1. 搜索求解

a) 启发式搜索

b) 对抗搜索

c) 蒙特卡洛树搜索

2. 搜索算法的形式化描述：

状态：城市

动作：从一个城市转移到下一个城市的操作

状态转移：某一状态进行某一动作后，到达的状态

路径：一系列状态的集合

测试目标：评估当前状态是否未所求解的目标状态

3. 启发式搜索(有信息搜索)

在搜索的过程中有与所求解问题相关的辅助额外信息

例如，辅助信息：任意一个状态与目标状态之间的距离

代表算法有：贪婪最佳优先搜索、A\*搜索

辅助信息：所求解问题之外、与所求解问题相关的特定信息或知识

评价函数f(n)：从当前节点𝑛出发，根据评价函数来选择后续节点

启发函数h(n)：计算从节点𝑛到目标节点之间所形成路径的最小代价值。这里将两点之间的直线距离作为启发函数。

评估函数 == 评价函数

启发函数 == 最小开销代价

4. 贪婪最佳优先搜索：评价函数f(n) = 启发函数h(n)

不足之处：

1. 贪婪最佳优先搜索不是最优的。经过Sibiu到Fagaras到Bucharest的路径比经过Rimnicu Vilcea到Pitesti到Bucharest的路径要长32公里。
2. 启发函数代价最小化这一目标会对错误的起点比较敏感。考虑从Iasi到Fagaras的问题，由启发式建议须先扩展Neamt，因为其离Fagaras最近，但是这是一条存在死循环路径。
3. 贪婪最佳优先搜索也是不完备的。所谓不完备即它可能沿着一条无限的路径走下去而不回来做其他的选择尝试，因此无法找到最佳路径这一答案。
4. 在最坏的情况下，贪婪最佳优先搜索的时间复杂度和空间复杂度都是O(𝑏𝑚)，其中𝑏是节点的分支因子数目、𝑚是搜索空间的最大深度。

5. A\*算法：评价函数f(n) = 当前最小开销代价g(n) + 后续最小开销代价h(n)

如果A\*算法将节点n选择作为具有最小代价开销的路径中一个节点，则𝑛一定是最优路径中的一个节点。即最先被选中扩展的节点在最优路径中

证明：反证法。假设上述结论不成立。则存在一个未被访问的节点𝑛′位于从起始节点到节点𝑛的最佳路径上。根据非递减性质，存在𝑓(𝑛) ≥𝑓(𝑛′) ，则𝑛′应该已经被访问过了（expanded）。因此，无论什么时候，一旦一个节点被访问到，它一定位于从起始节点到它自己之间的最佳路径上。

6. 智能体(agents)

7. 零和博弈

所谓零和博弈是博弈论的一个概念，属非合作博弈。指参与博弈的各方，在严格竞争下，一方的收益必然意味着另一方的损失，博弈各方的收益和损失相加总和永远为“零”，双方不存在合作的可能。与“零和”对应，“双赢博弈”的基本理论就是“利己”不“损人”，通过谈判、合作达到皆大欢喜的结果。

8. 对抗搜索/博弈搜索

最小最大搜索(Minimax Search)

Alpha-Beta剪枝搜索(Pruning Search)

蒙特卡洛树搜索(Monte-Carlo Tree Search): 通过采样而非穷举方法来实现搜索

9. 最小最大搜索(Minimax Search)

Complete ? Yes (if tree is finite)

Optimal ? Yes (against an optimal opponent)

Time complexity ? O(𝑏𝑚)

Space complexity ? O(b×𝑚) (depth-first exploration)

m 是游戏树的最大深度，在每个节点存在b个有效走法

缺点：如果搜索树极大，则无法在有效时间内返回结果

10. Alpha-Beta剪枝搜索(Pruning Search)

Alpha值(𝛼)：MAX节点目前得到的最高收益

Beta值(𝛽)：MIN节点目前可给对手的最小收益

𝛼和𝛽的值初始化分别设置为−∞和∞

Alpha值(𝛼)：

玩家MAX（根节点）目前得到的最高收益

假设𝑛是MIN节点，如果𝑛的一个后续节点可提供的收益小于𝛼，则𝑛及其后续节点可被剪枝

Beta值(𝛽)：

玩家MIN目前给对手的最小收益

假设𝑛是MAX节点，如果𝑛的一个后续节点可获得收益大于𝛽，则𝑛及其后续节点可被剪枝

𝛼和𝛽的值初始化分别设置为−∞和∞

•剪枝本身不影响算法输出结果

•节点先后次序会影响剪枝效率

•如果节点次序“恰到好处”，Alpha-Beta剪枝的时间复杂度为O(𝑏𝑚2)，最小最大搜索的时间复杂度为O(𝑏𝑚)

11. 蒙特卡洛树搜索(Monte-Carlo Tree Search): 通过采样而非穷举方法来实现搜索

alphago：深度学习、强化学习、蒙特卡洛树搜索

🟎单一状态蒙特卡洛规划： 多臂赌博机 (multi-armed bandits)

🟎上限置信区间策略 (Upper Confidence Bound Strategies, UCB)

🟎蒙特卡洛树搜索 (Monte-Carlo Tree Search)

🟎UCT (Upper Confidence Bounds on Trees)

12. 单一状态蒙特卡洛规划： 多臂赌博机 (multi-armed bandits)

k个赌博机，1个玩家

多臂赌博机问题是一种序列决策问题

这种问题需要在利用(exploitation)和探索(exploration)之间保持平衡

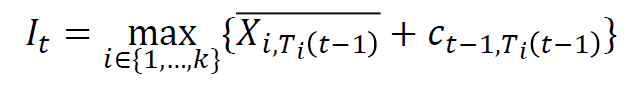
利用(exploitation) ：保证在过去决策中得到最佳回报

探索(exploration) ：寄希望在未来能够得到更大回报

悔值函数：上帝视角 - 玩家选择

一个良好的多臂赌博机操作的策略是在不同人进行了多次玩法后，能够让悔值函数的方差最小

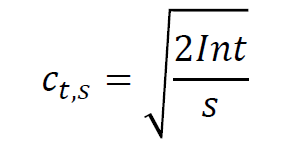
13. 上限置信区间策略 (Upper Confidence Bound Strategies, UCB)



𝑋𝑖,𝑇𝑖(𝑡−1)来记录第𝑖个赌博机在过去𝑡−1时刻内的平均奖赏

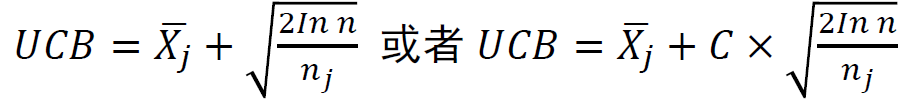
𝑇𝑖(𝑡)为在过去时刻（初始时刻到𝑡时刻）过程中选择第𝑖个赌博机的次数总和。

其中𝑐𝑡,𝑠取值定义如下：



利用：X

探索：C

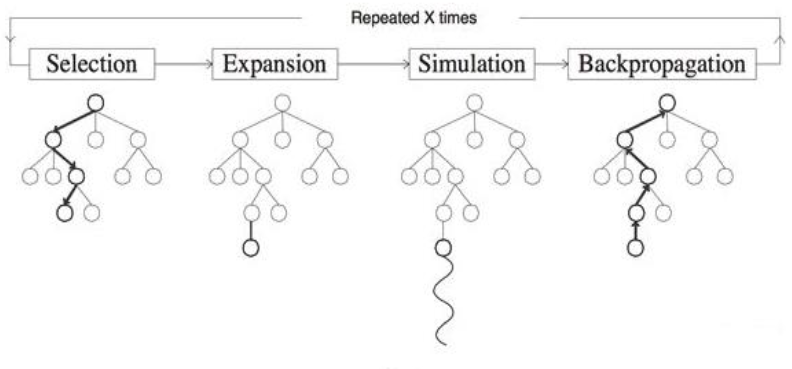


在探索-利用（exploration-exploitation）之间寻找平衡：既需要拉动在过去时间内获得最大平均奖赏的赌博机，又希望去选择那些拉动臂膀次数最少的赌博机。

14. 蒙特卡洛树搜索

将上限置信区间算法UCB应用于游戏树的搜索方法，由Kocsis和Szepesvari在2006年提出

包括了四个步骤：选举(selection)，扩展(expansion)，模拟(simulation)，反向传播(Back-Propagation)



选择：

在UCB基础上，在利用和探索之间选择最具潜力的后续节点

拓展：

在未被扩展的节点上，拓展节点

模拟：

模拟仿真游戏对抗，直到博弈游戏结束

反向传播：

用模拟所得结果来回溯更新导致这个结果的每个节点中获胜次数和访问次数

两种策略学习机制：

搜索树策略: 从已有的搜索树中选择或创建一个叶子结点（即蒙特卡洛中选择和拓展两个步骤）.搜索树策略需要在利用和探索之间保持平衡。

模拟策略：从非叶子结点出发模拟游戏，得到游戏仿真结果。

Monte-Carlo Tree Search (MCTS)：蒙特卡洛树搜索基于采样来得到结果、而非穷尽式枚举（虽然在枚举过程中也可剪掉若干不影响结果的分支）。

15. 蒙特卡洛树搜索算法 (Upper Confidence Bounds on Trees , UCT)



16. AlphaGo 1.0算法解读：

输入：3000万人类选手的对局局面

两个会下棋的智能软件

David Silver, et.al., Mastering the game of Go with Deep Neural Networks and Tree Search, Nature, 529:484-490,2016

🟎将每个状态（局面）均视为一幅图像

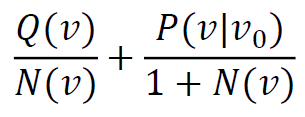
🟎训练策略（policy）网络和价值（value）网络

🟎𝑝𝜎,𝑝(𝑎|𝑠)表示当前状态为𝑠（局面）时，采取行动𝑎后所获胜的概率； 𝑣𝜃(𝑠’)表示当前状态为𝑠′时，整盘棋获胜的概率。

基于监督学习来先训练策略网络

再基于强化学习来训练策略网络

在通过深度学习得到的策略网络和价值网络帮助之下，如下完成棋局局面的选择和搜索。给定节点𝑣0，将具有如下最大值的节点𝑣选择作为𝑣0的后续节点



🟎这里𝑃(𝑣|𝑣0) 的值由策略网络计算得到。

🟎在模拟策略阶段(default policy)，AlphaGo不仅考虑仿真结果，而且考虑价值网络计算结果。

🟎策略网络和价值网络是离线训练得到的。

Mastering the game of Go without human knowledge, Nature, volume 550, pages 354–359 (19 October 2017)

AlphaGo Zero （一张白纸绘蓝图）

🟎不需要人类选手对决的棋面进行训练

🟎策略网络和价值网络合并

🟎深度残差网络

经过40天训练后，Zero总计运行约2900万次自我对弈，得以击败AlphaGo Master，比分为89比11

17. 逻辑与推理(1)

命题逻辑

谓词逻辑

知识图谱推理：一阶归纳推理算法

因果推理

18. 命题逻辑

🟎命题逻辑(proposition logic)是**应用一套形式化规则对以符号表示的描述性陈述进行推理**的系统。

🟎在命题逻辑中，一个或真或假的描述性陈述被称为**原子命题**，对原子命题的内部结构不做任何解析。

🟎若干原子命题可通过逻辑运算符来构成复合命题。

可通过命题联结词(connectives）对已有命题进行组合，得到新命题。这些通过命题联结词得到的命题被称为复合命题(compound proposition）

