开发文档

修改时间：2017年4月26日星期三

一、预期成果

1.1 完成象棋巫师软件上240个残局挑战

象棋巫师软件上的趣味象棋模块有240个残局，整体难度偏大，本任务具有挑战性。其中有100左右个残局的最优解法呈现出“红方只将军”的现象，在做搜索时，可以只生成将军着法来做，从而大大减小搜索树，但剩下的140左右个残局必须用有效的着法排序、搜索方法来增强处理。总的来说，引擎处理这240个残局的能力可代表着处理普通中局局面的能力，因此本任务有很大的意义。

本任务要求引擎每一步不能超过10秒，即单步时限为10秒。除外，黑方引擎选用象眼(eleeye)且选择特级大师级别。如果黑方无棋可走或认输，且黑方未出现致命错误，则算挑战完成。对于无法获胜的局面，要求能够进入和局。

1.2 打败象眼的大师级别

象眼一共有9个级别，分别是傻瓜、菜鸟、新手、入门、业余、专业、大师、特级大师、无敌。本任务目标是打败第7个级别，即大师级别。比赛采用标准赛制，局面总时间限制为20分钟，超时后时限为每步30秒。尽量采用软件控制时间。如果大师级别挑战失败，则至少打败专业级别。

1.3 思考并实现新的搜索算法

为了与《博弈树搜索算法在中国象棋中的应用》一文中所用的局面数据一致，数据采用黄少龙著的《象棋中局精妙战法》中的20个中局局面。主要研究如何减少搜索平均节点数及搜索平均时间。控制单一变量，测试结果以表格形式直观呈现。

要提升引擎效率，主要有两个方法：更深、更准。更准体现在评价函数上，更深体现在着法排序、搜索方法上。本引擎的评价函数完全采用象眼的方法，因此要效率，关键在如何提升着法排序及搜索方法的能力。

对于着法排序，主要研究如何提升象眼的排序方法，找出其不足之处而改进，并思考如何应用模拟退火等自然算法来提升。对于搜索方法，思考并实现更有效的新的搜索方法，目前有一方法是：博弈树转二叉树。值得一提的是，研究搜索方法是本次研究的关键，也切题。

二、各文件及类、函数

2.1 lazyboy.cpp

2.1.1 int main ( int argc, char \*argv[] )

功能：引擎与界面的接口

1. 接收引擎的指令。调用ucci.cpp/h中的CommEnum BootLine ( void )及CommEnum IdleLine ( CommDetail &Command, int /\* bool \*/ Debug )来获取指令
2. 处理引擎的指令。根据指令来调用引擎其他模块的函数，主要处理棋盘及新游戏初始化、调用主搜索函数。
3. 向界面返回反馈。返回反馈指令”bye”及”readyok”，调用search.cpp/h中的void MainSearch ( void )来返回”bestmove …”。

2.2 position.cpp/h

2.3 move.cpp/h

2.4 hash.cpp/h

2.5 rollback.cpp/h

2.6 time.cpp/h

2.7 ucci.cpp/h

2.8 base.h、debug.h

2.9 movesort.cpp/h

2.10 evaluate.cpp/h

2.11 search.cpp/h