四子棋机器对弈项目实验报告

——计 62 李祥凡 2016011262

一． 项目简介

我的对弈策略基于极大极小值搜索和 alpha-beta 剪枝，以当前棋盘状态为根节点向下搜索，将对手假想为最聪明的选手，他的每一步都会做出对自己最有利的选择，当产生对弈结果或搜索达到一定层数后，返回对弈结果或由评价函数得出的分值，并由子节点的分值更新父节点的分值，最终得出预期胜率最高的走法。

二． 算法实现

1. 评价函数

由于每一步的时间限制，搜索到产生胜负结果是不现实的，所以只能规定搜索层数，当搜索到最大层数时，若仍未产生对弈结果，就返回一个由评价函数对当前棋盘状态的估计，该估计值是 long long int 型，正数表示对我有利，负数表示对对手有利，下面简要介绍我的评价函数。

评价函数及与其相关的函数都在文件 Evaluate.cpp 中。

函数 transToLine 将当前二维棋盘数组 board，按四个方向（横，竖，左上-右下，左下-右上）划分为一系列的一维 vector（在不可落子点会有断开，且每个 vector 的首尾均添上了一个-1，便于判断边界），并返回以这些 vector 为元素的 vector<vector<int>>;

评价函数的原理：

对每四个相邻的棋格（四个方向上），统计这四个格子上的落子情况：

若这四个格子上双方的棋子都有，那么双方均不可能以这四个格子相连的方式取得胜利，评价函数在这四个格子上返回的数值为 0（不对任一方有利）。

若这四个格子上只有某一方的棋子，那么认为该情形对这一方有利（有可能以这四个格子相连的方式取得胜利），当调用的是这一方的评价函数时，根据棋子的数量返回相应的值；若调用的是另一方的评价函数，返回 0。

总评价函数 evaluate 将我方与对手对该棋盘的评价函数的返回值相减，即得到该棋盘状态的最终评价值。

2. 搜索函数

搜索函数 alpha-beta 以当前棋盘状态为根节点向下进行递归、回溯的搜索，参数 h 记录当前搜索到达的层数，并将要求的结果 x 和 y 作为参数以引用传入，当 h==0 且返回了更大的评价值时,更新 x 和 y。

当未进行剪枝时，在规定的时间限制下,最多能搜索 4 层，进行了 alpha-beta 剪枝后，能搜索到第 5 层，节约了大量不必要的搜索量，该剪枝是可以隔代进行

的。

在搜索过程中，若产生我方胜利的结果，返回一个很大的 long long 值（视为正无穷），若产生对手胜利的结果，返回一个很小的 long long 值（视为负无穷），若产生平局结果，返回 0.若搜索到第 5 层对局仍未结束，则返回评价函数得出的评价值。

三． 项目总结

经过测试，我的博弈策略在与下发的 50 个策略对弈时，先后手对弈共 100 次，取得 78 场胜利，效果还行。

但我的策略仍存在不少缺憾：

1.评价函数不够高明，我的评价函数借鉴了普通的五子棋的评价方法，但五子棋的落子是自由的（未落子处均可落子），而该四子棋只能在每一列的列顶落子，我的评价函数只将当前棋盘上的棋子相连情况作为评价的依据，而没有考虑规则对可落子点的制约，这显然是不够高明的。若能采用适用该四子棋游戏的科学的评价函数，该策略的智能化将得到很大提升。

2.尽管采用了 alpha-beta 剪枝，策略也只能搜索到五步之后，策略的预见性不足，大局观不够好，在对弈较强的策略时，很容易被对手牵制，导致最后无法兼顾多处的防守，陷入必输的局面。