今天的内容

- 期末复习

- 复习内容:
  - 搜索算法
  - 约束满足问题求解
  - 博弈树
  - 命题逻辑
  - 概率方法

智能体(Agent)有关概念

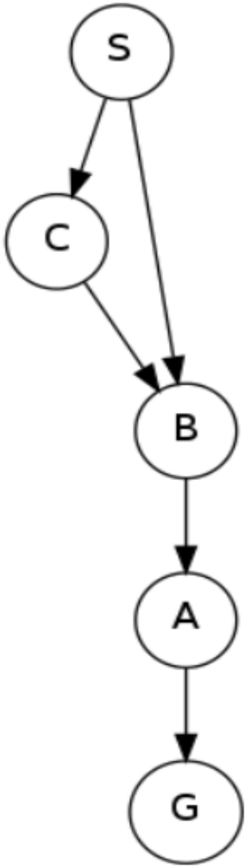
- 智能体(Agent)
  - 一个能感知和行动的实体；关键是如何实现一个智能体函数
- 智能体函数(Agent function)
  - 给定每个可能的感知序列，规定了智能体相应的行动
- 智能体程序(Agent program)
  - 结合机器的物理框架，程序实现了一个智能体函数
- 合理性(Rationality)
  - 一个属性，根据最大化的期望利益值来选择行动
  - 举例：只能采集到环境状态的部分信息的智能体不可能是非常合理的?
- 自主性(Autonomy)
  - 一个属性，智能体的行为是由自己的经验决定的，而不是仅由初始程序决定的

搜索问题里的概念

- 状态
  - 智能体所在的一个处境。两类状态：真实世界的，和抽象代表的（被智能体用于推测行动的）
- 搜索树
  - 一个树，根节点是开始状态，子节点代表可以通过行动达到的状态
- 搜索节点
  - 搜索树里的一个节点
- 目标状态
  - 智能体想要到达的状态
- 转移模型
  - 描述智能体给定一个状态后的选项（包括一组(行动，状态)对，即行动可以达到的状态）
- 分支因子
  - 搜索树中，智能体可选的行动数

搜索树

- 画出搜索树

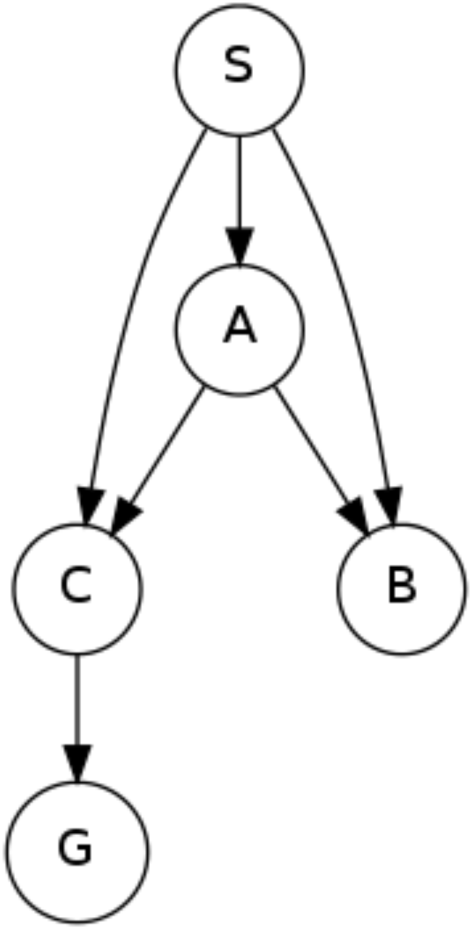


树搜索 vs 图搜索

- 图搜索把扩展过的节点存下来，以后不会再扩展这些节点；而树搜索则有可能会重复扩展这些节点。

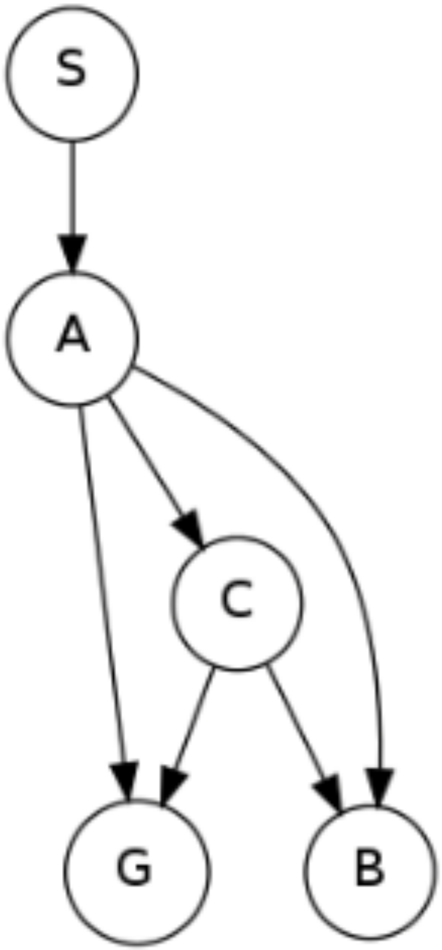
深度优先图搜索

- 返回路径



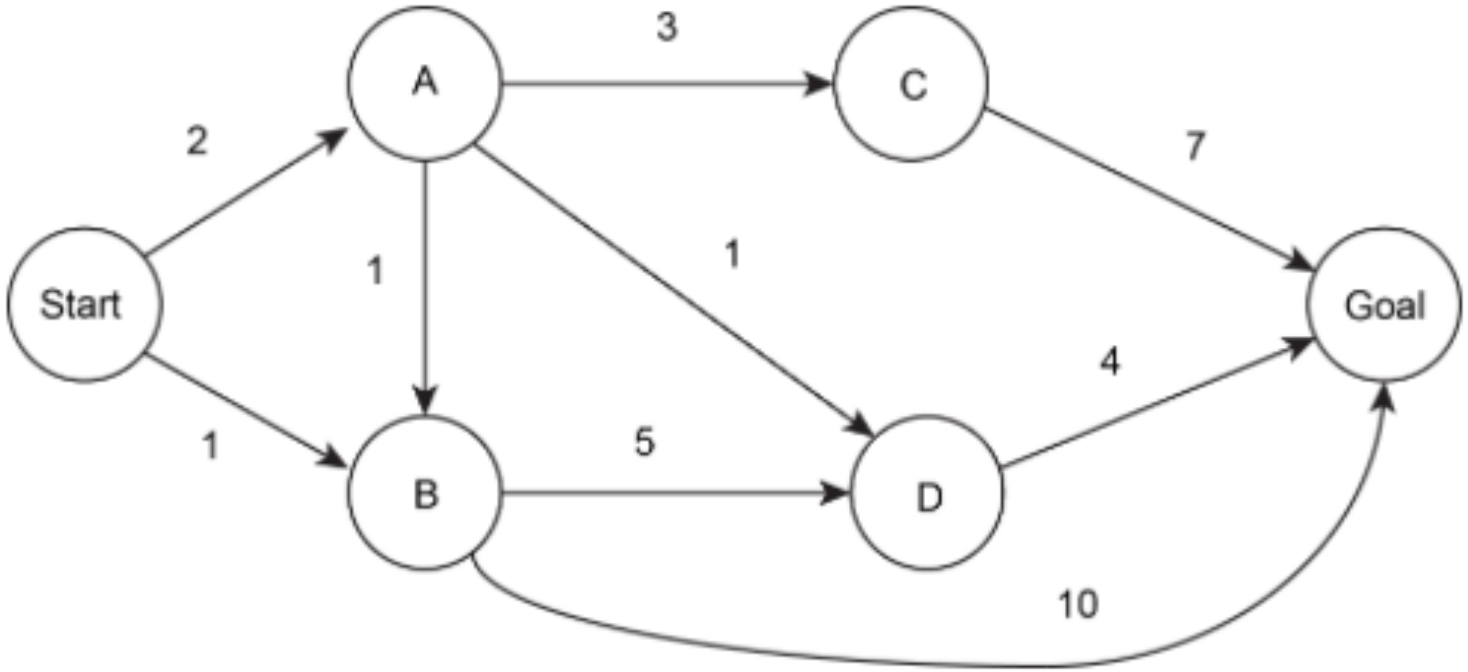
广度优先图搜索

- 返回路径



基于成本的统一图搜索

作业习题

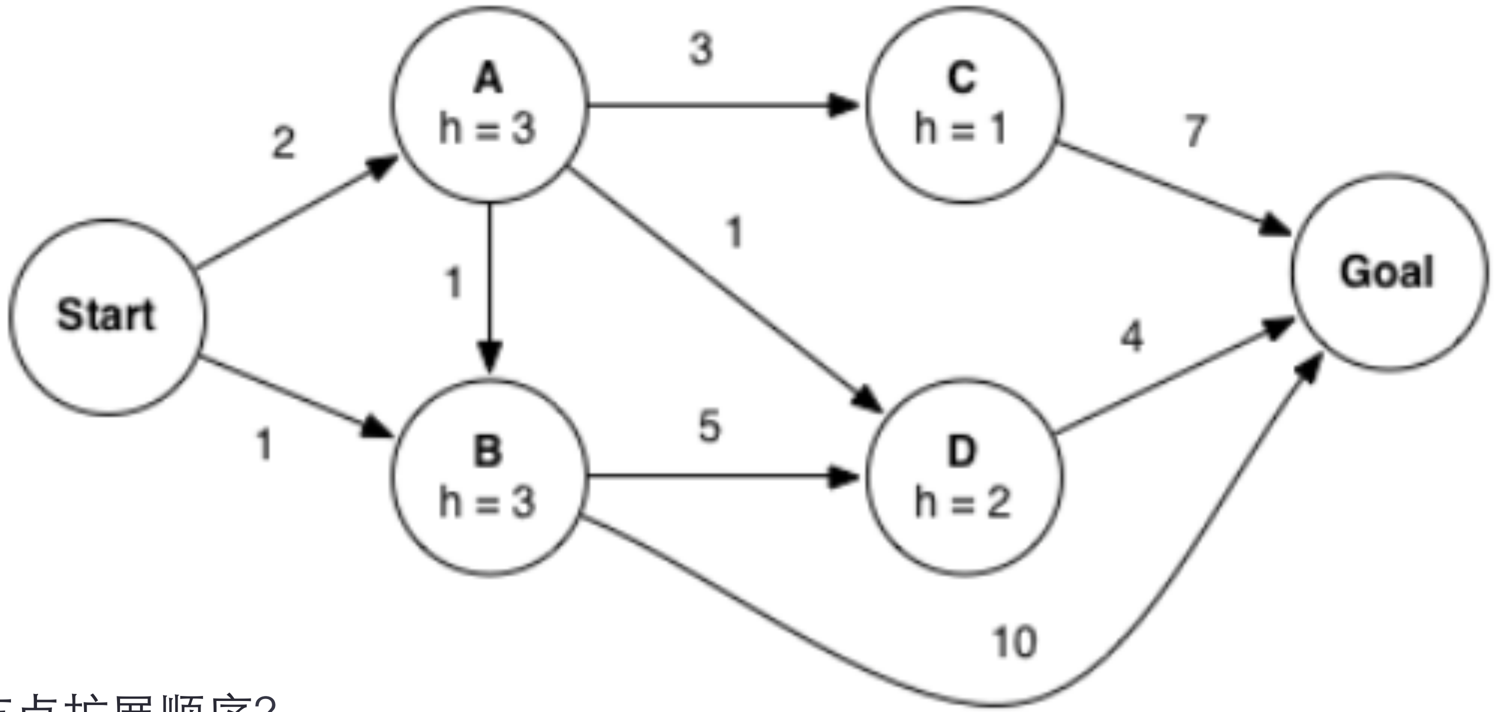


状态节点扩展顺序?

返回路径?

A* 图搜索

作业习题



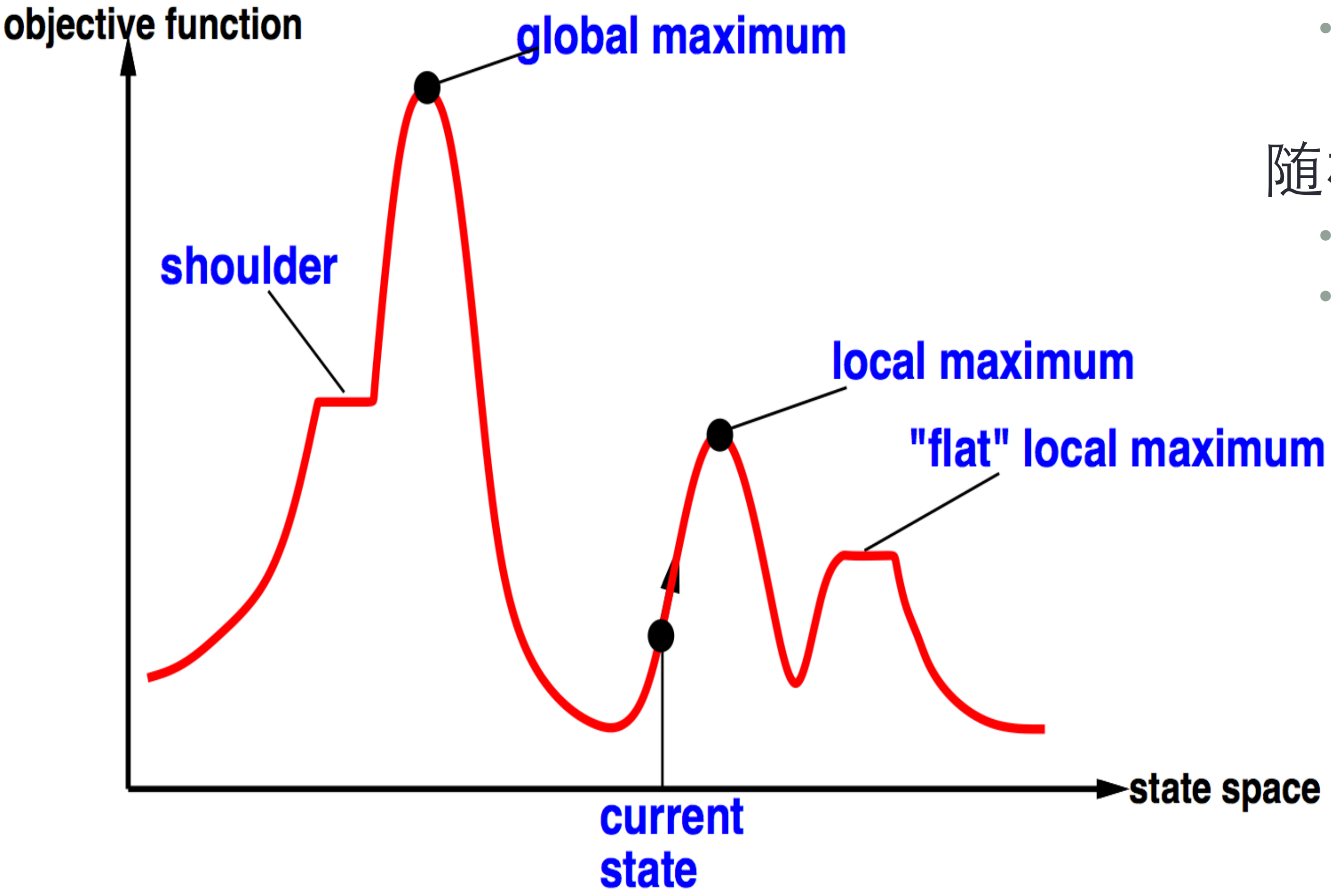
状态节点扩展顺序?

返回路径?

局部算法

- 爬山算法
  - 不断向值增加的方向移动（爬山），知道到达一个顶峰，周围邻居的值都比它的小。
- 优势
  - 不用维护一个搜索树，只需知道当前状态节点的值
- 不足
  - 返回的可能是局部最优解
- 改进方法
  - 随机重新开始
  - 随机水平移动

全局和局部最优解



随机开始点

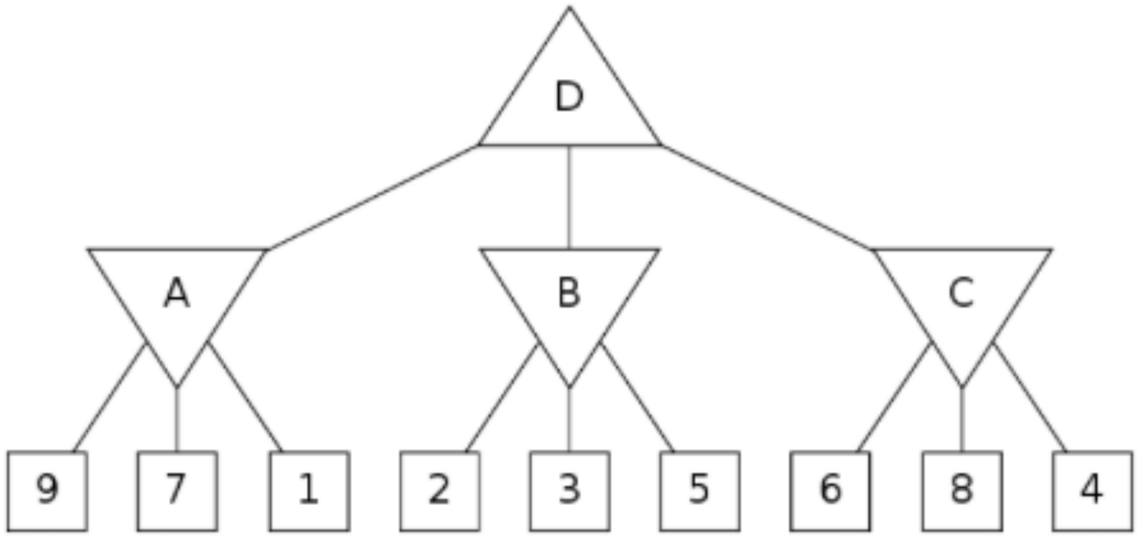
- 找到全局最优

随机水平移动

- 跳出“shoulder”
- 无限循环在“flat local maxima”

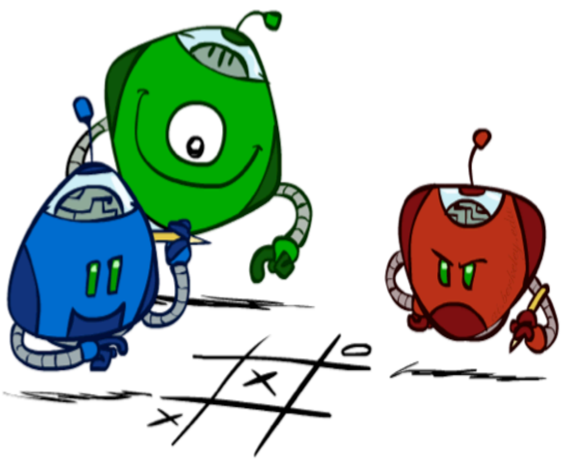
最小最大值算法

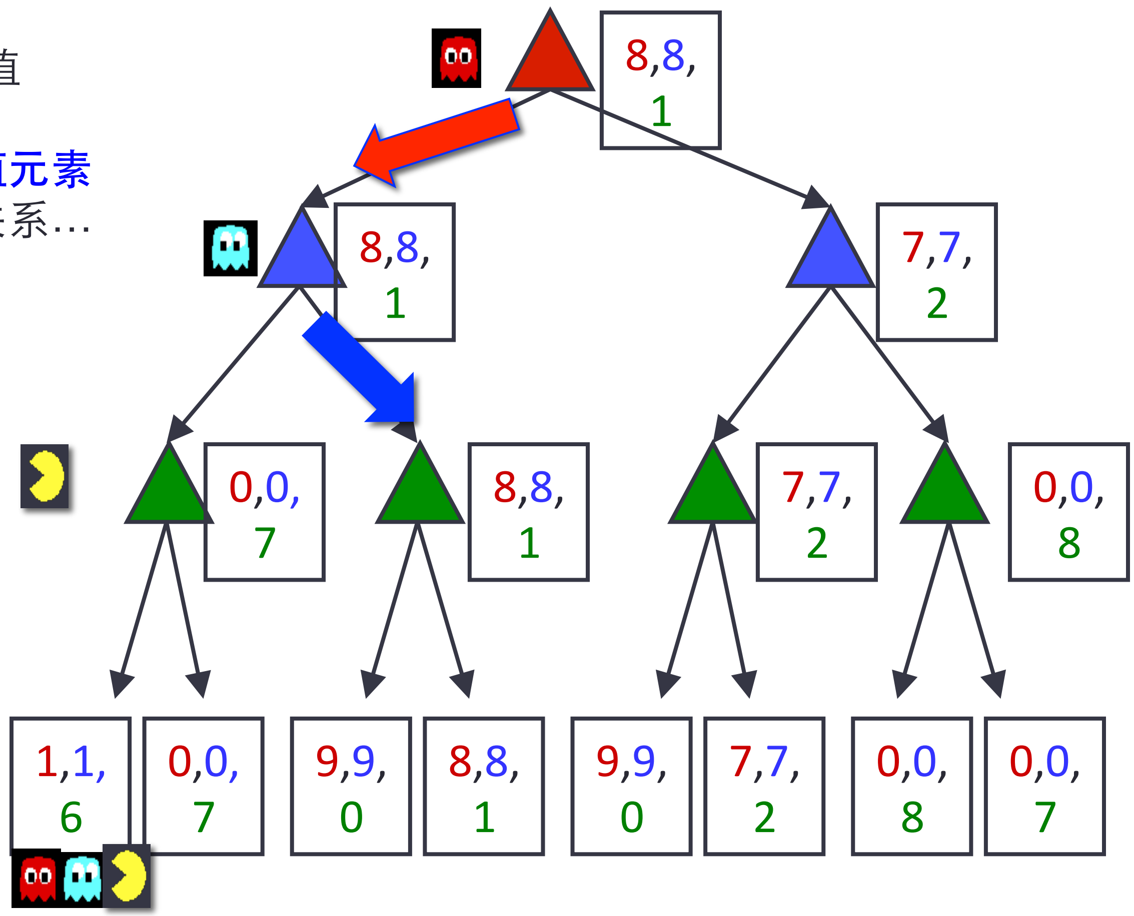
- 假设MAX和MIN是优化选择它们的行动
- 提供的是节点的下界值



普遍化的最小最大值法（minimax）

- 如果不是零和游戏，或有超过两个玩家?
- 普遍情况:
  - 终局（叶结点）的值是 一组值
  - 中间节点的值也是 一组值
  - **每个玩家最大化自己的利益值元素**
  - 能够动态产生协作和竞争等关系…



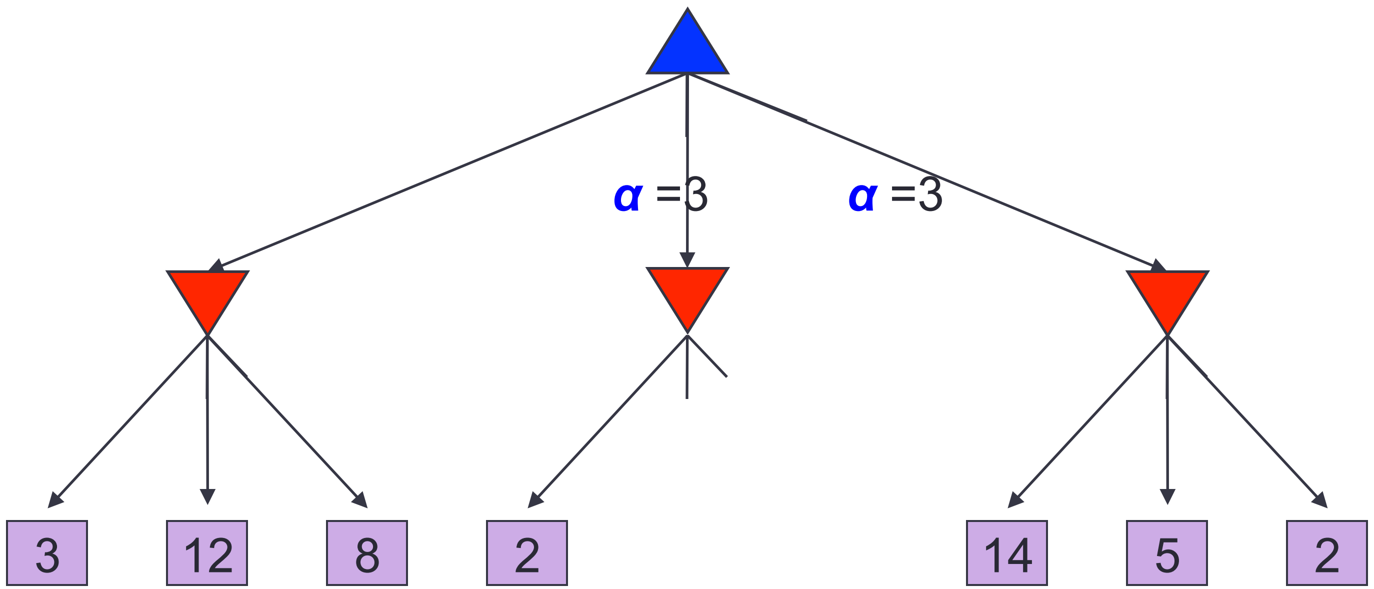


次优化策略

- 假设玩家MAX和玩家MIN在玩一个有限回合的零和博弈游戏。**MAX通过最小最大算法计算的树根节点的值是M**。假设每个玩家在每一轮都至少有2个可能的行动方向。而且，不同行动路线导致不同的终局节点值。
- 判断以下论断：
  - 假设玩家MIN在每一轮的行动是次优化的，并且玩家MAX知道它的对手行动是次优化的，那么存在一种策略使得玩家MAX能够在终局取得一个大于M的结果值？
  - 另一种假设，玩家MIN的行动选择在每一轮都是随机性的（且其概率是均匀分布的），并且玩家MAX知道这一点。对于玩家MAX来说，存在一种策略能够使得玩家MAX的期望终局结果值好于M？

Alpha-Beta 剪枝例子

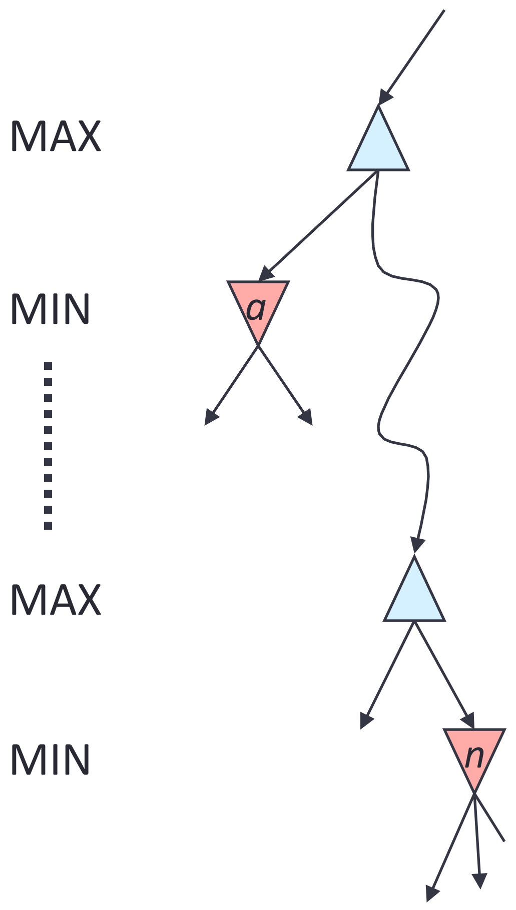
α = 在当前路径上所有MAX节点中最大的值



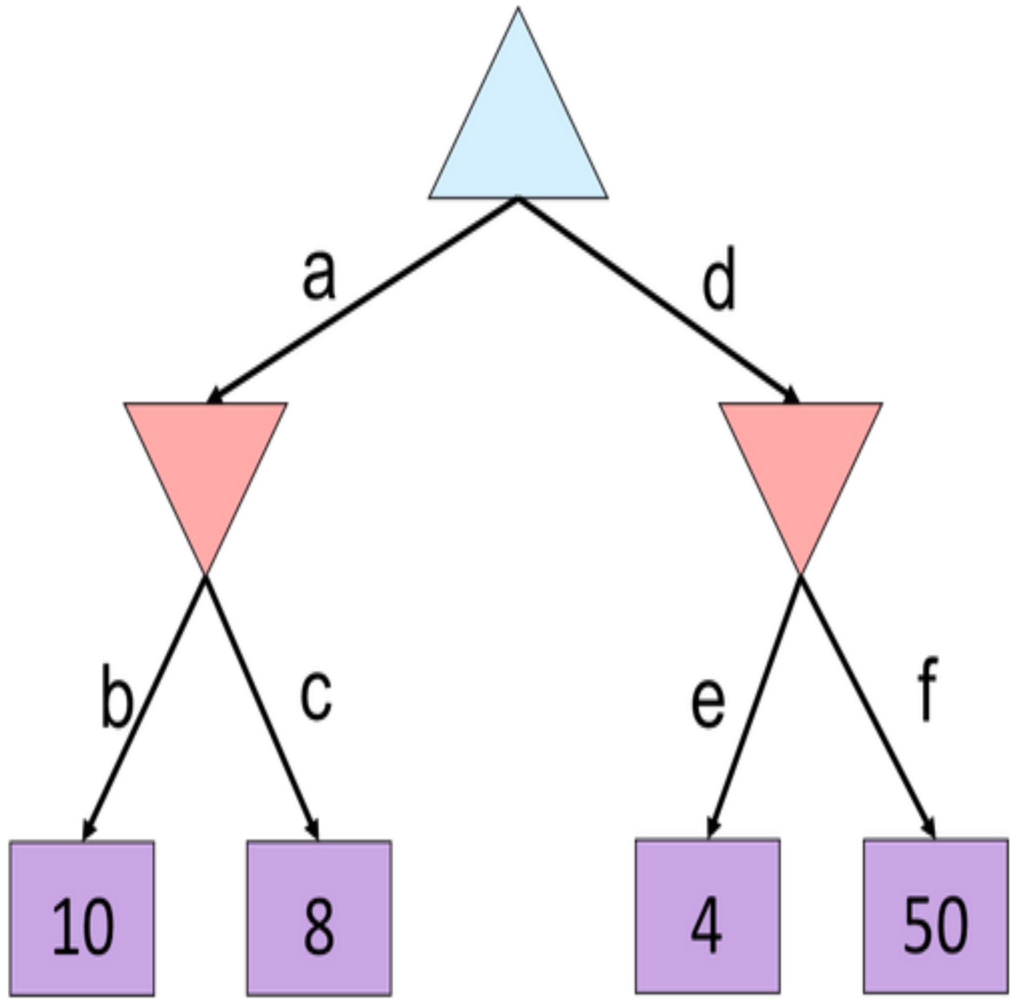
节点产生的顺序与结果有关: 可能导致不同的可被剪掉的节点数量

Alpha-Beta 剪枝

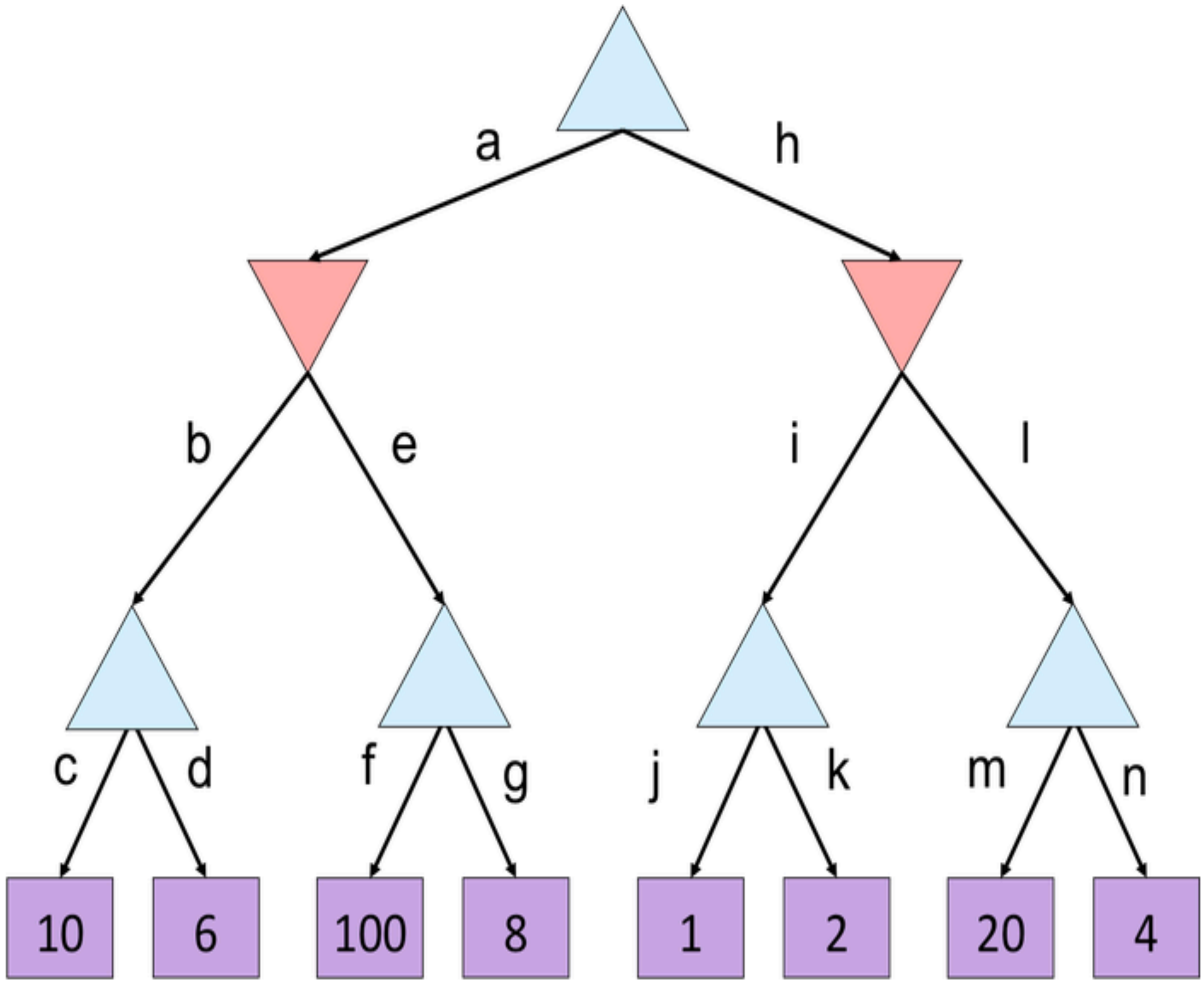
- 假定修剪MIN节点的子节点
  - 假设正在计算节点 *n* 的 最小-值
  - *n* 节点的值在检查其子节点的过程中逐渐减小
  - 令 **α** 是从根到当前节点路径上的MAX分支节点所能达到的最大值
  - 如果 *n* 的当前值比 **α** 的小，那么路径上的 MAX 分支节点将会避开这条路径，所以我们可以剪掉(不去检查) *n* 的其他子节点
- 对MAX 节点剪枝是对称的
  - 令 **β** 是从根到当前节点路径中的 MIN 分支节点所能达到的最小值



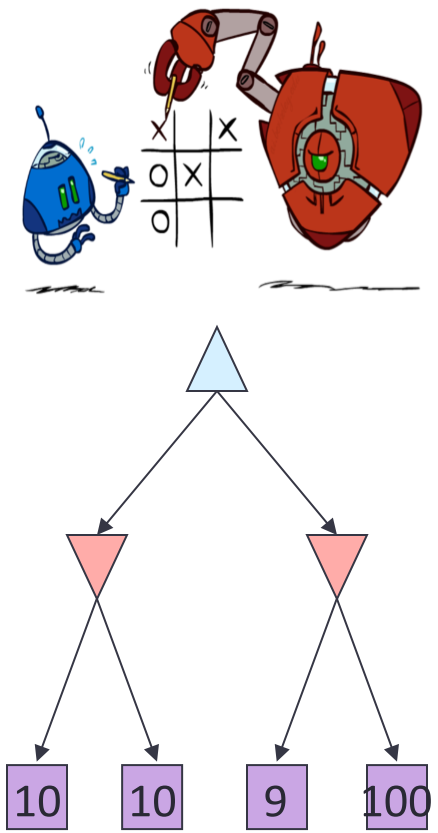
Alpha-Beta 剪枝测试



Alpha-Beta 剪枝测试2

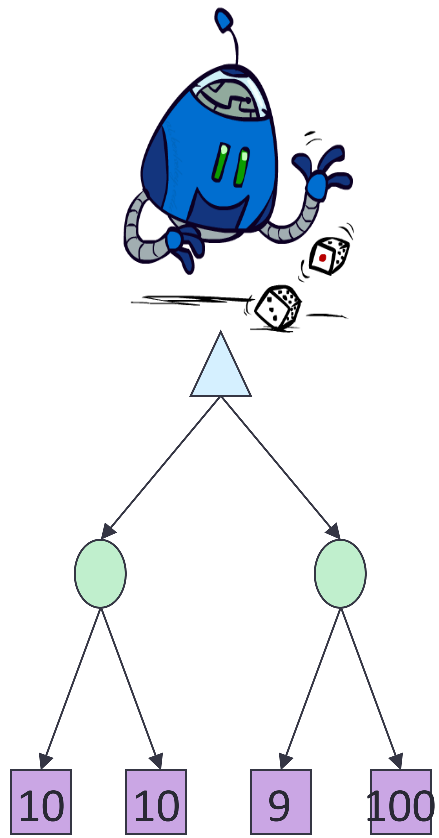


搜索树中可能有机遇节点



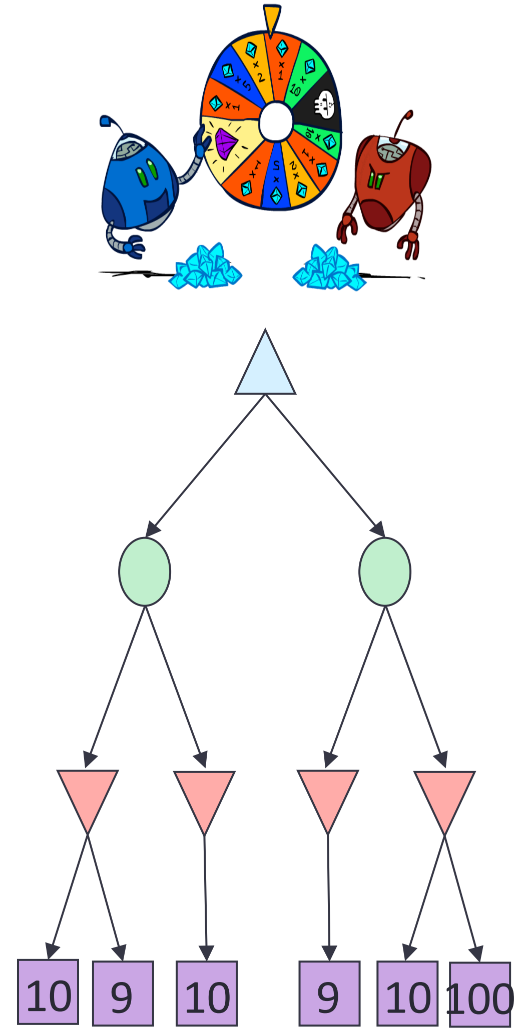
井字游戏

最小最大值(*Minimax*)



投资游戏

期望最大值(*Expectimax*)

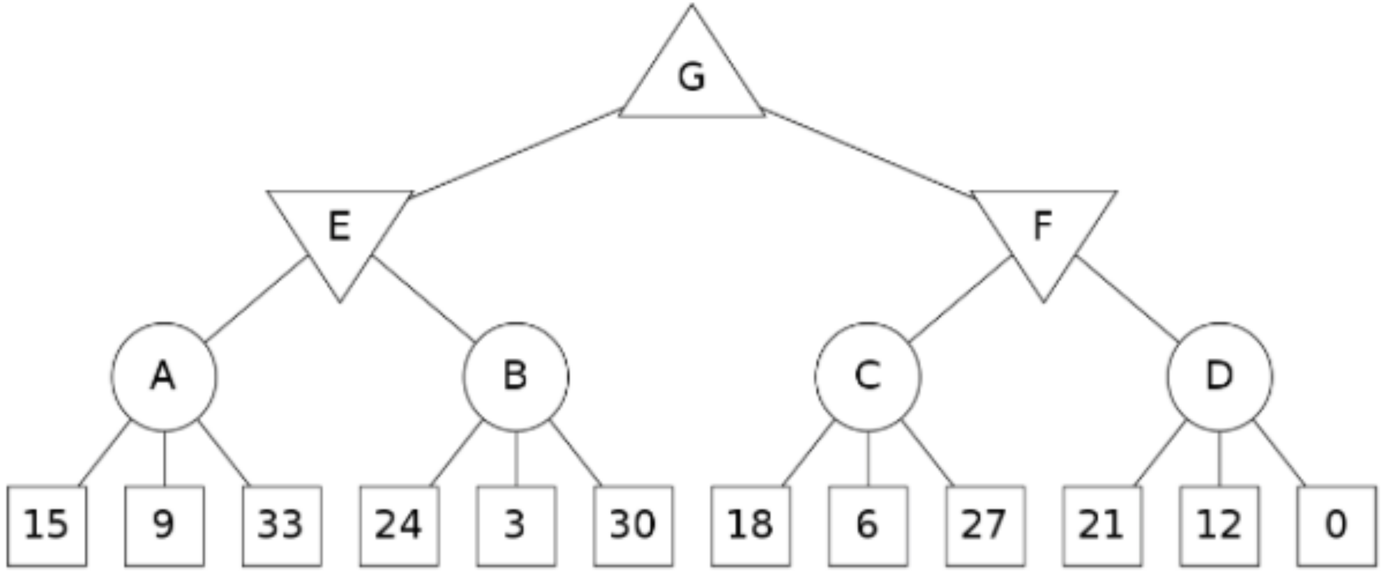


大富翁游戏

期望最小最大值(*Expectiminimax*)

期望最小最大算法(ExpectiMiniMax)

作业习题

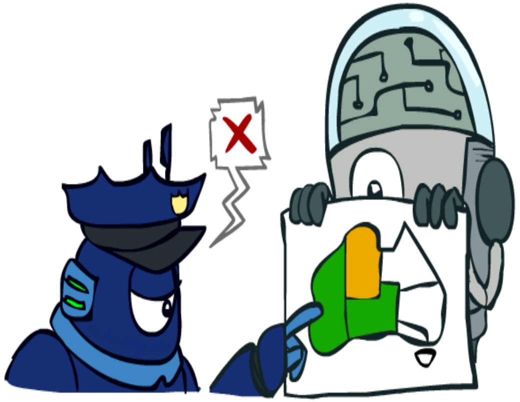


约束满足问题

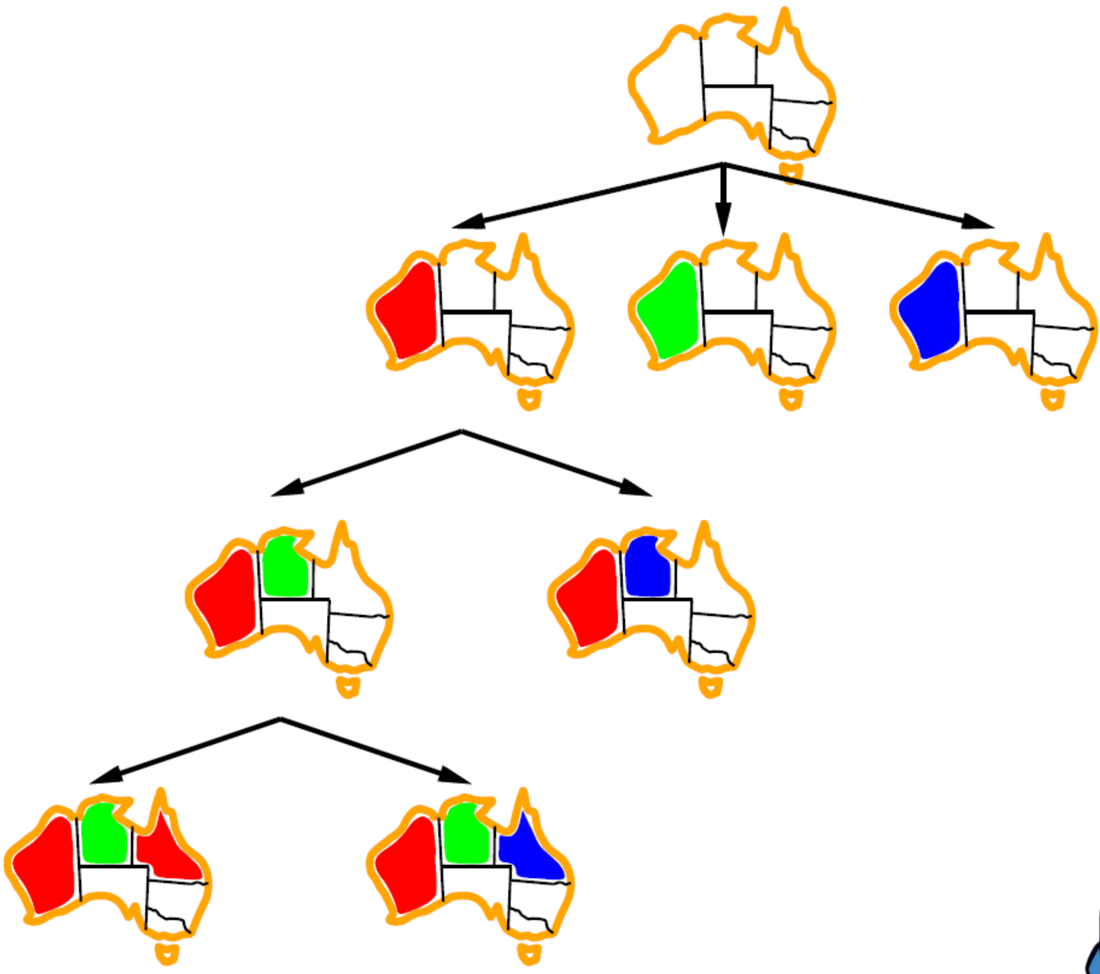
- 计算机课程被安排在每周一，三，五。总共有5门课程，3名教师，每名教师在一个时间只能教一门课。如何安排每门课程的教师？假设：
  - 课程包括：
    - 课程1 – 计算机编程简介， 时间 8：00 – 9：00AM
    - 课程2 – 人工智能导论， 时间 8：30 - 9：30AM
    - 课程3 – 软件工程， 时间 9：00 – 10：00 AM
    - 课程4 – 计算机视觉， 时间 9：00 – 10：00 AM
    - 课程5 – 机器学习， 时间 10：30 – 11：30AM
  - 教师包括：
    - 教师A，能够教课程1，2，5
    - 教师B，能够教课程3，4，5
    - 教师C，能够教课程1，3，4
- 描述为一个约束满足问题（假设一门课程是一个变量），指明每个变量的值域，及变量间的约束关系；约束关系可以是隐式表达形式。

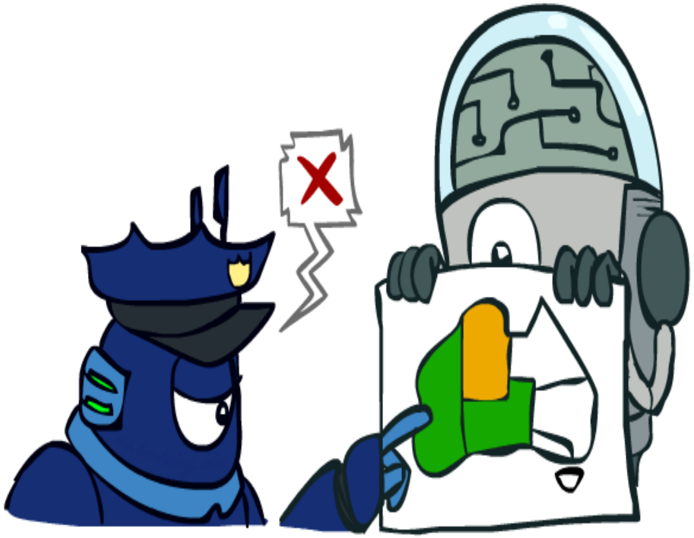
回溯搜索

- 回溯搜索是基本的无启发式信息的算法，用来求解CSP问题
- 想法 1: 一次探索一个变量
  - 在每一步只需考虑给一个变量配值
- 想法 2: 一边探索一边检查约束条件
  - 探索过程中检查当前的变量配值是否满足约束条件
  - 也许需要花费一些计算来检查约束条件是否满足
  - “逐步增加的目标测试”
- 深度优先搜索结合这两点改进，就叫作 回溯搜索
- 能够解决 n-皇后问题，直至 n = 25

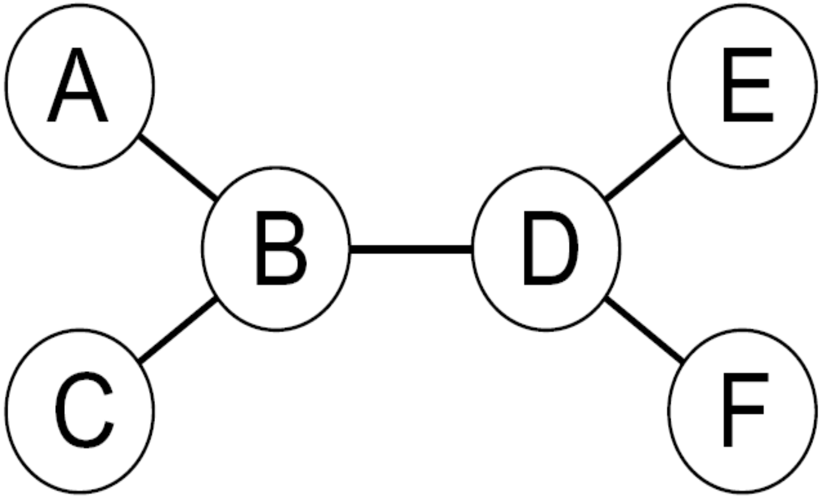


回溯搜索举例





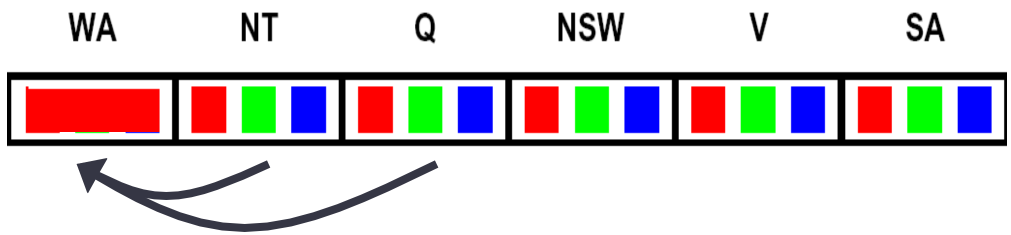
树结构的 CSPs



- 定理: 如果约束图无环，则其对应的约束满足问题的求解时间复杂度是 $O(n\ d^2)$
  - 比较一般的 CSPs, 最差时间复杂度是 $O(d^n)$

有用的概念: 弧的一致性（连贯性; Consistency）

- 一个弧 X → Y 是 一致的 当且仅当 对于X中的 每一个 x ，Y中存在 某个 y 值 不违背任何一个约束条件

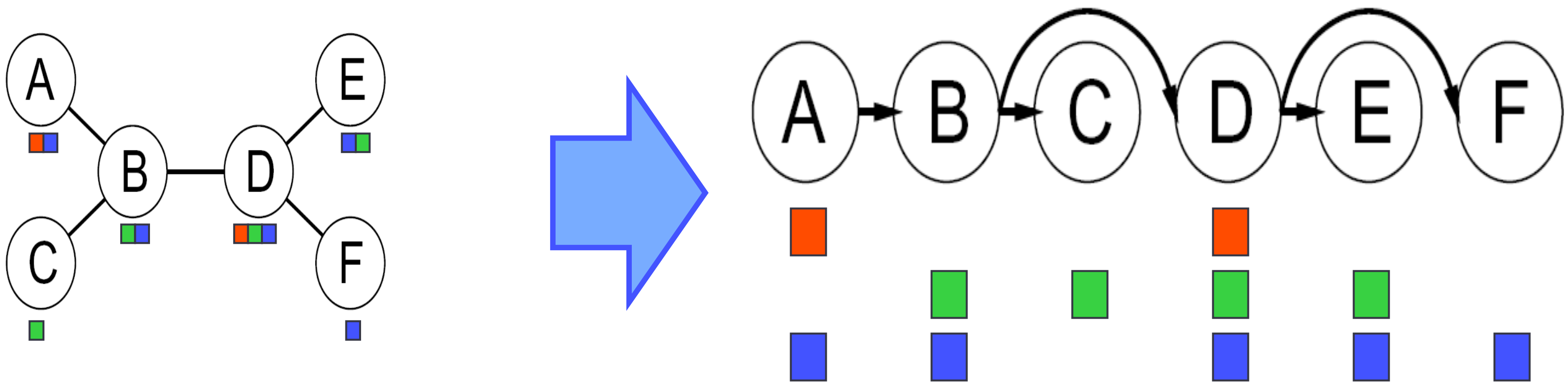


从弧的尾部删除

- 前向检查: 强制检查剩余未赋值变量指向新赋值变量的弧的一致性

树结构的 CSPs

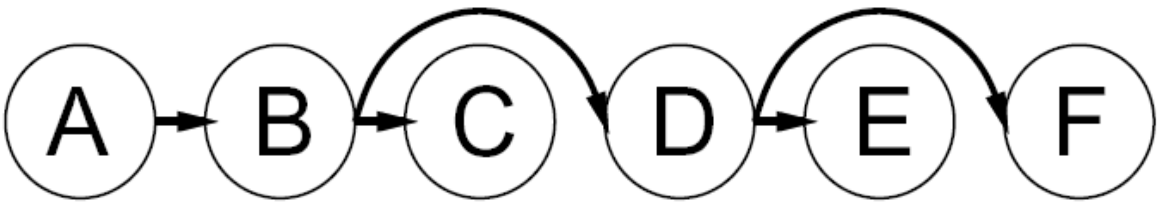
- 树结构的CSPs的求解算法:
  - 排序: 选一个根节点, 把变量线性排序， 使得父节点排在子节点之前



  - 从后向前删除: For i = n : 2, 应用 删除不一致的值(父节点(X_i),X_i)
  - 从前向后赋值: For i = 1 : n, 赋值 X_i 和父节点 Parent(X_i) 相一致的值

树结构的 CSPs

- 声明 1: 在从后向前的一致性检查后, 所有根到叶的弧都是一致性的
- 证明: 每个 X->Y 如果是一致性的，那么 Y的值域以后不会被减小 (因为 Y 的 子节点在Y之前先被处理过)

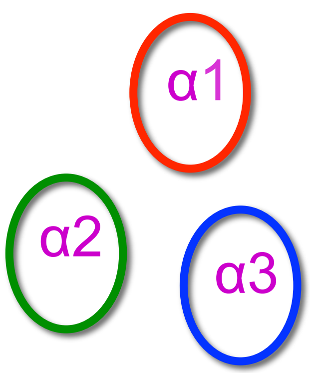


- 声明 2: **如果从根到叶的弧都是一致性的**, 那么从前向后的赋值过程将不会有回溯
- 证明: 归纳法

- 为什么这个算法不适用于约束图中有环的情况?

逻辑

- 语法: 定义句子
- 语义:
  - 可能的世界有哪些?
  - 这些句子在哪些世界里为 真 ? (句子真实性的定义)

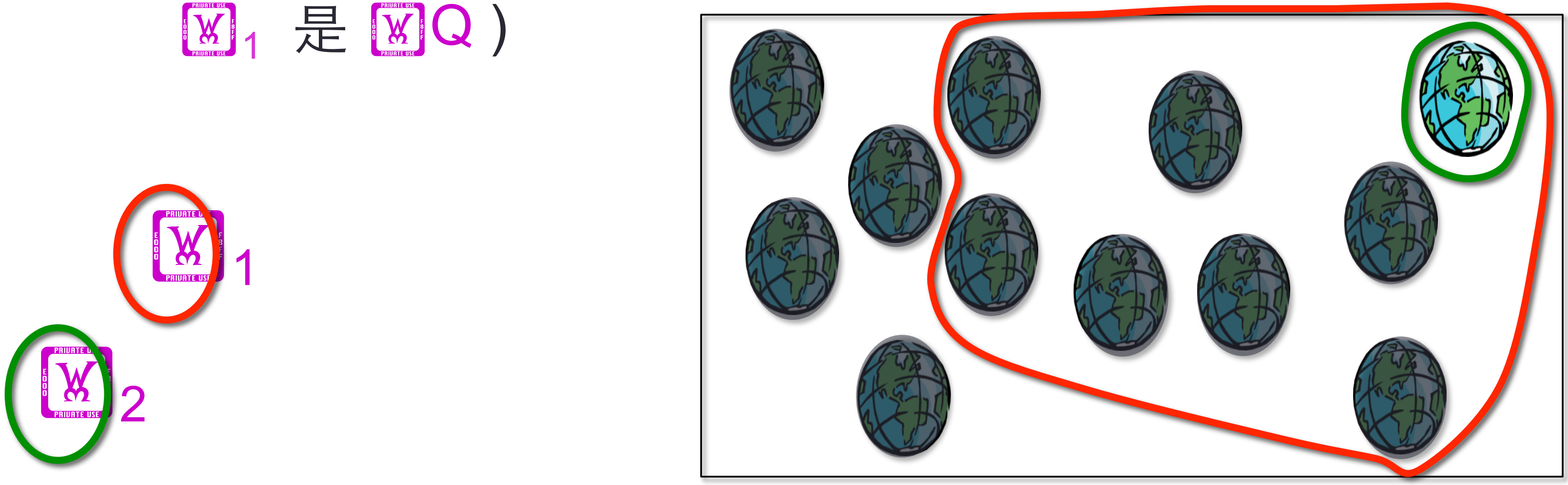


语法空间

语义空间

推理的内容: 蕴涵(entailment)

- **蕴涵**: □ ⊨ □ (“□ 牵涉(entails) □” or “□ 遵循于(follows from) □”)当且仅当在 □ 为真的每个世界里, □ 也是真
  - 换句话说， □-的世界（为真的那些世界） 是 □-的世界的一个子集 [***models***(□) □ ***models***(□)]
- 例如, □$_2$ ⊨ □$_1$
- (比如 □$_2$ 是 □Q □ R □ S □ W
  □$_1$ 是 □Q)



命题逻辑：语法

- 给定: 一组 命题字符 {X_1,X_2,…, X_n ,P, Q, R, North,…} （可为真或假）
  - (True 和 False 也是，真值固定)
- X_i 是一个句子（原子句）
- 复杂句
  - 如果 □ 是句子，那么 □□ 是一个句子 （否定）
  - 如果□ 和 □是句子，那么□ □ □是一个句子 （结合）
  - 如果□ 和 □是句子，那么□ □ □是一个句子 （分离）
  - 如果□ 和 □是句子，那么□ □ □是一个句子 （暗含,或条件的if … then）
  - 如果□ 和 □是句子，那么□ □ □是一个句子 （双向条件的 if and only if ）
  - 逻辑连接符，和（）的组合
- 文字(literal): 原子语句和否定的原子语句
- 没有其他样式的句子!

命题逻辑: 语义

- 给定一个模型（model），决定一个句子的真值
- 真值表

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

逻辑上的一致性

$$
\begin{array}{rcll}
(\alpha \land \beta) & \equiv & (\beta \land \alpha) & \text{commutativity of } \land \\
(\alpha \lor \beta) & \equiv & (\beta \lor \alpha) & \text{commutativity of } \lor \\
((\alpha \land \beta) \land \gamma) & \equiv & (\alpha \land (\beta \land \gamma)) & \text{associativity of } \land \\
((\alpha \lor \beta) \lor \gamma) & \equiv & (\alpha \lor (\beta \lor \gamma)) & \text{associativity of } \lor \\
\neg(\neg\alpha) & \equiv & \alpha & \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\beta \Rightarrow \neg\alpha) & \text{contraposition} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\alpha \lor \beta) & \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) & \equiv & ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)) & \text{biconditional elimination} \\
\neg(\alpha \land \beta) & \equiv & (\neg\alpha \lor \neg\beta) & \text{De Morgan} \\
\neg(\alpha \lor \beta) & \equiv & (\neg\alpha \land \neg\beta) & \text{De Morgan} \\
(\alpha \land (\beta \lor \gamma)) & \equiv & ((\alpha \land \beta) \lor (\alpha \land \gamma)) & \text{distributivity of } \land \text{ over } \lor \\
(\alpha \lor (\beta \land \gamma)) & \equiv & ((\alpha \lor \beta) \land (\alpha \lor \gamma)) & \text{distributivity of } \lor \text{ over } \land
\end{array}
$$

SAT问题

- 布尔满足问题
  - 例如，一个命题逻辑句子的可满足性
- $(A \land B) \lor (C \land D)$
  - 有多少可能的模型使该句为真?

可满足性 和 导出(蕴涵)

- 一个语句是 **可满足的** ， 如果它至少在一个世界里为真 (参见 CSPs!)
- 假设我们有一个超高效的 SAT solver; 我们如何能用它来测试蕴涵关系?
  - 假定 □ ⊨ □
  - 那么 □ □ □ 为真 在所有世界（演绎公理）
  - 因此 □(□ □ □) 为假 在所有世界
  - 因此 □ □ □□为假 在所有世界, i.e., 不可满足的(unsatisfiable)
- 所以, 把否定的结论添加到 所知道的语句里, 测试其不可满足性 (un)satisfiability; 也叫 归谬法(*reductio ad absurdum*)
- 高效的 SAT solvers 需要 合取范式(***conjunctive normal form)***

合取范式(CNF)

替换双向条件，用两个暗示条件

替换 $\alpha \Rightarrow \beta$ 用 $\neg\alpha$ v β

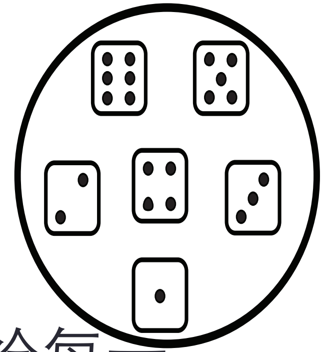
分配 v 到 $\wedge$

- 每个句子都能表达成一个子句
- 每个子句都是 文字(正的或否 符)的析取
- 到 CNF的标准变换:
  - At_1,1_0 ⇒ (Wall_0,1 ⇔ Blocked_W_0)
  - At_1,1_0 ⇒ ((Wall_0,1 ⇒ Blocked_W_0) ∧ (Blocked_W_0 ⇒ Wall_0,1))
  - ¬At_1,1_0 v ((¬Wall_0,1 v Blocked_W_0) ∧ (¬Blocked_W_0 v Wall_0,1))
  - (¬At_1,1_0 v ¬Wall_0,1 v Blocked_W_0) ∧ (¬At_1,1_0 v ¬Blocked_W_0 v Wall_0,1)

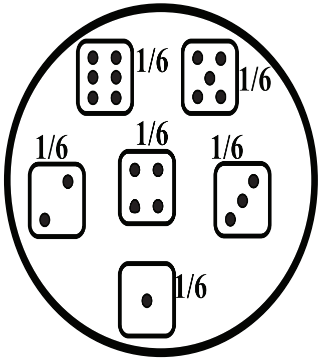
概率(Probability)

- 概率是对 复杂，不确定情况某种程度上的总结代表
  - 懒惰性(***laziness***): 太多的意外情况难以罗列, 等
  - 无知性(***ignorance***): 对某些情况缺乏了解和知识， 等
- 主观的(***Subjective***) or 贝叶斯(***Bayesian***) 概率:
  - 命题相关概率，根据自己的知识
  - 例如, $P(CatchPlane \mid A_{120}, \text{sunny}) = 0.92$
- 命题有关概率随新知识观察的变化:
  - 例如, $P(CatchPlane \mid A_{120}, \text{sunny}, NoDelaysOnBridge) = 0.96$

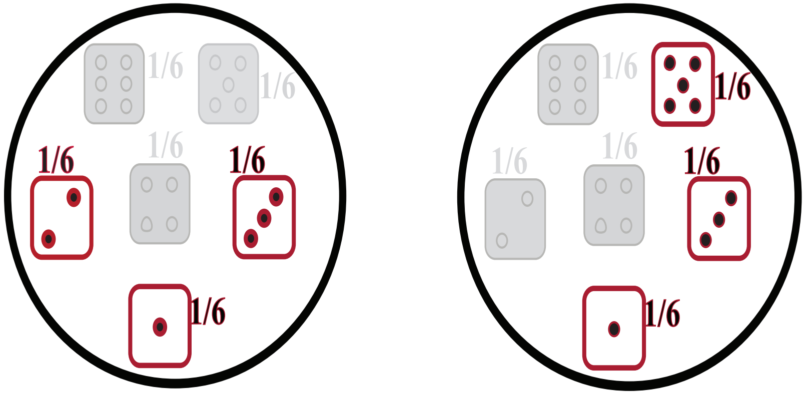
概率的基本法则



- 开始于一组可能世界的集合 Ω
  - 例如，一个骰子的6个可能结果, {1, 2, 3, 4, 5, 6}
- **概率模型(probability model)** 赋予一个数 $P(\omega)$ 给每一个世界 ω
  - $P(1) = P(2) = P(3) = P(5) = P(5) = P(6) = 1/6.$
- 这些数必须满足
  - $0 \le P(\omega) \le 1$
  - $\sum_{\omega \in \Omega} P(\omega) = 1$



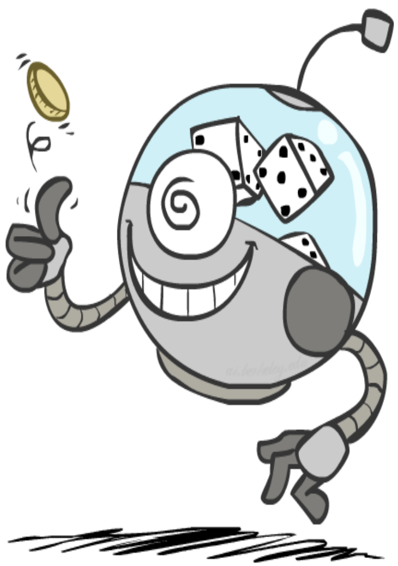
基本法则（继续）



- 一个**事件*(event)*** 是Ω 的一个子集
  - “投数 < 4” 是集合 {1,2,3}
  - “投数是奇数”， {1,3,5}
- 一个事件的概率是在对应世界概率数值之和
  - $P(A) = \sum_{\omega \in A} P(\omega)$
  - $P(\text{投数} < 4) = P(1) + P(2) + P(3) = 1/2$

随机变量(Random Variables)

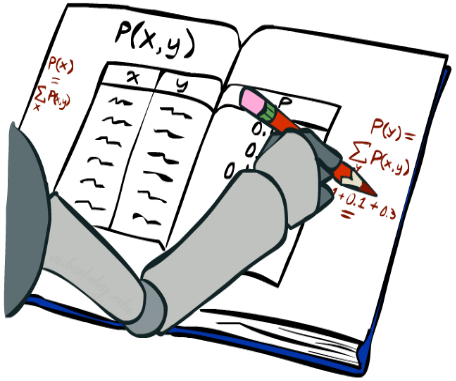
- 一个随机变量描述了世界中我们可能不确定的某个方面（正式的讲，是Ω 上的一个决定性的函数）
  - *R* = 天是否将会下雨?
  - *Odd* = 骰子的投数是否将会是一个奇数?
  - *T* = 天气是热还是冷?
  - *D* = 花费多长时间能够到达机场?



- 大写字母开头
- 随机变量也有值域
  - *Odd* in {true, false} e.g. *Odd*(1)=true, *Odd*(6) = false
    - 通常把事件 Odd=true 写成 odd, Odd=false 写成 ¬odd
  - *T* in {hot, cold}
  - *D* in [0, ∞)

边缘分布(Marginal Distributions)

- 边缘分布式消除掉某些变量后的子表
- 边缘化 (加和): 通过求和来合并行



$P(T,W)$

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

$P(T)$

T	P
hot	0.5
cold	0.5

$P(W)$

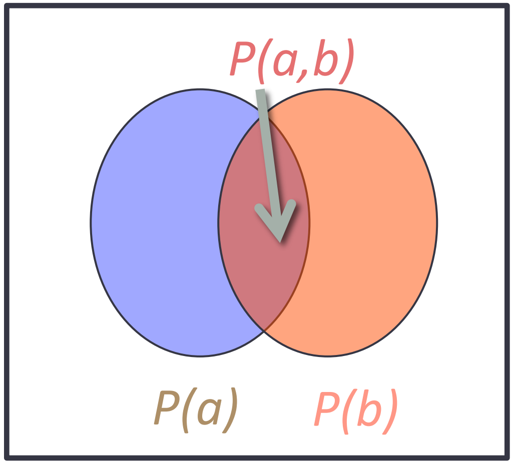
W	P
sun	0.6
rain	0.4

$$P(X_1 = x_1) = \sum_{x_2} P(X_1 = x_1, X_2 = x_2)$$

条件概率(Conditional Probabilities)

- 联合概率和条件概率间的简单关系
  - 一个条件概率的定义

$$P(a|b) = \frac{P(a,b)}{P(b)}$$



$P(T,W)$

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

$$P(W=s|T=c) = \frac{P(W=s,T=c)}{P(T=c)} = 0.4$$

$$= P(W=s,T=c) + P(W=r,T=c)$$
$$= 0.2 + 0.3 \quad = 0.5$$

条件分布（Conditional Distributions）

- 某些变量当其他变量的值固定的时候，的概率分布

条件分布

$$P(W|T)$$

$$P(W|T = hot)$$

W	P
sun	0.8
rain	0.2

$$P(W|T = cold)$$

W	P
sun	0.4
rain	0.6

联合分布

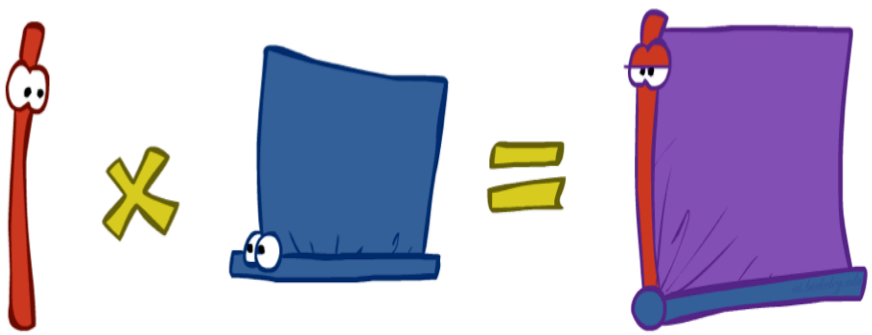
$$P(T, W)$$

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

乘法规则(The Product Rule)

- 已有条件分布，想要计算联合分布

$$P(y)P(x|y) = P(x,y) \quad \Longleftrightarrow \quad P(x|y) = \frac{P(x,y)}{P(y)}$$



乘法规则

$$P(y)P(x|y) = P(x,y)$$

- 举例:

$P(W)$

R	P
sun	0.8
rain	0.2

$P(D|W)$

D	W	P
wet	sun	0.1
dry	sun	0.9
wet	rain	0.7
dry	rain	0.3

$P(D,W)$

D	W	P
wet	sun	[illegible]
dry	sun	
wet	rain	
dry	rain	[illegible]

链式法则(The Chain Rule)

- 更普遍化的，任何联合分布可以写成条件分布的增量相乘

$$P(x_1, x_2, x_3) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2)$$

$$P(x_1, x_2, \ldots x_n) = \prod_i P(x_i|x_1 \ldots x_{i-1})$$

贝叶斯法则(Bayes' Rule)

- 两种方法因式分解一由两个变量组成的联合分布:

$$P(x,y) = P(x|y)P(y) = P(y|x)P(x)$$

- 相除后，我们得到:

$$P(x|y) = \frac{P(y|x)}{P(y)}P(x)$$

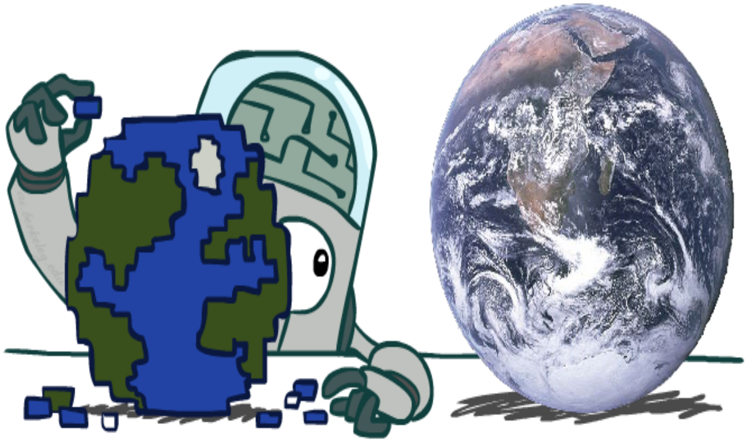
- 为什么这个有用?
  - 让我们计算一个条件概率，从它的相反的形式
  - 通常一个条件概率很难计算，但是相对应的另一个却很简单
  - 许多人工智能系统的基础

- 最重要的人工智能公式之一!

概率模型(Probabilistic Models)

- 模型描述的是世界（或某一部分）是如何工作的

- 模型总是一种简化
  - 可能忽略了某些变量和之间的交互关系
  - “所有模型都是错的; 但某些是有用的.”
    – George E. P. Box



- 概率模型能用来做什么?
  - 我们(或我们的人工智能体) 需要对未知变量进行推理，当给定一些证据后
  - 例如: 解释 (诊断推理)
  - 例如: 预测 (因果推理)
  - 例如: 基于期望利益值的决策

- 如何建立模型，并避免 d^n 的复杂性?

条件独立性（条件无关）Conditional Independence

- 无条件的 (绝对的) 独立性非常稀少 (为什么?)
- 条件独立性是我们对于不确定环境的最基本和强健的知识蕴藏形式
- X 是 条件独立于(conditionally independent) Y, 给定 Z

  当且仅当:

  $$\forall x,y,z \qquad P(x \mid y,z) = P(x|z)$$

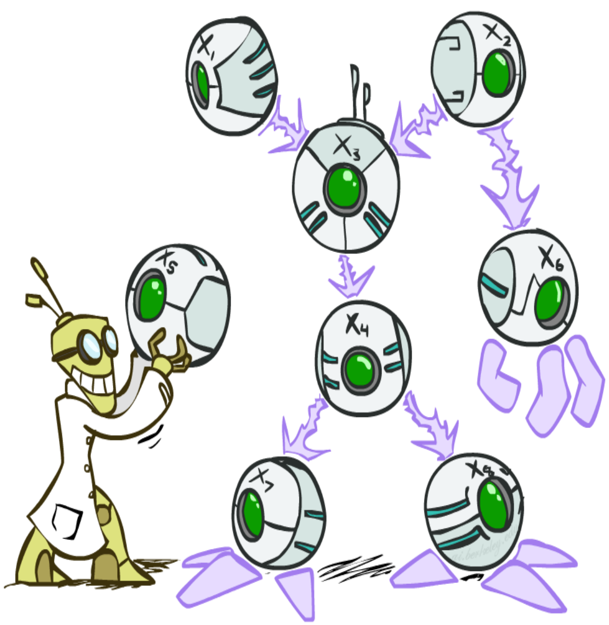
  或，等价地, 当且仅当

  $$\forall x,y,z \qquad P(x,y \mid z) = P(x|z)P(y|z)$$

贝叶斯网络: 宏观介绍

- 完全的联合分布表可以回答每一个问题，但是:
  - 表的大小是变量数的指数级
  - 需要大量的例子来学习相应的概率
  - 用列举法 (加和消掉隐藏变量)进行推理太慢

- 贝叶斯网络(Bayesian networks):
  - 表达了一个由变量组成的领域里所有的条件独立性关系
  - 联合分布因式分解为小规模条件概率分布的乘积
  - 分布表达的量级从指数减少为线性
  - 从较少的例子快速学习出模型
  - 快速推理 (在某些重要实例里可以达到线性时间复杂度)
  - “*Microsoft’s competitive advantage lies in its expertise in Bayesian networks*”
    -- Bill Gates, quoted in LA Times, 1996

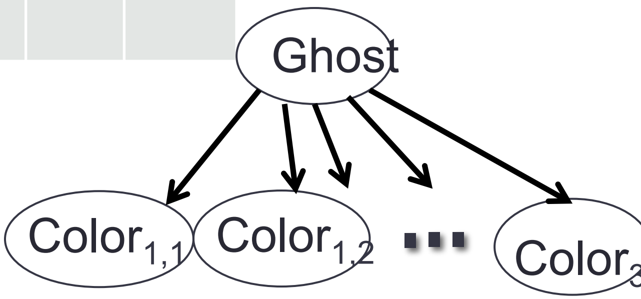


贝叶斯网络语法

- 一个节点对应一个变量 X_i
- 一个有向, 无环图
- 一个条件概率分布，对每个节点 给定图中它的 **父节点**
  - ***CPT***: 条件概率分布表:
    - 每一行是子节点的一个分布，在给定父节点的一个配置以后
  - 一个近似的“因果”过程的描述

P(Ghost)			
(1,1)	(1,2)	(1,3)	…
0.11	0.11	0.11	…

Ghost	P($Color_{1,1}$ | Ghost)			
	g	y	o	r
(1,1)	0.01	0.1	0.3	0.59
(1,2)	0.1	0.3	0.5	0.1
(1,3)	0.3	0.5	0.19	0.01
…				



贝叶斯网络 = 拓扑结构(图形) + 局部条件概率

贝叶斯网络的全局语法

- 贝叶斯网络整体表达了（编码） 联合分布，作为每一个变量上条件分布的乘积：

$$P(X_1,..,X_n) = \prod_i P(X_i \mid Parents(X_i))$$

贝叶斯网络里的概率

- 为什么我们可以保证以下公式反映的是正确的联合分布

$$P(X_1,..,X_n) = \prod_i P(X_i \mid Parents(X_i))$$

- 连锁法 (对所有分布有效): $P(X_1,..,X_n) = \prod_i P(X_i \mid X_1,\ldots,X_{i-1})$

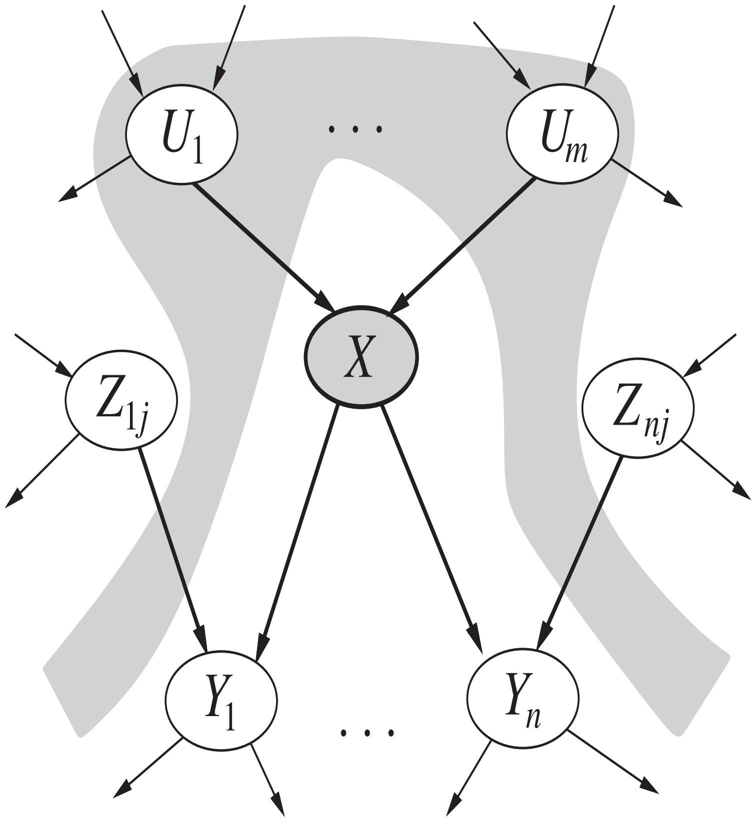
- <u>假定</u> 条件独立性: $P(X_i \mid X_1,\ldots,X_{i-1}) = P(X_i \mid Parents(X_i))$
  - 当加入节点 X_i, 保证了其父节点 “屏蔽” 它与其他祖先节点的联系

→ 结果: $P(X_1,..,X_n) = \prod_i P(X_i \mid Parents(X_i))$

- 所以，网络的拓扑结构暗示着肯定的条件独立性的成立

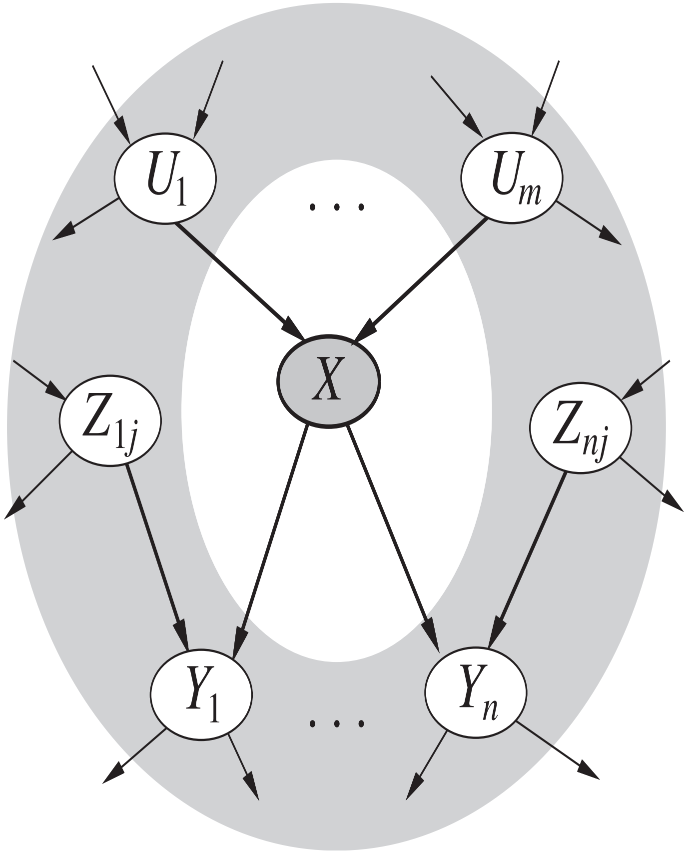
条件独立性的语义

- 每个变量在给定它的父变量节点情况下，则是条件独立于它的非 后代变量



马尔科夫毯(Markov blanket)

- 一个变量的马尔可夫毯包括父节点，子节点, 子节点的其他父节点
- 每个变量给定它的马尔科夫毯，则是条件独立于所有其他变量



贝叶斯网络推理

- 精确推理
  - 列举法
  - 变量消除法
- 近似推理（采样法）
  - 先验采样(Prior Sampling)
  - 拒绝抽样(Rejection Sampling)
  - 似然性/可能性加权(Likelihood Weighting)
  - 吉布斯采样(Gibbs Sampling)

有关期末考试

- 时间：第18周的周一（12/29）
- 题型
  - 选择
  - 填空
  - 判断
  - 分析计算
  - 简答
- 主要内容
  - 智能体相关概念
  - 搜索算法
  - 博弈树
  - 约束满足问题
  - 命题逻辑
  - 概率