基于极大极小值搜索 $alpha-beta$ 剪枝的重力四子棋 AI 博弈程序

2021310746-张学峰

November 16, 2021

1 作业介绍

本次作业是实现一个基于极大极小值搜索 $alpha-beta$ 剪枝算法的重力四子棋 AI 博弈程序，程序无需输入输出，只需要根据提供的参数补全 Strategy.cpp 中的策略函数 getPoint 即可。本次作业在 saiblo 平台上对全部偶数样例进行了测试，最高胜率可以达到 85%, 平均胜率 78%, 经过和不同水平的测试样例进行分别先后手 10 轮共 20 轮对局，最高可以和 84.dylib 胜率五五开。

2 作业内容

2.1 算法基本思路

首先编写出极大极小值搜索算法框架，为了让程序运行的更快，对博弈树进行 alpha-beta 剪枝。测试发现，规定时间内博弈树最多可以搜索到第五层。本次作业的核心部分是价估函数如何进行决策，大致思路为对局面的不同角色统计进攻得分，防守得分，截断得分。因为第五层的棋权在我方手中所以还要根据角色给定不同的分值，具体将在算法设计部分详细介绍。

2.2 算法设计

2.2.1 极大极小值搜索

极大极小值搜索是在一颗博弈树上寻求最优当前决策的过程。搜索树分为极大值层和极小值层，极大值层的节点希望估值结果越大越好，极小值层的节点则反之。每层的每一个节点代表一个局面，叶子结点为搜索树最深的局面。需要对搜索树的叶子结点代表的局面进行评估，极大值节点选取最大孩子节点值并返回，极小值点选取最小孩子节点并返回。

总的来说，极大极小值搜索在假定双方都足够聪明的条件下进行搜索，使得根结点得到未来 n 步内（搜索树层数）的最佳局面。当然局面价估函数的好坏将会对判断结果有决定性的影响。

2.2.2 alpha-beta 剪枝

一颗极大极小值搜索树的规模是成指数级别上涨的，那么就有必要进行高效的剪枝，缩小搜索范围。alpha-beta 剪枝的主要思想就是根据目前已经获得的搜索信息，判断当前局面是否还有深搜的必要，如果没有必要就进行剪枝。

例如当一个极大值点已经知道了一个子局面的最优结果是 3，那么意味着该极大值点最差也有 3 的收益，相当于确定了下限是 3，搜索另外一个子局面时发现一条子链的搜索结果是 2，那么这整个子局面就没有再搜索的必要，因为极大值点确定了下限为 3。

2.2.3 价估函数

对于局面的价估策略有三点：

1. 双方的进攻得分。统计横竖斜各个方向的相邻 4 个位置中我方棋子的个数，赋予不同的依次升高的得分。

2. 双方的防守得分。统计横竖斜各个方向中连续 5 个位置内，我方棋子成功围住敌方棋子的个数，赋予不同的依次升高的得分。

3. 双方的截断得分。统计横竖斜各个方向中，我方成功截断敌方棋子的个数，赋予不同的依次升高的得分。

一些细节：

1. 如果连续 n 个棋子的两头被围住，称这种为 die，如果只有一头被围住称为 half-live，如果两头都没有被围称为 live，要给三种情况赋予不同的依次变大的得分。

2. 本次作业的极大极小值搜索树深度为 5，在叶子节点我方掌握棋权，所以同样场景应赋予我方棋子更高的得分，例如：敌 2 < 我 2 « 敌 3 < 我 3 « 敌 4 < 我 4 。这种分值设置意味着同等局面，因为轮到我方落子，我方应该赋予更好的分值使得我方更有利连成 4 子。

3. 因为竖着的局面更容易被拦截，所以竖着的局面赋予更低的分值。但是考虑到横着或者斜着的局面可能需要填补某一列而更不容易被连成 4 子，所以要设置一个分数的衰减系数，每当横/斜着的情况出现一个纵向空位要填补，得分就乘一次衰减系数，因为衰减系数的存在，横向纵向的初始得分应该高一点，以防衰减到过低分数。

4. 我方先手落子要先占据中间位置，我方后手时要防守敌方，并且要放在空位更多的一侧。

3 实验

3.1 实验结果

Figure 1: saiblo 测试最高胜率（因为一天只能测 200 个 token，zxf_test 是创建的小号）

alpha-beta 代码实现在 saiblo 在线测评平台的全部偶数样例多次测试后，最高胜率为 85%，平均胜率为 78%。下表所示为本次作业程序在本地和不同等级测试样例进行 10 轮测试每一轮分别先后手共 20 局的胜率。

批量测试 #15222

Figure 2: saiblo 在线测试平台批量测试结果 2

批量测试 #15223

Figure 3: saiblo 在线测试平台批量测试结果 3

测试样例	胜率
2.dylib	100%
10.dylib	100%
20.dylib	100%
30.dylib	100%
40.dylib	95%
50.dylib	95%
60.dylib	70%
70.dylib	70%
80.dylib	65%
82.dylib	50%
84.dylib	50%
86.dylib	35%
88.dylib	20%
90.dylib	25%
96.dylib	30%
98.dylib	35%

3.2 模型改进方法与深入思考

- 因为横向和斜向更不容易连成 4 子（容易出现某列空特别多的情况），引入了衰减系数，不算连续位置在内，每当纵向存在一个空位需要填补就乘一次衰减系数。
- 因为重力四子棋的局面价估有空位影响，所以价估函数可能对复杂局面估计不准确，可以考

虑使用蒙特卡洛搜索树的方法，这样只需要考虑局面胜负即可。

- 应该充分考虑不可落子点的影响，例如先手落子等，尽量偏离不可落子点。
- 在测试样例到 80.dylib 以上时，会使用一些双 3 策略，如图所示，所以应该对局面特殊判断，发掘一些四子棋的技巧性，设计必胜局面，设置双 3 等判断。

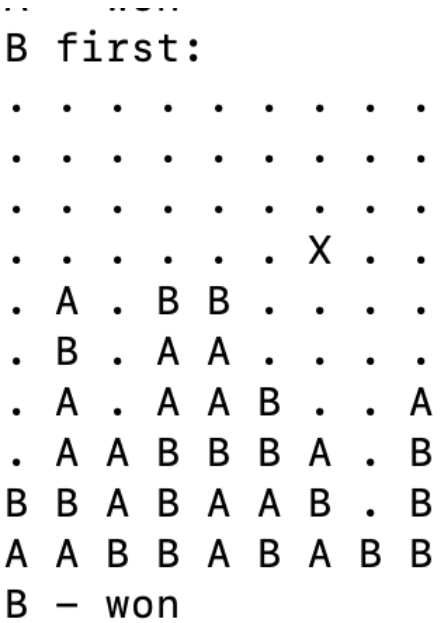

Figure 4: 80.dylib 的双 3 策略

3.3 实验收获

本次实验实现了基于极大极小值搜索 alpha-beta 剪枝算法的重力四子棋程序，并且实现了局面价估函数，使用搜索的方法需要有很强的技巧性，要求掌握下棋的策略，编写更优的局面价估函数。蒙特卡洛搜索树的做法可以直视到最终局面，但是有很大的随机性，极大极小值搜索这种方法的局限性在于搜索深度受限，无法直视到最终局面的好坏，但是根据已有信息进行尽可能多的正确剪枝的这种思想有必要深入理解和牢牢掌握。