搜索

- 无信息搜索
- 启发式搜索

搜索算法基础

搜索基本问题和求解

- 状态
- 动作
- 状态转移
- 路径和代价
- 目标测试

搜索算法的评价指标

1. 完备性：是否能找到一个解
2. 最优性：搜索算法是否能保证找到的第一个解是最优解
3. 时间复杂度
4. 空间复杂度

符号	含义
b	分支因子，即搜索树中每个节点最大的分支数目
d	根节点到最浅的目标节点的路径长度
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

启发式搜索（有信息搜索）

启发函数$h(n)$：用于估计节点n距离达成还需付出多少代价
评价函数$f(n)$：决定了搜索算法拓展节点n的优先度

贪婪最佳优先搜索

评价函数:$f(n)=h(n)$,即每次都选收益最大的情况

- 特点：完备（即一定能得到一个解），但并不一定是最优的
- 最坏情况下时间复杂度：$\mathcal{O}(b^n)$

- 最坏情况下空间复杂度：$\mathcal{O}(b^n)$

A^*搜索

评价函数：$f(n)=h(n)+g(n)$
$g(n)$取值为从起始节点到节点n的路径代价
特点：如果启发函数满足**可容性**和**一致性**，那么由A^*算法得到的一定是最优解

算法性能分析

- 可容性
  对任意节点n,有$h(n)\leq h^*(n)$，对目标节点来说$h(n)=0$
- 一致性
  $h(n)\leq c(n,a,n')+h(n')$
  满足一致性一定满足可容性
- 引理1：启发函数满足一致性条件时，评价函数对搜索树中找到的节点序列中的每一个节点按照先后次序依次单调非减
- 引理2：在结点n被加入搜索树后，若对应状态t的任何结点n'出现在边缘集合中，则必有$g(n)\leq g(n')$
- 算法最优性：对于任意一个状态t,它第一次被加入搜索树的时候的路径必然是最短路径。
- 最坏情况下空间复杂度：$\mathcal{O}(b^{d+1})$，甚至大于广度优先搜索，但由于在实际过程中可以剪枝，所以大概率可以的减少时空复杂度。

对抗搜索

最大最小搜索

$$minimax(s)=\begin{cases}utility(s) & if\ terminal\_test(s)\text{（如果已经下完了）}\\ \max_{a\in actions(s)} minimax(result(s,a)) & player\ Max\\ \min_{a\in actions(s)} minimax(result(s,a)) & player\ Min\end{cases}$$

即在计算搜索树中的每个节点分数之后，每一步由玩家根据自己的角色来选择使得分数最大或分数最小的行动。

- 如果对手总是理性地按照最优策略来交替采取行动，那么最大最小算法得到的解是最优的
- 时间复杂度 $\mathcal{O}(b^m)$,m为游戏树的最大深度
- 空间复杂度 $\mathcal{O}(bm)$

Alpha-Beta 剪枝搜索

用于减少最大最小搜索中没有必要访问的结点，以最大可能地提高最大最小搜索的效率

- 剪枝搜索不影响最大最小搜索最后的结果
- 最优效率下拓展的节点数量$\mathcal{O}(b^{\lceil\frac{m+1}{2}\rceil})$,m为偶数时，为$\mathcal{O}(b^{\frac{m}{2}})$

蒙特卡洛树搜索

探索与利用机制的平衡

多臂赌博机问题

- 状态：s
- 动作：a
- 状态转移：
- 奖励(reword)：$D_i, \mu_i, l_t, \hat{r_t}, D_{l_t}$.
  其中$\hat{r_t}$被称为第t次行动的奖励
- 悔恨(regret)：问题求解的目标即在于最小化悔恨函数的期望
  根据智能体前T次动作，可以定义悔恨函数：

$$\rho_T = T\mu^* - \sum_{i=1}^{T} \hat{r_i}$$

其中

$$\mu^* = \max_{i=1,...,K} \mu_i$$

要想实现悔恨函数的最小化，与先前的策略不同，需要一边探索一边调整自己的策略，争取获得最高的利益。

贪心算法：

- 在t次行动时，计算$\bar{x}_{i,T_{(i,t-1)}}$最大的赌博机进行摇动。
  （即选择t次后平均值高的赌博机猛摇）
- 缺点：很可能陷入到某个值还算高的陷阱中，而因其余的赌博机探索数量很少，以至于来不及“展现”出其收益的优越性而被该算法放弃。
- 探索利用困境：**探索**和**利用**之间存在对立关系，需要采取其他的方法来平衡两者之间的关系。

ϵ-贪心算法

- 实现方法：
  以$1-\epsilon$的概率去选择过去平均回报最大的赌博机
  以另外ϵ的概率去随机选择一个赌博机，即

$$l_t = \begin{cases} \arg\max_i \hat{x}_{i,T_{(i,t-1)}} & 以1-\epsilon的概率 \\ 随机的i \in \{1,2,...,K\} & 以\epsilon的概率 \end{cases}$$

- 不足：与被探索的次数无关，可能存在一个给出更好奖励期望的动作，但因为探索的次数少，而未被发现。

上限置信区间算法(UCB1)

- 实现方法：
  为每个动作的奖励期望起算一个估计范围，优先采用估计范围上限较高的动作。
- 霍夫丁不等式：

$$P(\mu_i - \bar{x}_{i,T_{(i,t-1)}} > \delta) \leq e^{-2T_{(i,t-1)}\delta^2}$$

令$t^{-4} = e^{-2T_{(i,t-1)}\delta^2}$, 反解出: $\delta = \sqrt{\frac{2\ln t}{T_{(i,t-1)}}}$

由于选不同的幂函数差一个乘积项参数C，所以最终的选择策略：

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

可以证明，UCB1算法悔恨函数的期望上界

$$O(\frac{K\ln T}{\Delta}), \Delta = \min_{i=1,\ldots,K;\mu_i<\mu^*} \mu^* - \mu_i;$$

蒙特卡洛树搜索算法（上限置信区间树搜索）

实现方法：

选择：从根节点出发，向下选择子节点，直至到达叶子结点或到达具有还未被拓展过的子节点L（选择子节点的过程可以采用$UCB1$算法）

扩展：如果节点L不是一个终止节点，则随机拓展它的一个未被拓展过的后继边缘节点。

模拟：从节点M出发，模拟拓展搜索树，直到找到一个终止节点（模拟进行一局游戏）。

反向传播：用模拟所得结果 回溯更新模拟路径中M以上节点的奖励均值和被访问次数。

优势：

并非基于穷尽式枚举，而是基于采样来决定搜索树的扩展方式。
既能保证搜索的效率，又能保证在一定程度上准确地评估每个分支的优劣，从而找到**近似**最优解。