搜索与求解

搜索算法基础

搜索：依据已有信息来寻找满足约束条件的待求解问题

<状态、动作、状态转移、路径/代价、目标测试>

完备性	当问题存在解时，算法能否保证找到一个解
最优性	搜索算法能否保证第一个找到的解是最优解

- [ ] 时间复杂度和空间复杂度通常用**big O notation**来描述

符号	**含义**
b	分支因子，即搜索树中每个节点的最大分支数目
d	根节点到最浅的目标结点的路径长度
m	搜索树中路径的最大可能长度
n	状态空间中状态的数量

树搜索：不具有完备性（环路）

图搜索：具有完备性 （边缘集合中所有产生环路的节点都要被剪枝，但不会排除所有潜在的可行解）

bfs:

- 时间复杂度：$O(b^d)$
- 空间复杂度：$O(b^d)$
- 完备性：yes
- 最优性：depends on the cost (Yes, if cost =1 per step, not optimal in general)

  个人理解：cost=1时，因为是一层一层找的，bfs可以保证最优性

dfs:

- 时间复杂度： $O(b^m)$
- 空间复杂度：$O(bm)$ 到达目标的layer不会高于m，假设恰为到了最底，上面只把每个节点的第一个孩子节点作为后继继续向下挖，而遍历了所有的孩子节点
- 完备性：No 有环，自己绕
- 最优性：No

bfs: 先进先出

dfs: 后进先出

启发式搜索

1 启发函数与评价函数

评价函数f(n)：从当前点n出发，根据评价函数来选择后续结点

启发函数h(n)：计算从结点n到目标节点之间所形成路径的最小代价值

贪婪最佳优先搜索：个人理解就是每次探索相邻available节点，每步都选择最小代价的

在该视角下，启发函数 == 代价函数

时间空间复杂度均为$O(b^m)$

代价函数g(n)表示从起始节点到结点n的开销代价值

A*算法：

选择下一个结点的标准变为了比较评价函数（f = g + h）的大小，比较的时候是对所有结点都进行比，因此可以产生“回退”的效果

算法分析详见ppt和补档（不考））

对抗搜索

最小最大搜索

- 完备性：yes (if the tree is finite)
- 最优性：yes (against an optimal opponent)
- 被视为depth-first exploration
- 时间复杂度：$O(b^m)$
- 空间复杂度：$O(bm)$

α剪枝：

更高的min层收益为α，其兄弟的儿子的儿子（也为min层）收益如果也小于α，可剪枝该节点的剩余儿子

β剪枝正好反过来

观察能收到收益的两个节点都属于一个类层，然后根据这个类，如果上位者更能**向上传递**（满足相反的需求），下位者的剩余孩子可被剪枝

- min层下位者能够收到的收益小于α
- max层下位者能够收到的收益大于β

算法流程：

根结点（MAX结点）的α值和β值分别被初始化为$-\infty$和$+\infty$。

子节点继承父节点的α值和β值，再按照以下规则进行更新

对于MAX节点，如果其孩子结点（MIN结点）的收益大于当前的α值（极大层的下界），则将α值更新为该收益；对于MIN结点，如果其孩子结点（MAX结点）的收益小于当前的β值（极小层的上界），则将β值更新为该收益。

随着搜索算法不断被执行，每个结点的α值和β值不断被更新。大体来说，每个结点的$[\alpha,\beta]$从其父结点提供的初始值开始，取值按照如下形式变化：α逐渐增加、β逐渐减少。不难验证，如果一个结点的α值和β值满足$\alpha > \beta$的条件，则该结点尚未被访问的后续结点就可以被剪枝（结合前面两个引理加以说明）。

α初始为负无穷，β初始为正无穷

max节点要更新α，往大了更新

min节点要更新β，往小了更新

更新都是通过对孩子结点的实值和当前节点的α或者β而进行的

- [ ] 随便写一个alpha beta剪枝树，两个人练习一下

蒙特卡洛树搜索

悔值(regret)函数

$\rho_T = Tu^* - \sum_{t=1}^{T} \hat{r}_t$

将前T次操作中**最优**策略的期望得分减去智能体的实际得分，就是悔值函数的结果。显然，问题求解的目标为最小化悔值函数的期望，该悔值函数的取值取决于智能体所采取的策略。

上限置信区间算法（UCB）

用霍夫丁不等式估计一个上限

在第t次时选择使得下面式子取值最大的动作a_{l_t}，其中l_t由如下式子计算得到：

$$l_t = \text{argmax}_i \, \bar{x}_{i,T_{(i,t-1)}} + C\sqrt{\frac{2\ln t}{T_{(i,t-1)}}}$$

在过去第t次已经对动作a_i探索了$T_{(i,t-1)}$次，在当前问题中对应摇动了第i个赌博机的臂膀$T_{(i,t-1)}$次，执行动作a_i所收到收益分数的均值为$\bar{x}_{i,T_{(i,t-1)}}$

- 选择(selection)：从搜索树的根节点开始，向下递归选择子节点，直至到达**叶子节点**或者到达**具有还未被扩展过的子节点的节点L。**这个向下递归选择过程可由**UCB1算法**来实现，在递归选择过程中记录下每个节点被选择次数和每个节点得到的奖励均值
- 扩展(expansion)：如果节点L不是一个终止节点（或对抗搜索的终局节点），则**随机扩展**它的一个未被扩展过的后继边缘节点M
- 模拟(simulation)：从节点M出发，模拟扩展搜索树，直到找到一个终止节点，模拟过程使用的策略和采用UCB1算法实现的选择过程并不相同，前者通常使用简单的策略，如**随机策略**
- 反向传播(back propagation)：用模拟所得结果（终止节点的代价或游戏终局分数）回溯更新模拟路径中M以上（含M）节点的**奖励均值**和**被访问次数**

反向传播时：MIN层结点现有总分加上终局得分分数， MAX层结点现有总分 减去终局得分分数

> 6.16:
>
> 查UCB都是查子的UCB，因为是个选择的过程，计算均值后都是加次数评测，ln内都是看当前点的访问次数
>
> 对于书上的例子，max起手，获胜max-1，min+1，失败不变