CS 520: Introduction to Artificial Intelligence

Prof. Louis Steinberg

Lecture 18:

Constructing Bayes Nets

Utility and Decisions

Acting under uncertainty

- **Probability: methods for representing and reasoning about uncertainty of knowledge**
- **Utility theory: a way to represent and reason about the** *desirability* **of different outcomes**
- **Decision theory: A way to combine probability and utility to decide what to do.**

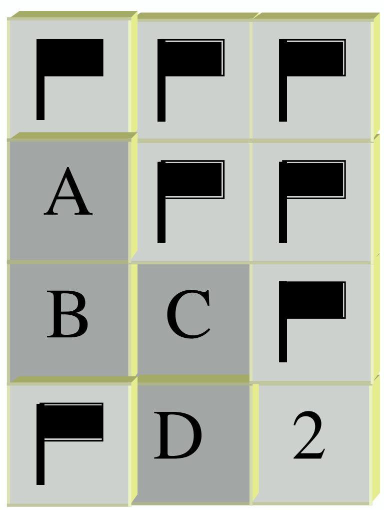
Review:Joint Probability

- **The state of a single situation / domain may be represented by a *set* of random variables.**
- **Joint probability distribution (or just "the joint"): probability distribution over all possible sets of values for the state variables.**

Example of Joint Probability Distribution

Mine Sweeper

MinesLeft = 1



A	B	C	D	Probability
F	F	F	F	0
F	F	F	T	1/2
F	F	T	F	1/2
F	F	T	T	0
F	T	F	F	0
F	T	F	T	0
F	T	T	F	0
F	T	T	T	0
T	F	F	F	0
T	F	F	T	0
T	F	T	F	0
T	F	T	T	0
T	T	F	F	0
T	T	F	T	0
T	T	T	F	0
T	T	T	T	0

Diagnosis Problem

- **Inferring Causes from Effects**
  - Given P(E|C), P(E|~C), P(C)
  - Find P(C|E)
- **How?**

  P(C|E) = □ P(E|C)*P(C)

  P(~C|E) = □ P(E|~C)*P(~C)

  □ makes P(C|E)+P(~C|E)=1 (normalization)
- **Two pieces of evidence,**
  - assuming E_1 and E_2 conditionally independent, given C

  P(C|E_1 ^ E_2) = □ P(C)P(E_1|C)P(E_2|C)

Example: screening for AIDS

- **C=AIDS, E=positive blood test**
- **Suppose $P(C) = 10^{-5}$, $P(E|C) = 1$, $P(E|{\sim}C) = 10^{-3}$**
- **$P(C|E) = \square\, P(E|C)*P(C) = \square *1* 10^{-5}$**
- **$P({\sim}C|E) = \square\, P(E|{\sim}C)*P({\sim}C) = \square*10^{-3}*(1- 10^{-5})$**
- **$\square \approx 10^{3}$**
- **$P(C|E) = .01$**
- **$P({\sim}C|E) = .99$**

Why?

- $P(E|C), P(E|\sim C)$ causal, more natural, can experiment
- $P(C)$ non-conditional
- Cause = fire, Effect = alarm sounds
  - $P(C)$: Fire department records
  - $P(E|C)$: Light fires in test rooms
  - $P(E|\sim C)$: Don't light fires in test rooms
    - C is rare but can be forced to happen or can be searched for; E can't

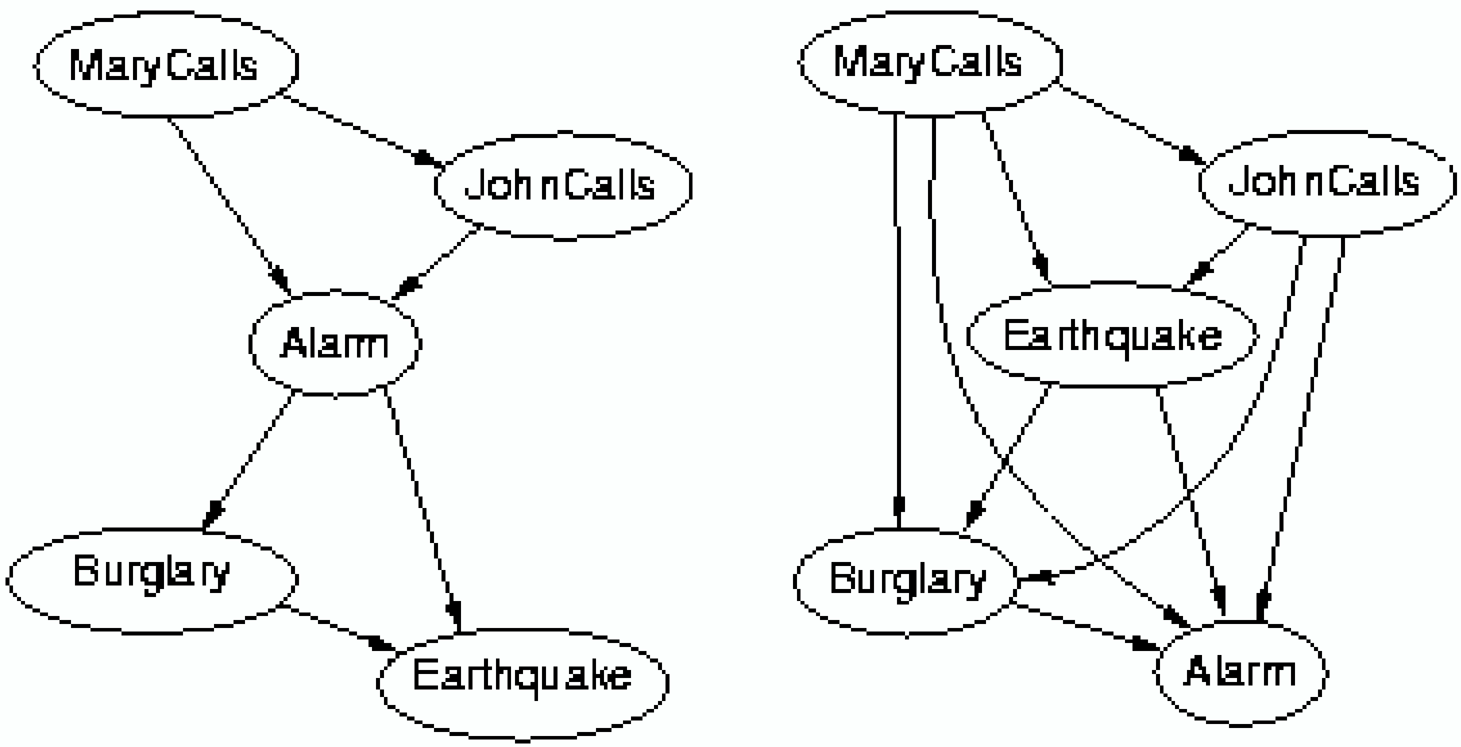
Belief Networks

- **A directed acyclic graph (DAG).**
- **Each node represents a random variable.**
- **Each arc represents a *direct* dependence of two variables on each other.**
- **Structure encodes conditional independence.**

  Node X, direct parents $P_1,\ldots,P_n$, other ancestors $A_1,..,A_m$

  X conditionally independent of ancestors given parents
- **Arcs represent dependence, not cause**
  - **Any given arc could go in opposite direction, if other changes made**
  - **But graph is more compact if arcs go in causal direction**

Network Structure Depends on Order of Introduction



Constructing a Belief Network

1. Choose a set of variables.
2. Choose an ordering of the variables.

3 While (Some variables are left)

   a. Choose a variable, ``Xi''.

   b. Add a node for Xi to the network.

   c. Set Parents(i) to be the minimal set of nodes already in the network satisfying the conditional independence property.

   d. Define the conditional probability table for node Xi.

Acting under uncertainty

- **Probability: methods for representing and reasoning about uncertainty of knowledge**
- **Utility theory: a way to represent and reason about the** *desirability* **of different outcomes**
- **Decision theory: A way to combine probability and utility to decide what to do.**

Representing an Agent's Preferences

- **Let A and B be states:**

$A \succ B$ **Agent prefers A to B.**

$A \prod B$ **Agent is indifferent between A and B.**

Representing Lotteries

- If A and B are states, then [p,A ; 1-p,B] is the lottery with two possible outcomes.
  - State A is the outcome with probability p.
  - State B is the outcome with probability 1-p.

Generalize to N possible outcomes.

Generalize to lotteries of lotteries.

Constraints on Rational Preferences

- Orderable:
  - (A < B) v (B < A) v (A = B)
- Transitive:
  - (A < B) ^ (B < C) => (A < C)
- Continuous:
  - A < B < C => (\exists p) [p,A;1-p,C] = B
- Substitutable:
  - A = B => [p,A;1-p,C] = [p,B;1-p,C]
- Monotonic:
  - A < B => (p > q <=> [p,A;1-p,B] < [q,A;1-q,B])
- Decomposable:
  - [p,A;1-p,[q,B;1-q,C]] = [p,A;(1-p)q,B;(1-p)(1-q),C]

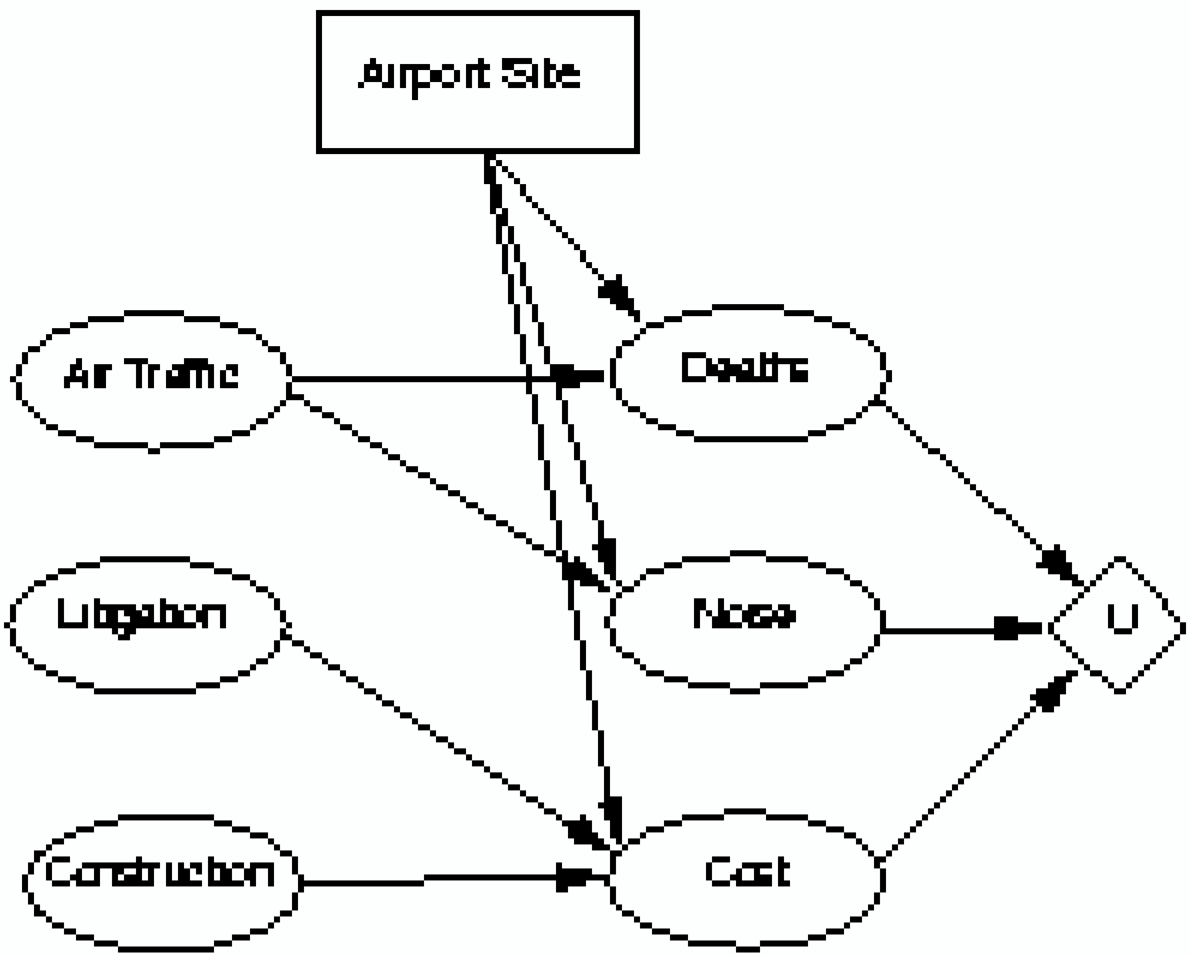
Representing Preferences by Utility Functions

- **If an agent's preferences satisfy the constraints on rationalpreferences, then his preferences can be represented by a**
- **utility function U:States => Reals.**
  - **U(A)>U(B) <=> A < B**
  - **U(A)=U(B) <=> A = B**
- **The utility of a lottery is the expected utility of the outcome**

One-Shot Decision Networks

- **Decision Nodes: (Rectangle) Controllable variables.**
- **Chance Nodes: (Ellipse) Random variables.**
- **Utility Node: (Diamond) Special random variable.**

Decision Network for Airport Siting Problem



Evaluating Decision Networks

- Set evidence variables from observations.
- For each possible action do:
  - Set decision nodes for the action.
  - Calculate posteriori probability for each value of
  - each parent of the utility node.
  - Calculate expected utility of the action.
- Return an action with highest expected utility.

Sequential Decision Problems

- Agent's utility depends on a sequence of decisions.
- Early decisions may effect choices available for later decisions.
- Early decisions may effect outcomes of later decisions.

Types of Environments

- **Deterministic: If agent knows the current state, he can predict the effect of each action.**
- **Stochastic: Even when agent knows the current state, he cannot predict the of each action.**
- **Accessible: The agent always knows or can determine the current state.**
- **Inaccessible: The agent has only partial information about the current state.**

Markov Decision Problem

- **Finding an optimal policy for an accessible, stochastic environment.**
- **A ``policy'' is a mapping from states to actions.**
- **Assume environment has ``Markov Property'', i.e., the transition probabilities don't depend on history.**

Finite and Infinte Horizons

- Finite Horizon: The agent makes a finite sequence of N decisions
- taking him through a series (S_0,…S_n) states:
  - Utility = U(S_1,…S_n)
- Infinite Horizon: The agent makes an infinite sequence of decisions
- taking him through a potentially infinite series S_0,…S_n,\dots states.

Value Iteration

- Algorithm for Solving Infinite Horizon Markov Decision Problems.
- Initial utility of each state is the immediate reward.
- Repeatedly update utilities by making optimal local decisions.
- Quit when utilities converge.

Policy Iteration

- Algorithm for Solving Infinite Horizon Markov Decision Problems.
- Initial policy is optimal for immediate rewards.
- Repeatedly compute states' utilities given current policy.
- Simultaneously update policy as well.
- Quit when policy does not change.

•