C语言五子棋by余健

> 五子棋代码分为一个头文件`wzq.h`，8个.c文件`wzq-s1.c` `ban.c` `ban2.c` `ai.c` `ai_choose.c` `score.c` `score2.c`，编译时不编译`score.c`下面详细介绍各个文件的内容与作用。总代码约6000行

头文件：wzq.h

- include一些常用的自带的头文件如`<stdio.h>`等
- 定义一些常量，包括棋盘大小SIZE以及用于评分的常数

  > 注：这些常量定义中关于评分的我分为上下两个部分，分别对应score.c和score2.c，原因是我写了两个评分函数，第二个更加完备，但是都贴上去了

- 定义一些用于评分的结构
- 声明在各个文件都要用到的`arrayForInnerBoardLayout`数组和`BLACK_WHITE`函数

主函数：wzq-s1.c

- 包括了主程序`main`，用于识别输入、显示棋盘
- 包含了`initRecordBorard`、`innerLayoutToDisplayArray`、`displayBoard`这些用于显示棋盘的函数
- 包含了`int win(int n, int x, int y)`函数用于判断胜负与否

禁手判断函数：ban.c, ban2.c

- 包含`isBANNED`函数，判断是否有长连、三三、四四
- 包含`isCHANGLIAN`，`isSANSAN`，`isSISI`函数，用于判断是否有长连禁手、三三禁手、四四禁手
- 包含`formHUOSAN`，`formHUOSI`，`formCHONGSI`，获得当前坐标活三、活四、冲四的个数，辅助禁手判断
- 考虑复杂禁手：

  > 由于存在禁手破坏可下棋的点造成假活三，假活四的情况，必然要进行一次禁手的迭代。一开始我考虑递归，但是可能会造成一次禁手计算太多时间，于是我决定判断两次，再写一个与`isBANNED`的函数几乎一样的函数`isBANNED_2`，用后者判断前者的复杂禁手情况，避免多次迭代的发生

  `isBANNED_2`与`isBANNED`几乎一样，但是删除了迭代部分

💡 接下来是关于如何实现五子棋AI的部分。我进行了两种尝试，发现后面一种更强，用后面一种参与了比赛。接下来两种都进行介绍

第一种评分：ai.c, score.c

- 采用了武老师实验课提到的：用结构记录各个维度的信息，获得总比分

- `ai.c`中的`updateScore`函数对所有空位进行评分，记录到`myBoardScore`和`oppBoardScore`数组中，最后返回总共的评分。
- `score.c`中的`Score`函数仅对一个点进行评分，返回评分值
- 两者的评分思想一致，对于某一个点分别向四个维度进行延申，获得连子数，和障碍数，然后我对武老师的思想做了一个小小的优化，多检索一个空格，可以获得跳子的信息，同时判断这个维度的可利用宽度是否小于5，如果是，这个维度显然无法成5，直接0分。比较两者，后者仅对单点，利于我们缩小检索范围等，对算法进行优化

  > 中间的注释是同时计算自己的分数和对方分数，实际操作时发现只要写一种而后修改函数参数即可，于是注释掉

> 但是我在实际写代码和运行中发现，这种评分方式稍微有些粗略，用while循环获得连子数和跳子数，没有对假活三等进行特别精细的分类。于是经过我和在别的班的好朋友@张笑晗的讨论，决定直接暴力if语句写出所有棋型的情况，获得更加准确的得分。

第二种评分：score2.c

- 记录连子数，是否是活子，是否是跳三，是否是跳四
- 直接用if判断各种棋型：五连，活四，连四，跳四，连活三，跳活三等
- 识别假活三
- 最后打分，获得更加准确的分数

选择最佳点：ai_choose.c

- 首先对黑白子的第一步进行人工定义：黑子下中间，白子下对方下的附近
- 利用`minimax`算法的思想，写了四层的搜索，看四步后的情况决定下哪个位置。由于能力有限以及害怕出错，没有采用递归，而是定义了`Max1`，`Min2`，`Max3`，`Min4`手写了四层
- 为防止超时，只对有棋子的位置的附近空位进行搜索，同时采用alpha-beta剪枝算法
- 为了让ai在自己可以获胜或者对方可以获胜的位置立刻做出决定，在`minimax`算法的中间加一些判断，在立刻获胜时马上退出
- 为了再次节省时间，采用了一个粗糙的启发搜索，记录对方上一次下的位置，先在那个位置附近进行搜索，再搜索其他的，这样可以先获得比较有可能是最大值的点，结合上面的`alpha-beta`剪枝算法节省时间

  > 中间大段注释的内容是一开始写的两层

总结和致谢

代码介绍至此结束。虽然由于能力有限，我只写了四层，还有很大的提升空间，但是这次写五子棋的过程让我沉下心来运用上课的东西去进行一次大的实践，探索的过程中收获良多。

在此特别感谢武老师以及助教们的辛苦付出。

感谢一个github上的教程: 五子棋AI教程 这个教程教会了我minimax、alpha-beta剪枝等算法思想，我在理解这些典型的算法后动手操作获得收获

感谢我的好朋友张笑晗同学，我们一起讨论算法，一起进步