**2024年春季学期**

**数据结构课程设计赛道B实验报告**

张凯玄1 王峰宁1 刘张弛1

软件学院2022级10 班

# 分工与合作

三人的工作占比基本是均等的。有一些诸如调参，测试，搜集文献等工作在代码中不能直接体现，但这些工作同样重要。

张凯玄：负责评估函数中第二次评估算法（基于路的评估）与评估函数中第三次评估算法（基于棋形的评估）的部分内容，以及博弈树+Alpha-Beta 剪枝的初步建构。测试 bot 与校正权值。

王峰宁：负责我方两步落子的总体价值评估逻辑，即我方的模拟棋步SimuEvaluate1和敌方模拟棋步 SimuEvaluate2；负责博弈树+Alpha-Beta 剪枝内容的完善与最终确定。测试 bot 与校正权值。

刘张弛：负责第一次评估算法First\_Eva\_BasedNUM 及其附属函数，负责评估函数中基于棋形评估的用于实现三次价值评估的getValue算法。测试bot与校正权值。

# 算法思想

## 总体思路

**三次估值函数+模拟棋步+博弈树+Alpha-Beta 剪枝**

**通过第一次评估来粗略确定棋盘上各可能落子位置的价值，并将各可能落子位置存入按照价值降序排列的类容器中，利用博弈树来从此数据结构中依次取落子位置，然后基于我方两步落子的价值评估逻辑，通过二次与三次价值评估确定出对我方最有价值的两步落子。**

模拟与剪枝的具体实现：

通过博弈树模拟我方两步落子，基于我方两步落子的价值评判逻辑，调用二次与三次评估函数计算出这两步落子的一次价值（对于我方来说的进攻价值与对于我方来说的防守价值以及对于敌方来说的总体价值）。此时，我方这两步落子已经落在棋盘上，基于此基础，进行敌方的模拟两步落子，基于敌方两步落子的价值评判逻辑，调用二次与三次评估函数计算出敌方这两步落子的具体价值，确定为二次价值，不断从降序价值类容器中寻找敌方的两步落子并确定价值，力求找到一个最大值。最终在所有二次价值中确定一个最大值（敌方这两步落子是对敌方来说最好的一个场面），用一次价值（我方所获价值）-二次价值（敌方所获价值）得出一个最终价值，也就是我们这两步落子的最终价值，如果这个最终价值小于我们之前存在的一个最终价值，则说明如果我方下这两步，敌方总能找出另外两步，使得我方处于一个不如下之前的最终价值对应两步落子位置的一个境地，这时就直接剪枝，舍弃这两步。否则，我们就把最有价值的两步更新为这两步，因为无论下一步敌方怎么下，我们下这两步总会比之前处于一个更好的境地。重复此过程，最终确定出对我方来讲最有价值的两步落子确定为我们这一回合的两步落子。

每一步落子通过三次评估来确定价值：

基于局面的第一次估值->初步确定可供后续选择的位置。

基于路的第二次估值（基于模拟思想）->计算最终最优解。

基于棋形的第三次估值->与路估值同时进行，辅助判断与决策。

我方执黑白色分别采用不同的逻辑：

若我方为黑，则以贪心思想为主，在博弈树中浅层递归后就退出，优先保证己方价值，力求在比赛取得先机，牵制对手。

若我方为白，则在博弈树中模拟8 颗棋子（ 4 颗我方，4 颗敌方），经过Alpha-Beta 剪枝，最后决策出当下局面的最优棋步。

*对于一些算法的补充说明：*

由于六子棋每次落两子的缘故，在估值分析中便会分析两层，棋型分析也分为两类 ，一类棋型是将局势向己方有利的局面发展 ，另一类是下棋时会直接产生胜负。

我们在程序中采用常见棋型估值分析方法计算局面价值，减少搜索量，判断合理落子顺序。具体实现为通过定义chesscount1、spacecount1、chesscount2、spacecount2、spacecount3、chesscount3、spacecount4这7个临时变量来对某一路的具体棋形进行判断。 此代码中简单实现了以下几种常见棋形的判断：

1. 眠五：在同一直线上有5颗同色棋子，符合“对方用一手棋就能挡住或长连”的棋型；

2. 活四：在同一直线上有4颗同色棋子，符合“对方必须用两手棋才能挡住或长连”的棋型；

3. 眠四：在同一直线上有4颗同色棋子，符合“对方用一手棋就能挡住或长连”的棋型；

4. 活三：在同一直线上有3颗同色棋子，符合“在下一手就能形成活四”的棋型；

5. 活五：在同一直线上（包括对角斜线）有5颗同色棋子，符合“对方必须用两手棋才能挡住或长连”的棋型；

6. 朦胧三：在同一直线上有3颗同色棋子，符合“在下一手棋只能形成眠四，而如果在下两手棋的话就能形成活五”的棋型；

7. 眠三：在同一直线上有3颗同色棋子，符合“在下两手也只能形成眠五”的棋型；

8. 活二：在同一直线上有2颗同色棋子，符合“在下两手就能形成活四”的棋型；

9. 眠二：在同一直线上有2颗同色棋子，符合“在下两手也只能形成眠四”的棋型；

## 所用方法的特别、新颖或创新之处

1. **黑白方分别采用了两套算法逻辑**

经过我们小组成员多轮的测试，我们发现，在对局中，先手和后手面临的局面和目标不同。

黑方在开局可以主动占据棋盘的中心位置，因此需要侧重于在开局阶段形成有效的进攻布局，尽量控制中心区域，并且迅速发展壮大自己的势力范围。通过积极主动的进攻，建立起对对手的压制和威胁。于是我们减少了先手下棋时候的博弈树递归层数，优先保证自己方棋子的局面，通过激进的布局迫使对方时刻处于忙于拦截我方的零散状态下。并且以贪心思路为主，尝试打开两个活四，令对方拦截失败。

而白方通常需要以反击为主，寻找先手布局的破绽，并且通过巧妙的落子组合，逐渐扭转局势，形成有利的连线。在我们的白方逻辑中，博弈树的模拟和Alpha-Beta 剪枝是重中之重，这可以保证我们的算法能够有效兼顾我方和敌方的局面评估，在后发劣势中寻找机会反击，牵制对方的棋步，进而实现连六目标。

1. **打破传统的权值分配思路**

在诸多六子棋的论文文献中，往往重点关照活四，活五等有突破口的棋形，而忽视眠四等，朦胧三的等潜在价值却难以量化的局。

但是我们在多次实践中发现了这一思路的漏洞。我们已经知道，贪心思想重点在于选取局部最优，但不一定导致全局最优。

此外，棋步的选择可能受限于先前决策的影响。

在相当多的情况下，被单侧拦截住的棋子不一定没有其价值，相反他很有可能在另一个方向为我们打开局面。此时如果我们的选择是基于当前局部信息，而不考虑整体的影响，就可能会出现忽视重要突破口的棋步。因此，我们在第二次估值和第三次（棋形）估值的时候，重点关注了此类棋步。

3、**使用三轮估值函数**

不同于现在大部分算法的一次估值或者两次基于路的扫描估值，我们的算法在时间允许范围内采用了三轮评估。三轮评估并非所有情况都要经历（比如黑子和白子不同，估值方法也不同），但是可以在绝大多数情况下使得决策最合适。

4、**基于路的扫描评估函数复用**

**int** EvaluateROAD(Location location, **int** color, **int** dx, **int** dy) {

    vector<**int**> line;

**int** startX = location.x - dx \* 5;

**int** startY = location.y - dy \* 5;

**for** (**int** i = 0; i < 11; i++) {

**int** x = startX + i \* dx;

**int** y = startY + i \* dy;

**if** (x >= LeftBoard && x <= RightBoard && y >= TopBoard && y <= BottomBoard) {

            line.push\_back(gridInfo[x][y]);

        }

    }

**if**(MyColor == grid\_black){

**return** ViewlineBlack(line, color, location);

    }

**else** {

**return** Viewline(line, color, location);

    }

}

当 dx 和 dy 分别取不同的值时，影响了在棋盘上沿着哪个方向移动以评估路线。这个函数可以合并先前的四个基于路的扫描评估：

当 dx = 0 而 dy = 1：

这种情况下，函数沿着垂直于 y 轴的方向进行移动。dy 的正负值决定了是向上还是向下移动。因此，路线将会是一条垂直于 y 轴的直线。

当 dx = 1 而 dy = 0：

这种情况下，函数沿着垂直于 x 轴的方向进行移动。dx 的正负值决定了是向左还是向右移动。因此，路线将会是一条垂直于 x 轴的直线。

当 dx 和 dy 分别取 1 和 1 （ 1 和 -1 ）：

这种情况下，函数沿着斜线方向进行移动。dx 和 dy 的正负值决定了移动的方向。因此，路线将会是一条斜线。实现了左上右下（右上左下）。

**5、我方两步落子的价值评估逻辑与模拟棋步**

一般来说，传统的价值评判逻辑只会评估我方接下来的两步落子的一个价值，也不会进行全面的模拟棋步。而我们的在程序中敢于打破常规，敢为人先，把我方两步落子的价值评估逻辑完善为相当于八步落子的（四步我方模拟落子，四步敌方模拟落子）一个价值评估逻辑，并且进行了较为全面的模拟棋步，力求找到一个对我方来说最有利的局面，同时将这种思想与博弈树的思想结合，进一步优化了传统的博弈树，使博弈树的功能得到了进一步的升级与提升，通过实测，这种较为全面的考虑要大大优于传统的简单的价值评估逻辑所带来的结果。

# 总结

在完成本次六子棋博弈系统的过程中，我们遇到了诸如怎样统一基于路的估值和基于棋形的估值、如何调出并确定一个好的参数、如何将剪枝算法与具体的估值函数相结合、如何减少搜索和决策的时间等问题，在每一位组员的合作与努力下，我们通过大量的查阅文献、在botzone平上手动测试对局来寻找漏洞和调整参数、尝试改进传统的价值判断逻辑、黑白方采取不同判断逻辑等方式，成功解决了上述问题，并使我们的bot棋力得以有效提升，这些也同样是我们的创新与得意之处。

关于哪些方面可以改进，我们认为有两点，一是关于调参，通过程序员大量手动对局和观察进行调参的方式过于耗费时间和精力，而且介于参数繁多，手动调出的参数很可能不准确，因此我们希望能找到一个更高效的调参方式；二是我们的博弈树能够递归的深度有限（时间限制原因），因此我们希望能继续改进估值过程和搜索算法，以使我们的搜索和决策效率更高。

通过本次课程设计，我们利用有趣的六子棋游戏，加深了对于常见数据结构的认识、巩固了上学期所学过的常见算法，更重要的是我们学习并很好地掌握了如零和博弈、博弈树、极大极小搜索、剪枝算法等新的知识，在调试程序的过程中提升了自身代码水平，在查找文献的过程中提升了科研和学习能力，在合作中感受到了合作精神，收获颇丰。

# 参考文献

1. 张颖.六子棋启发式搜索算法的优化与设计.(2008).
2. 邓银莹.常郝.并行思想的六子棋博弈搜索算法设计.(2005).
3. 李果.基于遗传算法的六子棋博弈评估函数参数优化.(2007).
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.(2010).
5. 何轩.机器博弈主要技术分析——以六子棋为例.(2019).
6. 李学俊.六子棋博弈系统中基于路和棋型的混合搜索方法.(2016).
7. 李学俊.六子棋中基于局部“路”扫描方式的博弈树生成算法.(2015).
8. 韩逢庆.六子棋博弈的二次估值.(2009).