



**计算机仿真大赛报告**

**课题名称：** **西洋跳棋的仿真**

|  |  |  |  |
| --- | --- | --- | --- |
| **成员** | | | |
| **姓名** | **学号** | **学院** | **班级** |
| **周佳鹏** | **3114001013** | **自动化** | **14自动化卓越工程师班** |

**2017 年 5月 01日**

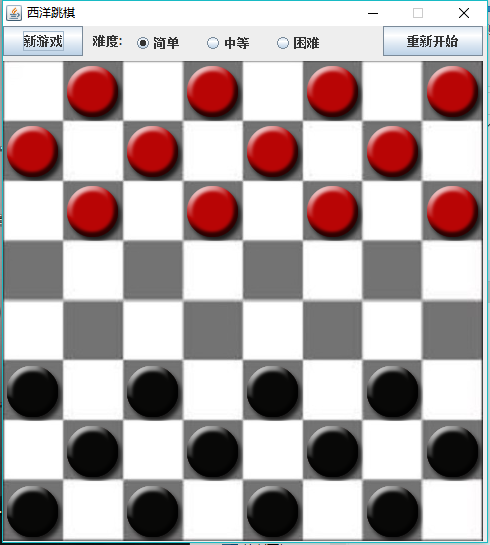
**目录**

1. 游戏介绍 ……………………………………………………………………………….3
2. 程序结构 ……………………………………………………………………………….4
3. 算法介绍 ……………………………………………………………………………….8
4. 总结 ……………………………………………………………………………….11
5. **游戏介绍**

[西洋跳棋](http://baike.baidu.com/item/%E8%A5%BF%E6%B4%8B%E8%B7%B3%E6%A3%8B)是一种两人玩家的棋，棋子都是沿斜角走的。棋子可跳过敌方的棋子并吃它。它的历史比囯际象棋长久，始祖为[中东跳棋](http://baike.baidu.com/item/%E4%B8%AD%E4%B8%9C%E8%B7%B3%E6%A3%8B)。西洋跳棋的玩法有很多。最流行的游戏形式是被称为[国际跳棋](http://baike.baidu.com/item/%E5%9B%BD%E9%99%85%E8%B7%B3%E6%A3%8B)的波兰跳棋，其次为[英国跳棋](http://baike.baidu.com/item/%E8%8B%B1%E5%9B%BD%E8%B7%B3%E6%A3%8B)。一直以来有很多数学家、电脑专家与英国跳棋专家研究破解[跳棋](http://baike.baidu.com/item/%E8%B7%B3%E6%A3%8B/325903)的程序。 直至2007年7月，加拿大计算机科学家正式宣布，英国跳棋已经被研透了，程序名为Chinook，他们表示程序可以找到该棋的最佳走法，若果双方都按照这最佳走法下棋，那么棋局将以和局收场。但更大变化的波兰跳棋、加拿大跳棋等跳棋还待破解。

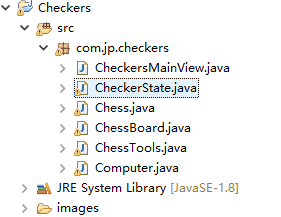
在这次的西洋跳棋仿真中，采用了一种传统的游戏策略，即alpha beta剪枝的极大极小值算法，程序中通过设置搜索的深度来控制电脑方的“智慧”,即游戏的难度。

游戏界面如下：



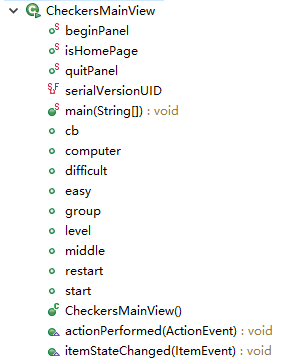
可通过选择界面上边菜单栏中的难度系数来设置游戏难度，无论是在游戏开始前，还是在游戏中，都可以随时改变游戏难度。在游戏中测试发现，游戏难度为困难是，电脑下棋的速度明显比游戏难度为简单和中等的速度慢。

1. **程序结构**

**程序结构如图：**

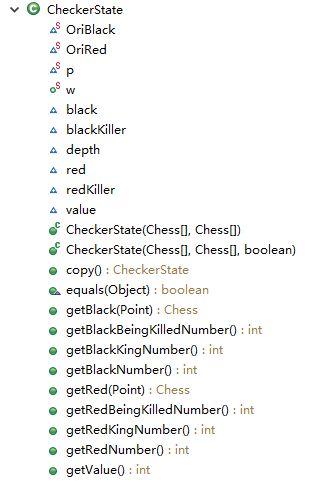
1. **CheckersMainView类**

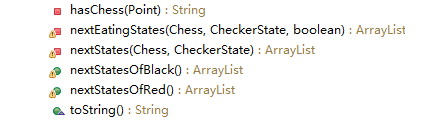
该类是程序的主类，入口函数main()函数在该类中。该类也是程序的主界面类，在该类中，主要设置游戏的界面和一些控件组件，以下为该类的结构



1. **CheckerState类**

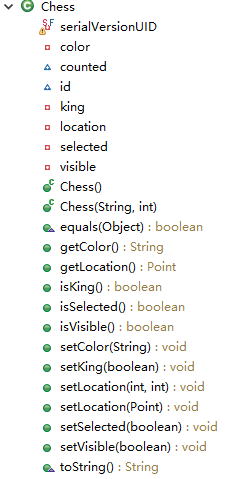
该类为棋盘状态类，主要保存棋局目前的状态，和进行棋局的分析，比如对当前局面的估值分析，对下一步所有可能走的状态的分析，以下为该类的结构





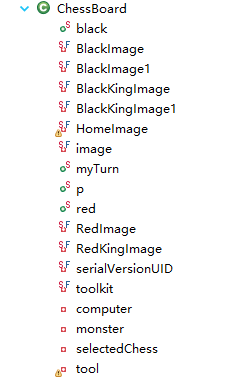
1. **Chess类**

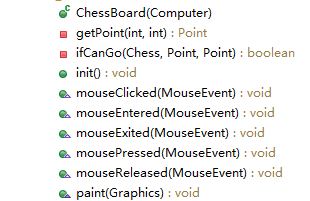
该类为棋子的实体类，保存了棋子的位置，颜色，可见性(是否被吃)和是否为king的状态，以下为该类的结构



1. **ChessBoard类**

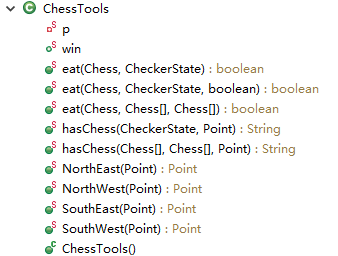
该类是棋盘的界面类，绘制显示棋盘和棋子，以下是该类的结构





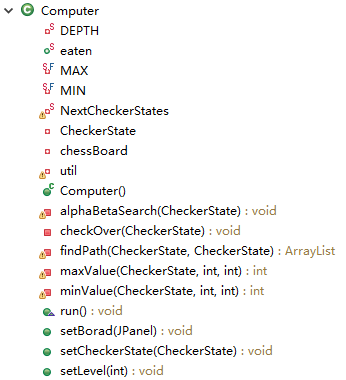
1. **ChessTools类**

该类是工具类，提供一些工具函数，包含判断是否可以吃子，判断某个位置是否有棋子等的函数，以下是该类的结构



1. **Computer类**

该类是电脑AI控制类，通过算法判断电脑方的落棋点，以下为该类的结构



1. **算法介绍**

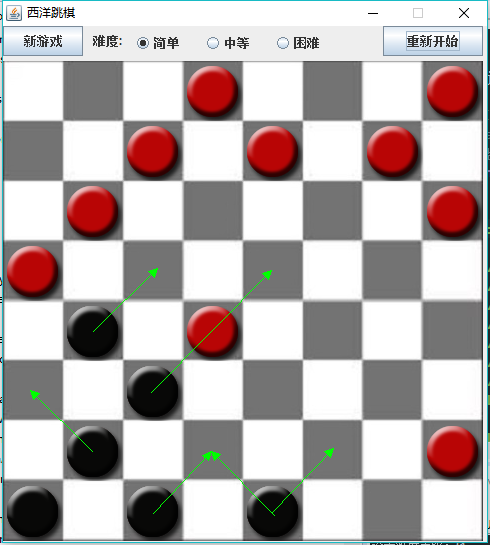
程序通过AlphaBeta剪枝算法来实现电脑自动对棋盘局面进行分析并选择最佳的落棋方案。AlphaBeta剪枝算法是一个搜索算法，旨在减少在其搜索树中，被极大极小算法评估的节点数。

算法使用了两个值，分别为alpha和beta，分别代表最佳得分的最小下界和最大上界，最初alpha，beta被设为无穷大和无穷小，随着搜索的进行，这个上界和下界逐渐收敛。

执行该算法分为三个步骤：1)搜索下一步可能的局面状态 2)对每个局面进行估值 3)通过AlphaBeta剪枝算法逐层判断，选取最优解

1. **搜索下一步可能的局面状态**

第一步，我们首先需要对双方下一步可能走的所有状态进行预测，对于不是king的棋子来说，可以向前走，也可以先前吃子，对于已经成为king的棋子来说，可以任意地向前或向后走和吃子举例来说，如下图，当前局面，对我黑棋来说，可能走的所有情况如箭头所示，黑棋可能有8中走法，搜索完所有的可能走法后，必须将这些情况保存起来，提供给后面的算法分析



1. **对每个局面进行估值**

对所有可能的走法搜索完成后，需要对每个可能出现的局面进行估值，这里的估值函数需要用到6个参数，分别为：

redNumber:红方非king的棋子数量

blackNumber:黑方非king的棋子数量

redKingNumber:红king的棋子数量

blackKingNumber:黑king的棋子数量

redBeingKilledNumber:红方将要被吃掉的棋子数量

blackBeingKilledNumber:黑方将要被吃掉的棋子数量

因为在本程序中，红方为电脑方，故只需对红方进行估值判断，估值函数为以上参数乘上对应的权值后相加的和

int[] w = new int[]{8,6,-6,8,-8,-4,4};//估值函数的权值

**value** = w[0] + w[1] \* redNumber + w[2] \* blackNumber + w[3] \* redKingNumber + w[4] \* blackKingNumber + w[5] \* redBeingKilledNumber + w[6] \* blackBeingKilledNumber;

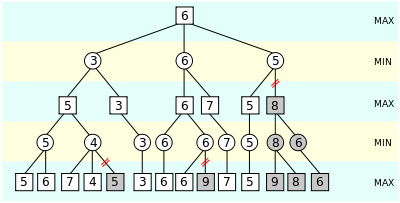
1. **通过AlphaBeta剪枝算法逐层判断，选取最优解**

算法使用了两个值，分别为alpha和beta，分别代表最佳得分的最小下界和最大上界，最初alpha，beta被设为无穷大和无穷小，随着搜索的进行，这个上界和下界逐渐收敛。

1） 任何max节点的alpha值大于其父节点的beta值，则把该节点剩余的枝剪去，因为这说明在该节点之前至少有一个节点的alpha值比它的上界要小，而父节点是min节点，所以这个节点明显不是min节点所想要的。或者说这个节点对对手有利。------Beta剪枝

2） 同理任何min节点的beta值小于其父节点的alpha值，则把该节点剩余的枝剪去。------Alpha剪枝

如下图所示



比如第4层MIN节点的4的 beta = 4 比 其父节点的 alpha值5要小，所以将其剩余的枝剪去，又如第二层MIN节点中的5的beta = 5比其父节点的alpha值6要小，所以将其剩余的枝剪去。

1. **总结**

总体来说，这次的西洋跳棋仿真是能够达到基本功能的，并且能够设置算法的搜索深度来改变游戏的难度,不足的地方是程序的UI界面略显简陋期待以后的改进。