28 天自制你的 AlphaGo（五）：蒙特卡洛树搜索（MCTS）基础

本文作者：彭博 2017-02-24 17:37

导语：蒙特卡洛树搜索（MCTS）是所有现代围棋程序的核心组件。

雷锋网(公众号：雷锋网)按：本文作者彭博，Blink·稟临科技联合创始人。文章由雷锋网整理自作者知乎专栏，获授权发布，未经允许禁止转载。

蒙特卡洛树搜索（MCTS）是所有现代围棋程序的核心组件。在此之上可以加入各种小技巧（如 UCT，RAVE/AMAF，Progressive Bias，Virtual win & lose，Progressive Widening，LGR，Criticality 等等）和大改进（如 AlphaGo 的策略网络和价值网络）。

网上很少见到关于 MCTS 的详细介绍，而且许多看似详细的介绍实际有错误，甚至许多人会混淆蒙特卡洛树搜索和蒙特卡洛方法。这两者有本质区别。用做过渲染器的朋友会理解的话来说：蒙特卡洛方法有偏差（Bias），而MCTS没有偏差（Bias）。我们在后文会解释。

一、极小极大（Minimax）搜索

先看传统的博弈游戏树搜索，著名的极小极大（Minimax）搜索，学过算法的朋友会清楚。看下图，假设现在轮到黑棋，黑棋有b1和b2两手可选，白棋对于b1有w1和w2两手可选，白棋对于b2有w3 w4 w5三手可选：

彭博

专栏作者

Blink·稟临科技 联合创始人

发私信

当月热门文章

最新文章

阿里巴巴年度技术总结：人工智能在搜索的应用和实践

回望2017，基于深度学习的NLP研究大盘点

清华博士生孙奕帆：行人再识别论文介绍及最新进展

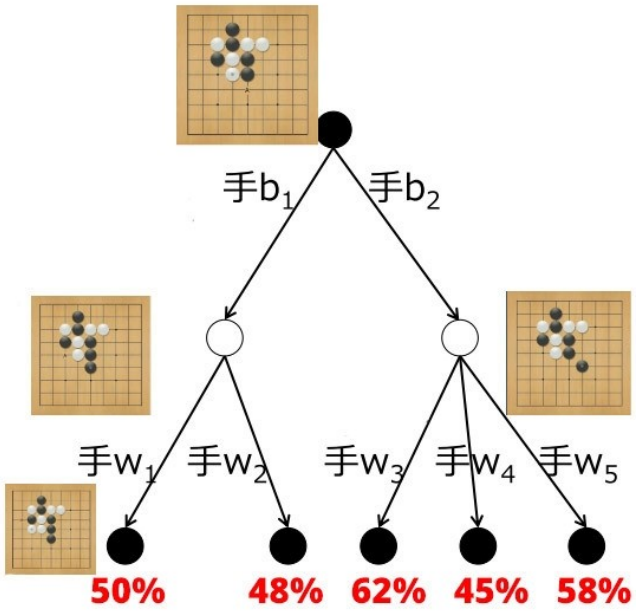
香港科技大学施行鸿：深度学习用于短临降雨预报的一个基准和一个新模型 | 分享总结

2017年度人工智能热门事件大盘点，哪些令你印象最深刻？

Geoffrey Hinton、Andrew Moore、吴恩达等人总结 2017 AI 大事件

热门搜索

融资 比特币



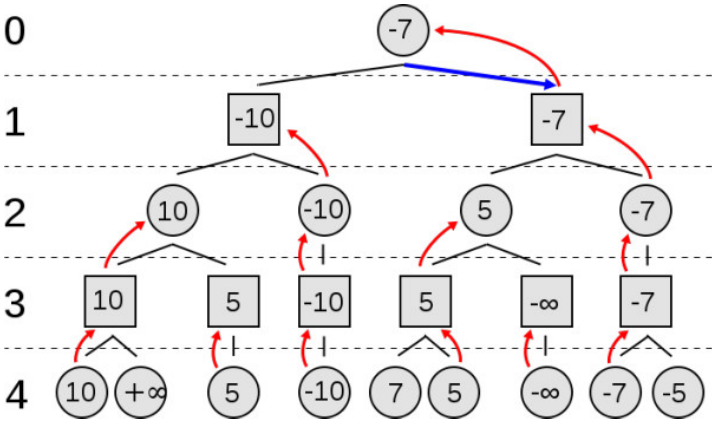
然后假设走完w1/w2/w3/w4/w5后，经过局面评估，黑棋的未来胜率分别是 50%/48%/62%/45%/58%（等一下，这些胜率是怎么评估出来的？我们后文会说这个问题）。

请问，黑棋此时最佳的着法是b1还是b2？如果是用蒙特卡洛方法，趋近的会是其下所有胜率的平均值。例如经过蒙特卡洛模拟，会发现b1后续的胜率是49% = (50+48)/2，而b2后续的胜率是55% = (62+45+58)/3。

于是蒙特卡洛方法说应该走b2，因为55%比49%的胜率高。但这是错误的。因为如果白棋够聪明，会在黑棋走b1的时候回应以w2（尽量降低黑棋的胜率），在黑棋走b2的时候回应以w4（尽量降低黑棋的胜率）。

所以走b1后黑棋的真实胜率是48%，走b2后黑棋的真实胜率是45%。黑棋的正解是b1。这就是 Minimax 搜索的核心思想：在搜索树中，每次轮到黑棋走时，走对黑棋最有利的；轮到白棋走时，走对黑棋最不利的。由于围棋是零和游戏，这就可以达到最优解。这是一个由底往上的过程：先把搜索树画到我们可以承受的深度，然后逐层往上取最大值或最小值回溯，就可以看到双方的正解（如果胜率评估是准确的）。而实际编程的时候，是往下不断生长节点，然后动态更新每个父节点的胜率值。

下图是一个更多层的例子：

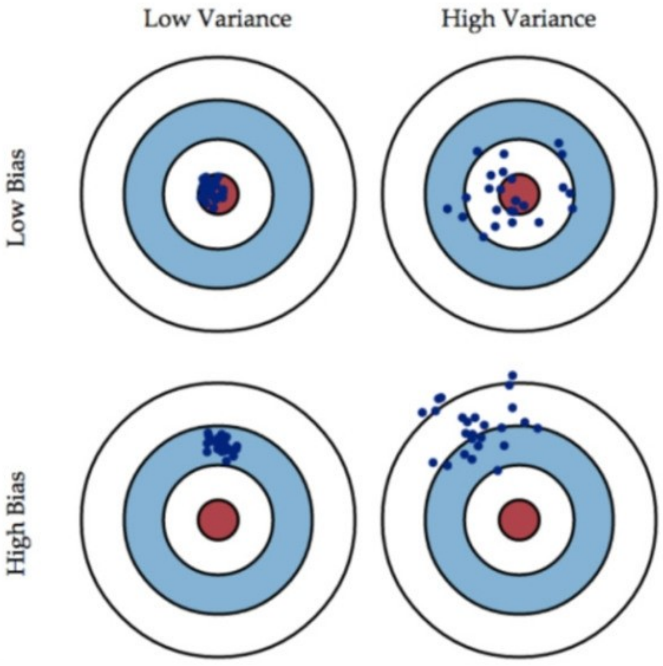


值得注意的是，在实际对局中，胜率评估会有不准确的地方，这就会导致“地平线效应”，即由于电脑思考的深度不够，且胜率评估不够准确，因此没有看见正解。

Minimax 搜索还有许多后续发展，如课本会说的 Alpha-beta 剪枝，以及更进一步的 Null Window / Nega Scout / MTD(f) 等等。可惜这些方法更适合象棋等棋类，对于围棋的意义不大（除非已经接近终局），请读者思考原因。

蒙特卡洛树搜索和蒙特卡洛方法的区别在于：

- 如果用蒙特卡洛方法做上一百万次模拟，b1和b2的胜率仍然会固定在49%和55%，不会进步，永远错误。所以它的结果存在偏差（Bias），当然，也有方差（Variance）。
- 而蒙特卡洛树搜索在一段时间模拟后，b1和b2的胜率就会向48%和45%收敛，从而给出正确的答案。所以它的结果不存在偏差（Bias），只存在方差（Variance）。但是，对于复杂的局面，它仍然有可能长期陷入陷阱，直到很久之后才开始收敛到正确答案。



二、蒙特卡洛树搜索

如果想把 Minimax 搜索运用到围棋上，立刻会遇到两个大问题：

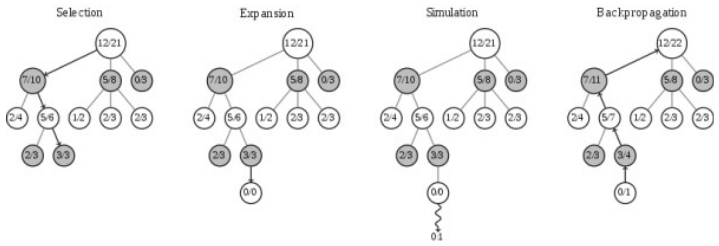
- 搜索树太广。棋盘太大了，每一方在每一步都有很多着法可选。
- 很难评估胜率。除非把搜索树走到终局，这意味着要走够三百多步（因为对于电脑来说，甚至很难判断何时才是双方都同意的终局，所以只能傻傻地填子，一直到双方都真的没地方可以走为止）。简单地说，搜索树也需要特别深。

蒙特卡洛树搜索的意义在于部分解决了上述两个问题：

- 它可以给出一个局面评估，虽然不准，但比没有强。这就部分解决了第二个问题。
- 根据它的设计，搜索树会较好地自动集中到“更值得搜索的变化”（注意，也不一定准）。如果发现一个不错的着法，蒙特卡洛树搜索会较快地把它看到很深，可以说它结合了广度优先搜索和深度优先

搜索，类似于启发式搜索。这就部分解决了第一个问题。

最后，随着搜索树的自动生长，蒙特卡洛树搜索可以保证在足够长的时间后收敛到完美解（但可能需要极长的时间）。下面看具体过程（需要指出，下图是网络上常见的一个图，但其实有错误）：



上图中每个节点代表一个局面。而 A/B 代表这个节点被访问 B 次，黑棋胜利了 A 次。例如一开始的根节点是 12/21，代表总共模拟了 21 次，黑棋胜利了 12 次。

我们将不断重复一个过程（很多万次）：

这个过程的第一步叫选择（Selection）。从根节点往下走，每次都选一个“最值得看的子节点”（具体规则稍后说），直到来到一个“存在未扩展的子节点”的节点，如图中的 3/3 节点。什么叫做“存在未扩展的子节点”，其实就是指这个局面存在未走过的后续着法。

第二步叫扩展（Expansion），我们给这个节点加上一个 0/0 子节点，对应之前所说的“未扩展的子节点”，就是还没有试过的一个着法。

第三步是模拟（Simluation）。从上面这个没有试过的着法开始，用快速走子策略（Rollout policy）走到底，得到一个胜负结果。按照普遍的观点，快速走子策略适合选择一个棋力很弱但走子很快的策略。因为如果这个策略走得慢（比如用 AlphaGo 的策略网络走棋），虽然棋力会更强，结果会更准确，但由于耗时多了，在单位时间内的模拟次数就少了，所以不一定会棋力更强，有可能会更弱。这也是为什么我们一般只模拟一次，因为如果模拟多次，虽然更准确，但更慢。

第四步是回溯（Backpropagation）。把模拟的结果加到它的所有父节点上。例如第三步模拟的结果是 0/1（代表黑棋失败），那么就把这个节点的所有父节点加上 0/1。

三、重要细节

怎么选择节点？和从前一样：如果轮到黑棋走，就选对于黑棋有利的；如果轮到白棋走，就选对于黑棋最不利的。但不能太贪心，不能每次都只选择“最有利的/最不利的”，因为这会意味着搜索树的广度不够，容易忽略实际更好的选择。

因此，最简单有效的选择公式是这样的：

$$score = x_{child} + C \cdot \sqrt{\frac{\log(N_{parent})}{N_{child}}}$$

其中 x 是节点的当前胜率估计（注意，如前所述，要考虑当前是黑棋走还是白棋走！），N 是节点的访问次数。C 是一个常数。C 越大就越偏向于广度搜索，C 越小就越偏向于深度搜索。注意对于原始的 UCT 有一个理论最优的 C 值，但由于我们的目标并不是最小化“遗憾”，因此需要根据实际情况调参。

我们看例子说明这是什么意思，就看之前的图吧。假设根节点是轮到黑棋走。那么我们首先需要在 7/10、5/8、0/3 之间选择：

1. 其中 7/10 对应的分数为 $7/10 + C \cdot \sqrt{\frac{\log(21)}{10}} \approx 0.7 + 0.55C$。
2. 其中 5/8 对应的分数为 $5/8 + C \cdot \sqrt{\frac{\log(21)}{8}} \approx 0.625 + 0.62C$。
3. 其中 0/3 对应的分数为 $0/3 + C \cdot \sqrt{\frac{\log(21)}{3}} \approx 0 + 1.00C$。
4. 可以注意到，C越大，就会越照顾访问次数相对较少的子节点。

如果 C 比较小，我们将会选择 7/10，接着就要在 2/4 和 5/6 间选择。注意，由于现在是白棋走，需要把胜率估计倒过来：

1. 其中 2/4 对应的分数为 $(1 - 2/4) + C \cdot \sqrt{\frac{\log(10)}{4}} \approx 0.5 + 0.76C$。
2. 其中 5/6 对应的分数为 $(1 - 5/6) + C \cdot \sqrt{\frac{\log(10)}{6}} \approx 0.17 + 0.62C$。

那么我们下一步肯定应该选 2/4。所以说之前的图是错误的，因为制图的人并没有注意到要把胜率倒过来（有朋友会说是不是可以认为它的白圈代表白棋的胜率，但这样它的回溯过程就是错的）。

最后，AlphaGo 的策略网络，可以用于改进上述的分数公式，让我们更准确地选择需扩展的节点。而 AlphaGo 的价值网络，可以与快速走子策略的模拟结果相结合，得到更准确的局面评估结果。注意，如果想写出高效的程序，这里还有很多细节，因为策略网络和价值网络的运行毕竟需要时间，我们不希望等到网络给出结果再进行下一步，在等的时候可以先做其它事情，例如 AlphaGo 还有一个所谓 Tree policy，是在策略网络给出结果之前先用着。

相信大家现在就可以写出正确的 MCTS 程序了（注意：最终应该选择访问量最大的节点，而不是胜率最高的节点，简单地说是因为访问量最大的节点的结果更可靠）。如果要写一个高效的程序，你会发现必须自己写一个内存池。关于 MCTS 还有很多话题可以说，这篇就到这里吧。

本系列已更新多篇，其它几篇的传送门：

28 天自制你的 AlphaGo（一）

28 天自制你的 AlphaGo（二）：训练策略网络，真正与之对弈

28天自制你的AlphaGo（三）：对策略网络的深入分析以及它的弱点所在

28天自制你的AlphaGo（四）：结合强化学习与深度学习的Policy Gradient（左右互搏自我进化的基础）

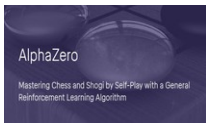
相关文章

蒙特卡洛树搜索 AlphaGo 深度学习 围棋

DeepMind 资深研究员黄士杰发表临别感言，宣布正式

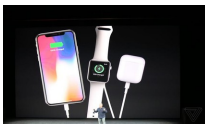
DeepMind 推出 AlphaGo 围棋教学工具，围棋学习新

马云称每天晚上都睡不好，担心阿里被淘汰；AlphaGo



「阿尔法狗」再进化！通用算法AlphaZero再攻克几种

火绒马刚口述：站着能挣钱，是我坚定的第二个信念



苹果警告：iPhone X无线充电可能会毁坏信用卡

爱奇艺泡泡社区发布战绩讲解背后社交逻辑

爱奇艺泡泡社区发布战绩讲解背后社交逻辑

文章点评：

我有话要说.....

同步到新浪微博

提交

热门关键字

热门标签 人工智能 机器人 机器学习 深度学习 金融科技 未来医疗 智能驾驶 自动驾驶 计算机视觉 激光雷达 图像识别 智能音箱 区块链 智能投顾 医学影像 物联网 IoT CV 微信小程序平台 微信小程序在哪 CES 2017 CES 2016年最值得购买的智能硬件 2016 互联网 小程序 微信朋友圈 抢票软件 智能手机 智能家居 智能手环 智能机器人 智能电视 360智能硬件 智能摄像机 智能硬件产品 智能硬件发展 智能硬件创业 黑客 白帽子 大数据 云计算 新能源汽车 无人驾驶 无人机 大疆 小米无人机 特斯拉 VR游戏 VR电影 VR视频 VR眼镜 VR购物 AR 直播 扫地机器人 医疗机器人 工业机器人 类人机器人 聊天机器人 微信机器人 微信小程序 移动支付 支付宝 P2P 区块链 比特币 风控 高盛 人脸识别 指纹识别 黑科技 谷歌地图 谷歌 IBM 微软 乐视 百度 三星s8 腾讯 三星Note8 小米MIX 小米Note 华为 小米 阿里巴巴 苹果 MacBook Pro iPhone Facebook GAIR IROS 双创周 云栖大会 优葩 智能硬件公司 智能硬件 QQ红包 支付宝红包 敬业福 小米手表 ripple tsp 三维扫描 微信血糖仪 windows 10 pro insider preview 小牛电动车官网 无人机集群 x5max拆机 树莓派搭建nas 特斯拉续航里程 2015科幻 小米和360摄像头哪个好 米mix 恩智浦 更多

0