算法课报告 1

61516220
卞悠悠

本次算法课要求完成一个五子棋 AI，我目前的思路是这样的：

首先我们要学会防守，我的想法是对于棋盘上的各个空位，统计它的 8 个方向，看看颜色相同的棋子连成线的个数，对每个位置进行遍历，根据个数给每个位置打个分 X（每个位置的分数由下了这步棋后连成的线中棋子的个数 n 以及可能连成的线的条数来决定。）

If n>=4 ,X+=10000;

If N=3 ,X+=4000;

If N=2 ,X+=2000;

If N=1 ,X+=500;

对于黑子和白子，我们各计算一遍，一个点下黑子时的分数记为 x1[i][j]（i，j 为其坐标），下白子的分数为 x2[i][j]，只要下在分数最高的地方即可。

再考虑下进攻：比如 AI 是下黑子，那么白子评分高的地方就是需要防守的地方，而黑子评分高的地方就是适合进攻的地方。

如何决定此时防守还是进攻呢？当 x2>=x1 时，我们就选择防守，反之则进攻。

这样基本的逻辑就完成了，接着我们会想办法提高这个 AI 的判断能力，我想到的办法就是尽可能多得把往后几步的情况推理出来，即得出它的博弈树。这种树同样需要一个打分机制来决定那条路比较好。如果前瞻步数无上限，那么可以假设叶字节点的评分标准是：AI 赢了得 1 分，输了得-1 分，平手得 0 分，但因为前瞻步骤有限，我们不能看到结局，只能看到几步后可能出现的各种情况，故我们还需要设计一个不同的评价函数，它决定了 AI 的思考能力（由此我们才能由可能情况推出前几步的得分）。

关于打分我们也需要分为两类：（运用部分之前的那个打分标准）

第一类是原本连成了一行（如图所示）（1 为已有的棋子）

1	1	1	1						

这种情况（A 情况）评分规则如下：（五个子连着简称五子）（情况指落子之前的情况）

情况	四子	三子	二子	一子
得分	100000	40000	20000	10000

如果其中一端有子（如图），则得分除以 10（除了四连的不受该条影响）

0	1	1	1	1	（下的位置）

第二种情况如下图：

1	1	（要下的位置）	
	1		
1			

这种情况下打分标准比较复杂：如下

相隔 45 度/90 度/135 度		相隔 180 度	
情况	分数	情况（两侧棋子相加的个数）	分数
双四子	1000000	四子	1000000
双三子	1000000	三子两端无子	1000000
三子和二子（三子的另一端有子）	1000	三子一端有子	1000
三子和二子（二子的另一端有子）	100000	三子两端有子	1
三子和一子（三子	50000	二子两端无子	10000

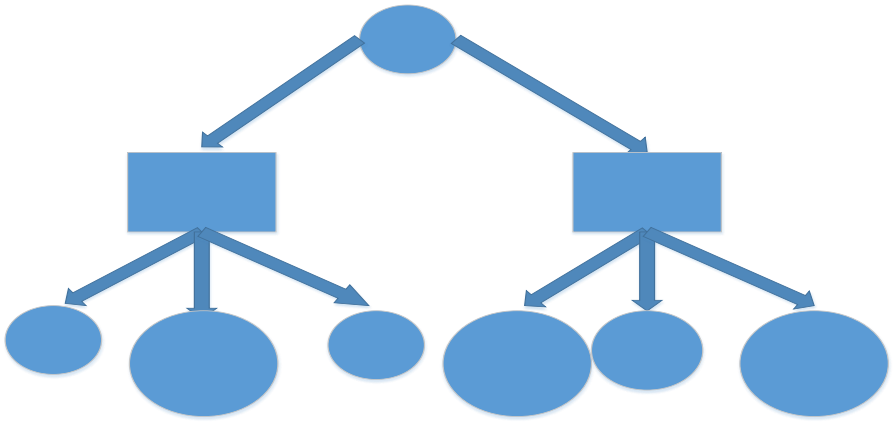
的另一端有子）			
三子和一子（一子的另一端有子）	100000	二子一端无子	100
二子和二子（）	5000	二子两端有子	1
二子和二子（一端有子）	2000		
二子和二子（两端有子）	100		
其余情况	1		

相隔 180 度情况如图：

1	1	（下的位置）	1

根据得分来决定如何下棋的方法：（假设我们的 AI 能够预见到未来三步）

假设圆圈是 AI 下棋的步骤，方块是玩家下的，那么在可前瞻到的圆圈处根据上面的打分情况我们会得到一个分数，方块步的玩家可以根据圆圈步的分数选择一个最小的分数，最顶层圆圈步的 AI 再选择方块步最大的，由此递推可以得出判断。



在两种情况同分的情况下进行随机选择。

说明 1：考虑到实际情形，三串子或者更多串数的子聚焦到一点的情况应该几乎不可能出现，比如如图所示情况：

1	1	1	1	（你现在可以下在这里）
			1	1
		1		1
	1			1

说明 2：在初始的时候，如果是玩家先落子，则 AI 随机落子在其周围。

Online 测试问题：我下了胡黛琳同学发的那个网站上的前几个但好像有两个报出来有毒，有个打开来只有个 exe 运行下啥都没有。

可能的改进方向：这个评分方案的科学性可能比较有限，是凭我自己对棋局的情况了解进行主观打分的，如果老师感觉总体思路可以的话下一步会读一些相关文献寻找出更加可靠的打分方法加以实现。同时，这个方法的时间开销较大，只能通过减少前瞻步骤来缩短 AI

反应时间，因此也需要读一些相关文献寻找算法加以利用。
不足之处：这个代码的水平是基于我对五子棋这个项目自己玩的时候的理解的，换言之在完全理性的情况下它的水平不可能超过我本身（当然它的优点是它不会犯错）。
代码：目前完成了情况一的判定，想让老师先看下思路可不可行再往下写。

```
int pingfen1()
{
    int a[15][15];
    int i, j,k,counts1=0,counts2=0,count, counth1 = 0, counth2 = 0,fen=0;
    for (i = 0, i <= 14, i++)
    {
        for (j = 0; j = 14; j++)
        {
            for (k = j, k <= j-4, k--)
                if (
                    int a[i][k] = 1)
                {
                    counts1++;
                }
                else { break; }
                for (k = j, k <= j + 4, k++)
                    if (
                        int a[i][k] = 1)
                    {
                        counts2++;
                    }
                    else { break; }
                    if (counts1 <= counts2) { count = counts2; }
                    else { count = counts1; }

        }
    }
    for (i = 0, i <= 14, i++)
    {
        for (j = 0; j = 14; j++)
        {
            for (k = i, k <= i - 4, k--)
                if (
                    int a[k][j] = 1)
                {
                    counth1++;
                }
                else { break; }
                for (k = j, k <= j + 4, k++)
                    if (
```

```
                    int a[k][j] = 1)
                {
                    counth2++;
                }
                else { break; }
                if (counth1 <= counth2&&counth2>count) { count = counth2; }
                else if(counth1>counth2&&counth1>count){ count = counth1; }

            }
        }
        if (count == 4)(fen += 100000);
        else if (count == 3)(fen += 40000);
        else if(count==2)(fen += 20000);
        else if (count == 1)(fen += 10000);
        return count;
}
```