蒙特卡罗树搜索(Monte Carlo Tree Search, MCTS)

- MCTS算法
- 比较alpha-beta 和MCTS

为什么引入MCTS?

围棋游戏说明了启发式alpha-beta树搜索的两个主要弱点：

- 围棋的分支因子为361，这意味着alpha-beta搜索将仅限于4或5层。
- 很难为围棋定义一个好的评价函数

MCTS: 主要思想

- 一个状态的值被估计为从该状态开始的完整博弈的多次模拟的平均效益。
- 一次模拟(simulation，也称为playout或rollout)首先为一个玩家选择动作，然后再为另一个玩家选择动作，一直重复直到到达一个终止位置。
- 在终止位置，博弈规则决定了谁赢或输，以及分数。
- 对于结果只是赢或输的游戏，“平均效益”即“胜率”。

模拟策略

我们如何选择在模拟中做什么动作？

- 如果我们只是随机选择，那么在多次模拟之后，我们得到了以下问题的答案："如果双方都随机玩，最好的动作是什么？"
- 但是我们关心的是"如果双方都玩得好，最好的动作是什么？"
- 所以我们需要一个偏向于好的动作的模拟策略。
- 对于围棋和其他游戏，通过使用神经网络，模拟策略可以成功地从自我博弈(self-play)中学习。
- 有时会使用特定游戏的启发式方法，如黑白棋中的"占领角落"。

选择策略

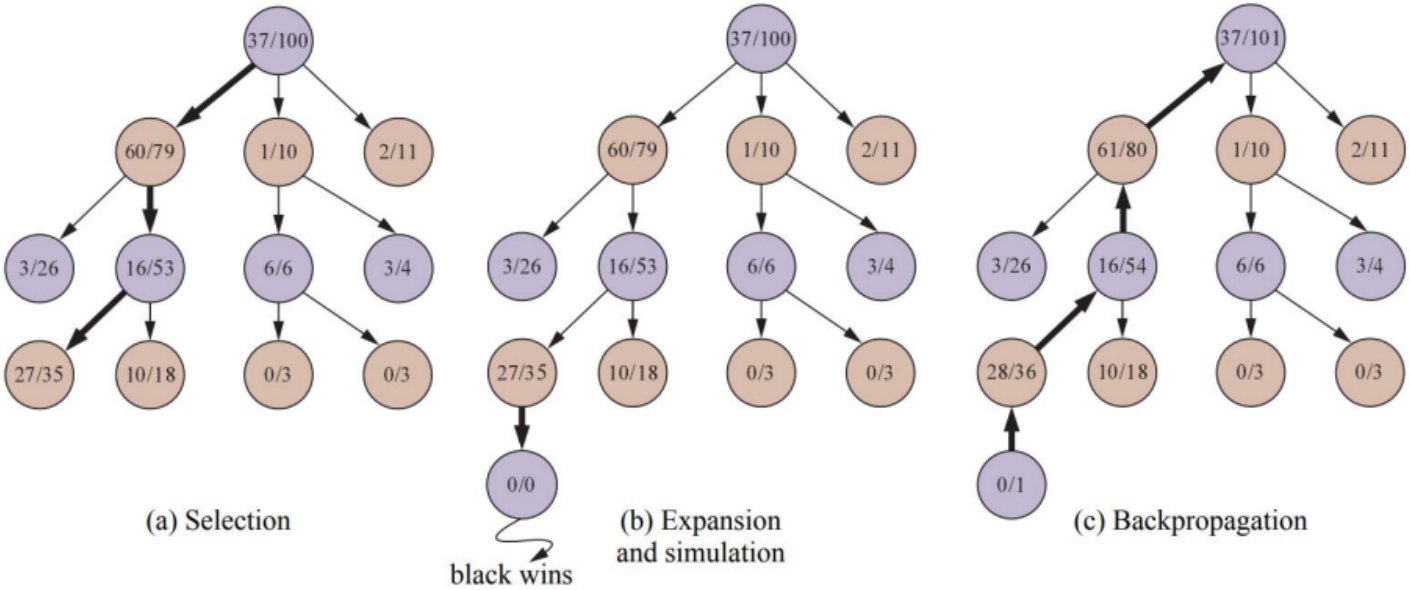
- 我们从什么位置开始模拟，并且我们分配给每个位置多少次模拟？
- 最简单的方法，称为纯蒙特卡罗搜索：从游戏的当前状态开始进行N次模拟，并记录从当前位置开始的哪些可能的动作具有最高的胜率。
- 然而，我们需要一个重点关注博弈树的重要部分的选择策略。
- 它平衡两个因素:探索模拟次数很少的状态，利用在过去的模拟中表现良好的状态，以更准确地估计它们的价值。

MCTS

MCTS维护一个搜索树，并在包含以下四个步骤的每次迭代中使其增长：

- 选择：从搜索树的根节点开始，（根据选择策略）选择一个动作，到达一个后继节点，并重复这个过程，向下移动到树的一个叶子。
- 扩展：通过生成所选节点的一个新子女来扩展搜索树；
- 模拟：从新生成的子节点进行一次模拟，根据模拟策略为两个玩家选择动作。
- 反向传播：使用模拟结果更新向上到根的所有节点。

一个迭代的例子



$U(n)/N(n)$:
$U(n)$是通过节点n的所有模拟的总效益，
$N(n)$是通过节点n的模拟次数

UCT选择策略

UCT(应用于树的上限置信区间)

根据一个称为UCB1的上限置信区间公式对每个可能的动作进行排序

$$UCB1(N) = \frac{U(n)}{N(n)} + C \times \sqrt{\frac{\log N(\text{PARENT}(n))}{N(n)}}$$

- $U(n)$: 通过节点n的所有模拟的总效益,
- $N(n)$: 通过节点n的模拟次数
- PARENT(n): 是树中n 的父节点

UCB1公式

$$UCB1(N) = \frac{U(n)}{N(n)} + C \times \sqrt{\frac{\log N(\text{PARENT}(n))}{N(n)}}$$

- $\frac{U(n)}{N(n)}$是利用项：n的平均效益。
- 带有平方根的项为探索项：
  - 对于只被探索过几次的节点，它将会很高
- C是一个平衡利用和探索的常数。
  - 有一个理论证明C应该是$\sqrt{2}$,
  - 但在实践中，程序员尝试C的多个值，并选择性能最好的值。

UCT MCTS算法

function MONTE-CARLO-TREE-SEARCH(*state*) **returns** *an action*
  tree ← NODE(*state*)
  while IS-TIME-REMAINING() **do**
    leaf ← SELECT(*tree*)
    child ← EXPAND(*leaf*)
    result ← SIMULATE(*child*)
    BACK-PROPAGATE(*result*, *child*)
  return the move in ACTIONS(*state*) whose node has highest number of playouts

为什么要返回模拟次数最多的节点呢?

- UCB1确保模拟次数最多的节点几乎总是胜率最高的节点,
- 因为随着模拟次数的增加，选择过程越来越倾向于胜率高的节点。

完成一个模拟的时间与博弈树的深度成线性关系，而不是指数的

MCTS的优点

- Alpha-beta选择一条路径到达一个评价函数得分最高的节点
- 因此，如果评价函数不准确，alpha-beta将是不准确的。
- 单个节点上的计算错误可能会导致alpha-beta错误地选择（或避免）到该节点的路径。
- 但是MCTS依赖于多次模拟的结果，因此并不容易受到单个错误的影响。

MCTS的优点

- MCTS可以应用于全新的博弈，对其没有经验可以用来定义一个评价函数。
- 只要我们知道博弈规则，MCTS不需要任何额外的信息。
- 选择和模拟策略可以充分利用手工的专家知识，但好的策略可以通过在自我博弈上训练的神经网络来学习。
- 长期以来，人们一直认为alpha-beta搜索更适合于具有低分支因子和良好评价函数的国际象棋等博弈，
- 但近年来，蒙特卡罗方法在国际象棋和其他博弈中取得了成功。

MCTS的缺点

- 很可能一个单个的动作可以改变博弈的进程，但由于其随机性，MCTS可能不会考虑到这个动作。
- 可能有一些博弈状态是“明显”的获胜状态（根据人类知识和评价函数），但在模拟中需要很多动作。