第一章. 搜索算法

	完备性	最优解	时间复杂度	空间复杂度	Frontier
BFS	✓	若均匀搜索代价	$O(b^d)$	$O(b^d)$	Queue
DFS	✗ 节点无限深就找不到	✗ 多解就寄	$O(b^m)$	$O(bm)$ 空间复杂度小·占用内存少	Stack
IDS	✓	若均匀搜索代价	$O(b^d)$	$O(bd)$	Stack
UCS	✓	✓			Priority Queue 按 g(n) 排序 → 走过的距离
Greedy	✗	✗	最坏 $O(b^m)$ 最好 $O(bd)$	最坏 $O(b^m)$ 最好 $O(bd)$	Priority Queue 按 h(n) 排序 → 离目标估计
A*	✓	✓			Priority Queue 按 h(n)+g(n) 排序

搜索参数

b: 一个节点最多分支
m: 整个图最大深度
d: 解所在的深度

启发式属性　实际距离　估算距离

Admissible: $d(n) \geq h(n)$

Consistent: $d(n,p) \geq h(n)-h(p)$

$h(n)=0$ 将同时满足以上 2 个属性

CSP 约束满足问题

前提假设: single agent + fully observable + deterministic + discrete environment

局部搜索:

从完全放置开始，此时不需要满足条件

尝试重新分配变量来满足 constraint

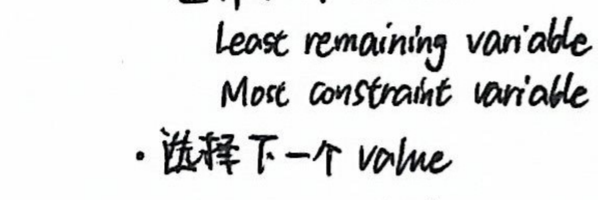
每次分配一个变量，使得冲突最少

启发式 Backtracking

- 选择下一个 variable
  - Least remaining variable (LRV)
  - Most constraint variable (MCV)
- 选择下一个 value
  - Least constraint assignment (LCA) 选择产生限制最少的值
- 早期失败检测
  - Forward checking 检查所有变量至少还有一个可选值
  - Arc consistency

第二章. Game

- Minimax & α-β 剪枝 (DFS 算法)



只要发生更新，就需要与父节点比较。
比父节点小就会被剪枝。
因为 Max 只会选更大的

	时间复杂度
Minimax	$O(b^m)$
α-β	$O(b^{m/2})$

- 纳什均衡

囚徒困境 (鼓励不合作)

	A招	A不招
B招	-5,-5	-10,0
B不招	0,-10	-1,-1

猎鹿游戏 (鼓励合作)

	A鹿	A兔
B鹿	2,2	1,0
B兔	0,1	1,1

示弱游戏

	A直	A转
B直	-10,-10	-1,1
B转	1,-1	0,0

- 定理: 任何博弈，只要选择有限，则毕然有一个纳什均衡

If a player has dominant strategy. 纳什为 this player use dominant, the other one use best respond

If both player has dominant strategy. 纳什为两人都用自己的 dominant strategy

第三章. 贝叶斯

- 基本公式 $P(A|B)=\frac{P(A\cap B)}{P(B)} \Leftrightarrow P(A\cap B)=P(A|B)P(B)=P(B|A)\cdot P(A)$

$P(A\cap B)=P(A)\cdot P(B)$ 相互独立

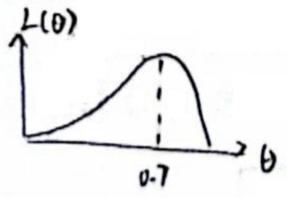
$P(B)=P(B|A_1)P(A_1)+P(B|A_2)\cdot P(A_2)\cdots P(B|A_n)\cdot P(A_n)$

$P(A|B)=\frac{P(B|A)\cdot P(A)}{P(B|A)\cdot P(A)+P(B|\neg A)\cdot P(\neg A)}$

- 最大类似估计 (MLE)　扔硬币，7次人上，3次数字上

$\theta=\arg\max P(X_0|\theta)$　$L(\theta)=\theta^7(1-\theta)^3$

具体事件　概率



- 最大后验概率 (MAP)　垃圾 / 非垃圾　$P(H|X)=\frac{P(X|H)P(H)}{P(X)}$

$\theta=\arg\max P(\theta|X_0)$

$\Downarrow$

$\theta=\arg\max P(X_0|\theta)\cdot P(\theta)$

$P(X|H)=\frac{25}{30}=\frac{5}{6}$

$P(H)=\frac{30}{100}=\frac{3}{10}$

$P(X)=\frac{35}{100}=\frac{7}{20}$

$\Rightarrow P(H|X)=\frac{5}{7}$

总100，正常70，垃圾30
非正常中出现16
垃圾中出现25
非垃圾的概率为 5/7

- 朴素贝叶斯

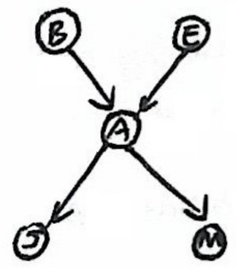
$P(X=(0,2)|Y=1)=P(X=0|Y=1)\cdot P(X=2|Y=1)=\frac{6}{16}$

$P(X=(0,2)|Y=0)=P(X=0|Y=0)\cdot P(X=2|Y=0)=\frac{1}{36}$

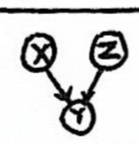
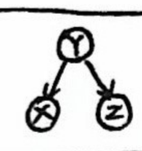
谁大 $X=(0,2)$ 就是哪一类 Y

第四章.

- 贝叶斯网络



- 基本概念

	
$P(Z,X)=P(Z)P(X)$ 是独立	$P(Z,X)=P(Z\|Y)P(X\|Y)P(Y)$ 不是独立
$P(Z,X\|Y)=\frac{P(Y\|X,Z)P(X)P(Z)}{P(Y)}$ $\neq P(Z\|Y)P(X\|Y)$ 不是条件独立	$P(Z,X\|Y)=P(Z\|Y)P(X\|Y)$ 是条件独立

链式法则: $P(A_1,A_2\cdots A_n)=\prod_{i=1}^{n}P(A_i|Parent(A_i))$

$=\prod_{i=1}^{n}P(A_i|A_1,\cdots A_{i-1})$

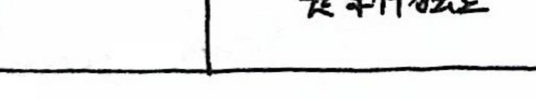
$=P(A_1)\cdot P(A_2|A_1)\cdot P(A_3|A_1,A_2)\cdots P(A_n|A_1\cdots A_{n-1})$

$P(J,M,A,B,E)=P(B)\cdot P(E)\cdot P(A|B,E)\cdot P(J|A)\cdot P(M|A)$

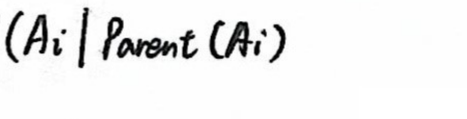
$P(B,J)=\sum_E\sum_A\sum_M P(B,E,A,J,M)$

隐马尔可夫模型 HMM

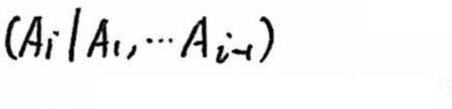
- Filtering: 根据观察序列 $e_{1:t}$ 推测当前状态 X_t 的分布



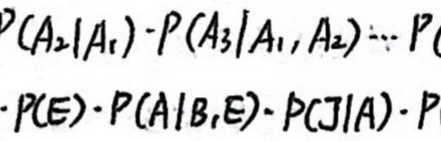
- Smoothing: 根据观察序列 $e_{1:t}$ 推测某一状态 X_k 的分布



- Evaluation: 计算特点 $e_{1:t}$ 序列的可能性



- Decoding: 根据 $e_{1:t}$ 计算最可能的 $X_{0:t}$

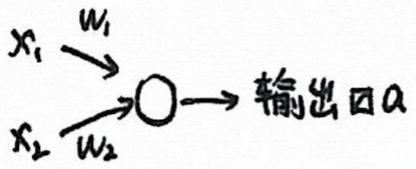


Transition Matrix 状态转移矩阵: 从一个 state 到另一个 state 的概率

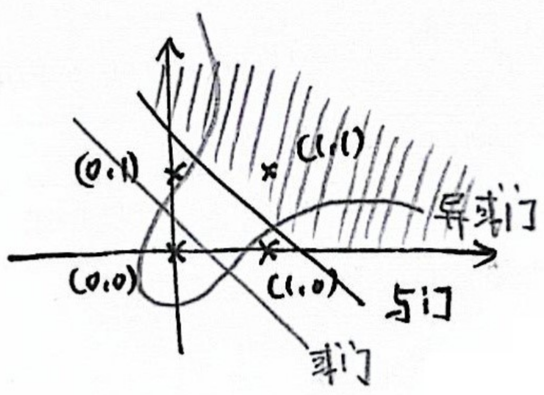
Emission Probability 映射矩阵: 从一个 state 到观测现象的概率

第四章

- 感知机 Perceptron



$$\text{输出} = \begin{cases} 0, & \vec{w}\cdot\vec{x}+b \le 0 \\ 1, & \vec{w}\cdot\vec{x}+b > 0 \end{cases}$$ ← 域值



可做为以下门电路

与门	AND Gate
与非门	NAND Gate
或门	OR Gate

但无法做为异或门 XOR

第五章

- 马尔可夫决策过程 MDP

组成：
- state s
- Action a　每个state s 有可选的行动 A(s)
- Transition model $P(s'|s,a)$
- Reward function $R(s)$

解 solution：
- Policy $\pi(s)$

- 贝尔曼公式

$$U(s) = R(s) + \gamma \max \sum P(s'|s,a)\,U(s')$$

$\gamma \in (0,1)$

$$U(s_0,s_1,s_2\cdots) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots = \sum_{t=0}^{\infty} \gamma^t R(s_t) \le \frac{R_{max}}{1-\gamma}$$

- 强化学习

Model-based: Learn the model of MDP (transition probability and reward) and try to solve the MDP correctly

Model-freed: Learn how to act without explicitly learning the transition probabity $P(s'|s,a)$

Q-learning: learn an action-utility function $Q(s,a)$ that tells us the value of doing action a in state s

Model-based:

Exploration 探索（采取一个后果未知的 action）

好处：{获取更准确的环境模型 / 可能发现最多的 reward}

坏处：{探索不是 maximize utility / 可能遇到坏情况}

Exploitation 利用（Go with the best strategy found so far）

处好：{maximize utility / 避免遇到坏情况}

坏处：可能没探索到未知的最佳策略

Mode-free

Q-learning

TD (Temporal difference)

$$Q^{local}(s,a) = R(s) + \gamma \max Q(s',a')$$

$$Q^{new}(s,a) = (1-\alpha)\,Q(s,a) + \alpha \cdot Q^{local}(s,a)$$

第六章

- 深度学习

如何训练神经网络.

1. 从 training dataset 中随机选择一个 traing token (x_i, y_i)
2. 计算 neural net prediction $f(x_i)$
3. 计算 loss 例如 $L = -\log f_{y_i}(x_i)$
4. Back-propagate 寻找梯度 $\frac{\partial L}{\partial W^{(2)}}$ 和 $\frac{\partial L}{\partial W^{(1)}}$
5. 梯度更新 $W^{(l)} \leftarrow W^{(l)} - \eta \frac{\partial L}{\partial W^{(l)}}$
6. 重复过程直至 loss 足够小.