**5.1博弈**

两人参与的游戏：

玩家：MAX、MIN

组成部分：

\* S0：初始状态，规范游戏开始时的情况。

\* PLAYER(s): 定义此时该谁行动。

\* ACTIONS(s)：返回此状态下的合法移动集合。

\* RESULT(s,a)：转移模型，定义行动的结果。

\* TERMINAL-TEST(s): 终止测试，游戏结束返回真，否则返回假。游戏结束的状态称为终止状态。

\* UTILITY(s,p)：效用函数，**以MAX的效益计数**

**5.2博弈中的优化决策**

**5.2.1极小极大算法**

确定性、完备信息的完美游戏

执行完整的深度优先探索，递归算法自上而下一直前进到树的叶结点，然后随着递归回溯通过搜索树把极小极大值回传

在每个结点合法的行棋有b个，那么极小极大算法的时间复杂度是O(b^m)

一次性生成所有的后继的算法，空间复杂度是 O(bm),

**若树是有限的：完备**

**最优**

**5.2.2多人博弈时的最优决策**

将结点的单一值替换为值向量（分量为对各玩家的效用值）

结点n的倒推值是对在该点进行选择的玩家来说(对应分量)效用值最大的值向量

**5.3 α-β剪枝**

时间复杂度是O(b^(m/2))

α = 到目前为止路径上发现的**MAX**的最佳(即极大值）选择  **至少**

β = 到目前为止路径上发现的 **MIN** 的最佳(即极小值）选择 **至多**

递归搜索时，子节点继承父节点的α和β值，

MAX层：将所有可能的行动a依次丢给MIN层来决策，得到效用值v2(和MIN决策move2)

比较v2与当前最佳效用值v，若v2>v,，则更新v为v2，更新最佳动作move为a.

最后将最大的v更新给α

剪枝条件：若α≥β，直接返回v, move

MIN层: 将所有可能的行动a依次丢给MAX层来决策，得到v2(和MAX决策move2)

比较v2与当前最佳效用值v，若v2<v,，则更新v为v2，更新最佳动作move为a

最后将最小的v更新给β

剪枝条件：若β≤α，直接返回v, move

**5.4不完美的实时决策**

蒙特卡洛树搜索

1. ​**选择（Selection）​**：
   * 从搜索树的根节点开始，按照某种策略（如 UCB1）选择子节点，直到到达一个未完全探索的节点（叶子节点）。
   * 选择策略的目标是平衡**探索**​（尝试新节点）和**利用**​（选择已知较好的节点）。
2. ​**扩展（Expansion）​**：
   * 如果当前叶子节点不是终止状态，则扩展它，生成一个或多个子节点，并将其加入搜索树。
3. ​**模拟（Simulation）​**：
   * 从**新扩展的节点开始**，通过**随机模拟（蒙特卡洛方法）进行游戏**，直到达到终止状态（如游戏结束）。
   * **模拟过程中不需要构建搜索树，而是通过随机选择行动来快速完成。**
4. ​**回溯（Backpropagation）​**：
   * 将模拟结果（如胜利或失败）沿着搜索路径回溯，**更新所有经过节点的统计信息**（如访问次数和胜率）。

搜索空间几乎无限，找出的是局部最优

**5.4.1评估函数**

博弈程序的性能严重依赖于评估函数的质量

要求：

1. 评估函数对终止状态的排序应该和**真正的效用函数**的**排序结果一样**：贏状态的评估值一定要好于平局，而平局一定要好于输的状态。
2. 评估函数的计算本身不能花费太长时间
3. 对于**非终止状态**，评估函数应该和**取胜几率**密切相关

**5.4.2截断搜索**

**5.4.3向前剪枝**

**5.4.4搜索与查表**

**5.5随机博弈**

**期望极小化极大算法**