东 韶关 512005)

摘要:本文针对我校参加全国大学生博弈大赛完成的六子棋博弈系统,首先简要介绍本系统的一些关键构成.并重点论述了对现有的六子棋系统一个更进一步的优化,在 MTD(f)算法上增加自适应宽度调整和在评估函数上增加 TD_BP 算法自我学习方法,使系统"思考"性的能力更加强.

关键词:六子棋;计算机博弈;评估函数;开局库;TD_BP 算法
中图分类号:TP312;TP183　　**文献标识码**:A　　**文章编号**:1007-5348(2014)10-0017-06

六子棋是一种黑白棋游戏,是一种双人对弈的游戏,对战双方各执一种颜色的棋子,对弈过程中,一方先于另外一方先连成六颗子获胜,所以叫六子棋,六子棋的棋盘有的大小可以是无限大,但一般都以 19 乘以 19 的方格作为棋盘.现在研究六子棋的领域越发成熟,在网络上出现好多六子棋博弈的系统,例如文献[1]提到的深思、文献[2]提到的深蓝、网络上的 X6 等,这些系统都有自己的特色,它们的构成都大同小异,但他们都没有使用自我学习的算法.本文主要简单论述以下六子棋系统的构成,然后针对六子棋博弈系统增加自我宽度调整和自我学习进行一个比较详细的论述.

1　六子棋系统构成

1.1　棋盘和棋局的表示

在六子棋计算机博弈系统中,只有黑白两种颜色的棋子,棋盘的表示有数组棋盘和比特棋盘,但由于六子棋的棋盘比较大,而且棋类没西洋跳棋、中国象棋那么复杂,所以采用了数组棋盘.六子棋的棋盘看起来像一个二维数组,因此用一个二维数组表示六子棋棋盘.如代码 1 所示,首先枚举棋子的类型,分别是黑棋、白棋、空子,这样就可以用不同的数值表示棋盘上的棋子状态,然后定义一个 1919 的六子棋棋盘,关键代码为:

```
#define ROW_MAX 19
#define COL_MAX 19
enum CellType {// 棋子类型
    CELL_TYPE_BLACK,
    CELL_TYPE_WHITE,
    CELL_TYPE_EMPTY,
    CELL_TYPE_NOT_A_CELL};
typedef CellType Board[ROW_MAX][COL_MAX];// 定义的二维数组棋盘.
```

1.2　着法生成

着法生成是指根据当前棋局产生所有有效的着法.本六子棋采取常用的棋盘扫描法,在六子棋的规则

收稿日期:2014-06-11
基金项目:韶关学院大学生创新创业训练计划国家级立项建设项目(201310576-007).
作者简介:于江明(1963-),男,湖南邵阳人,韶关学院计算机科学学院副教授,主要从事知阵论及计算机算法研究.

内和棋盘范围内,进行不断的扫描,从而约束条件和下子情况.

1.3 MTD(f)算法

MTD(f)算法代码如代码 2 所示,该算法在文献[2]里面负极大值算法(Negamax)的基础上设定了搜索的边界,一个是上边界,另外一个是下边界 ,还设置了一个猜测值.文献[3]提到过 MTD(f)算法的效率主要是来自于空窗搜索的高剪枝率,目的是通过猜测值修改上下边界来逼近真实值,当下边界的值大于或等于上边界时,就可以结束搜索返回了.如果猜测值跟真实值越接近,那么搜索的次数越小,最好的情况是刚开始的猜测值就是真实值,进行两次的空窗搜索两个边界相等就能结束搜索了.因为 MTD(f)算法搜索过程中可能会重复一些节点,因此结合置换表使用的话更高效率.MTD(f)算法可以调整精度,增加一个步进值使搜索的幅度增大,这个步进值可以是固定值,也可以是一个动态值,这个值要根据博弈系统的评估函数来调整,关键代码为:

```
int MTDf (bool 棋方, int 深度)
{
int 真实值, 猜测值;
int 上边界 = 正无穷;  int 下边界 = 负无穷;
while( 下边界 <上边界 )
{
if ( 猜测值 == 下边界)
真实值= 猜测值 + 1;
else
真实值 = 猜测值;
猜测值= 负极大值算法(棋方, 深度, 真实值–1 , 真实值);
if (真实值 < 当前猜测值 )
上边界= 当前猜测值;
else
下边界= 当前猜测值;   }
return 猜测值; }.
```

1.4 评估函数

上述的算法里面,都有用到评估函数值,评估函数对于一个博弈系统来说占了一比较重要的地位,评估函数的值可以决定所选择着法的优劣程度.现在博弈系统一般取用的是静态评估函数,因为静态评估函数是比较容易实现,而且根据各种文献[4]的经验可以很快的调整它值.评估函数表示了己方的有利程度的有利成度,对一般的,取胜局面的评估函数的值为+∞,失败局面的评估函数值–∞.评估函数值的大小对取胜的影响很大.参考文献[5]的评估函数中棋型的分类内容本系统的分为 C6、DW、L5、D5、L4、D4、L3、S3、D3、L2、D2、L1、D1、V14 种棋型,这样就可以根据棋局算出评估函数值,而且根据评估函数值我们可以大概看到当前的棋局对己方是否有利.本系统采用数据匹配的方式,把各种棋型的评估值放到 txt 文件里面,运行时,会在系统中把数据存到一个表上面去,利用二进制进行一个匹配棋型,不论是白起还是黑棋都一样.因为是一文件的方式读入,所以维护里面的参数也比较方便,通过实际的对弈可以进行人工的修改.

还有一种静态评估方法是基于文献[4]所提出的“路”,把它做成 Segment 表,连续六格表示一个棋型,表中有六种情况,所以这个静态评估值有六种值,跟第一种评估函数不一样.假设第一种评估的为 Evalua-

tor,第二种为 segmentScore,那么总的静态评估值 val 可以表示成公式:

$$val=A\times Evaluator+B\times segmentScore \quad (1)$$

公式(1)中 A 和 B 的值是两种评估函数在博弈系统中所占的比重,也就是权重,我们设置它们的值都为 1.也就是说两种评估都占了相同的地位.

2　六子棋的优化方法

2.1　开局库的引用

文献[2]提到常用的定式是一些常用的,但这些远远不够用,有些好的棋局需要计算机算上好长一段时间,几小时或者几天都有可能.利用开局库来存储这个好的定式.如果当前棋局的局面在开局库有的话,就可以直接提取出来而不用搜索引擎进行树的搜索,这样可以避免由于深度不够造成下子的失误,同时也增加了搜索的效率.

本博弈系统采用数据库的形式,使用了 SQL 的开放数据库互连(ODBC).开局库存储的内容是棋局的棋子坐标.对于当前局面,首先连接数据库,然后向数据库发送语句查询信号,查询数据库中有没有对应的开局,有的就返回对应着法数据,对数据转换成坐标.

在数据库中创建两个表,一个是黑棋着法的数据库,另一个是白棋着法的数据库,两个表内存的定石除了一些文献[2]说到常用的开局,还有通过自己跟别人对弈得到的一些比较好的棋局.黑棋跟白棋都可以算到第六手.

2.2　自适应确定宽度值

上述的 MTD(f)算法,利用迭代深化,用一个时间的限制来规定搜索引擎搜索的深度,在时间内搜到好的棋局就回返回, 搜不到好的棋局就会一直搜到时间限制的结束再从搜出的这么多节点当中挑出最好的那个返回.算法里面还设置了一个步进值,例如下述代码所示这个步进值是修改搜索上下边界的宽度,这个值是协同评估函数值的,精度高的评估函数,颗粒小的话需要多次搜索才会接近真实值.设置步进值可以适当减少搜索次数,这样使搜索的效率更高一些,关键代码为:

```
int MTDf ( int 原始猜测值, int 深度)
{
While(深度<最大深度||时间>最大时间){
int 真实值,     猜测值=原始猜测值 ;
int 上边界= 正无穷;   int 下边界 = 负无穷;
int 步进值 = 设定固定值;
While(深度<最大深度){
while( 下边界< 上边界)
{ 步进值 = 上一轮搜索中最好的评估值;
if ( 猜测值 == 下边界 )
窗口值 = 猜测值 + 1+步进值;
else
窗口值= 猜测值–步进值;
猜测值 = 负极大值算法( 窗口值 – 1, 窗口值,深度 );
if (猜测值 < 窗口值 )
上边界= 猜测值 ;
```

else

下边界= 猜测值;

return 猜测值; }.

2.3 自适应宽度的调整的取值分析

自适应确定宽度值的取值怎么取呢？在本博弈树中，我们利用一个比较大的数 1 000 000 000 代表 ∞，这个也是评估函数的极限值，因此 step 值的取值范围是 0~1 000 000 000,但这个 step 值不能取太小，太小的话搜索引擎需要搜索多次，太大的话虽然搜索很快，但可能会错过了一次比较好的着法，因此效率和着法质量是一个反比的关系. 取宽度值为 0、150 000 和动态值三个值进行多次测试， 整理数据分析得出，当 step 值为 0 时，搜索的次数明显要给 step 的值为固定值或动态值的要多.Step 值为动态值的时候，搜索的次数大部分是给 step 值为 0 或固定值小.

3 TD_BP 自我学习评估

3.1 TD_BP 自我学习的介绍

传统的搜索算法一般是采用静态评估函数结合的博弈系统， 为了改进系统， 增加 TD_BP 自我学习系统，这是一个增强学习的系统，文献[6]所提出的 BP 算法分为有导师和无导师的的学习方法，如果要采用导师学习的方法，需要手工输入棋局和对弈结果，需要费很大的人力，所以是不合理的.所以本系统采用无导师的非监督的学习方法，它能从环境的反馈学习，不断尝试进行动作选择，并从环境的反馈结果调整评估值，等到最优策略.

3.2 评估函数 $P(s,a)$

对于前面提到过的静态评估，参考文献[6]对于局面 P 的估值过程都可以形式上的描述为一个线性的函数 $V(p)$:

$$V(p)=W\cdot F^T=w_1\times f_1+w_2\times f_2+\cdots+w_n\times f_n. \tag{2}$$

其中，W 是常量分量构成的权重向量$<w_1, w_2, w_3, \ldots, w_n>$，$f$ 是由特征值构成的 n 维特征向量$<f_1, f_2, f_3,\ldots,f_n>$.对于上述的静态评估权重向量的取值都是 1.

而对于自我学习系统的评估函数递归公式为 $P(s, a)$:

$$P(s,a)=r(s,a)+\gamma\cdot\max\ P(\delta(s,a), a'), \tag{3}$$

上式 s 为每个局面的状态，a 为动作，$P(s, a)$为评估值.

由公式(3)P 值函数可以被看做是基于单步前瞻的训练值，可变为：

$$P^{(1)}(s_t, a_t)=r_t+\gamma\cdot\max\ \hat{P}(S_{t+1}, a). \tag{4}$$

基于 n 步前瞻的回报 $P^{(n)}(s_t, a_t)$:

$$P^{(n)}(s_t, a_t)=r_t+\gamma\cdot r_{t+1}+\cdots+\gamma^{(n-1)}\cdot\max_a\ \hat{P}(S_{t+1}, a). \tag{5}$$

在公式(3)上增加一个 $0\leqslant\lambda\leqslant1$ 合并不同前瞻距离的评估值 $P(s_t, a_t)$:

$$P^{(\lambda)}(s_t, a_t)=r_t+\gamma[(1-\lambda)]\max_a\hat{P}(s_t, a_t)+\lambda\cdot P^{\lambda}(s_{t+1}, a_{t+1}). \tag{6}$$

自我学习算法 $TD(\lambda)=P^{\lambda}(s_t, a_t)$，对于博系统，上式(6)中，当 $\lambda=0$ 时，$P^{(1)}$,只考虑单步差异的话，对估值没什么影响，需要考虑更远的前瞻，这样对当前的估计更准确，训练更好.

3.3 BP 网络的结构设计与神经元网络的参数选择

映射所有连续函数的可以采用单隐藏层的前馈网络，不连续函数的学习时需要两个隐藏层，设计中一

个隐藏层的节点数很多时,往往网络性能还得不到很好的改善,就会再增加一个隐藏层.本系统采用输入层,隐藏层,输出层 3 层的网络结构.

用三层的神经网络作为六子棋的评估函数, 网络相邻之间的传输函数对网络的的拟合性能影响很大.转换函数可以选线性函数或非线性函数.使用线性函数的话,评估函数会存在很大的不足,是因为六子棋的局面错综复杂. 因此构造网络采用非线性函数 S 型 sigmoid 函数. 任意逼近任何连续函数需用两层 sigmoid 函数,反向传播误差时更容易.

本博弈系统中应用神经元网络必须提取棋局的的局面特征作为神经元网络的输入, 所选的棋型特征必须符合以下原则:

(1)对结果影响比较大而且方便取出的的局面特征.

(2)特征之间互不影响或影响性很小.

上述的静态评估中已经提到过把六子棋的棋型分为 C6、DW、L5、D5、L4、D4、L3、S3、D3、L2、D2、L1、D1、V 共 14 种,但对于特征原则的选取,只需要其中的 8 种,对于神经元的输入,必须要对棋局特征进行一个编码.例如局面中的'L3'棋型用 4 个输入单元表述,4 个输入单元取值用 4 维向量表示,(1,0,0,0)表示黑方'L3'棋型的个数为 0 个时;(0,1,0,0)表示黑方'L3'棋型的个数为 1 个时;(0,0,1,0)表示黑方'L3'棋型的个数为 2 个时;(0,0,0,1)表示黑方'L3'棋型的个数为 3 个或 3 个以上时;类似的,要描述黑棋需要 32 个输入单元,一个局面总共需要 64 个输入单元.

神经元网络有一个输出单元,表示当前局面的估值,输出越越向 1,局势越对黑方有利,输出越越向 0,局势对白方有利.根据得到的输出单元的值进行选择着法,生成最优着法.

3.4 隐藏层节点数的选取

隐藏层节点的数目会影响到神经网络.隐藏层节点过少,学习就会受到限制,这样就看不出规律来了;隐藏层节点过多,会使训练时间增加,而且可能会将样本中非规律行的内容存储进去,降低了学习的能力.本博弈系统选取了 18 个隐藏单元.最终确定本系统 BP 神经元网络(见图 1)输入层为 64 个单元,输出层为 1 个单元,隐藏层为 18 个单元.

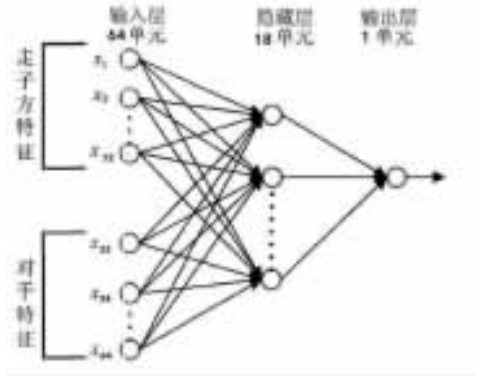

图 1　BP 神经元网络的结构设计

3.5 神经网络权值 W 的修改和参数 a 和设置对系统的影响

在神经网中主要是通过网络权值 W 的修改达到学习的目的,因为本系统采用的是无监督学习方法,所以不要输入样本和输出数据,因此 W 权值修改的公式可以表示为:

$$\triangle w_t=a(P_{t+1}-P_t)\sum_{k=1}^{t}\lambda^{t-k}\nabla_w P_t\,, \tag{7}$$

式(7)中 λ 为衰减因子,a 是学习速率,$P_{t+1}-P_t$ 是相邻的两个预测值,$\nabla_w P_t$ 是 P_t 对 w 每个分量的偏导数.

在 TD_BP 算法中,a 和是常量,a 是神经元反向传递学习的学习速率,$a \in [0, 1]$,a 的值过小,神经元的学习就会过于缓慢,很难得到提升到最大的性能,a 的值过大.尽管学习速度好快,但却降低网络的最大性能,难于拟合到真实的局面估值函数.是神经网络的衰减因子,$\lambda \in [0, 1]$,λ 的值越接近 0,表示越远离当前时刻 t 的预测,对时刻 t 的预测的影响就越小.因此对于参数 a 和 λ 的取值要经过测试取适当的值.各自取不同的值进行测试,参数 a=0.3 和 λ=0.7 时胜率提高比其他参数好.

增加 TD_BP 算法的本系统通过不断的跟其他六子棋系统不断的对弈进行自我反馈, 动态调整评估函数使搜索引擎有更好的寻优,相对于没有增加 TD_BP 算法的本系统胜率会更高.

4 结语

阐述了六子棋博弈系统的几个核心部分:棋局表示、着法生成、搜索引擎、评估函数.搜索引擎使用了 MTD(f)算法,在 MTD(f)算法的基础上加入自适应宽度的调整,有效地减少搜索时间,进一步提高搜索效率.在评估函数的基础上加入 TD_BP 的自我学习评估.增强系统“思考”的能力,能更好的提高棋子的棋力.

参考文献:

[1]李果.基于遗传算法的六子棋博弈评估函数参数优化[J].西南大学学报:自然科学版,2007,29(11):138-142.
[2]闵文杰.六子棋计算机博弈关键技术研究[D].重庆:重庆交通大学,2010.
[3]陈其.PC 游戏编程-人机博弈[M].重庆:重庆大学出版社,2002.
[4]陈光年.基于智能算法的六子棋博弈行为选择的应用研究[D].重庆:重庆理工大学,2011.
[5]徐长明.基于连珠模式的六子棋机器博弈关键技术研究[D].沈阳:东北大学,2010.
[6]李新星.六子棋中基于 BP_TD 学习的局面估值方法研究[D].沈阳:东北大学,2009.

Machine learning research and design of algorithm Connect6 game system

YU Jiang-ming , YUAN Bin-jie , HE Can-hui , DUAN Zhuo-hua
(Institute of Computer Science, Shaoguan university, Shaoguan 512005, Guangdong, China)

Abstract: This article is aimed to analyze Shaoguan University's students who participatd in the national Computer Games finished. It briefly analyzed the system structure and some key opponents with a focus on the Connect6 for a further optimization. It also increases adaptive width adjustment on the MTD (f) algorithm and evaluation function to increase TD_BP on self-learning algorithm which would strengthen system's "thinking" ability.

Key words: Connect6;computer game;evaluation function;opening data;TD_BP algorithm

(责任编辑:欧 恺)