doi:10. 16055/j. issn. 1672-058X. 2021. 0002. 002

基于 Java 的五子棋博弈平台研究*

曹风云[1,2], 赵卫华[1]
(1. 合肥师范学院 计算机学院,合肥 230601;
2. 合肥师范学院 电子信息系统仿真设计安徽省重点实验室,合肥 230601)

摘　要:针对当前五子棋博弈平台中自身功能差异和对引擎的编写语言有限制等问题,设计实现一个通用的五子棋博弈运行平台,平台以 AlphaBeta 剪枝算法为内置博弈引擎算法基础,融合了迭代加深、Zobrist 缓存和启发式搜索算法,建立集对战、算法引擎加载和引擎交互一体化通用五子棋博弈平台;平台利用加载算法引擎文件可自动进行五子棋算法间的对局,且提供调用引擎的接口和引擎设计的参考,方便用户快速设计编写出自己的五子棋算法引擎;通过大量对战测试实验表明,平台在自动博弈模拟、人机博弈、引擎加载、比赛规则方面表现优异,大大提高了博弈算法设计效率。

关键词:计算机博弈;JavaFX;JNA;AlphaBeta 剪枝
中图分类号:TP311. 5　　**文献标志码**:A　　**文章编号**:1672-058X(2021)02-0010-06

0　引　言

针对当前博弈比赛中的五子棋博弈平台[1-8]诸多问题,为五子棋算法设计实现一个完善的比赛平台、平台调用算法的接口及实现此接口的五子棋算法引擎。实现的五子棋算法运行平台能够全程在没有人工干预的情况下调用五子棋算法引擎完成多类别比赛规则的自动对局。支持指定开局、三手交换、五手 N 打、禁手等规则,另外具有控制对局的暂停、继续与强制结束等功能,并附有多步悔棋的功能,同时开启限时,可以在平台界面实时显示棋手单步和单局的耗时,也可以保存和加载平台保存的棋局文件恢复对局。另外,平台具有简易的引擎接口函数并提供对应的 API、引擎框架及示例引擎,通过这些可以极大方便快速编写出在此平台下运行的五子棋算法引擎。

1　博弈平台总体设计

采用 Java 作为平台编写语言,可选 Java、C 或 C++ 作为算法引擎的编写语言,采用 JavaFX 作为平台界面的编写框架,采用 Maven 作为此项目的构建工具。另外,使用 dom4j 解析引擎配置文件,使用 fastjson 读取存储棋局、平台设置,使用 JNA 调用动态链接库中的函数。

设计主要分为平台、引擎和公共类 3 部分,引擎可选 Java、C 或 C++作为编写语言,每个部分具有一个独立的项目。平台部分负责选择对局引擎、设置五子棋对局规则、发送规则至引擎、计时、接收与发送引擎落子、判定引擎落子、控制对局等,引擎部分需要实现平台调用引擎的接口,除了平台内已有的

收稿日期:2020-04-01;**修回日期**:2020-05-23.
* **基金项目**:电子信息系统仿真设计安徽省重点实验室开放基金项目(2019ZDSYSZY06);2019 年度高校优秀青年人才支持计划(GXYQ2019068);大学生创新创业训练计划项目(201914098144).
作者简介:曹风云(1986—),男,安徽合肥人,讲师,硕士,从事数字图像与模式识别与计算机视觉研究.

引擎,开发者可以根据平台的接口实现自己的引擎。引擎根据平台发送的棋规及接收平台发送过来的对方的落子或者整个棋局的所有落子决策出一个优势落子作为引擎执子方的落子。公共类部分是将平台与 Java 编写的引擎中存在的都需要或者可以使用的类抽取出来单独作为一个项目存在,在平台及引擎的项目中引入项目作为依赖。这样既简化引擎接口的参数及返回类型,也方便平台直接调用引擎中的函数,提高了调用引擎的性能。平台架构如图 1 所示。

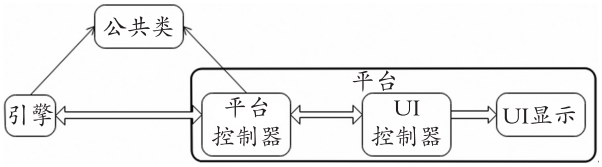

图 1　博弈平台架构

Fig. 1　Game platform architecture

2　平台主体设计

2.1　平台的界面

界面采用 JavaFX 作为编写框架,除棋盘面板外,其他面板均使用 fxml 来编写界面达到快速编写界面的目的且方便修改,另外使用 CSS 来渲染界面也达到美化的作用。主界面如图 2 所示。

图 2　平台主界面

Fig. 2　Platform main interface

平台启动时会将各面板的。fxml 文件加载到内存并进行唯一一次初始化,任何一个面板对应的。fxml 文件都需要一个对应的控制类,通过这个控制类一方面实现各个界面的初始化与控件的调整和变化,另一方面与整个平台运行的逻辑控制类实现数据交互与传递。

棋盘面板不借助 fxml 来实现界面显示,完全通过一个专门的类来达到目的。在这个类中利用 JavaFX 中的画布与图形上下文来实现绘制网格和棋子。通过堆栈来顺序存储落子,方便实现撤回操作也方便获取最后的一个落子,从而方便在界面中体现棋手的最后一个落子。

2.2　对局的设计

从点击右面板的“新一局”按钮到对局结束的简化流程如下:

```
创建双方棋手对象
检查规则
if(时间限制){
    创建新的计时线程计时
    while(未超时){
        时间累加、更新界面时间显示
    }
    to "end step"
}
创建新的对局线程
if(正式比赛规则){
    向棋手请求五手 N 打打点数、请求开局落子
    等待开局落子后,向另一个棋手发送开局落子并
    请求是否进行三手交换
    if(交换){
        执行交换过程
    } else {
        请求白四落子
    }
    请求黑五的几个落子
    while(黑五落子无效){
        请求黑五的几个落子
    }
    将黑五的所有落子发送给白手并获取黑五的实际落子
    通知黑手黑五实际落子
    while(未定胜负){
        轮流向白黑手请茫落子
        while(落子无效){
            通知棋手棋子无效、同时请求新的落子
        }
    }
} else {
    while(落子总数<9){
        轮流向黑白手请求落子
        while(落子无效){
            通知棋手棋子无效、同时请求新的落子
        }
    }
    while(未定胜负){
        向下一个落子棋手请求落子
        while(落子无效){
            通知棋手棋子无效、同时请求新的落子
        }
```

```
    }
}
```

end step：停止向棋手请求落子、提示结束本局、保存结束信息、销毁相关对象等

点击“新一局”按钮后，利用平台控制器中的监听器通知棋盘面板控制器开始新的一局，调用当前对局状态类中相应函数恢复对局状态类至新的一局初始状态；调用棋手库类中创建棋手对象的函数依据当前选定的黑白手棋手创建对应的棋手对象；检查棋规，根据棋规决定是调用引擎中还是由棋手手动选择五手 N 打数量及开局落子；利用监听器通知UI控制器开始新的一局，更新界面的一些控件状态，通过创建并启动一个GameThread线程对象来不断进行黑白手轮流落子；最后利用log4j输出开始新一局的通知信息。右面板接收到开始新的一局的通知时，会先更新相关控件的状态，根据是否限时设置右面板显示时间的控件及设置计时类是否工作的标志。在GameThread中使用JDK API中的ExecutorService配合Future来获取棋手函数的执行结果。

2.3 引擎加载与调用

平台目前可加载实现平台引擎规范和接口的.jar、.dll、.so文件。其中后两者通过C/C++编写项目并导出成.dll或.so文件，前者是利用Maven将Java引擎项目打包的.jar文件。

对于.jar格式引擎，需要一个专门的Java引擎加载器来加载并创建对应的引擎对象。引擎加载器首先获取.jar文件中保存引擎配置信息文件EngineConfigs.xml的文件输入流，接着使用dom4j解析文件生成所有的引擎配置信息对象。因为一个.jar引擎文件中的EngineConfigs.xml文件可能存储多个引擎配置信息，所以一个.jar文件可以被解析出多个可用的引擎配置信息。使用URLClassLoader类加载器加载引擎配置信息中存储的引擎入口类并得到对应的Class对象，并利用此Class对象和反射创建该引擎入口类的对象，因为平台与引擎的项目都依赖公共类项目定义的棋手基类，所以平台可以直接调用Jar引擎中引擎类的函数无需通过反射进行间接调用。

对于动态链接库格式引擎，规定动态链接库文件名必须规范化将其表示的引擎配置信息体现在其中，平台通过解析文件名生成对应的引擎配置信息，平台通过此配置信息创建DllEngine类来表示动态链接库格式的引擎。具体来说需要创建一个接口继承com.sun.jna.Library类，在接口中添加动态链接库中需要被调用的函数的声明，函数名必须与动态链接库中被调用的函数名完全一致。平台调用DllEngine中接口函数，DllEngine再调用接口中对应的函数，JNA在底层会自动调用动态链接库中相应函数。这样就实现平台调用外置C++引擎函数的目的。

因为Java与C/C++在数据类型上无法通用，如果平台直接把表示棋手落子的对象作为参数传递给动态链接库引擎中的函数，函数是无法识别的，导致无法成功调用对应函数的。所以在平台内和平台外都需要对函数的参数类型和执行结果进行转化，转化的结果使两者都可以识别，在已发布的C/C++引擎接口项目中具有相应的函数直接供调用。上述过程如图3所示。

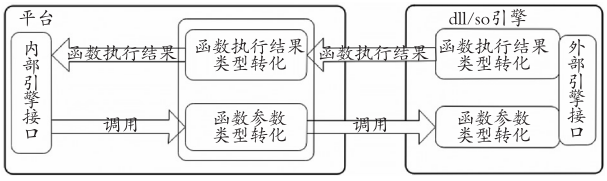

图3 平台与外置动态链接库引擎交互

Fig. 3 Platform interacts with external DLL engine

2.4 引擎框架设计

平台内外都有一个实现引擎接口的引擎框架，在此框架的基础上，可以减少算法外其他代码的编写，缩短编写算法的时间，同时也减少实现引擎接口函数的数量。内外引擎中的引擎框架在结构和逻辑上是一样的。引擎框架的架构如图4所示。

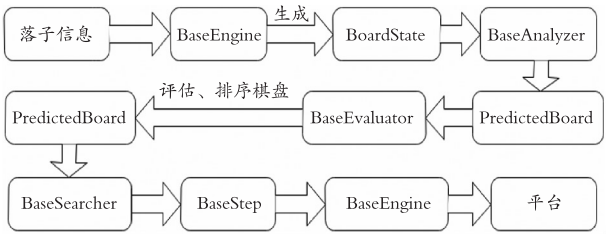

图4 引擎框架架构

Fig. 4 Engine framework architecture

在架构中，BaseEngine预先实现好平台规定的接口函数，转而只预留一个生成下一步落子的函数需要继承BaseEngine的类去手动实现。在自己实现的引擎类中可以覆盖BaseEngine中接口函数中的获取五手 N 打打点数的函数、获取开局落子的函数、获取黑手引擎第五手的所有落子的函数和决定

对方第五手落子的函数。BaseAnalyzer 负责分析棋局产生值得落子、对我方具有威胁和我方有利的位置。BaseEvaluator 负责对棋局上某个位置和整个棋局进行评估。PredictedBoard 表示预测的落子产生的新的棋局。BaseSearch 负责搜索棋局,并将产生最佳落子发送给引擎。最后平台通过接口函数获取引擎的落子。

当接收到平台发送过来的对方落子,首先根据存储的所有落子列表,生成一个棋局状态,即 BoardState 对象,这个棋局状态除了用一个二维数组来表示当前的棋子分布情况,还需要用一个填满 64 位随机整数的三维数组,三维数组可看成由 2 层大小为 15×15 的二维数组组成,两层分别表黑白棋,该数组就是 Zobrist 数组,另外还需要一个 64 位随机值来唯一表示当前棋局,可称这随机值为当前棋局的 ID。每下一步棋,则用当前 ID 异或 Zobrist 数组里对应位置的随机数,得到的结果即为新的 ID。如果是删除棋子或悔棋,则再异或一次即可。使用过 Zobrist 数组方便后面使用 Zobrist 缓存。生成棋局后,还需要给将当前的棋局发送给棋盘分析器、估值类及博弈树搜索引擎。另外还需要一个四维数组表示每个位置周围落子情况。因为可以基于框架基础上实现的示例引擎,所以引擎中的棋局表示工作在框架中已经做好了。

算法开始下子初,首先调用棋盘分析器分析棋局的形式。从棋盘的[A,1]位置开始分析每个位置周围的落子来填充那个表示每个位置周围落子情况的四维数组,同时,如果位置有落子,分析位置对我方及对方的有利及威胁程度,如对方的活三、冲四等,我方的活四、冲四、活三等。如果满足条件,将对应棋型的对策落子添加到一个列表中,并为之形成的新的预测棋局设置一个初始估值,分析完棋盘后直接将该列表返回做下一步处理。如果没有上述棋型,再接着分析棋局上值得的落子的地方,这些地方首先一定是空位置并且不能离有落子地方太远,否则要么落子无效要么落子的意义不是很大还会增大计算时间,只要分析以某个棋子为中心,周围 2 个棋子范围内的位置即可。

对每个值得分析的位置,需要评估出该位置的分数,等对棋盘上所有值得的位置做出评估得到对应分数后,默认使用启发式搜索,即将它们的评分从高到低进行排序,认为分数越高对我方越有利就越先搜索,但是这个有利并不是绝对的。最后将得到的新的顺序的落子产生的对应预测棋局返回,到这里棋局分析的工作完成。同样自己实现的引擎可以直接调用这一部分的函数,也可以通过重写框架中的棋局分析函数实现自己的棋局功能。

2.5　示例引擎设计

平台中的默认引擎在上述框架的基础上进行编写的,引擎以广泛使用的 α-β 剪枝算法为搜索博弈树的基础算法,并加上迭代加深、启发式搜索、Zobrist 缓存这些优化方法,结构如图 5 所示。

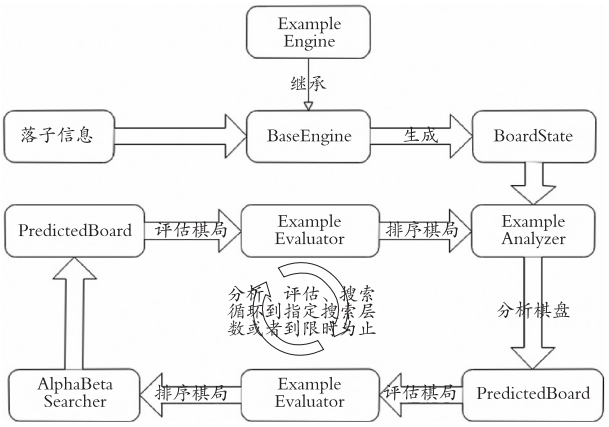

图 5　示例引擎结构

Fig. 5　Example engine structure

2.5.1　估　值

对棋盘上某个位置估值时,在分析有利威胁落子的时候,对应的预测棋型根据产生棋局的棋子是活四、冲四还是活三分别设置固定的分数。但是在分析值得落子位置时,使用另一种估值的方法。

方法简单来说根据当前被评估位置某个方向能形成连五的数量来评估该位置的分数。如图 6 所示,当前的执子颜色是黑色。估值开始时,获取某位置某方向棋子数组的第 $i(1>=i>=5)$ 个棋子,然后再获取该棋子后面的 $j(i+1<=j<=i+4)$ 棋子,根据是空位置还是当前分析的棋手执子颜色,如果是的,对应的计数加一,再获取 j+1 棋子颜色,重复上述操作,直到 j+4 棋子结束,如果空位置的数量和我方颜色为数量之和为 5,则进行一次计分,否则必定有对方棋子掺杂进来一定无法构成连五,所以不进行计分。再获取该方向棋子数 i+2 棋子,接着重复上述操作直至第五个棋子结束。如果是对方的棋子颜色,不用再继续获取下一个棋子,直接获取方向棋子数 i+2 棋子,接着重复上述操作直至第五个棋子结束。

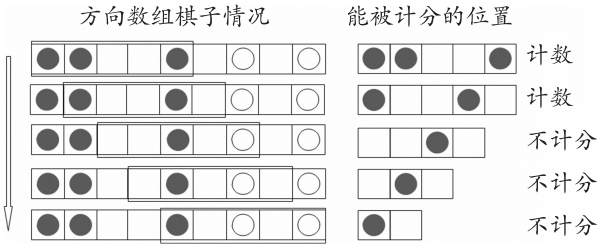

图6 估值计分

Fig. 6 Valuation score

2.5.2 基于启发式 α-β 搜索算法

α-β 剪枝搜索和剪枝与子节点的排列顺序有关。在MAX层节点的值是子节点中的最大值,按节点值从大到小的顺序排列MIN层节点的话,计算第一个MIN层节点后会更新其父节点即MAX层的最大下届,如果在计算后面MIN层更新其最小上届的值比其父节点MAX层的最大下届还要小,那么该MIN层后面的节点就没有继续搜索下去的必要了,因为MIN层最大值都比MAX层的最小值还要小,直接触发剪枝,没有必要在继续搜索下去。同理MAX层按子节点值从小到大排列也会有同样的效果。

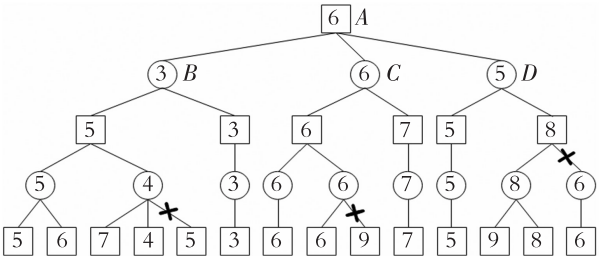

图7 α-β 剪枝

Fig. 7 α-β pruning

如图7中,第二层中的 B 和 C 节点,按照现在的顺序排列,因为 B 节点的值3比较小,更新父节点 A 的最大下届为3,而节点 C 第一个子节点的值是6比3大,还要继续计算下一个子节点的值,也就是需要完整计算第二层 C 节点的所有子节点。如果将它们互换位置,因为第一个节点 B 的值6比3大,计算完后更新父节点 A 的最小下届为6,计算 C 节点的第一个子节点后更新 C 节点的最大下届为5,比父节点 A 的最大下届还要小,那么 C 节点后面的子节点就没有继续搜素计算下去的必要了,触发了剪枝。所以,α-β 剪枝的效率和节点排序有很大关系,如果最优的节点能排在前面,则能大幅提升剪枝效率。

但是节点的值是通过递归计算出来的,也就是在之前是无法精确对节点进行排序的,对节点做出一个大致的排序。在博弈树中,每个节点代表一个预测的棋局,节点的值是该棋局的评分,所以对节点排序时,可以根据产生棋局的落子的评分进行排序,虽然只是一个大概的排序,但是会很好的提升搜索的速度和剪枝的效率。在每次分析完棋局后,首先将值得落子的评分临时设置为落子对应的预测棋局评分,因为采用负极大值的 α-β 剪枝搜索,所以这里将棋局按照其评分从大到小进行排序得到一个新的预测棋局列表,再将这个列表送至搜索引擎使用 α-β 剪枝进行搜索。

3 结 论

建立了通用五子棋博弈平台。采用Java作为平台主体的编写语言,并附有Java和C/C++编写的引擎接口。研究结果表明:

(1) 与现有的博弈平台相比,使用者只需将自己编写的引擎文件拷贝到自动被加载的目录或者通过平台菜单栏中的"加载引擎"按钮即可实现加载引擎。

(2) 平台即可代替人工全自动进行五子棋算法间的对局,且提供调用引擎的接口和引擎设计的参考,及方便快速编写出自己的五子棋算法引擎,也方便操作。且平台完全支持正式五子棋比赛使用的禁手、指定开局、三手交换等规则,在各类计算机博弈比赛五子棋赛中也具有较好的实用性。

参考文献(References):

[1] 李卓轩,李媛,冉冠阳,等. 基于PVS搜索算法的亚马逊棋博弈系统的设计[J]. 智能计算机与应用,2018,8(5):92—94
LI Z X, LI Y, RAN G Y, et al. Amazons Game System Based on PVS Search Algorithm[J]. Intelligent Computer and Application, 2018, 8(5): 92—94(in Chinese)

[2] 郭琴琴,李淑琴,包华. 亚马逊棋机器博弈系统中评估函数的研究[J]. 计算机工程与应用,2012,48(34):50—54
GUO Q Q, LI S Q, BAO H. Research on Evaluation Function Computer Game of Amazon. Computer Engineering and Applications, 2012, 48(34): 50—54(in Chinese)

[3] 涂智豪. 人工智能五子棋系统设计与实现[D]. 广州:华南理工大学,2016
TU Z H. System Design and Implementation of Gobang Artificial Intelligence [D]. Guangzhou: South China University of Technology, 2016(in Chinese)

[4] 王骄,徐心和. 计算机博弈:人工智能的前沿领域—全国大学生计算机博弈大赛[J]. 计算机教育,2012,163(7):14—18
WANG J, XU X H. Computer Game: The Frontier of Artificial Intelligence: National University Student Computer Game Contest[J]. Computer Education, 2012, 163(7):14—18(in Chinese)

[5] 张明亮,吴俊,李凡长. 五子棋机器博弈系统评估函数的设计[J]. 计算机应用,2012,32(7):1969—1972
ZHANG M L, WU J, LI F Z. Design of Evaluation-function for Computer Gobang Game System [J]. Journal of Computer Applications, 2012, 32(7): 1969—1972 (in Chinese)

[6] 王云霞. 智能五子棋博弈算法研究[J]. 江苏技术师范学院学报,2013(2):62—66
WANG Y X. The Algorithm Research of Intelligent Gobang Playgame [J]. Journal of Jiangsu Teachers University of Technology, 2013(2):62—66(in Chinese)

[7] 李虎阳,罗旭,常永虎. 基于可信度的多次重复博弈研究[J]. 重庆工商大学学报(自然科学版),2016,33(1):70—72
LI H Y, LUO X, CHANG Y H. Research on Multiple Repeat Games Based on Credibility [J]. Journal of Chongqing Technology and Business University (Natural Science Edition), 2016, 33(1):70—72(in Chinese)

[8] 周代平,李康奇,贺琳. 诱导信息条件下车辆路径选择:基于有限理性模糊博弈[J]. 重庆工商大学学报(自然科学版),2015,32(12):31—35
ZHOU D P, LI K Q, HE L. Research on the Model of Vehicle Routing Choice Based on the Condition of the Bounded Rationality Fuzzy Game with Inducing Information[J]. Journal of Chongqing Technology and Business University (Natural Science Edition), 2015, 32(12):31—35(in Chinese)

Research on Java-based Gobang Game Platform

CAO Feng-yun[1,2*], ZHAO Wei-hua[1]

(1. School of Computer Science and Technology, Hefei Normal University, Hefei 230601, China;
2. Anhui Key Laboratory of Simulation and Design for Electronic Information System, Hefei Normal University, Hefei 230601, China)

Abstract: Aiming at the current differences in functions of the Gomoku game platform and the limitation of the engine's design language, a universal Gomoku engine operating platform is designed and implemented to reduce the restrictions on the types of engine design languages. The platform is based on the AlphaBeta pruning algorithm as the built-in engine. It combines iterative deepening, Zobrist caching, and heuristic search algorithms, and establishes a set game platform, engine and engine interaction integrated game platform. The platform uses loading engine files to automatically play games between Gomoku algorithms, and provides a reference to the engine's interface and engine design, which is convenient for users to quickly write their own Gomoku algorithm engines. A large number of battle test experiments show that the platform excels in automatic game simulation, man-machine game, engine loading, and game rules, which greatly improves the efficiency of game algorithm design.

Key words: computer game; Java FX; JNA; Alpha Beta pruning

责任编辑:田　静

引用本文/Cite this paper:
曹风云,赵卫华. 基于 Java 的五子棋博弈平台研究[J]. 重庆工商大学学报(自然科学版),2021,38(2):10—15
CAO F Y, ZHAO W H. Research on Java-based Gobang Game Platform [J]. Journal of Chongqing Technology and Business University (Natural Science Edition), 2021, 38(2):10—15