Tutorial on Bayesian Networks

Jack Breese
Microsoft Research
breese@microsoft.com

Daphne Koller
Stanford University
koller@cs.stanford.edu

First given as a AAAI'97 tutorial.

Probabilities

- Probability distribution $P(X|\xi)$
  - X is a random variable
    - Discrete
    - Continuous
  - ξ is background state of information

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Discrete Random Variables

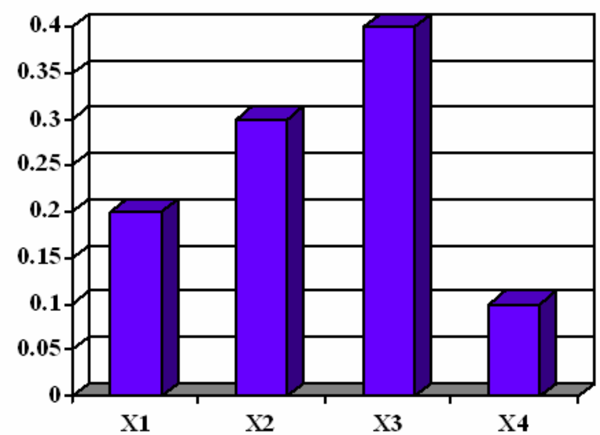
- Finite set of possible outcomes

$$X \in \{x_1, x_2, x_3, \ldots, x_n\}$$

$$P(x_i) \geq 0$$

$$\sum_{i=1}^{n} P(x_i) = 1$$

X binary: $P(x) + P(\overline{x}) = 1$



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

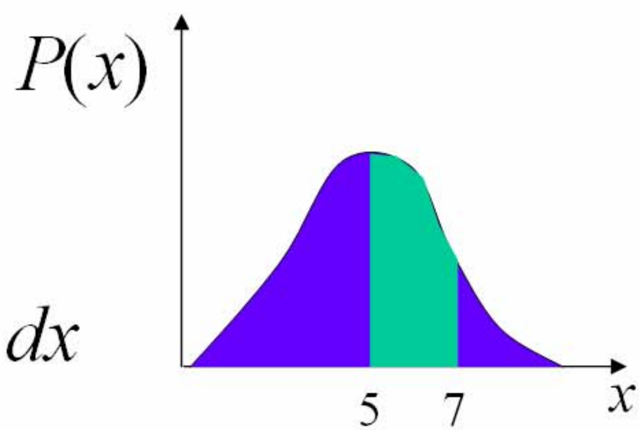
Continuous Random Variable

- Probability distribution (density function) over continuous values

$$X \in [0,10] \qquad P(x) \geq 0$$

$$\int_0^{10} P(x)\,dx = 1$$

$$P(5 \leq x \leq 7) = \int_5^7 P(x)\,dx$$



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Bayesian networks

- Basics
  - Structured representation
  - Conditional independence
  - Naïve Bayes model
  - Independence facts

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Bayesian Networks

$S \in \{no, light, heavy\}$

$C \in \{none, benign, malignant\}$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Bayesian Networks

$S \in \{no, light, heavy\}$

$C \in \{none, benign, malignant\}$

$P(S=no)$	0.80
$P(S=light)$	0.15
$P(S=heavy)$	0.05

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Bayesian Networks

$S \in \{no, light, heavy\}$

P(S=no)	0.80
P(S=light)	0.15
P(S=heavy)	0.05

$C \in \{none, benign, malignant\}$

Smoking=	no	light	heavy
P(C=none)	0.96	0.88	0.60
P(C=benign)	0.03	0.08	0.25
P(C=malig)	0.01	0.04	0.15

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Product Rule

- $P(C,S) = P(C|S)\ P(S)$

$S\Downarrow \quad C\Rightarrow$	*none*	*benign*	*malignant*
no	0.768	0.024	0.008
light	0.132	0.012	0.006
heavy	0.035	0.010	0.005

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Marginalization

$S\Downarrow$ $C\Rightarrow$	*none*	*benign*	*malig*	total
no	0.768	0.024	0.008	.80
light	0.132	0.012	0.006	.15
heavy	0.035	0.010	0.005	.05
total	0.935	0.046	0.019	

P(Smoke)

P(Cancer)

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Bayes Rule Revisited

$$P(S \mid C) = \frac{P(C \mid S)P(S)}{P(C)} = \frac{P(C,S)}{P(C)}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Bayes Rule Revisited

$$P(S \mid C) = \frac{P(C \mid S)P(S)}{P(C)} = \frac{P(C,S)}{P(C)}$$

$S\Downarrow \quad C\Rightarrow$	*none*	*benign*	*malig*
no	0.768/.935	0.024/.046	0.008/.019
light	0.132/.935	0.012/.046	0.006/.019
heavy	0.030/.935	0.015/.046	0.005/.019

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Bayes Rule Revisited

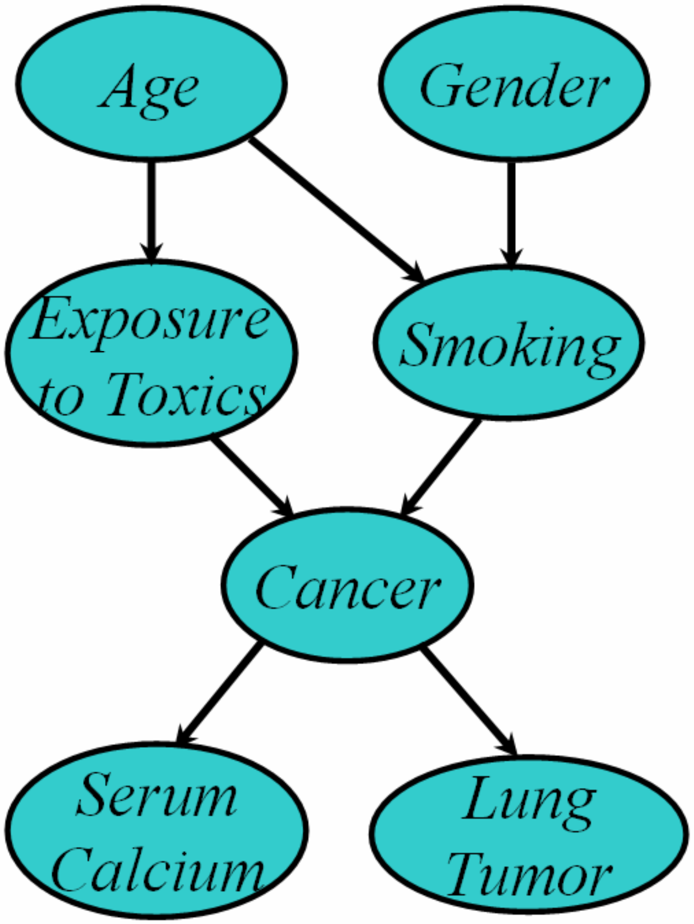
$$P(S \mid C) = \frac{P(C \mid S)P(S)}{P(C)} = \frac{P(C,S)}{P(C)}$$

$S \Downarrow \quad C \Rightarrow$	*none*	*benign*	*malig*
no	0.768/.935	0.024/.046	0.008/.019
light	0.132/.935	0.012/.046	0.006/.019
heavy	0.030/.935	0.015/.046	0.005/.019

Cancer=	*none*	*benign*	*malignant*
P(S=no)	0.821	0.522	0.421
P(S=light)	0.141	0.261	0.316
P(S=heavy)	0.037	0.217	0.263

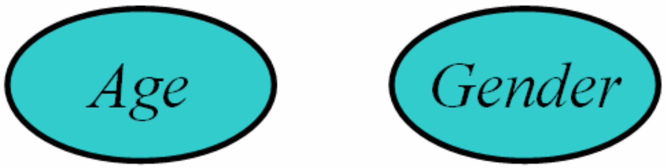
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

A Bayesian Network



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

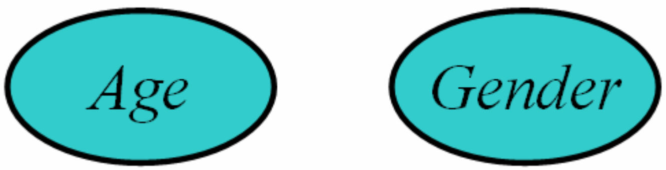
Independence



Age and *Gender* are independent.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Independence

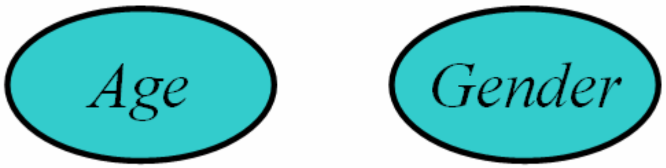


Age and *Gender* are independent.

$$P(A,G) = P(G)P(A)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Independence



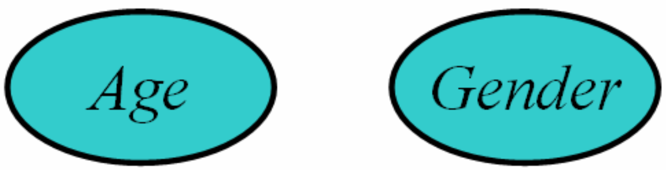
Age and *Gender* are independent.

$$P(A,G) = P(G)P(A)$$

$$P(A|G) = P(A) \quad A \perp G$$
$$P(G|A) = P(G) \quad G \perp A$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Independence



Age and *Gender* are independent.

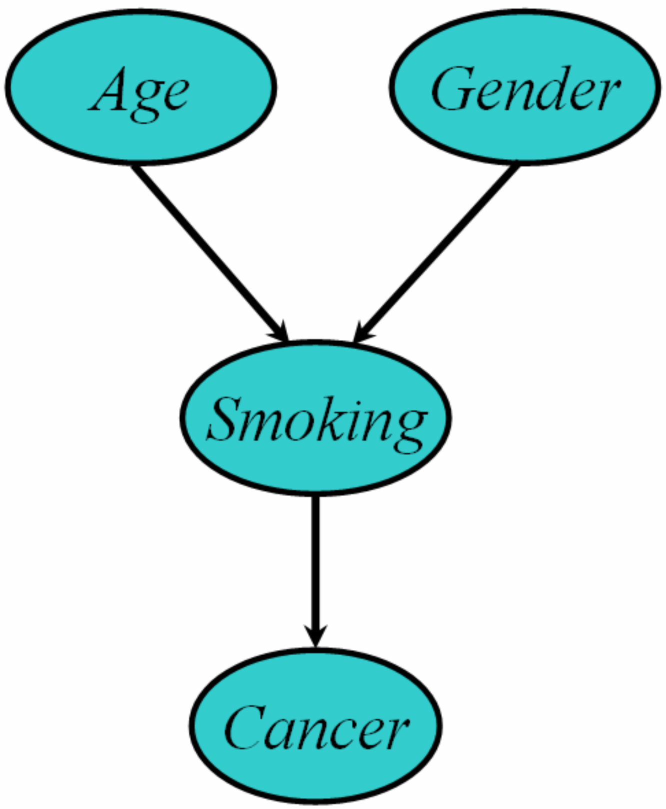
$$P(A,G) = P(G)P(A)$$

$$P(A|G) = P(A) \quad A \perp G$$
$$P(G|A) = P(G) \quad G \perp A$$

$$P(A,G) = P(G|A)\,P(A) = P(G)P(A)$$
$$P(A,G) = P(A|G)\,P(G) = P(A)P(G)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

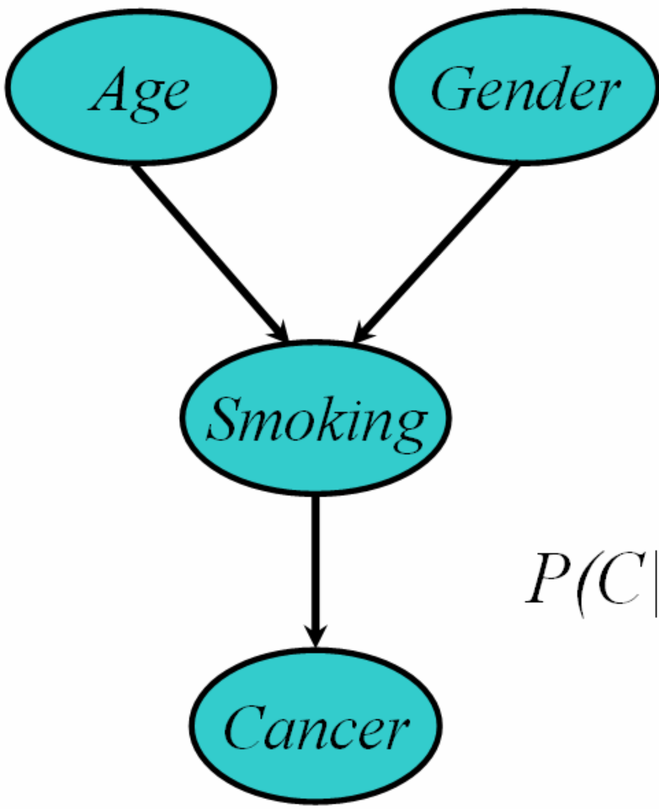
Conditional Independence



Cancer is independent of *Age* and *Gender* given *Smoking*.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Conditional Independence

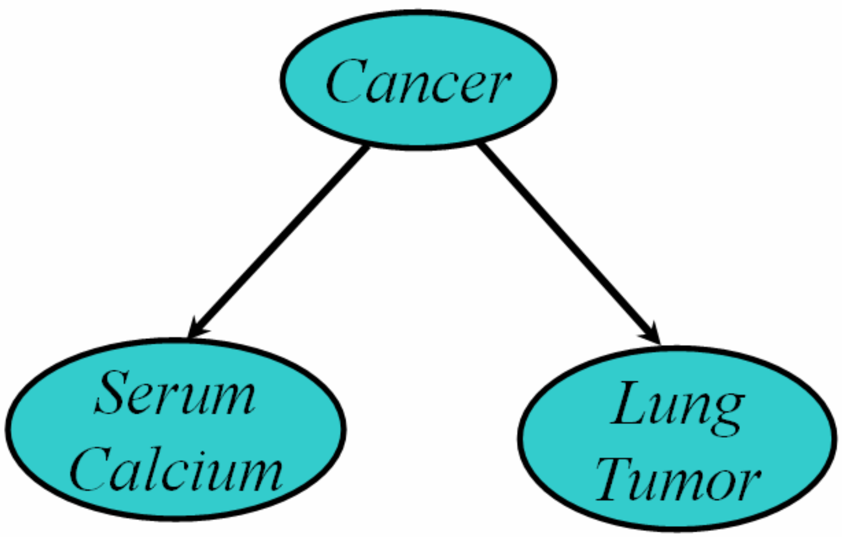


Cancer is independent of *Age* and *Gender* given *Smoking*.

$$P(C|A,G,S) = P(C|S) \quad C \perp A,G \mid S$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

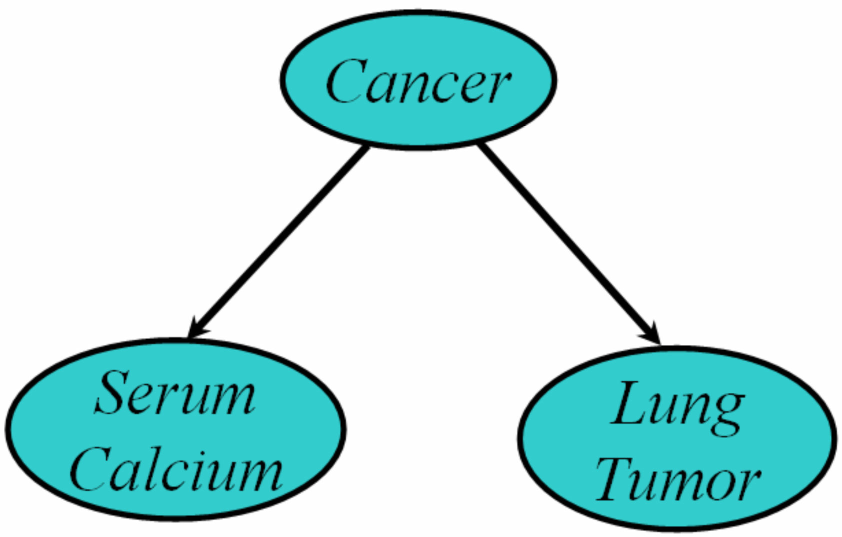
More Conditional Independence: Naïve Bayes



Serum Calcium and *Lung Tumor* are dependent

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

More Conditional Independence: Naïve Bayes



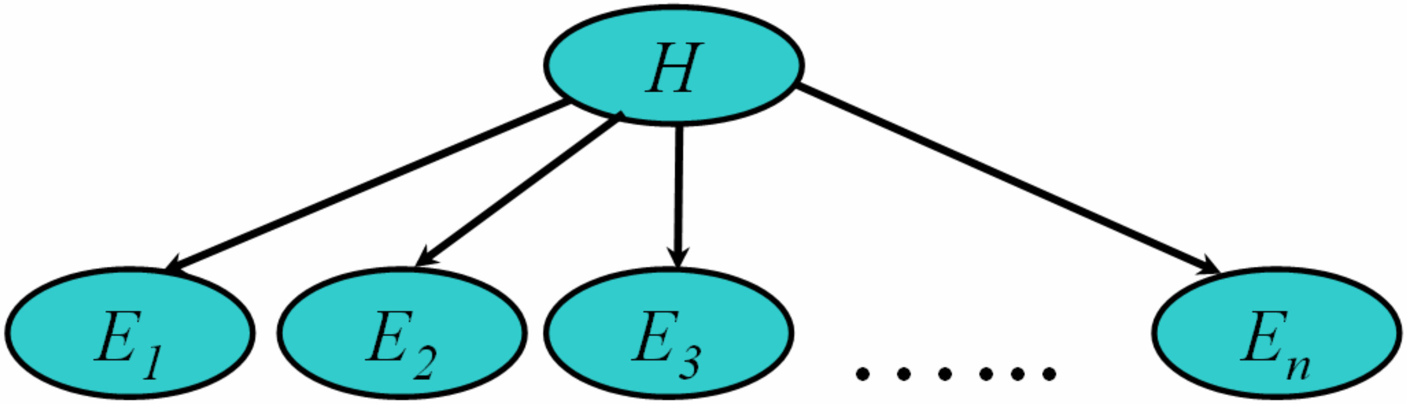
Serum Calcium and *Lung Tumor* are dependent

Serum Calcium is independent of *Lung Tumor*, given *Cancer*

$$P(L|SC,C) = P(L|C)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Naïve Bayes in general



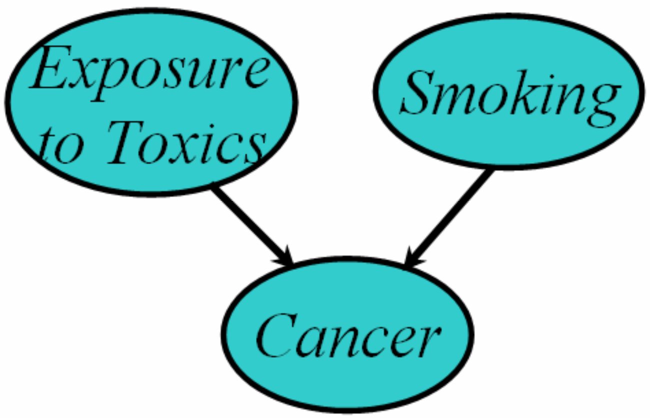
2n + 1 parameters:

$$P(h)$$

$$P(e_i \mid h), P(e_i \mid \bar{h}), \; i = 1, \ldots, n$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

More Conditional Independence: Explaining Away

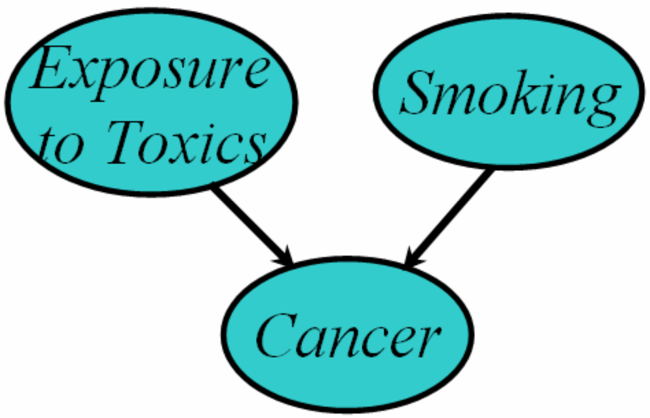


Exposure to Toxics and *Smoking* are independent

$$E \perp S$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

More Conditional Independence: Explaining Away



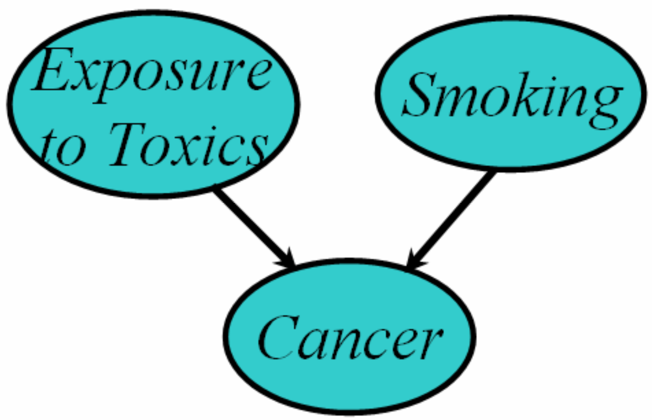
Exposure to Toxics and *Smoking* are independent

$$E \perp S$$

Exposure to Toxics is **dependent** on *Smoking*, given *Cancer*

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

More Conditional Independence: Explaining Away



Exposure to Toxics and *Smoking* are independent

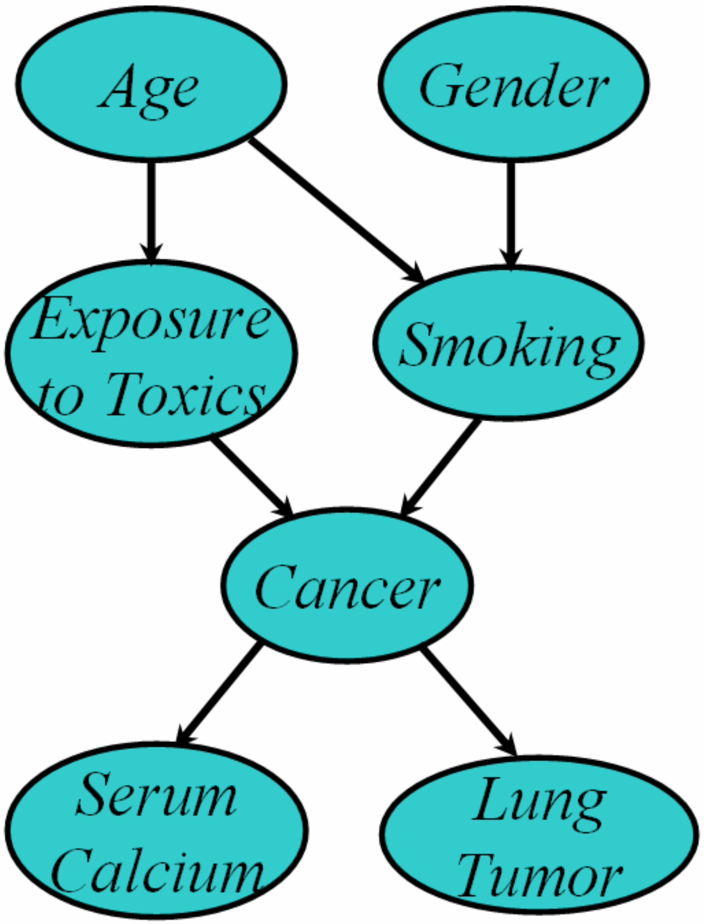
$$E \perp S$$

Exposure to Toxics is **dependent** on *Smoking*, given *Cancer*

$$P(E = heavy \mid C = malignant) >$$
$$P(E = heavy \mid C = malignant,\ S{=}heavy)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Put it all together



$$P(A, G, E, S, C, L, SC) =$$

$$P(A) \cdot P(G) \cdot$$

$$P(E \mid A) \cdot P(S \mid A, G) \cdot$$

$$P(C \mid E, S) \cdot$$

$$P(SC \mid C) \cdot P(L \mid C)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

General Product (Chain) Rule for Bayesian Networks

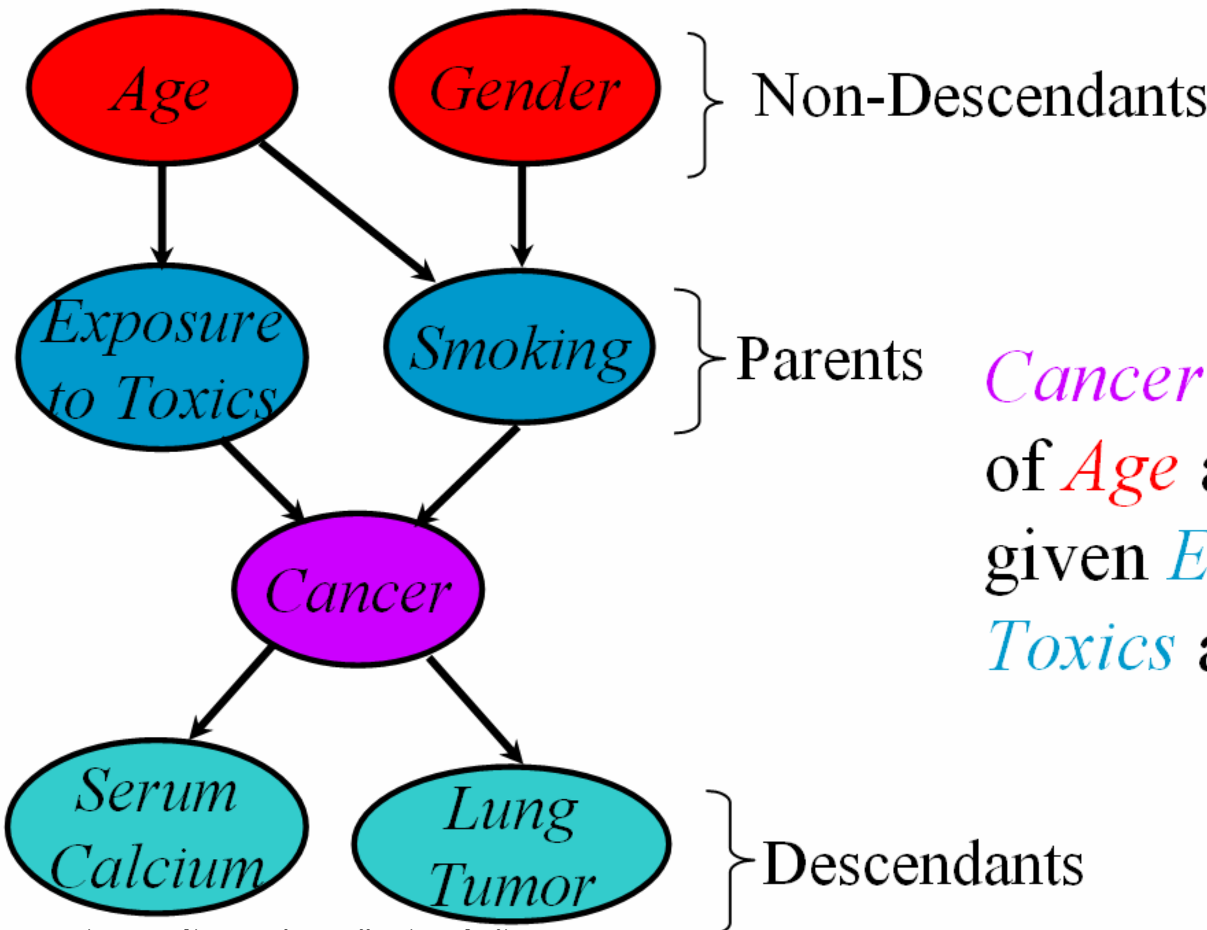
$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i \mid \boldsymbol{Pa_i})$$

$$\boldsymbol{Pa_i} = parents(X_i)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Conditional Independence

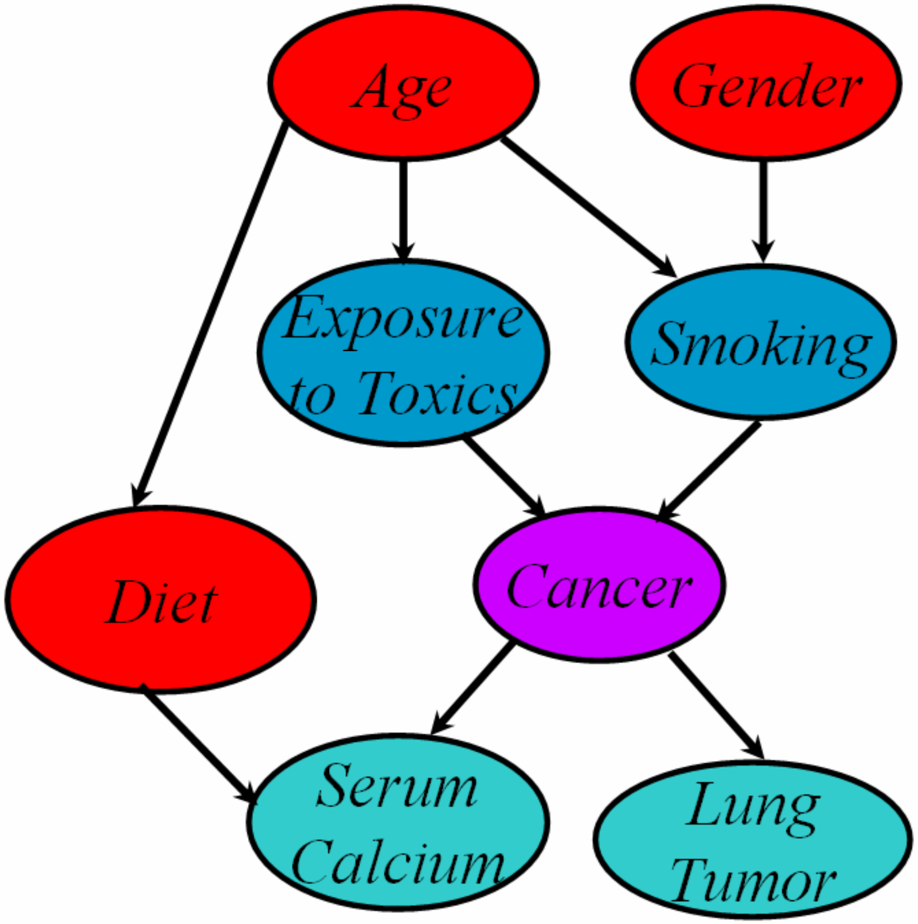
A variable (node) is conditionally independent of its non-descendants given its parents.



Cancer is independent of *Age* and *Gender* given *Exposure to Toxics* and *Smoking*.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Another non-descendant



Cancer is independent of *Diet* given *Exposure to Toxics* and *Smoking*.

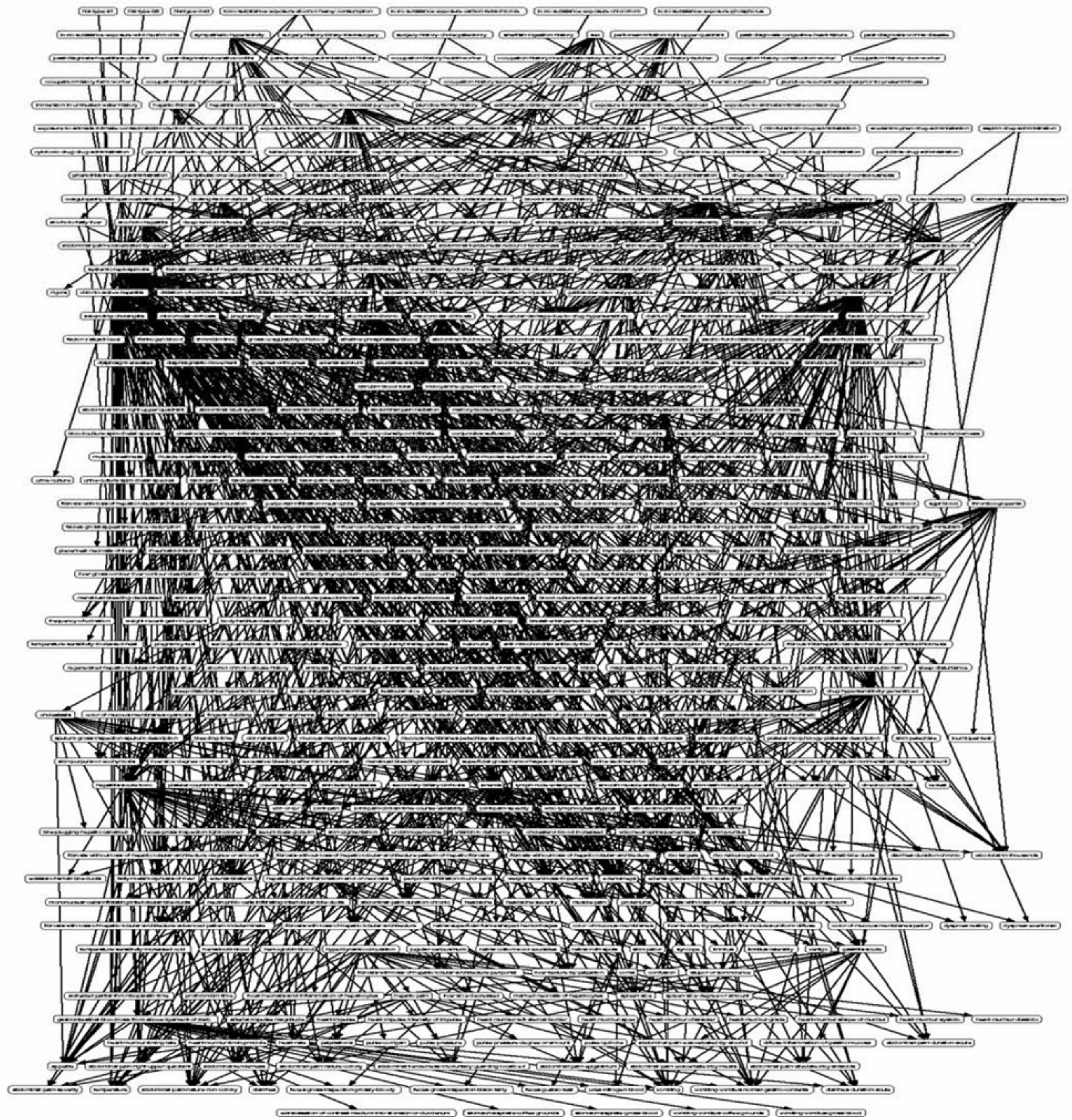
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Independence and Graph Separation

- Given a set of observations, is one set of variables dependent on another set?
- Observing effects can induce dependencies.
- d-separation (Pearl 1988) allows us to check conditional independence graphically.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

CPCS Network



© Jack Breese (Microsoft) & Daphne Koller (Stan

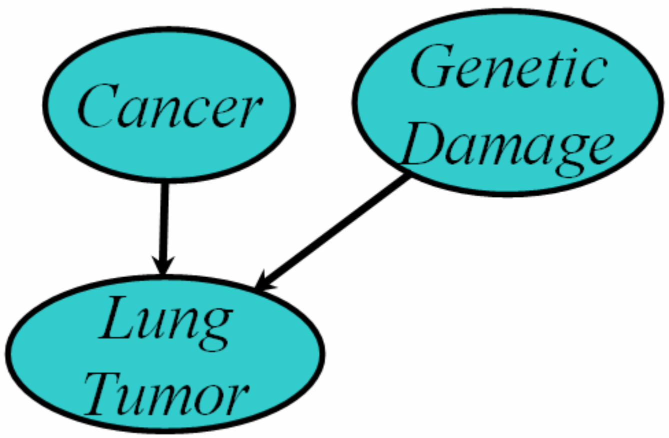
Structuring

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Structuring

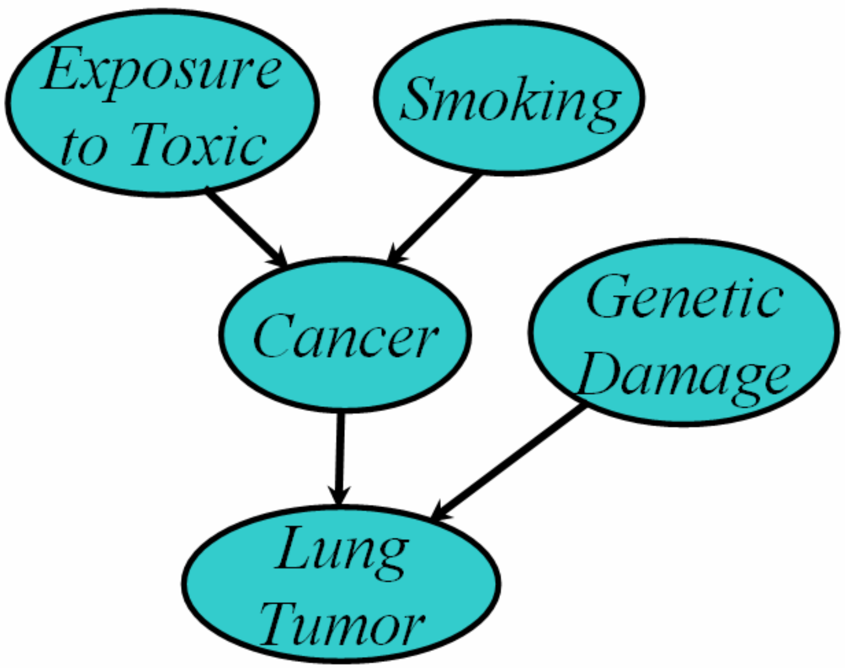
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Structuring



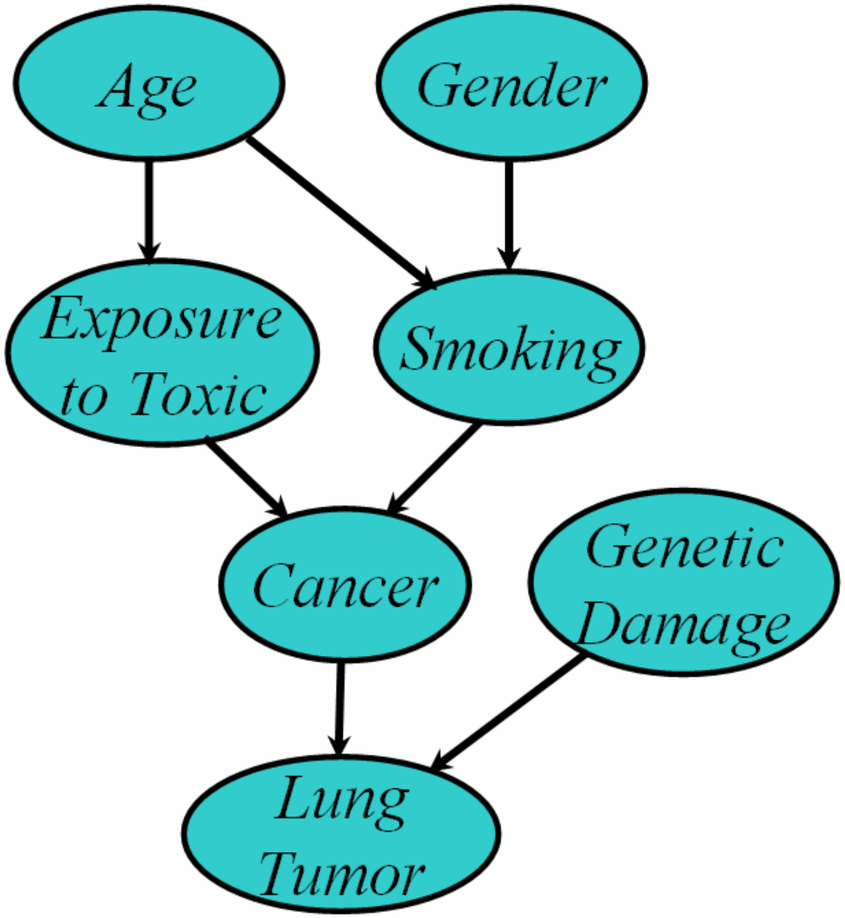
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Structuring



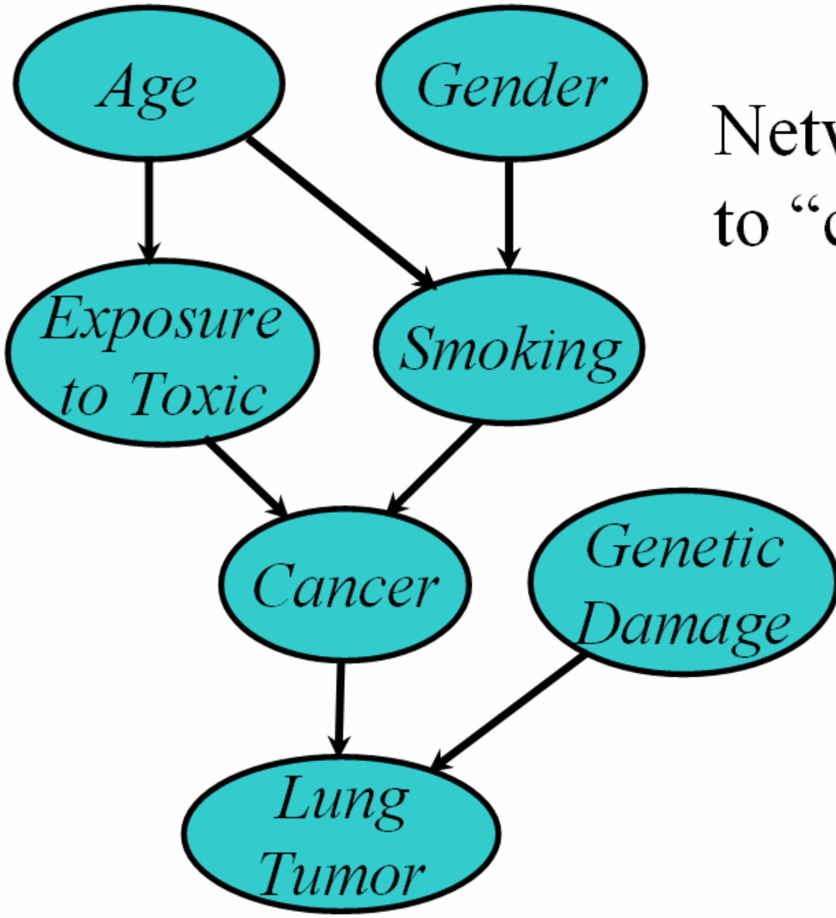
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Structuring



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

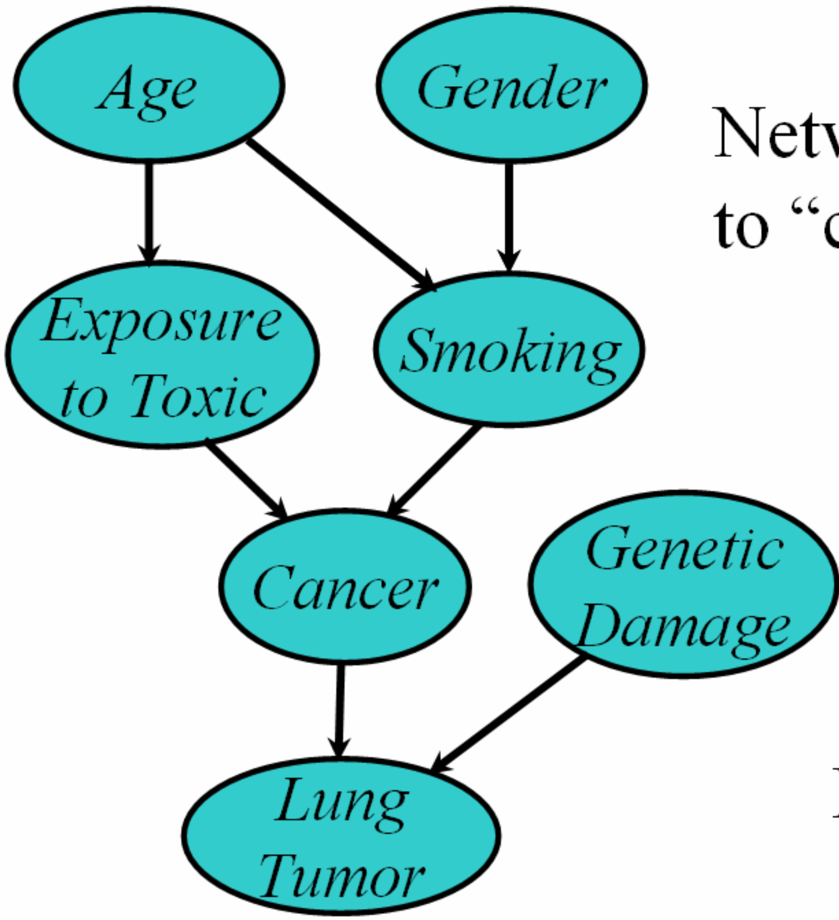
Structuring



Network structure corresponding to "causality" is usually good.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Structuring

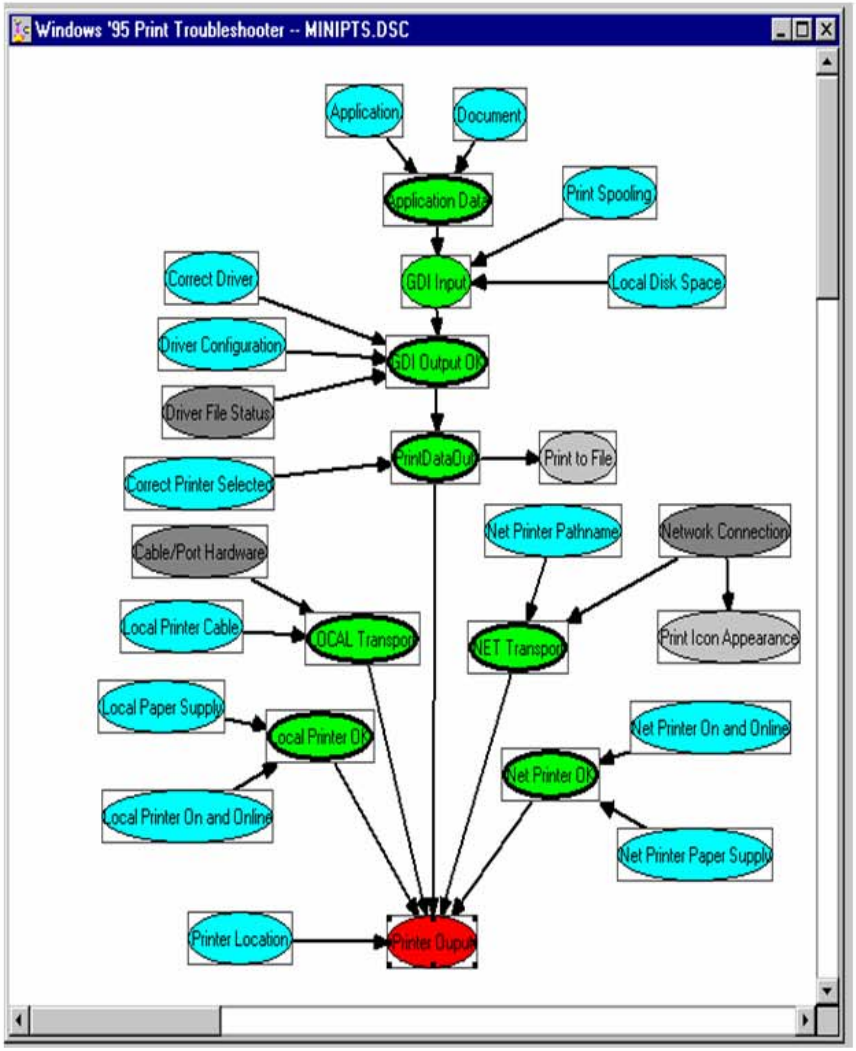


Network structure corresponding to “causality” is usually good.

Extending the conversation.

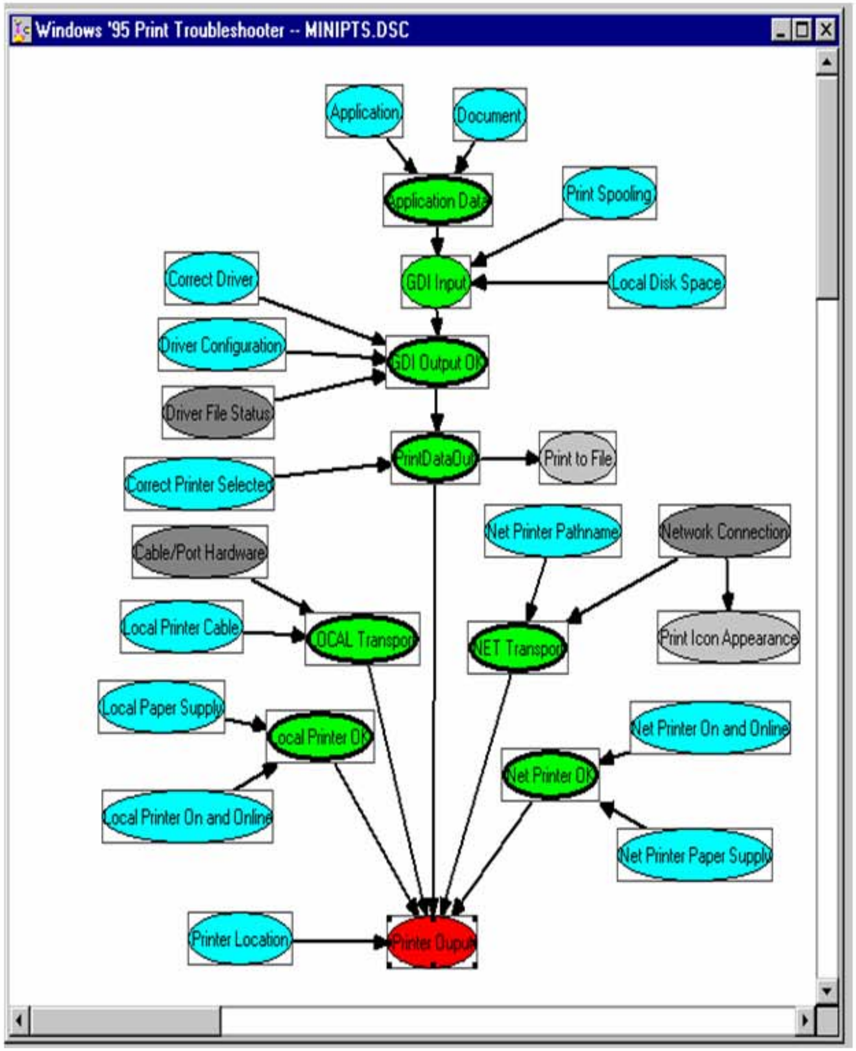
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Local Structure



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

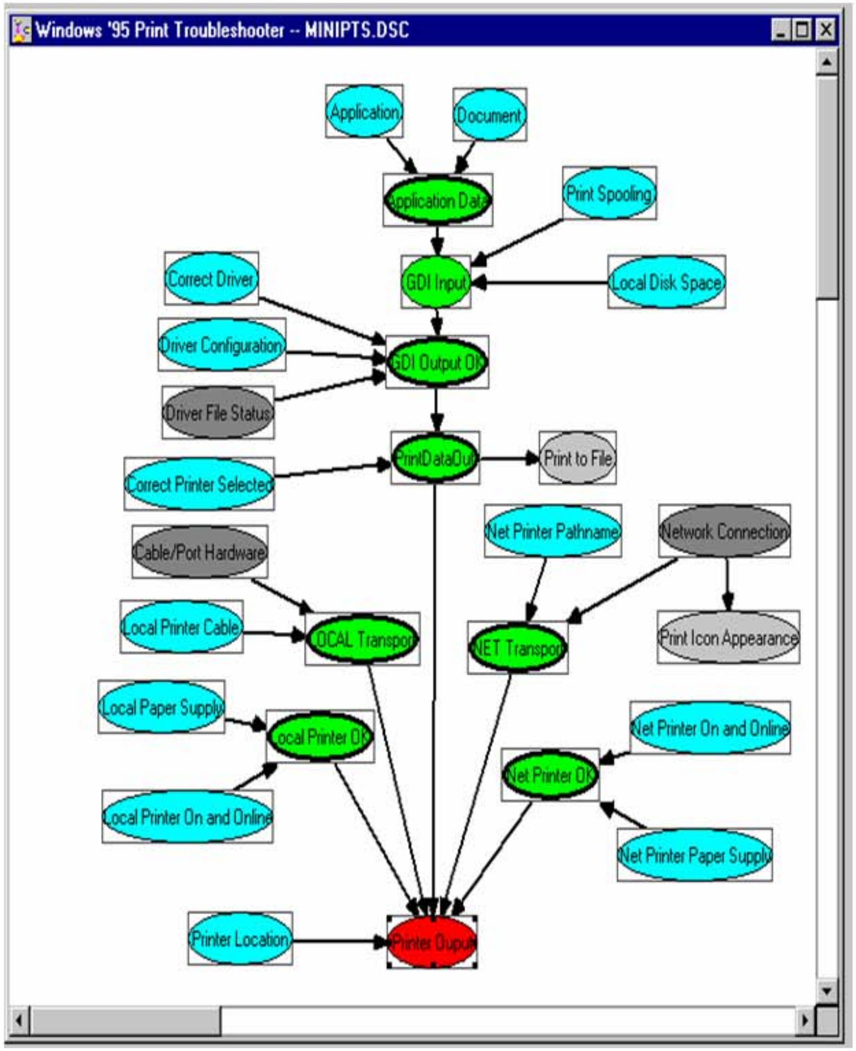
Local Structure



- Causal independence: from 2^n to $n+1$ parameters

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

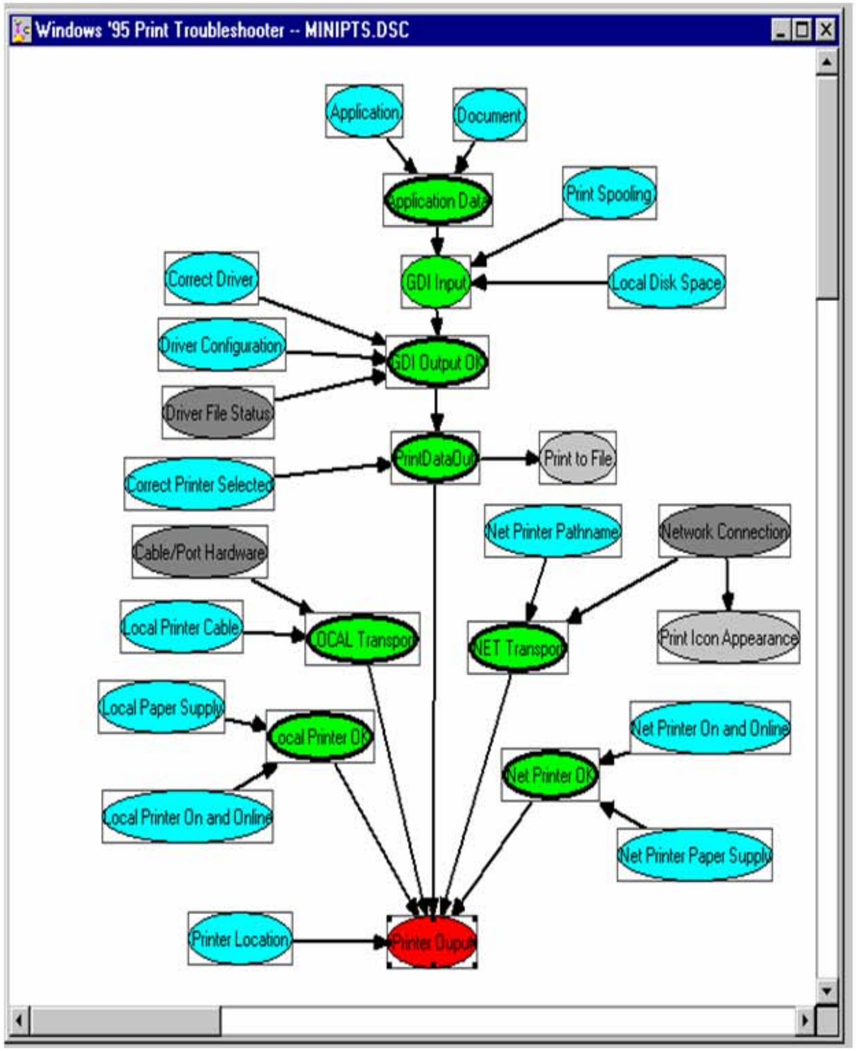
Local Structure



- Causal independence: from 2^n to $n+1$ parameters
- Asymmetric assessment: similar savings in practice.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

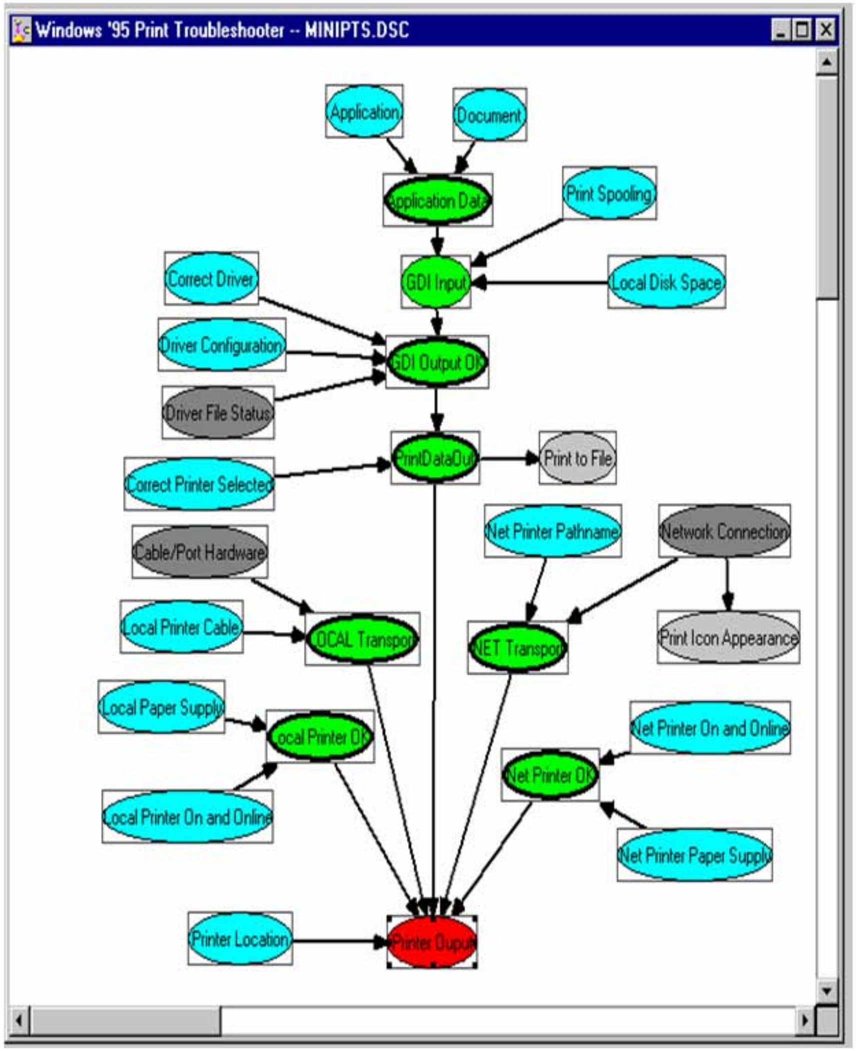
Local Structure



- Causal independence: from 2^n to $n+1$ parameters
- Asymmetric assessment: similar savings in practice.
- Typical savings (#params):

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

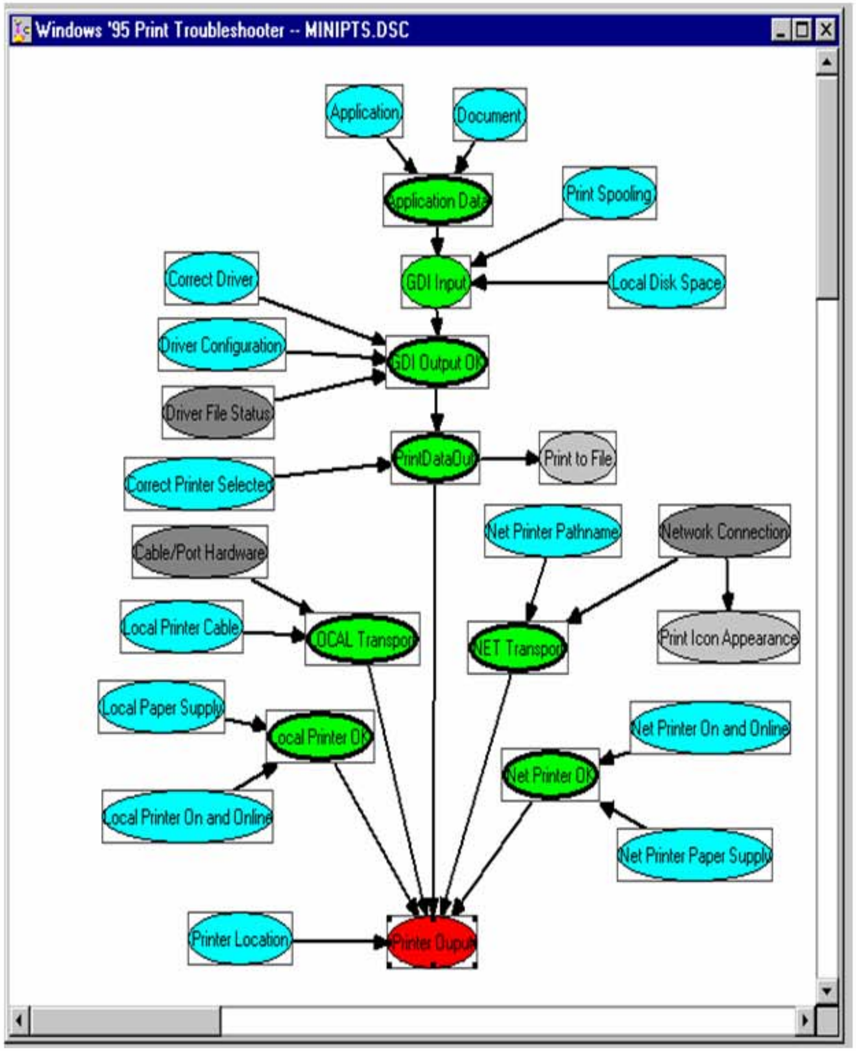
Local Structure



- Causal independence: from 2^n to $n+1$ parameters
- Asymmetric assessment: similar savings in practice.
- Typical savings (#params):
  - 145 to 55 for a small hardware network;

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Local Structure



- Causal independence: from 2^n to $n+1$ parameters
- Asymmetric assessment: similar savings in practice.
- Typical savings (#params):
  - 145 to 55 for a small hardware network;
  - 133,931,430 to 8254 for CPCS !!

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Course Contents

- Concepts in Probability
- Bayesian Networks
- » Inference
- Decision making
- Learning networks from data
- Reasoning over time
- Applications

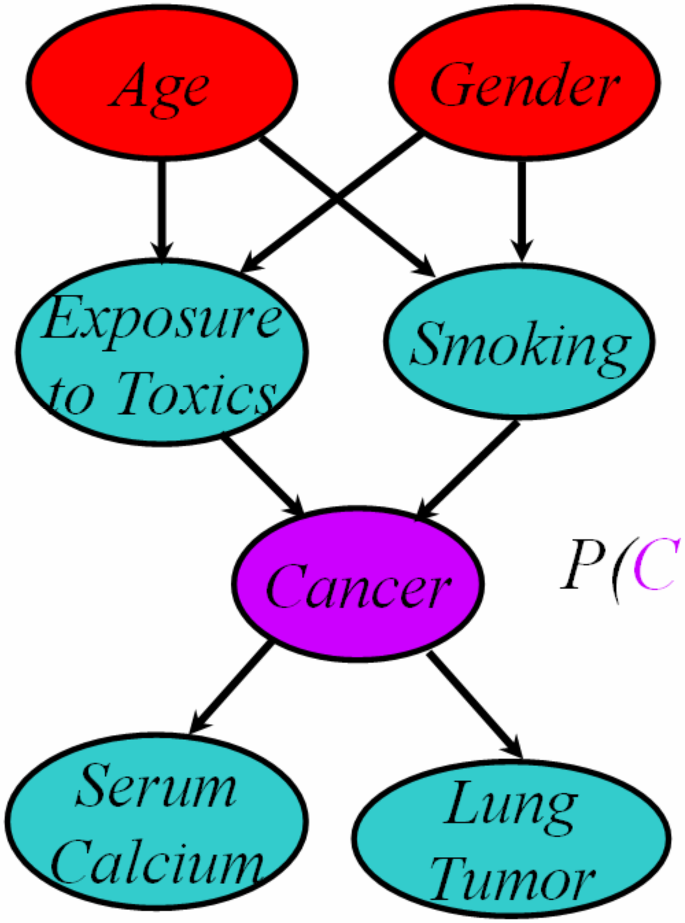
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Inference

- Patterns of reasoning
- Basic inference
- Exact inference
- Exploiting structure
- Approximate inference

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Predictive Inference

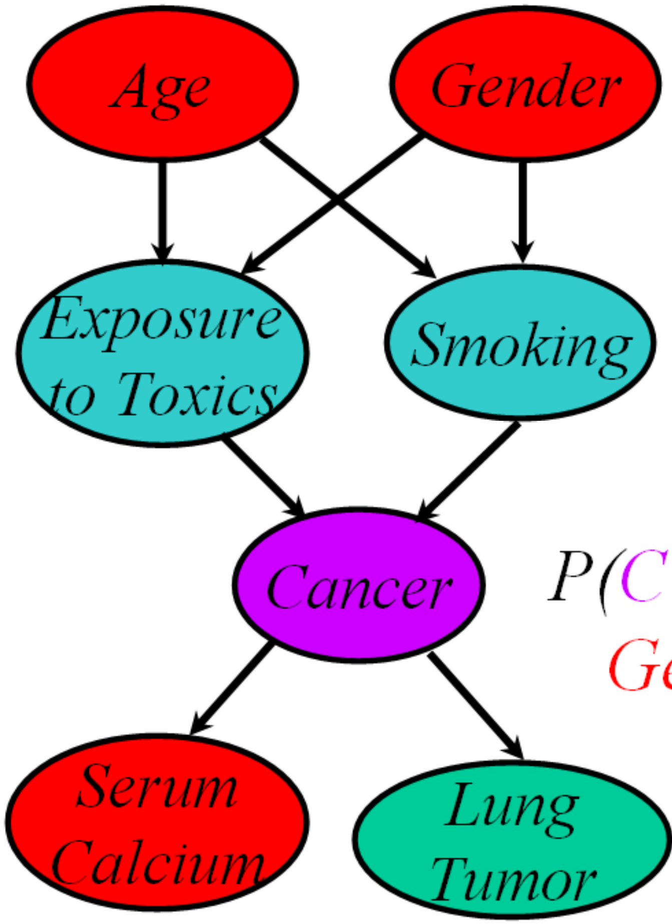


How likely are elderly males to get malignant cancer?

P(C=malignant | Age>60, Gender= male)

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Combined

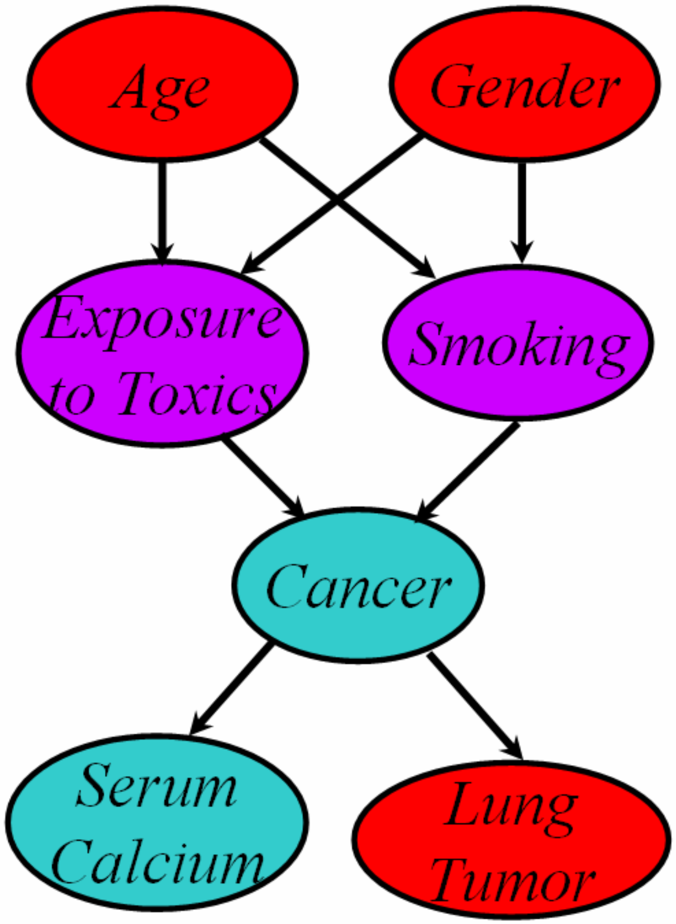


How likely is an elderly male patient with high Serum Calcium to have malignant cancer?

$$P(C = malignant \mid Age > 60, Gender = male, Serum\ Calcium = high)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

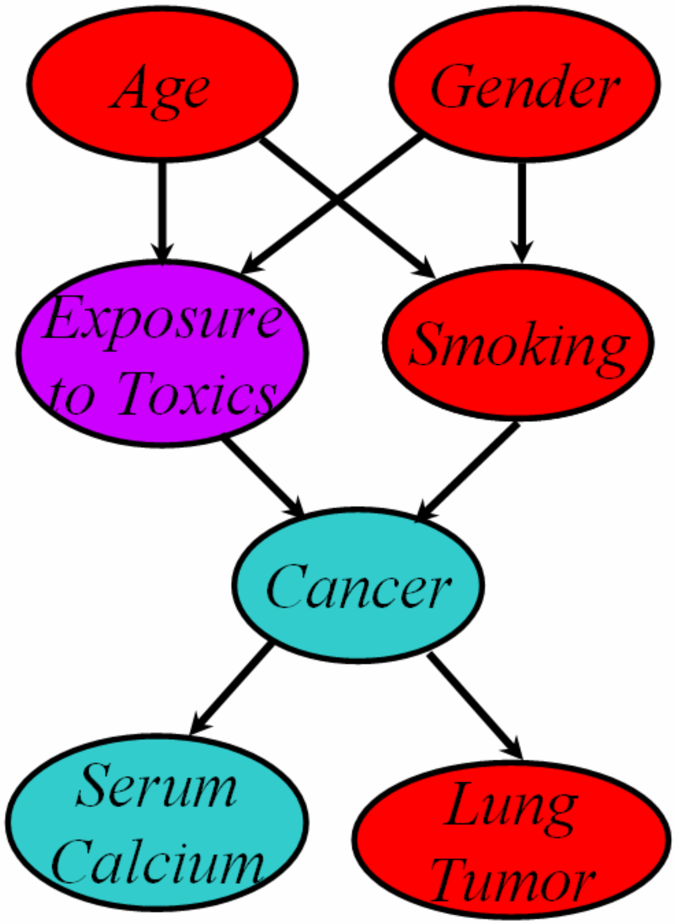
Explaining away



- If we see a lung tumor, the probability of heavy smoking and of exposure to toxics both go up.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Explaining away



- If we see a lung tumor, the probability of heavy smoking and of exposure to toxics both go up.
- If we then observe heavy smoking, the probability of exposure to toxics goes back down.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Inference in Belief Networks

- Find $P(Q{=}q \mid \mathbf{E}{=}\,\mathbf{e})$
  - Q the query variable
  - $\mathbf{E}$ set of evidence variables

$$P(q \mid \mathbf{e}) = \frac{P(q, \mathbf{e})}{P(\mathbf{e})}$$

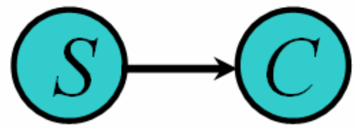
$X_1, \dots, X_n$ are network variables except Q, $\mathbf{E}$

$$P(q, \mathbf{e}) = \sum_{x_1, \dots, x_n} P(q, e, x_1, \dots, x_n)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference

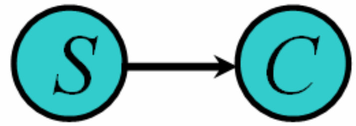
$P(c) = ?$



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference

$$P(c) = ?$$

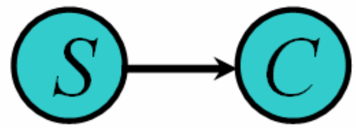


- $P(C,S) = P(C|S)\,P(S)$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference

$$P(c) = ?$$



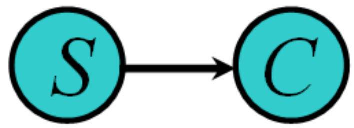
- $P(C,S) = P(C|S)\, P(S)$

Smoking=	no	light	heavy
P(C=none)	0.96	0.88	0.60
P(C=benign)	0.03	0.08	0.25
P(C=malig)	0.01	0.04	0.15

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference

$$P(c) = ?$$



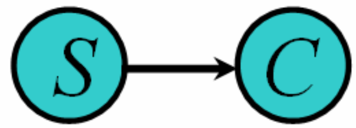
- $P(C,S) = P(C|S)\,P(S)$

Smoking=	no	light	heavy
P(C=none)	0.96	0.88	0.60
P(C=benign)	0.03	0.08	0.25
P(C=malig)	0.01	0.04	0.15

P(S=no)	0.80
P(S=light)	0.15
P(S=heavy)	0.05

Basic Inference

$P(c) = ?$



- $P(C,S) = P(C|S)\,P(S)$

Smoking=	*no*	*light*	*heavy*
P(C=none)	0.96	0.88	0.60
P(C=benign)	0.03	0.08	0.25
P(C=malig)	0.01	0.04	0.15

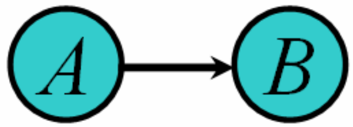
P(S=no)	0.80
P(S=light)	0.15
P(S=heavy)	0.05

$S\Downarrow$ $C\Rightarrow$	*none*	*benign*	*malig*	total
no	0.768	0.024	0.008	.80
light	0.132	0.012	0.006	.15
heavy	0.035	0.010	0.005	.05
total	0.935	0.046	0.019	

P(Cancer)

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

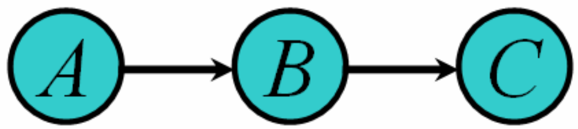
Basic Inference

$$A \rightarrow B \rightarrow C$$

$$P(b) = \sum_a P(a, b) = \sum_a P(b \mid a)\, P(a)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference



$$P(b) = \sum_a P(a, b) = \sum_a P(b \mid a)\, P(a)$$

$$P(c) = \sum_b P(c \mid b)\, P(b)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference

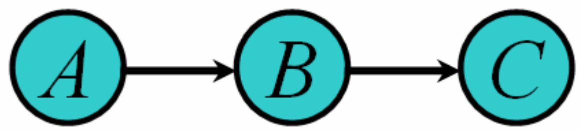
$$A \rightarrow B \rightarrow C$$

$$\underbrace{P(b)} = \sum_a P(a, b) = \sum_a P(b \mid a)\, P(a)$$

$$P(c) = \sum_b P(c \mid b)\, \overbrace{P(b)}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference



$$P(b) = \sum_a P(a, b) = \sum_a P(b \mid a)\, P(a)$$

$$P(c) = \sum_b P(c \mid b)\, P(b)$$

$$P(c) = \sum_{b,a} P(a, b, c)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference

$$A \rightarrow B \rightarrow C$$

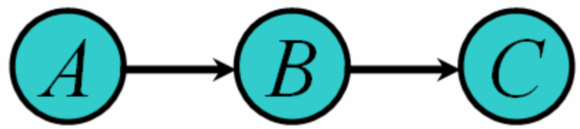
$$\underbrace{P(b)} = \sum_a P(a, b) = \sum_a P(b \mid a)\, P(a)$$

$$P(c) = \sum_b P(c \mid b)\, \overbrace{P(b)}$$

$$P(c) = \sum_{b,a} P(a, b, c) = \sum_{b,a} P(c \mid b)\, P(b \mid a)\, P(a)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference



$$P(b) = \sum_a P(a, b) = \sum_a P(b \mid a)\, P(a)$$

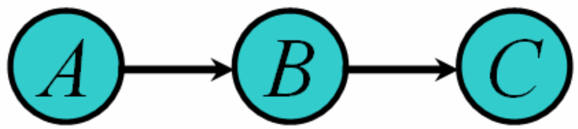
$$P(c) = \sum_b P(c \mid b)\, P(b)$$

$$P(c) = \sum_{b,a} P(a, b, c) = \sum_{b,a} P(c \mid b)\, P(b \mid a)\, P(a)$$

$$= \sum_b P(c \mid b) \sum_a P(b \mid a)\, P(a)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Basic Inference



$$\underbrace{P(b)} = \sum_{a} P(a, b) = \sum_{a} P(b \mid a)\, P(a)$$

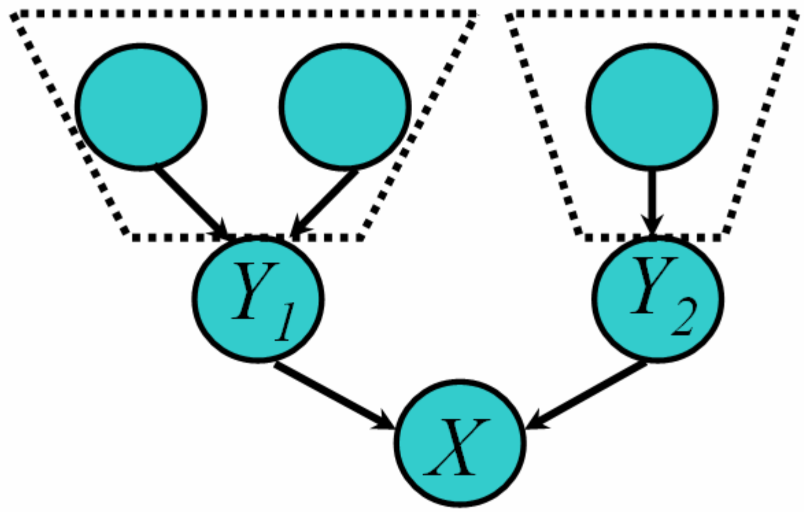
$$P(c) = \sum_{b} P(c \mid b)\, \overbrace{P(b)}$$

$$P(c) = \sum_{b,a} P(a, b, c) = \sum_{b,a} P(c \mid b)\, P(b \mid a)\, P(a)$$

$$= \sum_{b} P(c \mid b) \underbrace{\sum_{a} P(b \mid a)\, P(a)}_{P(b)}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

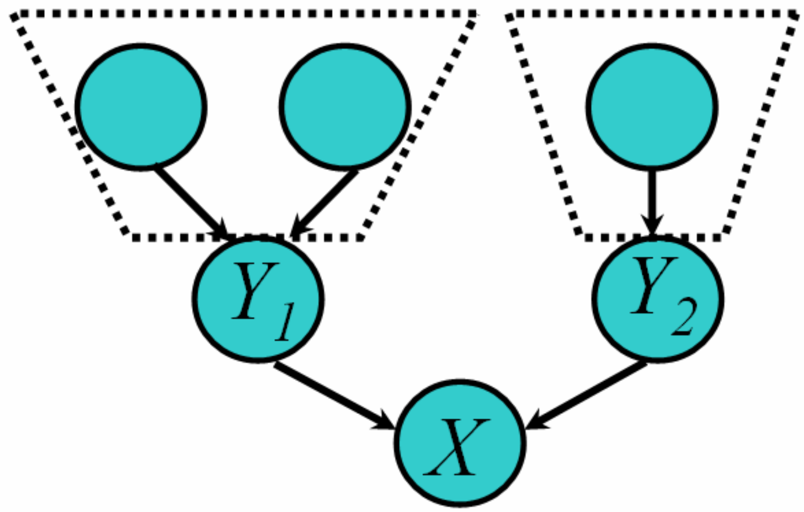
Inference in trees



$$P(x) = \sum_{y_1, y_2} P(x \mid y_1, y_2)\, P(y_1, y_2)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Inference in trees



$$P(x) = \sum_{y_1, y_2} P(x \mid y_1, y_2)\, P(y_1, y_2)$$

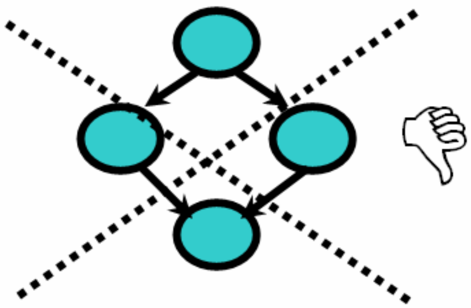
because of independence of Y_1, Y_2:

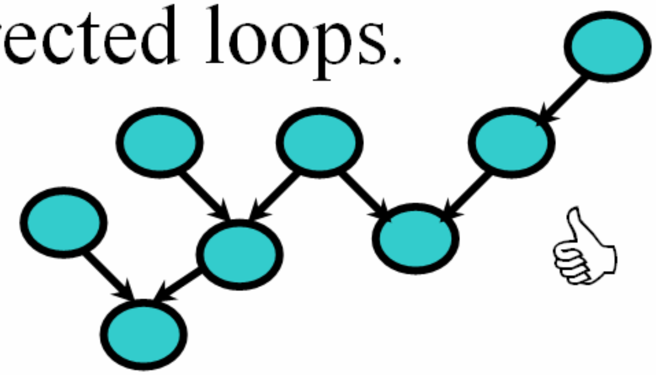
$$= \sum_{y_1, y_2} P(x \mid y_1, y_2)\, P(y_1)\, P(y_2)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Polytrees

- A network is *singly connected* (a *polytree*) if it contains no undirected loops.





Theorem: Inference in a singly connected network can be done in linear time*.

Main idea: in variable elimination, need only maintain distributions over single nodes.

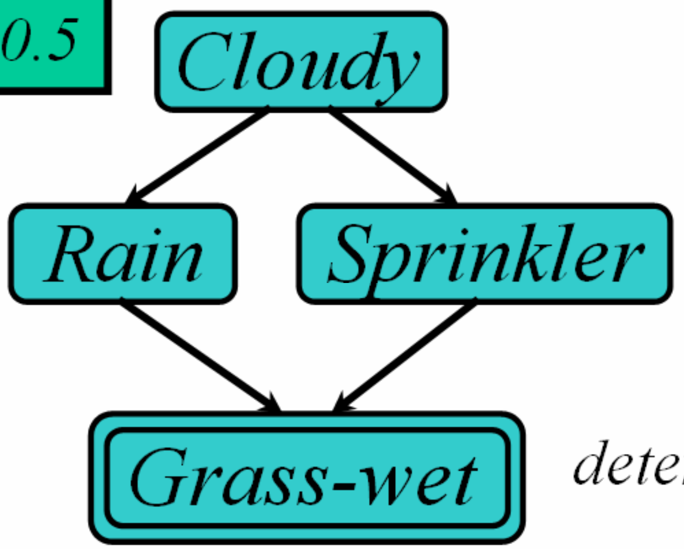
* in network size including table sizes.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The problem with loops

$P(c)$	0.5

	c	$\bar{c}$
$P(r)$	0.99	0.01



	c	$\bar{c}$
$P(s)$	0.01	0.99

deterministic or

The grass is dry only if no rain and no sprinklers.

$$P(\bar{g}) = P(\bar{r}, \bar{s}) \sim 0$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The problem with loops contd.

$$P(\bar{g}) =$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The problem with loops contd.

$$P(\bar{g}) = \quad P(\bar{g} \mid r, s)\, P(r, s) + P(\bar{g} \mid r, \bar{s})\, P(r, \bar{s})$$

$$+ P(\bar{g} \mid \bar{r}, s)\, P(\bar{r}, s) + P(\bar{g} \mid \bar{r}, \bar{s})\, P(\bar{r}, \bar{s})$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The problem with loops contd.

$$P(\bar{g}) = \overbrace{P(\bar{g} \mid r, s)}^{0} P(r, s) + \overbrace{P(\bar{g} \mid r, \bar{s})}^{0} P(r, \bar{s})$$

$$+ \underbrace{P(\bar{g} \mid \bar{r}, s)}_{0} P(\bar{r}, s) + \underbrace{P(\bar{g} \mid \bar{r}, \bar{s})}_{1} P(\bar{r}, \bar{s})$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The problem with loops contd.

$$P(\bar{g}) = \overbrace{P(\bar{g} \mid r, s)}^{0} P(r, s) + \overbrace{P(\bar{g} \mid r, \bar{s})}^{0} P(r, \bar{s})$$

$$+ \underbrace{P(\bar{g} \mid \bar{r}, s)}_{0} P(\bar{r}, s) + \underbrace{P(\bar{g} \mid \bar{r}, \bar{s})}_{1} P(\bar{r}, \bar{s})$$

$$= P(\bar{r}, \bar{s})$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The problem with loops contd.

$$P(\bar{g}) = \overbrace{P(\bar{g} \mid r, s)}^{0} P(r, s) + \overbrace{P(\bar{g} \mid r, \bar{s})}^{0} P(r, \bar{s})$$

$$+ \underbrace{P(\bar{g} \mid \bar{r}, s)}_{0} P(\bar{r}, s) + \underbrace{P(\bar{g} \mid \bar{r}, \bar{s})}_{1} P(\bar{r}, \bar{s})$$

$$= P(\bar{r}, \bar{s})$$

$$= P(\bar{r})\, P(\bar{s}) \sim 0.5 \cdot 0.5 = 0.25$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The problem with loops contd.

$$P(\bar{g}) = \overbrace{P(\bar{g} \mid r, s)}^{0} P(r, s) + \overbrace{P(\bar{g} \mid r, \bar{s})}^{0} P(r, \bar{s})$$

$$+ \underbrace{P(\bar{g} \mid \bar{r}, s)}_{0} P(\bar{r}, s) + \underbrace{P(\bar{g} \mid \bar{r}, \bar{s})}_{1} P(\bar{r}, \bar{s})$$

$$= P(\bar{r}, \bar{s}) \sim 0$$

$$= P(\bar{r})\, P(\bar{s}) \sim 0.5 \cdot 0.5 = 0.25$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The problem with loops contd.

$$P(\bar{g}) = \overbrace{P(\bar{g} \mid r, s)}^{0} P(r, s) + \overbrace{P(\bar{g} \mid r, \bar{s})}^{0} P(r, \bar{s})$$

$$+ \underbrace{P(\bar{g} \mid \bar{r}, s)}_{0} P(\bar{r}, s) + \underbrace{P(\bar{g} \mid \bar{r}, \bar{s})}_{1} P(\bar{r}, \bar{s})$$

$$= P(\bar{r}, \bar{s}) \sim 0$$

$$\neq P(\bar{r})\, P(\bar{s}) \sim 0.5 \cdot 0.5 = 0.25$$

problem

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

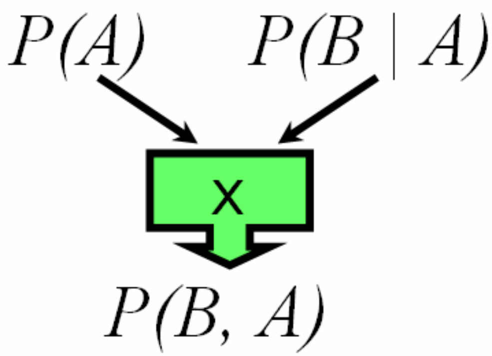
Variable elimination

$$P(c) = \sum_b P(c \mid b) \underbrace{\sum_a P(b \mid a)\, P(a)}_{P(b)}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Variable elimination

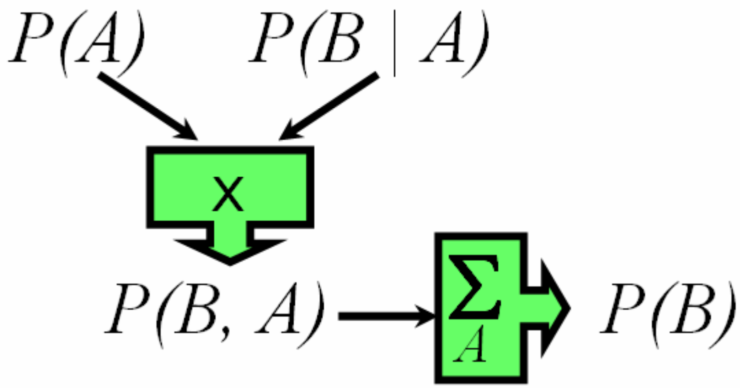
$$P(c) = \sum_b P(c \mid b) \underbrace{\sum_a P(b \mid a)\, P(a)}_{P(b)}$$



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Variable elimination

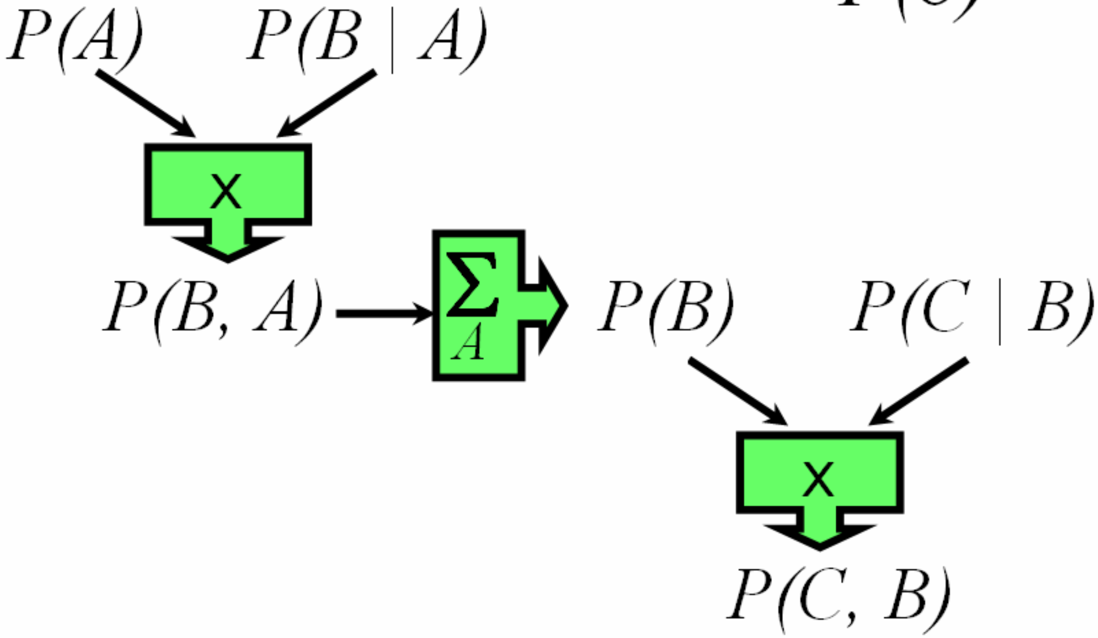
$$P(c) = \sum_b P(c \mid b) \underbrace{\sum_a P(b \mid a)\, P(a)}_{P(b)}$$



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Variable elimination

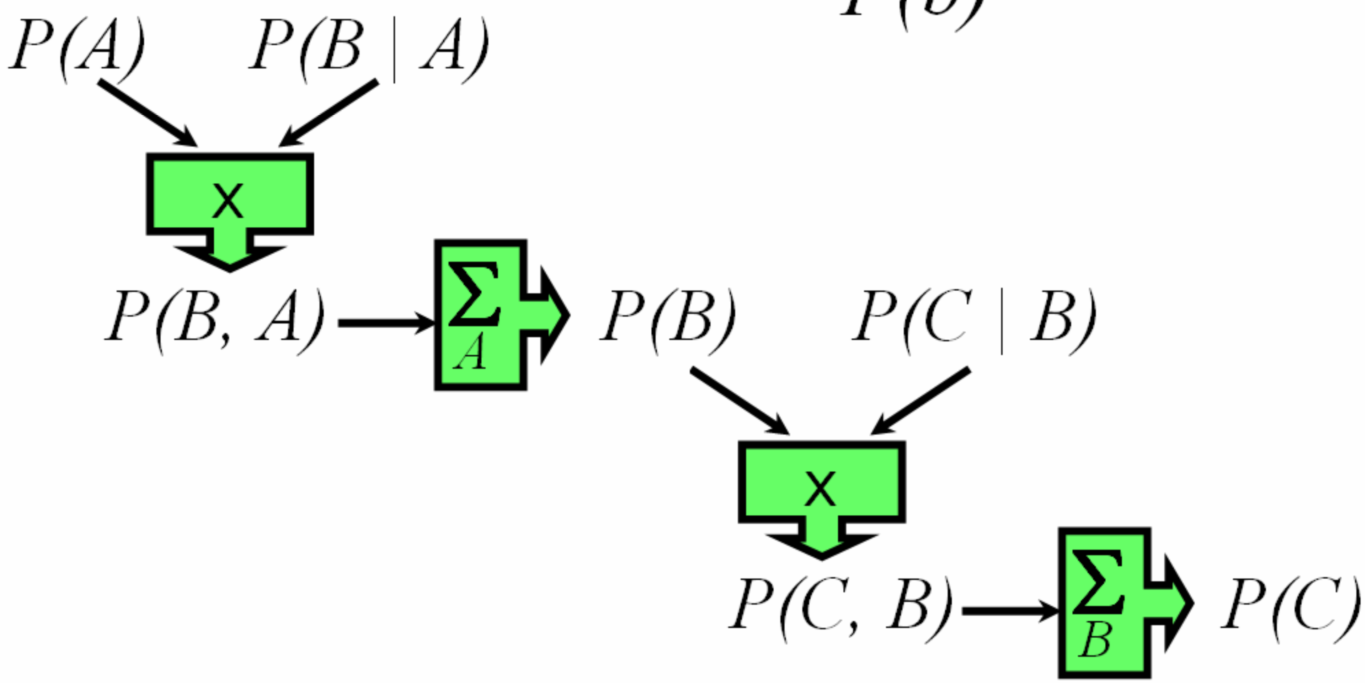
$$P(c) = \sum_b P(c \mid b) \underbrace{\sum_a P(b \mid a)\, P(a)}_{P(b)}$$



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Variable elimination

$$P(c) = \sum_b P(c \mid b) \underbrace{\sum_a P(b \mid a)\, P(a)}_{P(b)}$$



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Inference as variable elimination

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Inference as variable elimination

- A **factor** over X is a function from $val(X)$ to numbers in [0,1]:

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Inference as variable elimination

- A **factor** over *X* is a function from *val(X)* to numbers in [0,1]:
  - A CPT is a factor

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Inference as variable elimination

- A **factor** over ***X*** is a function from *val(**X**)* to numbers in [0,1]:
  - A CPT is a factor
  - A joint distribution is also a factor

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Inference as variable elimination

- A **factor** over X is a function from $val(X)$ to numbers in [0,1]:
  - A CPT is a factor
  - A joint distribution is also a factor
- BN inference:

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Inference as variable elimination

- A **factor** over *X* is a function from *val(X)* to numbers in [0,1]:
  - A CPT is a factor
  - A joint distribution is also a factor
- BN inference:
  - factors are multiplied to give new ones

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Inference as variable elimination

- A **factor** over *X* is a function from *val(X)* to numbers in [0,1]:
  - A CPT is a factor
  - A joint distribution is also a factor
- BN inference:
  - factors are multiplied to give new ones
  - variables in factors summed out

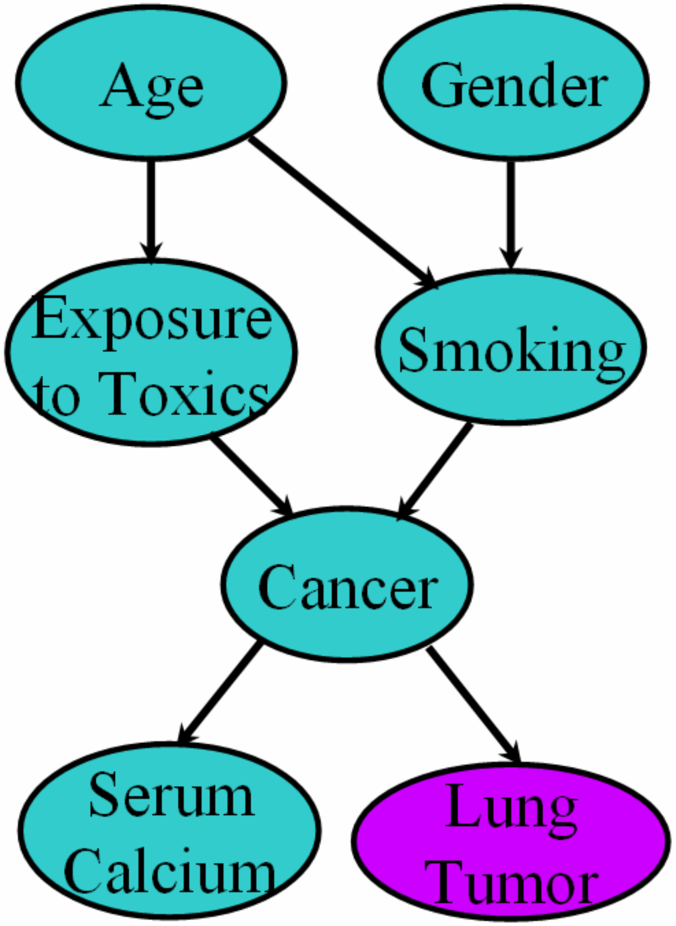
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Inference as variable elimination

- A **factor** over ***X*** is a function from *val(**X**)* to numbers in [0,1]:
  - A CPT is a factor
  - A joint distribution is also a factor
- BN inference:
  - factors are multiplied to give new ones
  - variables in factors summed out
- A variable can be summed out as soon as all factors mentioning it have been multiplied.

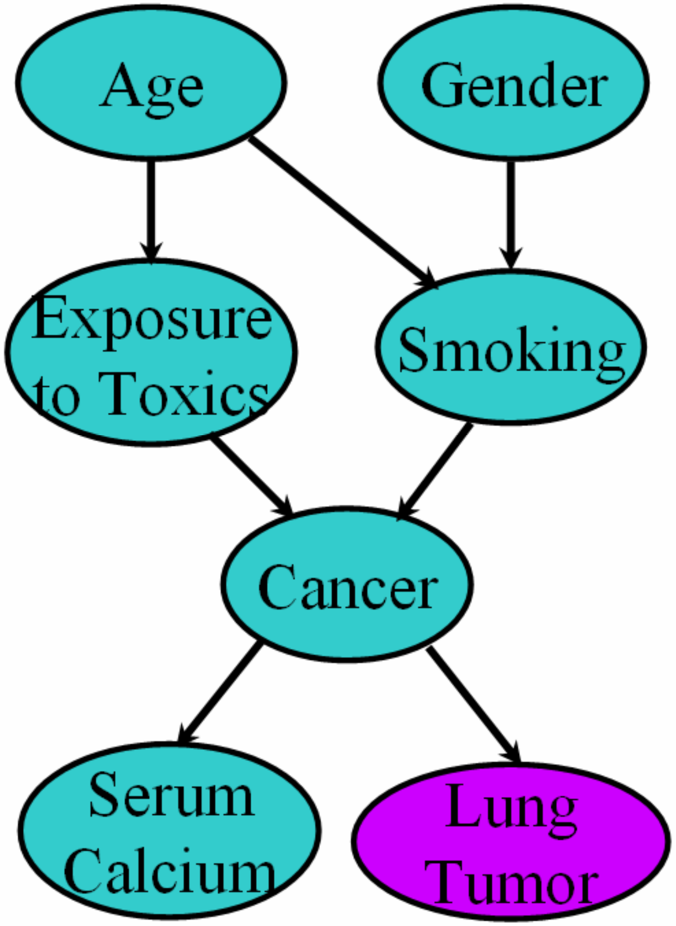
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

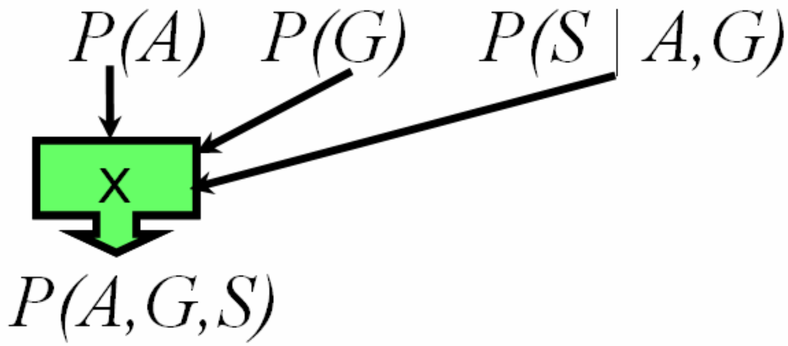
Variable Elimination with loops



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

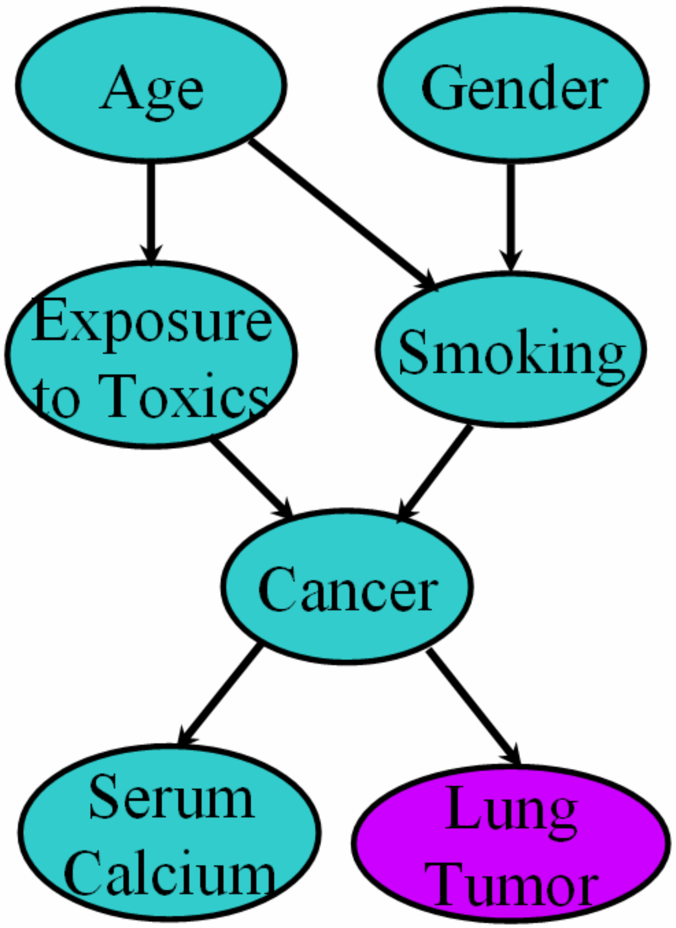
Variable Elimination with loops

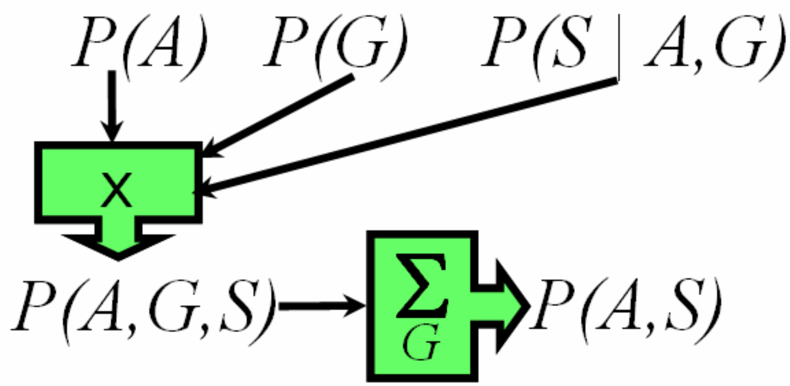




© Jack Breese (Microsoft) & Daphne Koller (Stanford)

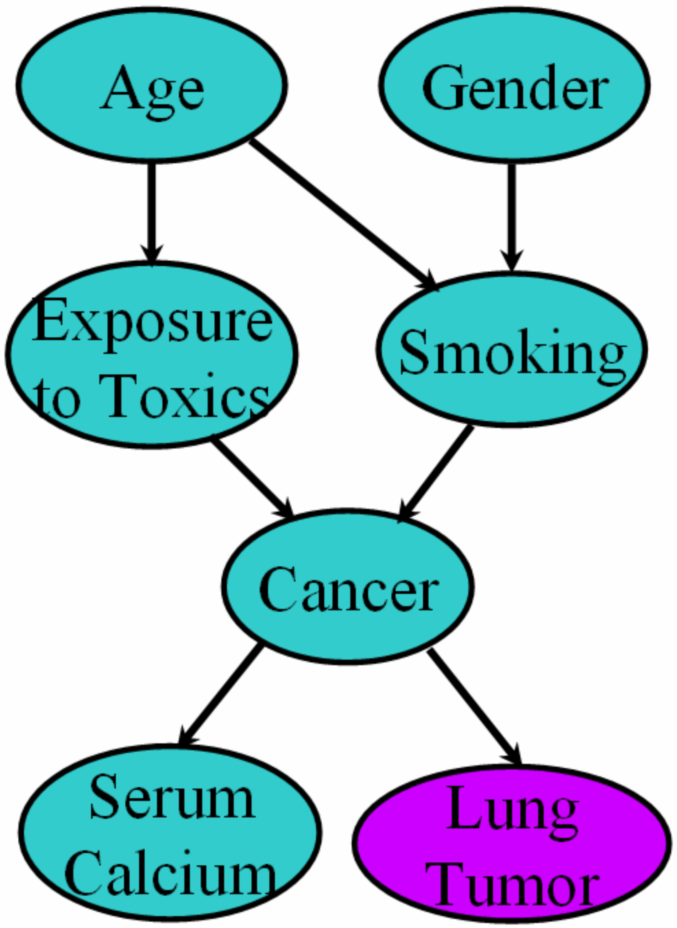
Variable Elimination with loops

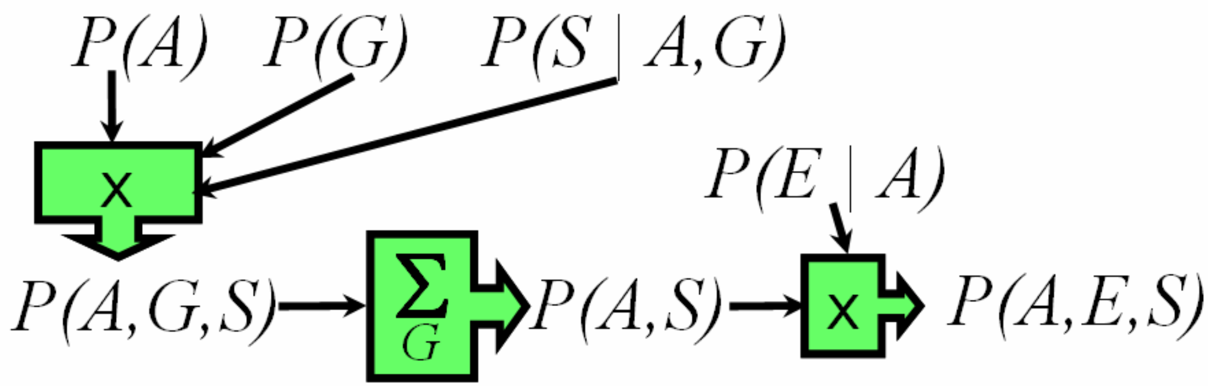




© Jack Breese (Microsoft) & Daphne Koller (Stanford)

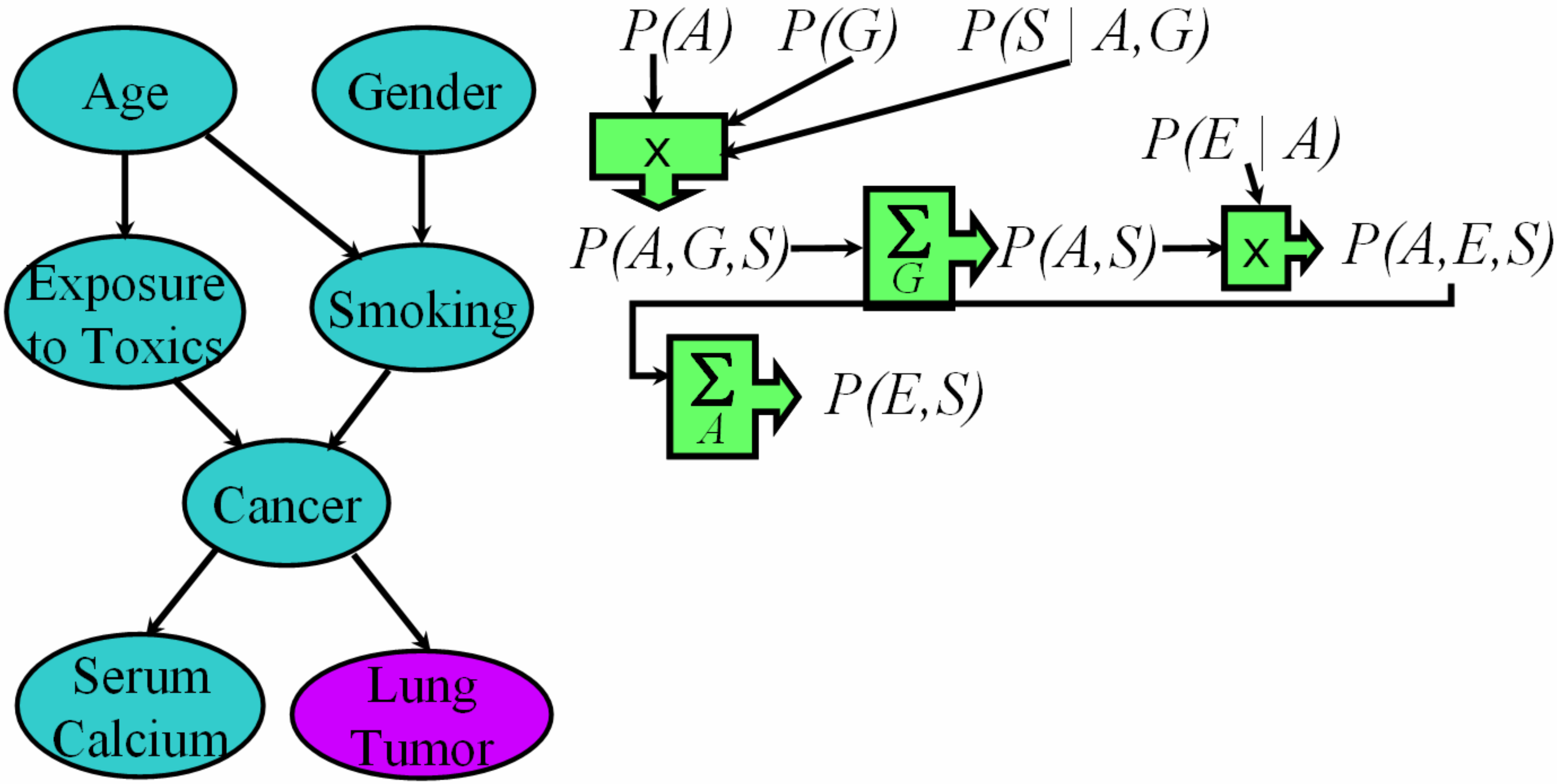
Variable Elimination with loops





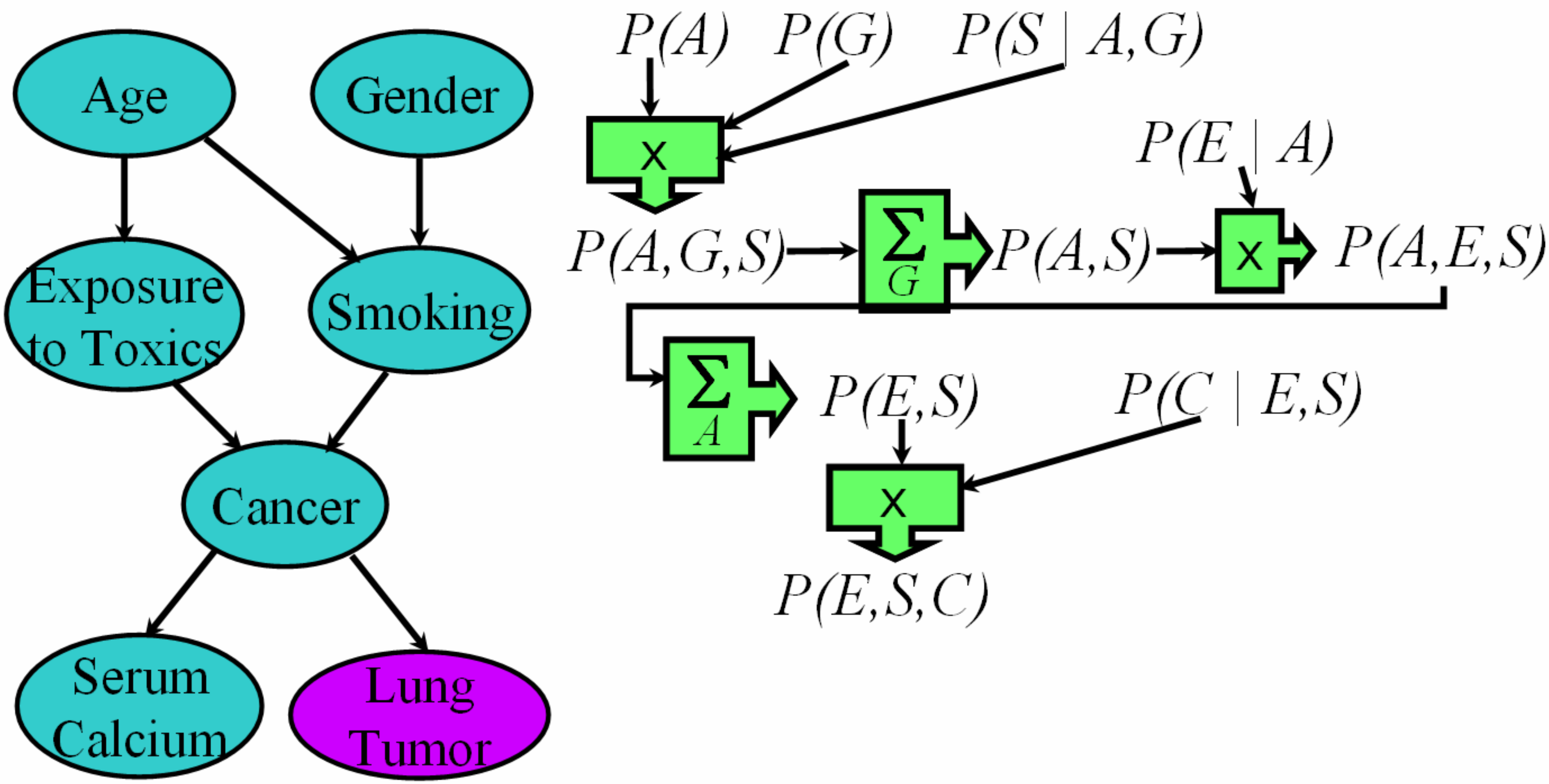
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Variable Elimination with loops



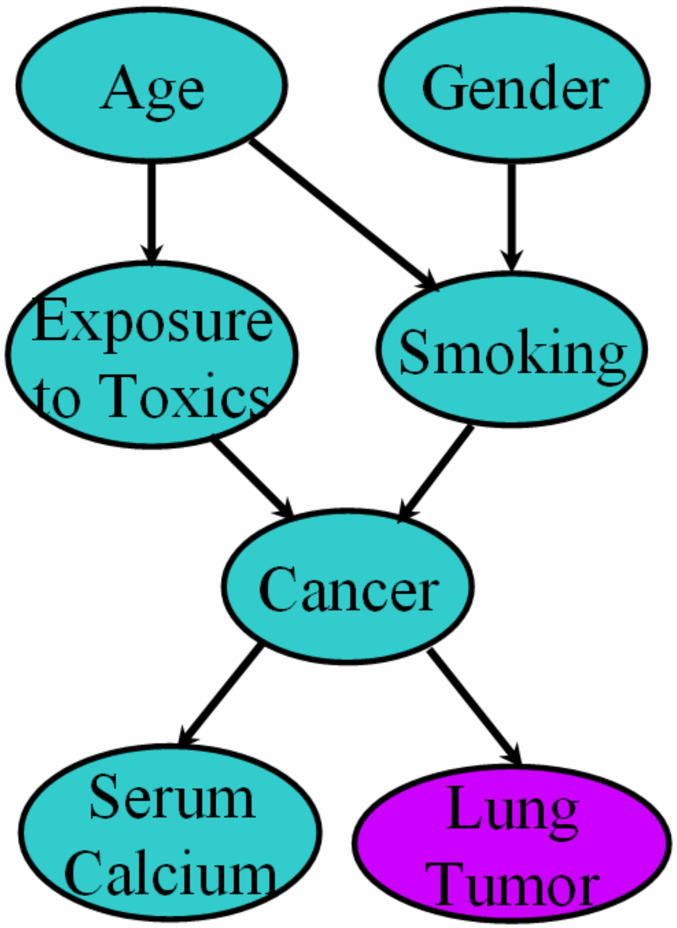
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

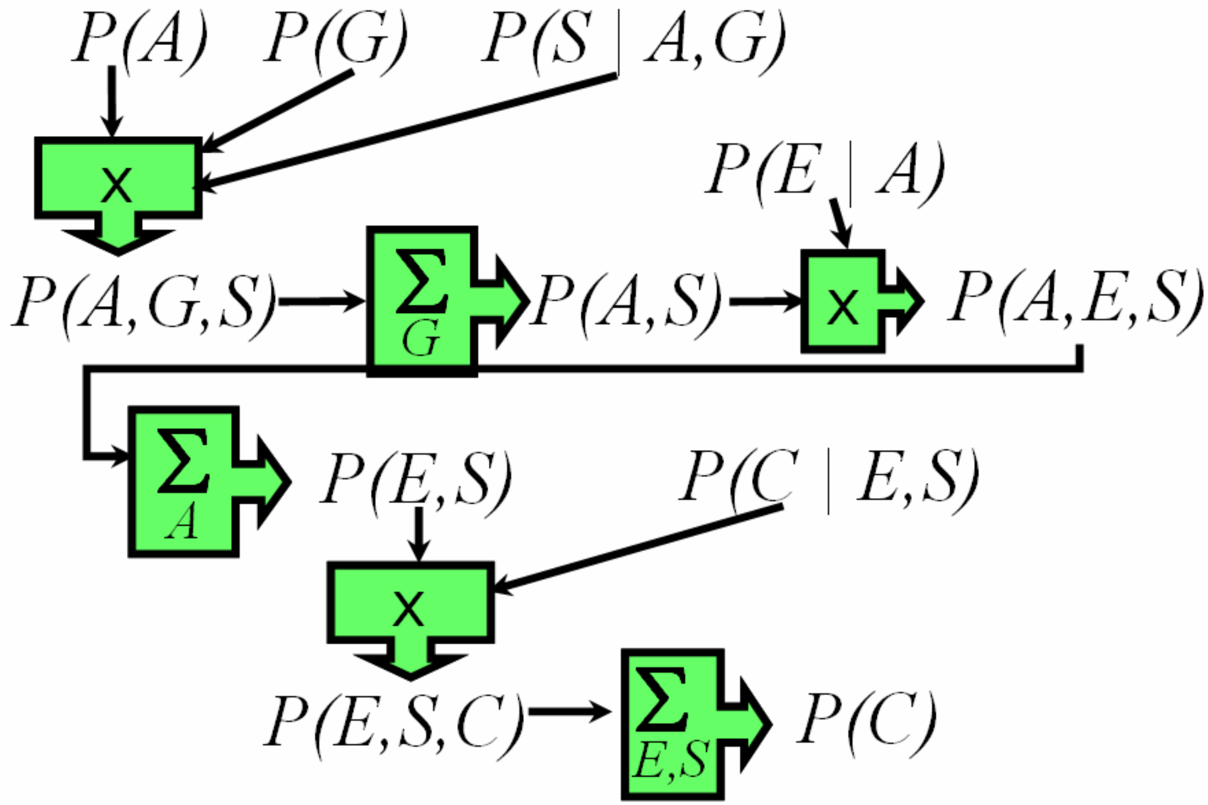
Variable Elimination with loops



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

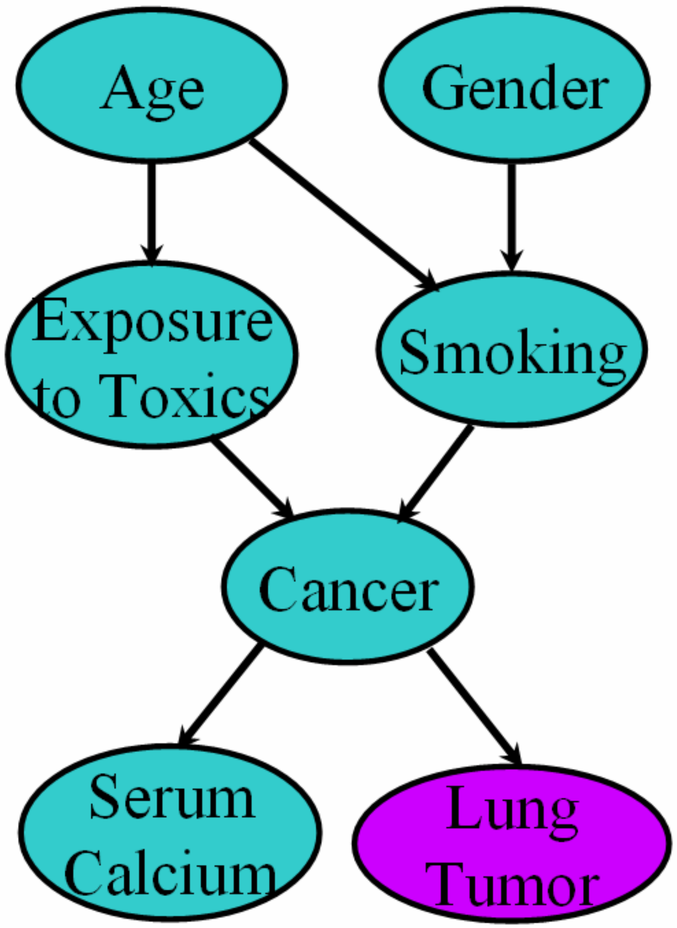
Variable Elimination with loops

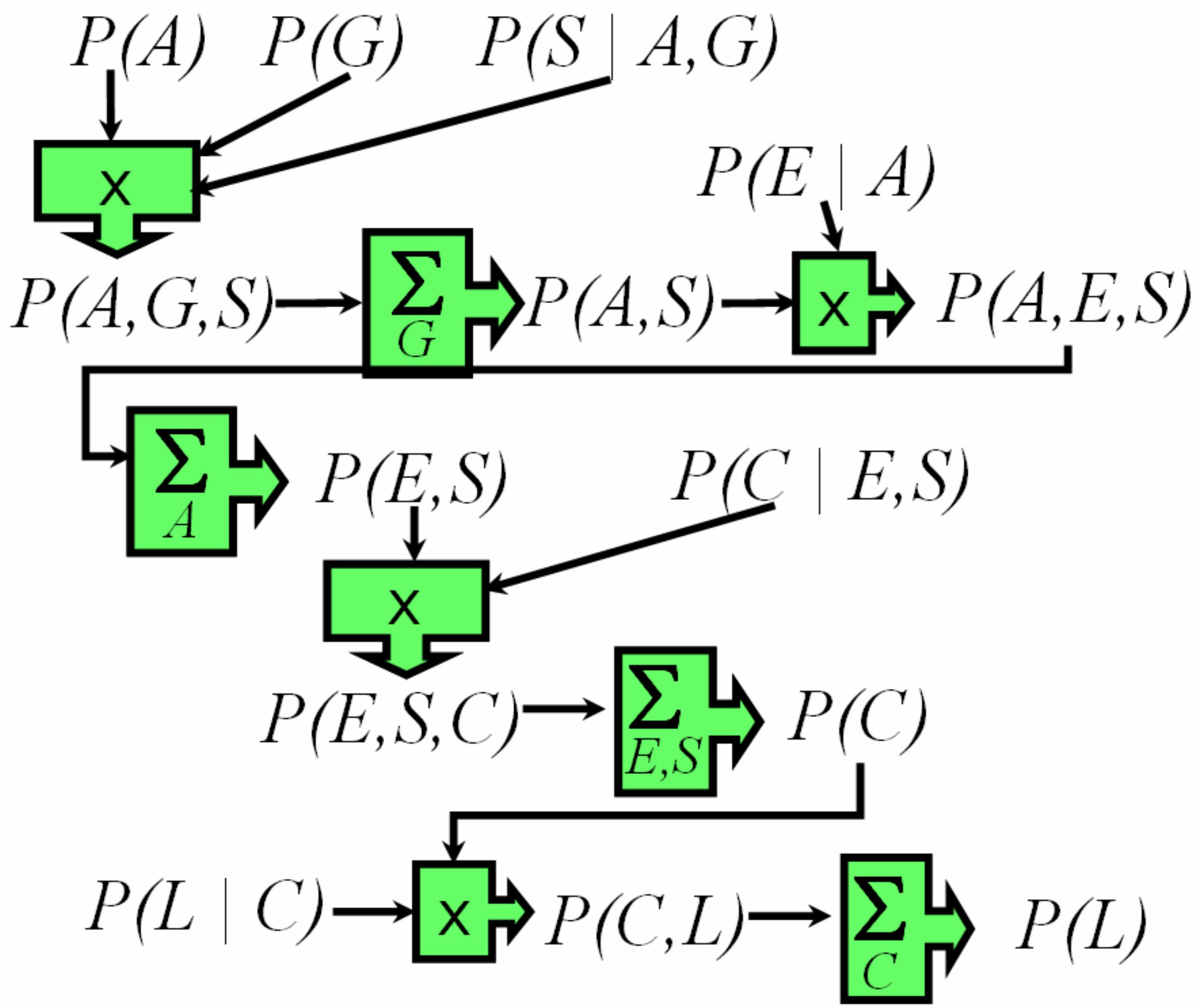




© Jack Breese (Microsoft) & Daphne Koller (Stanford)

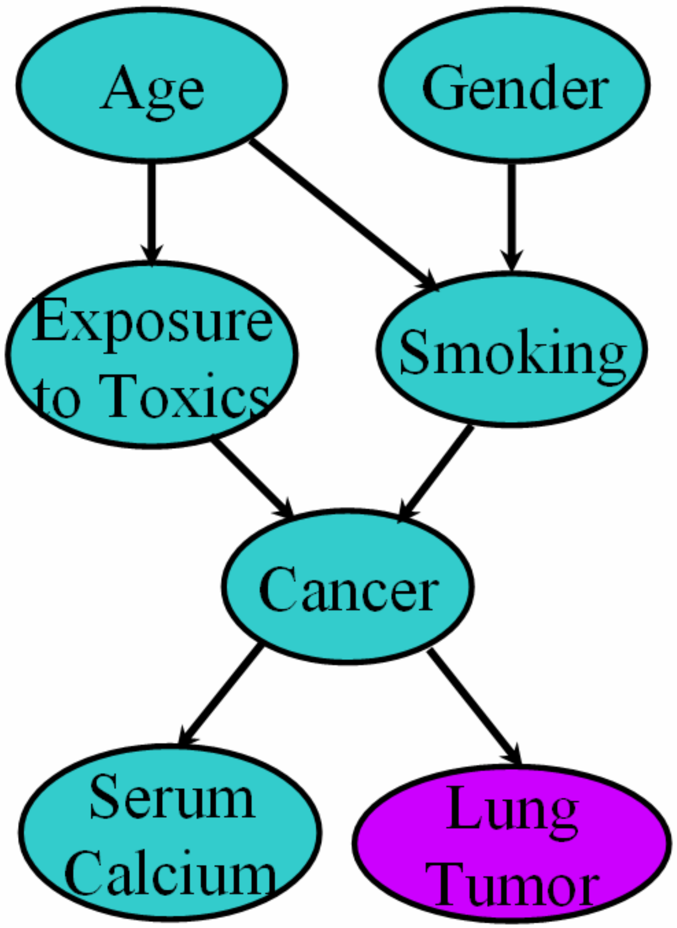
Variable Elimination with loops

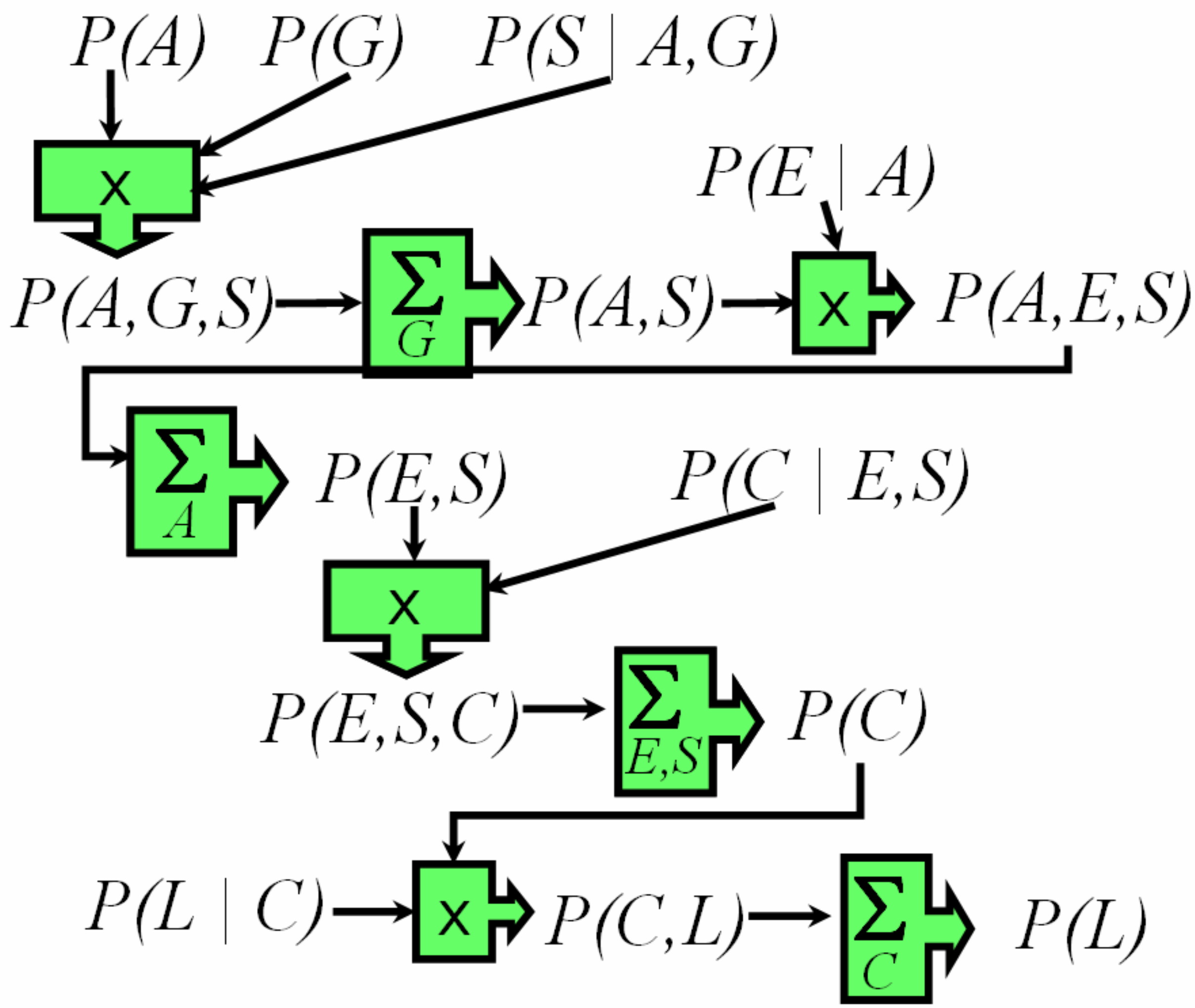




© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Variable Elimination with loops



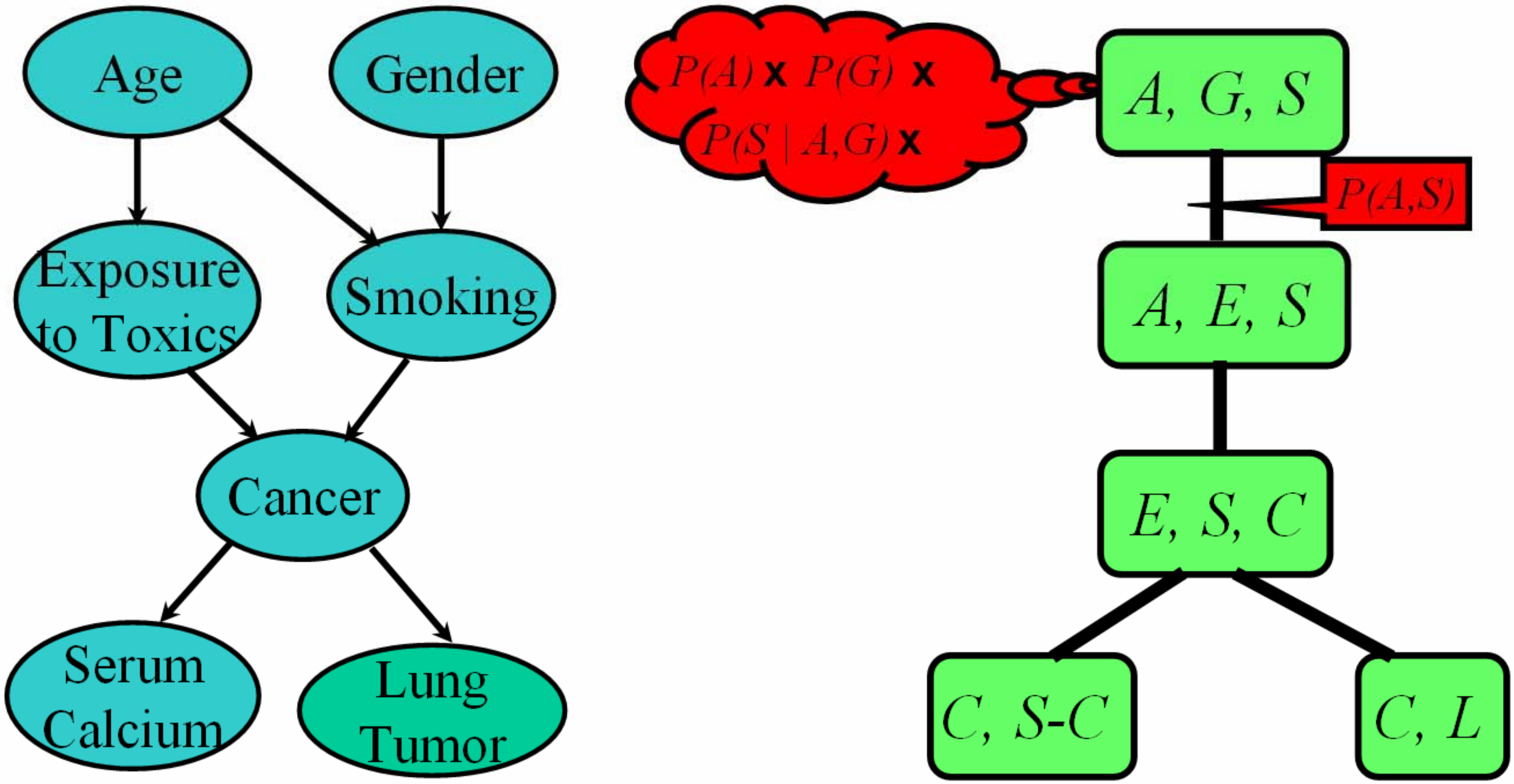


Complexity is exponential in the size of the factors

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Join trees*

A join tree is a partially precompiled factorization



* aka junction trees, Lauritzen-Spiegelhalter, Hugin alg., …

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Computational complexity

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Computational complexity

- **Theorem:** Inference in a multi-connected Bayesian network is NP-hard.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Computational complexity

- **Theorem:** Inference in a multi-connected Bayesian network is NP-hard.

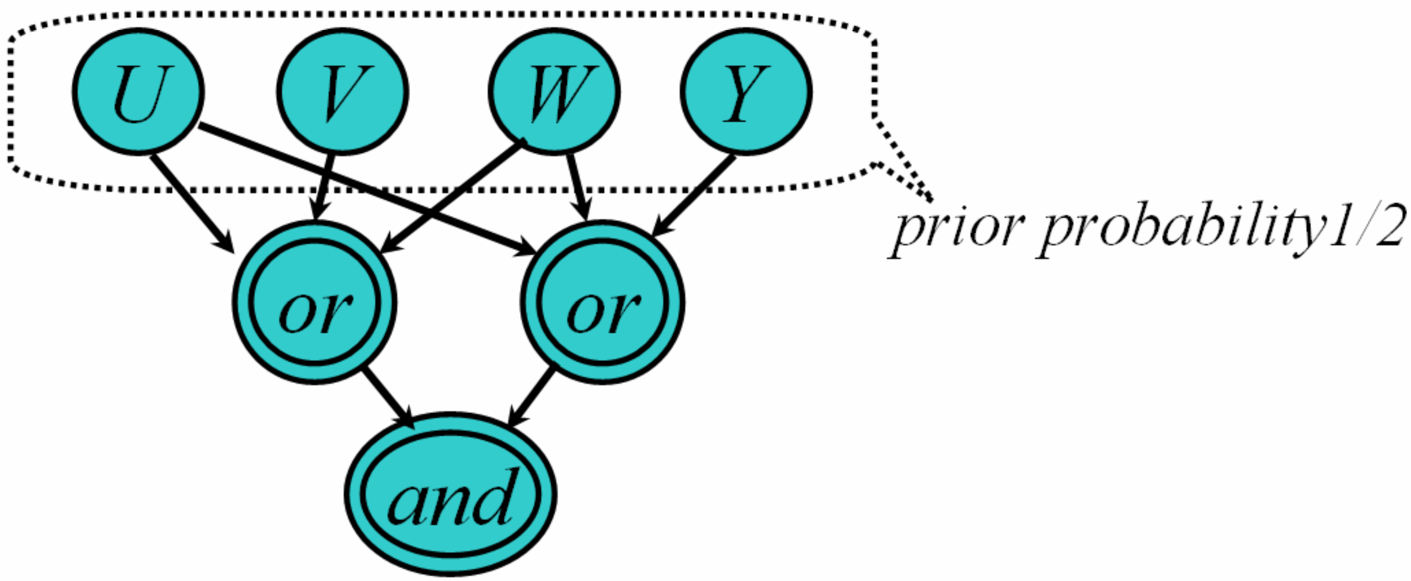
Boolean 3CNF formula $\phi = (u \vee \bar{v} \vee w) \wedge (\bar{u} \vee \bar{w} \vee y)$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Computational complexity

- **Theorem:** Inference in a multi-connected Bayesian network is NP-hard.

Boolean 3CNF formula $\phi = (u \vee \bar{v} \vee w) \wedge (\bar{u} \vee \bar{w} \vee y)$

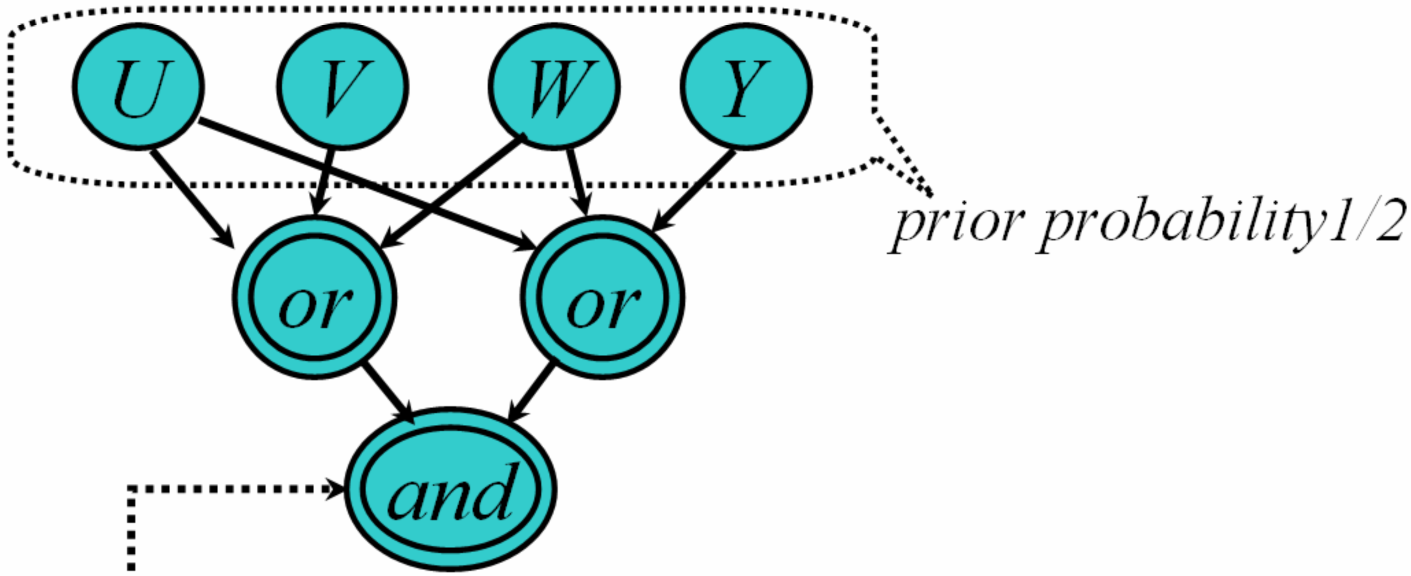


© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Computational complexity

- **Theorem:** Inference in a multi-connected Bayesian network is NP-hard.

Boolean 3CNF formula $\phi = (u \vee \bar{v} \vee w) \wedge (\bar{u} \vee \bar{w} \vee y)$



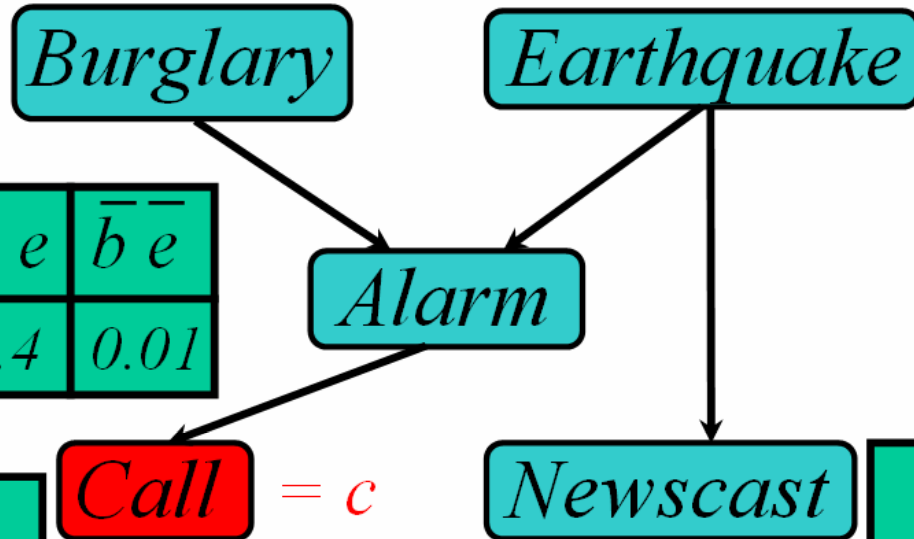
Probability ( ) = $1/2^n \cdot$ # satisfying assignments of ϕ

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03

$P(e)$	0.001



	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

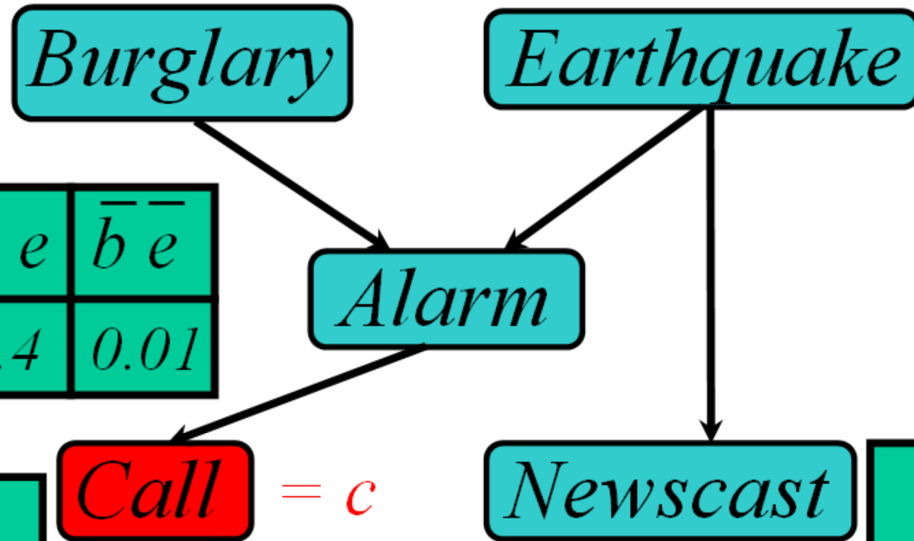
	e	$\bar{e}$
$P(n)$	0.3	0.001

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03

$P(e)$	0.001



	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

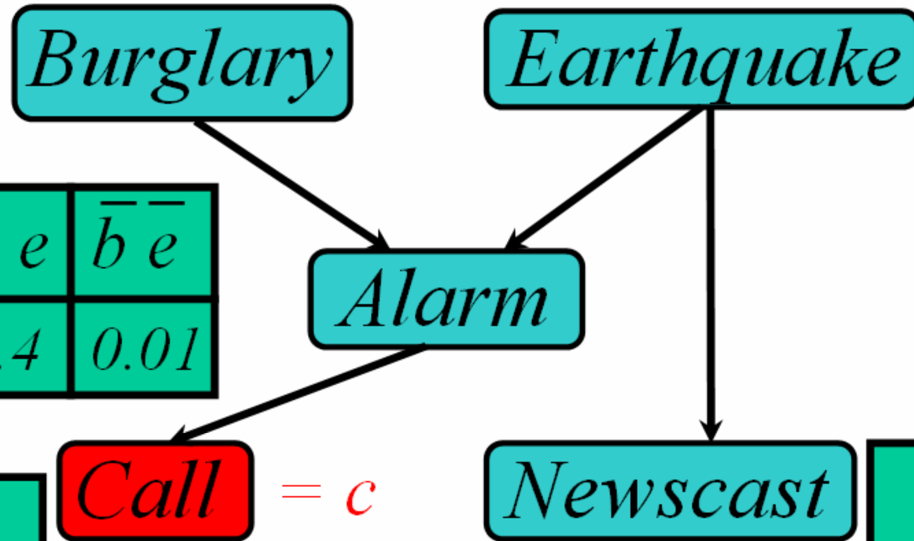
Samples:

B E A C N

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03



$P(e)$	0.001

	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

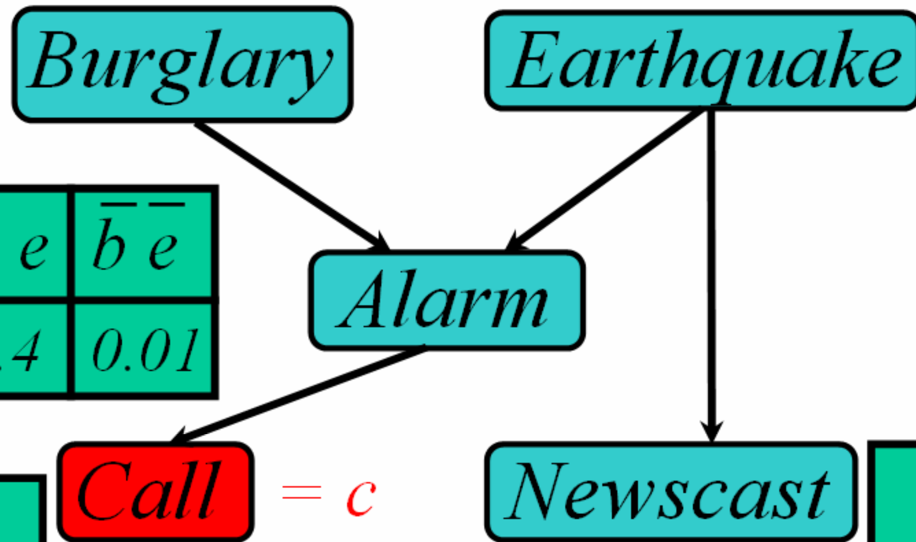
Samples:

B	***E***	***A***	***C***	***N***

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03



$P(e)$	0.001

	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

Samples:

B E A C N
$\bar{b}$

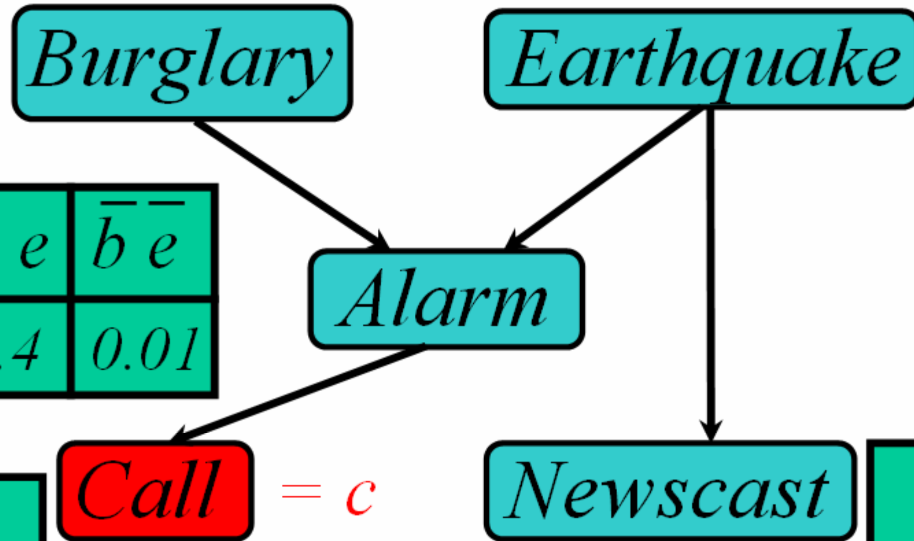
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03

$P(e)$	0.001

	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01



	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

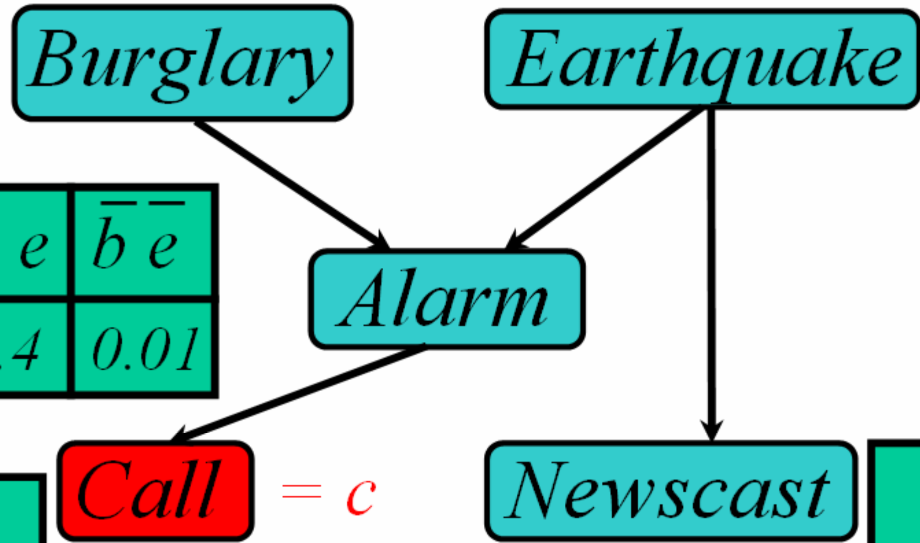
Samples:

B	E	A	C	N
$\bar{b}$				

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03



$P(e)$	0.001

	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

Samples:

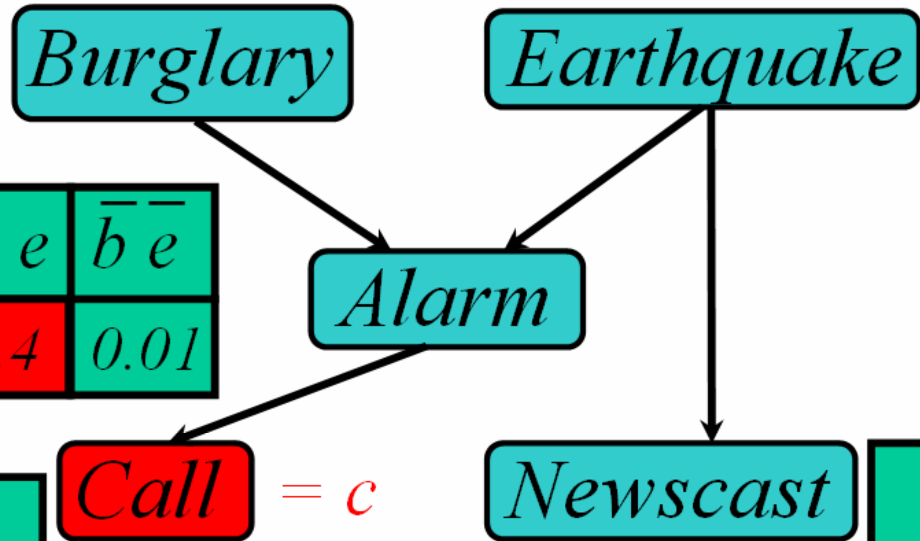
B	**E**	**A**	**C**	**N**
$\bar{b}$	e			

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03

$P(e)$	0.001



	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

Samples:

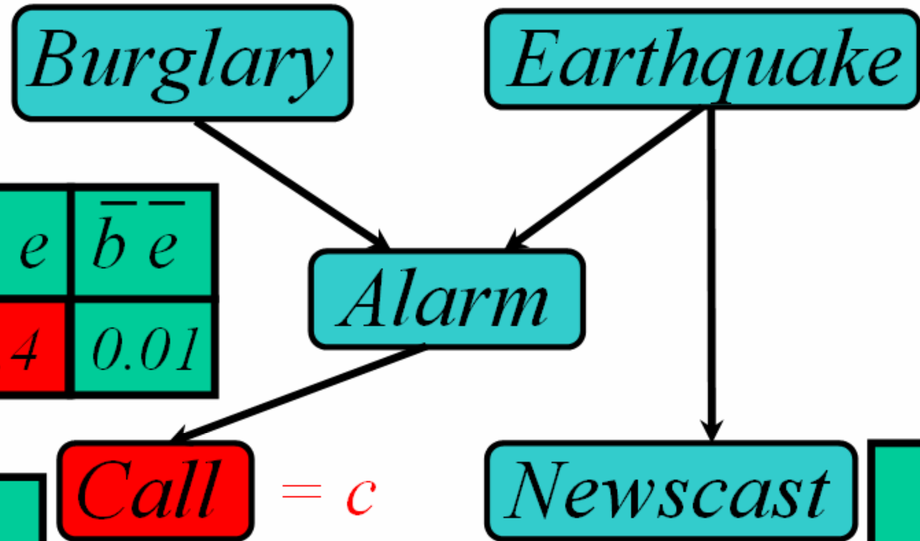
B	E	A	C	N
$\bar{b}$	e			

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03

$P(e)$	0.001



	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

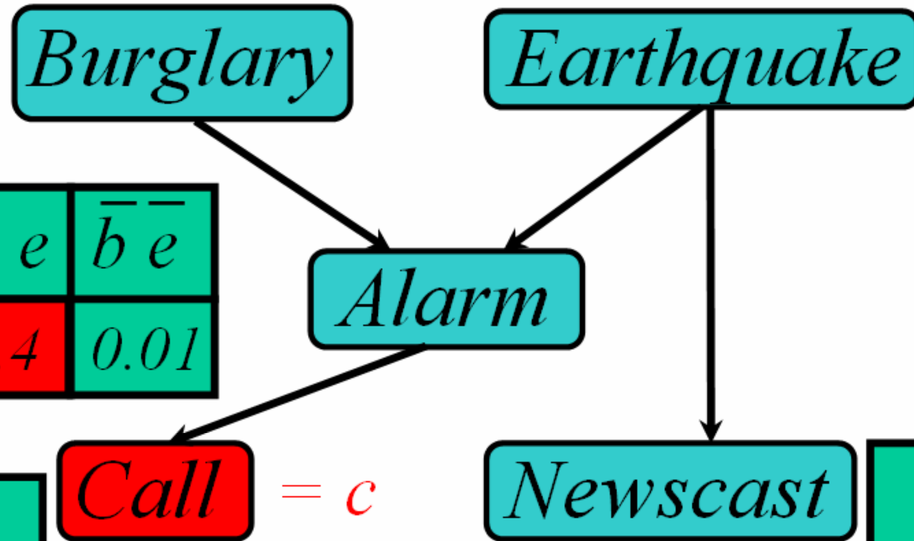
Samples:

B	***E***	***A***	***C***	***N***
$\bar{b}$	e	a		

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03



$P(e)$	0.001

	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

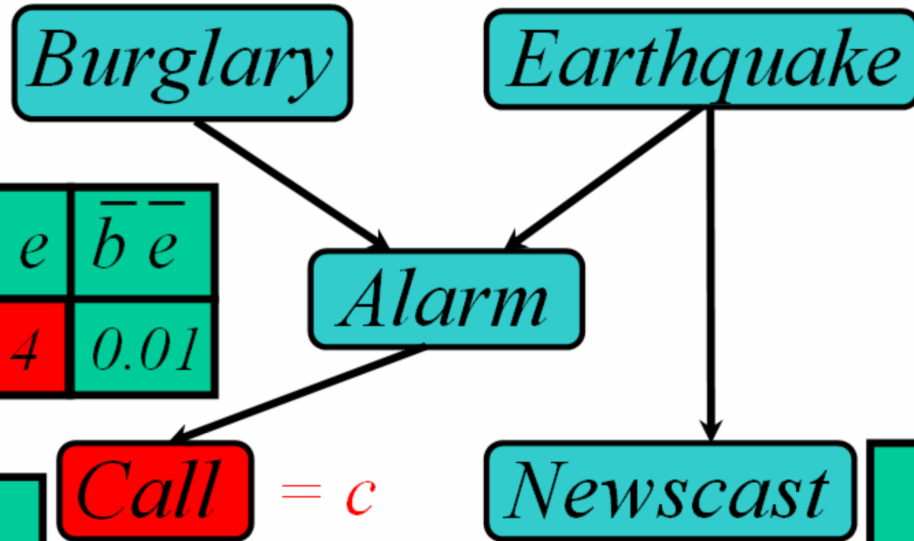
Samples:

B	E	A	C	N
$\bar{b}$	e	a		

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03



$P(e)$	0.001

	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

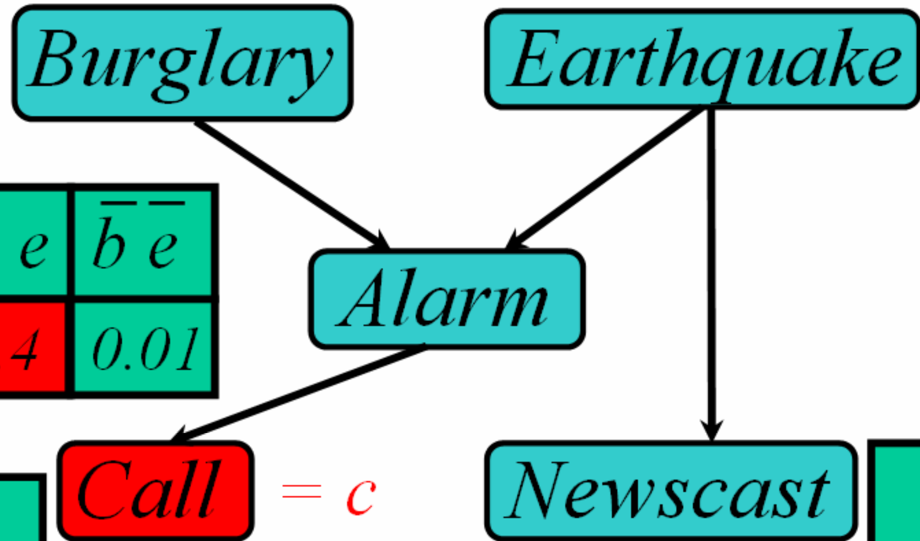
Samples:

B	***E***	***A***	***C***	***N***
$\bar{b}$	e	a	c	

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03



$P(e)$	0.001

	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

Samples:

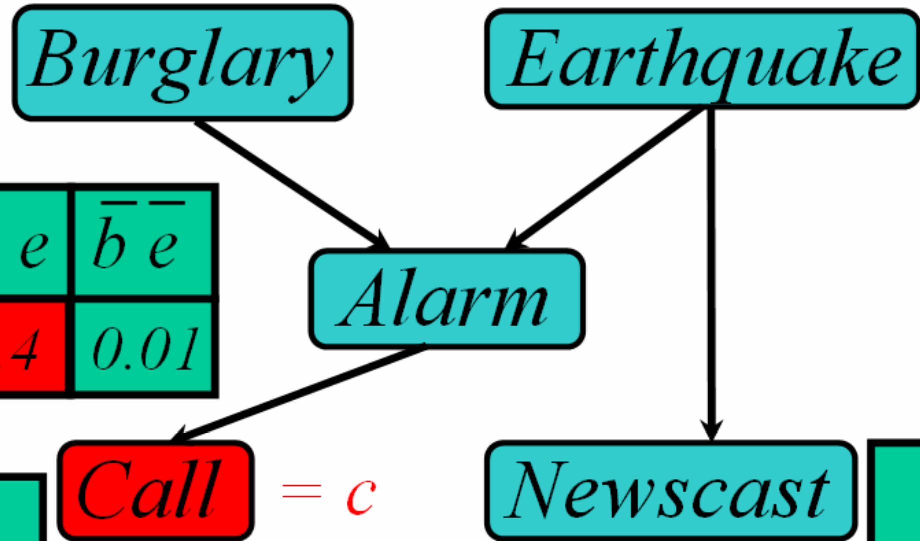
B	***E***	***A***	***C***	***N***
$\bar{b}$	e	a	c	

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03

	$b\,e$	$b\,\bar{e}$	$\bar{b}\,e$	$\bar{b}\,\bar{e}$
$P(a)$	0.98	0.7	0.4	0.01



$P(e)$	0.001

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

Samples:

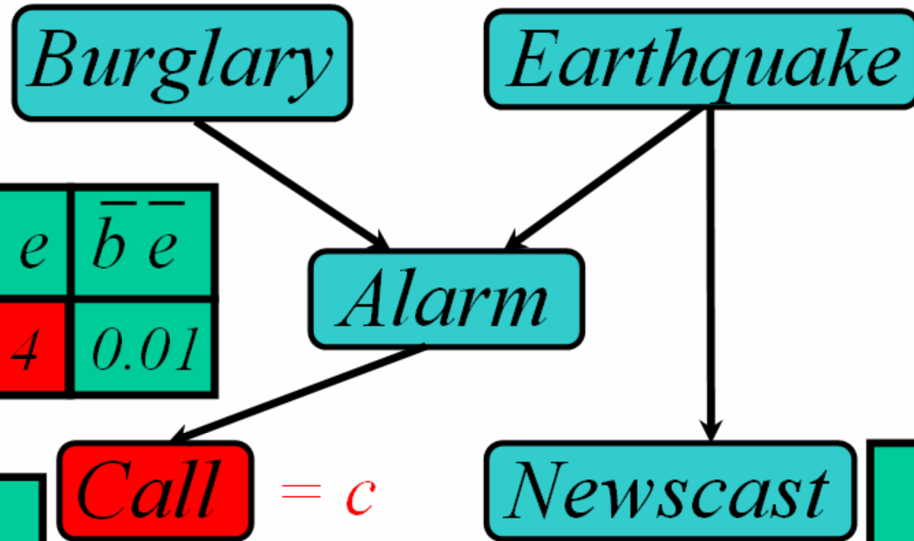
B	*E*	*A*	*C*	*N*
$\bar{b}$	e	a	c	$\bar{n}$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03

	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01



$P(e)$	0.001

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

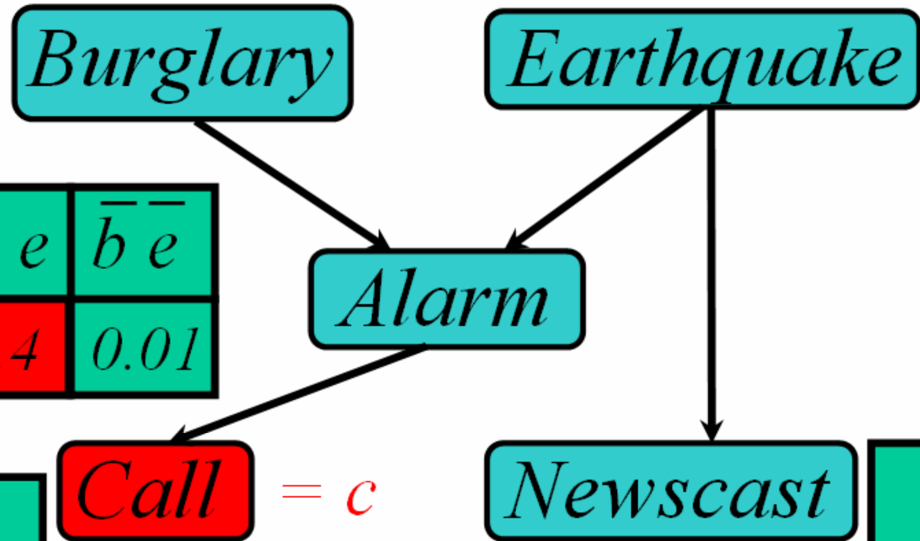
Samples:

B	***E***	***A***	***C***	***N***
$\bar{b}$	e	a	c	$\bar{n}$
b	$\bar{e}$	a	$\bar{c}$	n

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03



$P(e)$	0.001

	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

	e	$\bar{e}$
$P(n)$	0.3	0.001

Samples:

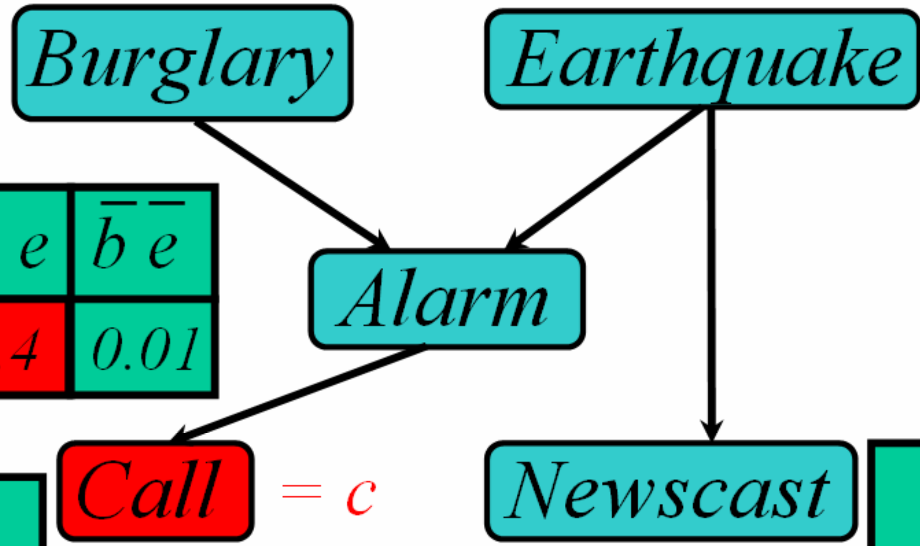
B	E	A	C	N
$\bar{b}$	e	a	c	$\bar{n}$
$\cancel{b}$	$\cancel{\bar{e}}$	$\cancel{a}$	$\cancel{\bar{c}}$	$\cancel{n}$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation

$P(b)$	0.03

$P(e)$	0.001



	$b\ e$	$b\ \bar{e}$	$\bar{b}\ e$	$\bar{b}\ \bar{e}$
$P(a)$	0.98	0.7	0.4	0.01

	a	$\bar{a}$
$P(c)$	0.8	0.05

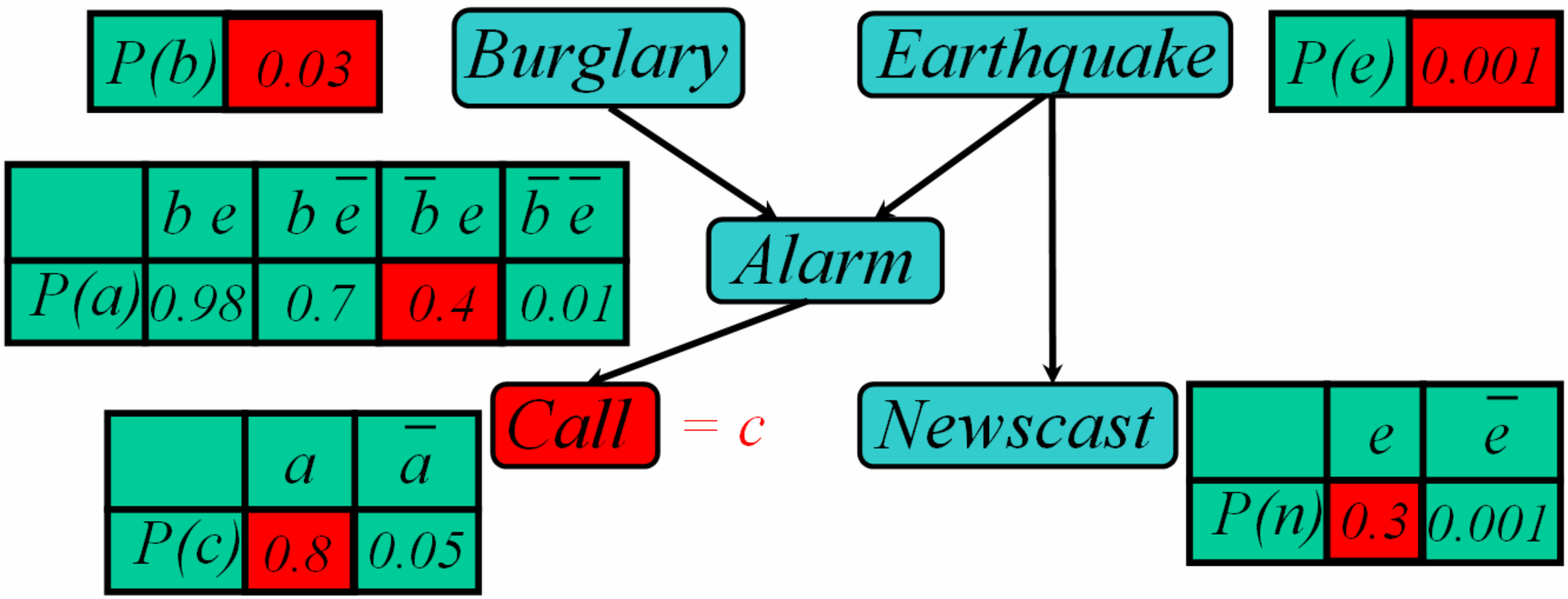
	e	$\bar{e}$
$P(n)$	0.3	0.001

Samples:

B	**E**	**A**	**C**	**N**
$\bar{b}$	e	a	c	$\bar{n}$
~~b~~	~~$\bar{e}$~~	~~a~~	~~$\bar{c}$~~	~~n~~
		⋮		

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Stochastic simulation



Samples:

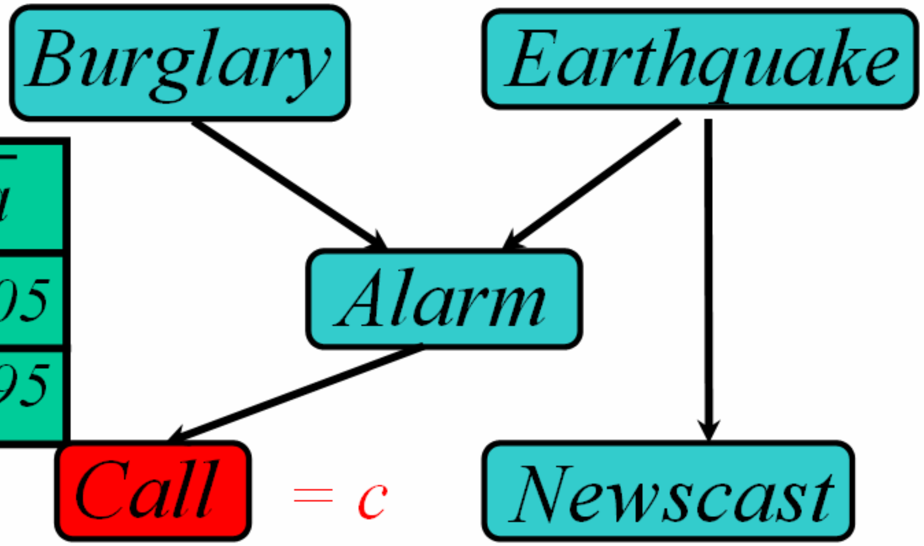
B	E	A	C	N
$\bar{b}$	e	a	c	$\bar{n}$
~~b~~	~~$\bar{e}$~~	~~a~~	~~$\bar{c}$~~	~~n~~
		⋮		

$$P(b|c) \sim \frac{\#\text{ of live samples with } B=b}{\text{total } \#\text{ of live samples}}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Likelihood weighting

	a	$\bar{a}$
$P(c)$	0.8	0.05
$P(\bar{c})$	0.2	0.95



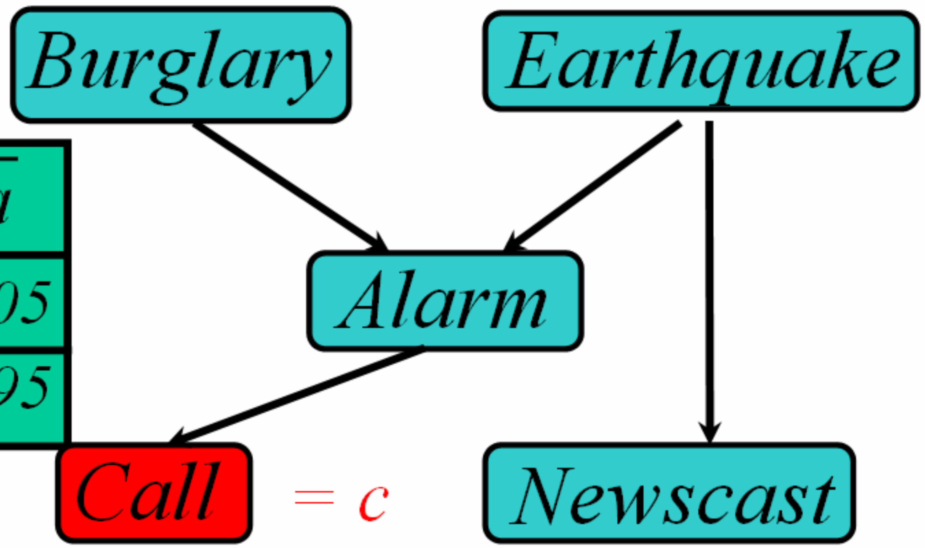
Samples:

B	E	A	C	N

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Likelihood weighting

	a	$\bar{a}$
$P(c)$	0.8	0.05
$P(\bar{c})$	0.2	0.95

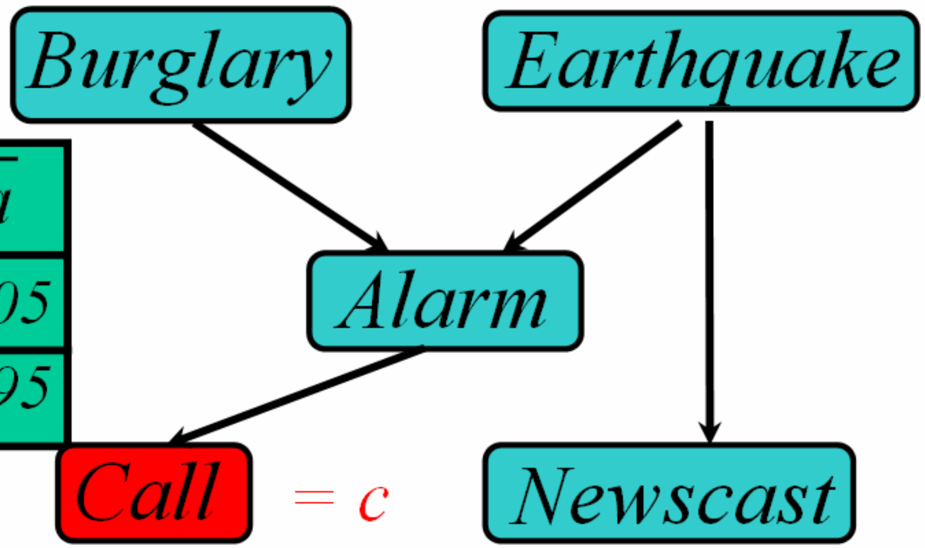


Samples:

B	*E*	*A*	*C*	*N*
$\bar{b}$				

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Likelihood weighting



	a	$\bar{a}$
$P(c)$	0.8	0.05
$P(\bar{c})$	0.2	0.95

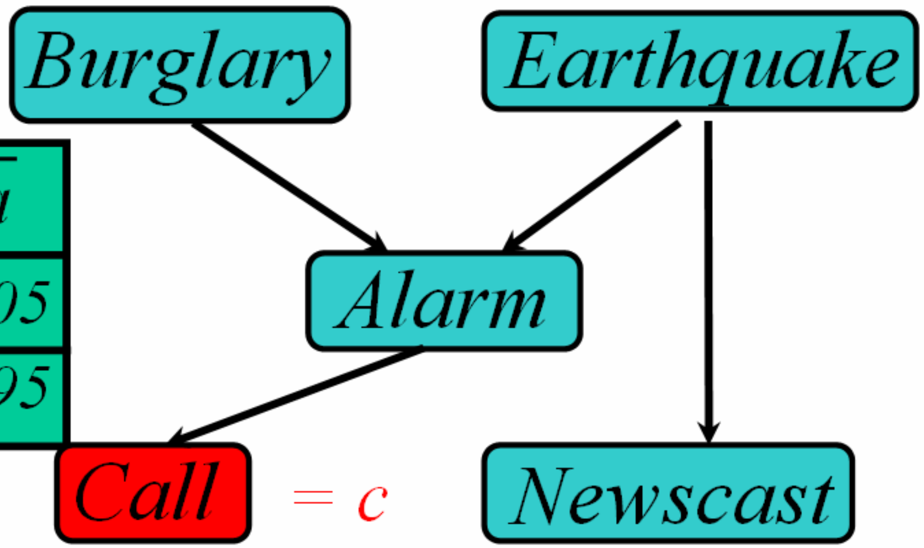
Samples:

B	**E**	**A**	**C**	**N**
$\bar{b}$	e			

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Likelihood weighting

	a	$\bar{a}$
$P(c)$	0.8	0.05
$P(\bar{c})$	0.2	0.95



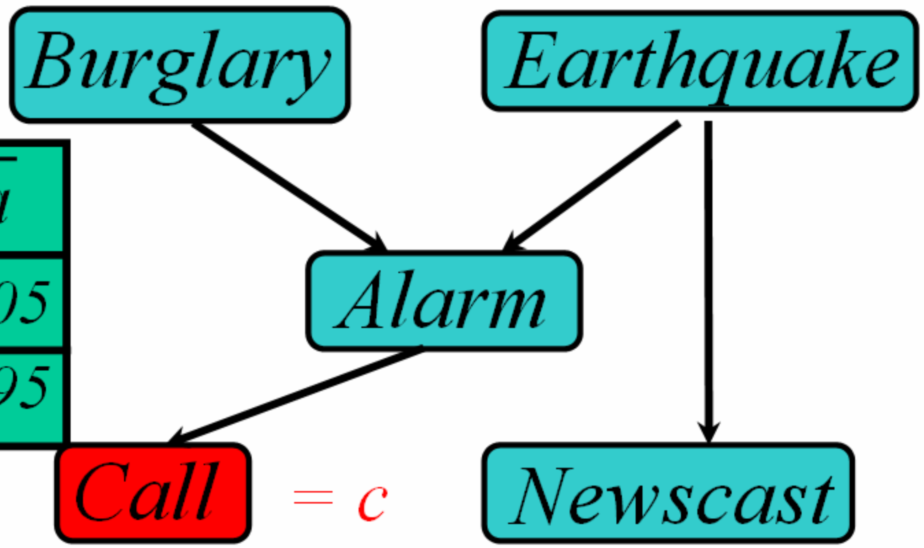
Samples:

B	***E***	***A***	***C***	***N***
$\bar{b}$	e	a		

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Likelihood weighting

	a	$\bar{a}$
$P(c)$	0.8	0.05
$P(\bar{c})$	0.2	0.95

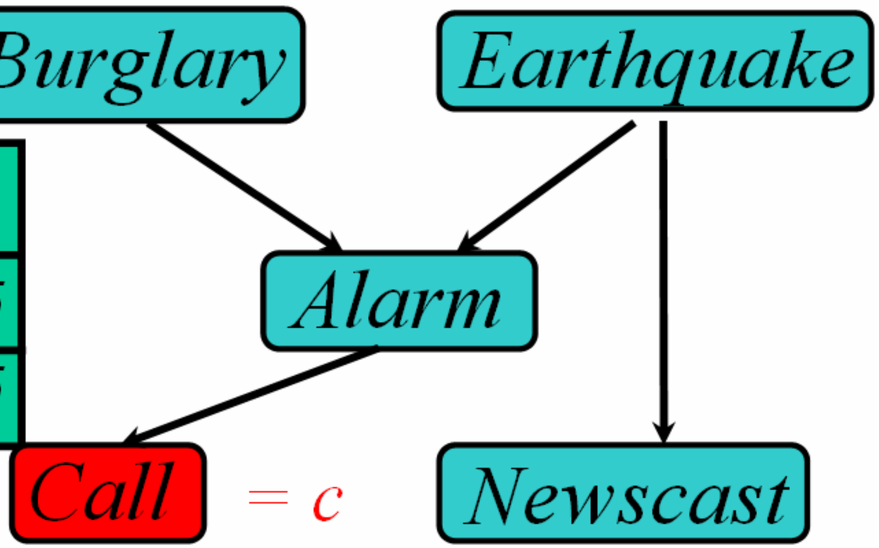


Samples:

B	***E***	***A***	***C***	***N***
$\bar{b}$	e	a	c	

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Likelihood weighting



	a	$\bar{a}$
$P(c)$	0.8	0.05
$P(\bar{c})$	0.2	0.95

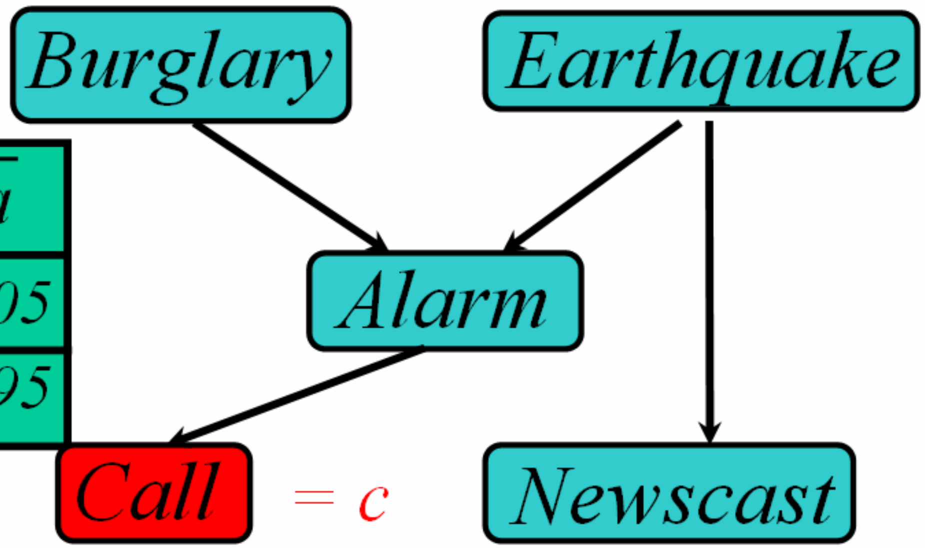
Samples:

B	**E**	**A**	**C**	**N**
$\bar{b}$	e	a	c	$\bar{n}$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Likelihood weighting

	a	$\bar{a}$
$P(c)$	0.8	0.05
$P(\bar{c})$	0.2	0.95

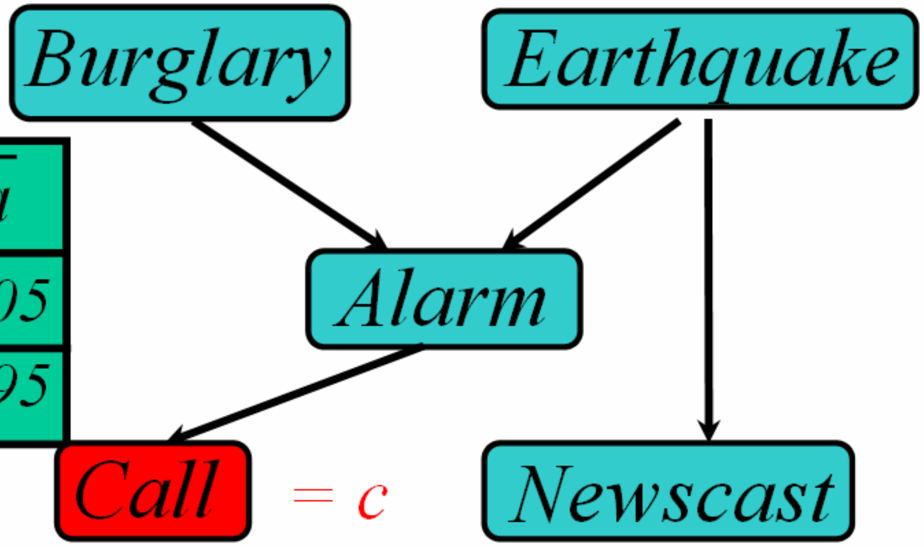


Samples:

B E A C N	weight
$\bar{b}$ e a c $\bar{n}$	0.8

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Likelihood weighting



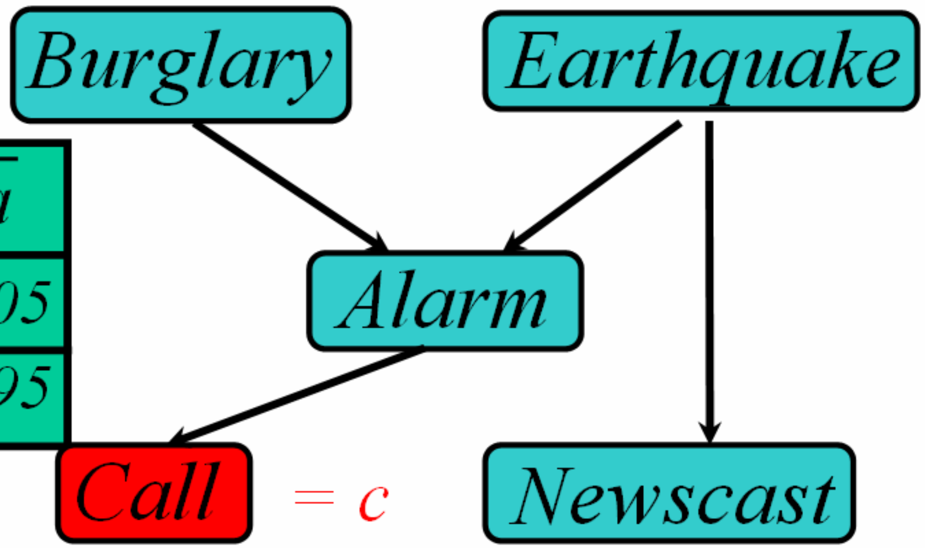
	a	$\bar{a}$
$P(c)$	0.8	0.05
$P(\bar{c})$	0.2	0.95

Samples:

B E A C N	*weight*
$\bar{b}$ e a c $\bar{n}$	0.8
b $\bar{e}$ $\bar{a}$ c n	0.05

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Likelihood weighting



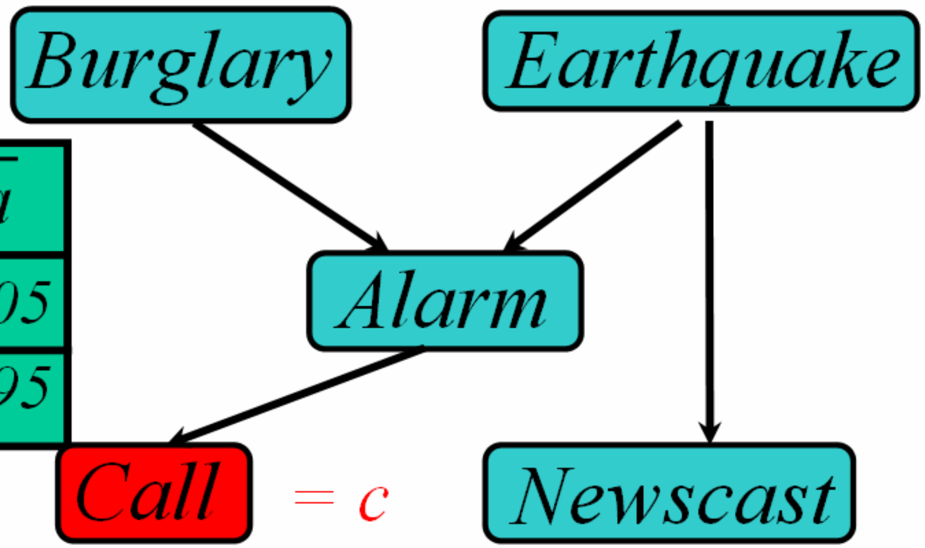
	a	$\bar{a}$
$P(c)$	0.8	0.05
$P(\bar{c})$	0.2	0.95

Samples:

B E A C N	*weight*
$\bar{b}$ e a c $\bar{n}$	0.8
b $\bar{e}$ $\bar{a}$ c n	0.05
⋮	

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Likelihood weighting



	a	$\bar{a}$
$P(c)$	0.8	0.05
$P(\bar{c})$	0.2	0.95

Samples:

B E A C N	*weight*
$\bar{b}$ e a c $\bar{n}$	0.8
b $\bar{e}$ $\bar{a}$ c n	0.05
⋮	

$$P(b|c) = \frac{\text{weight of samples with } B=b}{\text{total weight of samples}}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Markov Chain Monte Carlo

MCMC with Gibbs Sampling

Fix the values of observed variables

Set the values of all non-observed variables randomly

Perform a random walk through the space of complete variable assignments. On each move:

1. Pick a variable X
2. Calculate Pr(X=true | all other variables)
3. Set X to true with that probability

Repeat many times. Frequency with which any variable X is true is it's posterior probability.

Converges to true posterior when frequencies stop changing significantly

- stable distribution, mixing

Markov Blanket Sampling

How to calculate Pr(X=true | all other variables) **?**

Recall: a variable is independent of all others given it's Markov Blanket

- parents
- children
- other parents of children

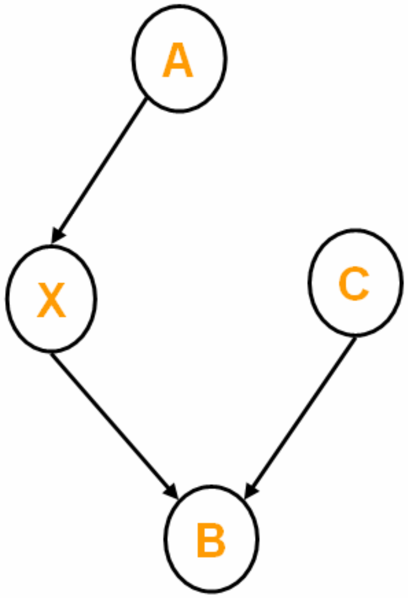
So problem becomes calculating Pr(X=true | MB(X))

- We solve this sub-problem exactly
- Fortunately, it is easy to solve

$$P(X) = \alpha P(X \mid Parents(X)) \prod_{Y \in Children(X)} P(Y \mid Parents(Y))$$

Example

$$P(X) = \alpha P(X \mid Parents(X)) \prod_{Y \in Children(X)} P(Y \mid Parents(Y))$$



$$P(X \mid A,B,C) = \frac{P(X,A,B,C)}{P(A,B,C)}$$

$$= \frac{P(A)P(X \mid A)P(C)P(B \mid X,C)}{P(A,B,C)}$$

$$= \left[\frac{P(A)P(C)}{P(A,B,C)}\right] P(X \mid A)P(B \mid X,C)$$

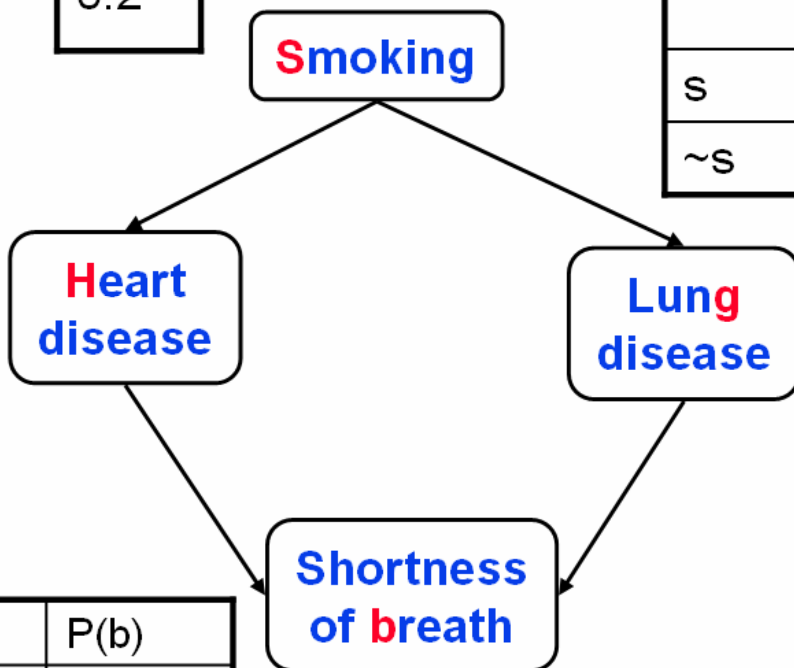
$$= \alpha P(X \mid A)P(B \mid X,C)$$

Example

P(s)
0.2

	P(s)
s	0.6
~s	0.1

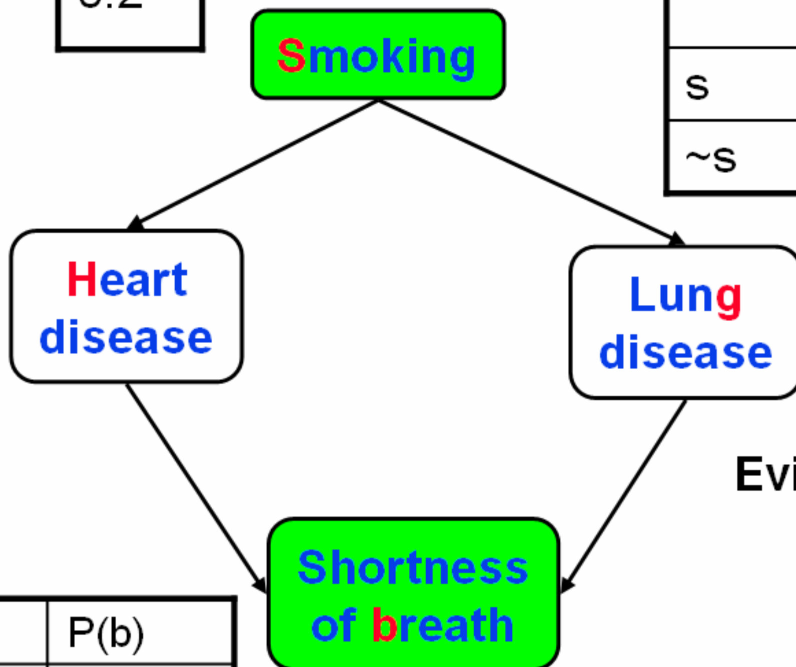
	P(g)
s	0.8
~s	0.1



H	G	P(b)
h	g	0.9
h	~g	0.8
~h	g	0.7
~h	~g	0.1

Example

P(s)
0.2



	P(g)
s	0.8
~s	0.1

	P(s)
s	0.6
~s	0.1

Evidence: s, b

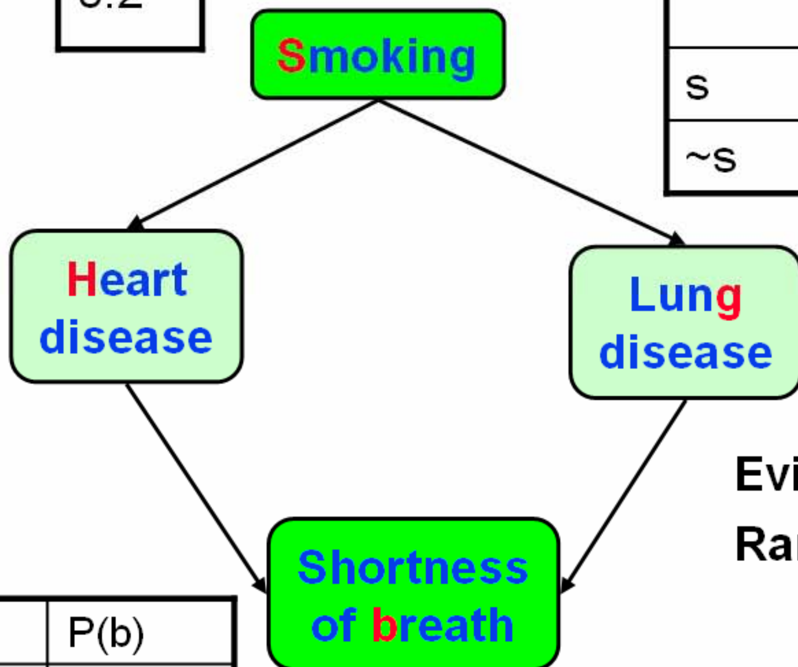
H	G	P(b)
h	g	0.9
h	~g	0.8
~h	g	0.7
~h	~g	0.1

Example

P(s)
0.2

	P(s)
s	0.6
~s	0.1

	P(g)
s	0.8
~s	0.1



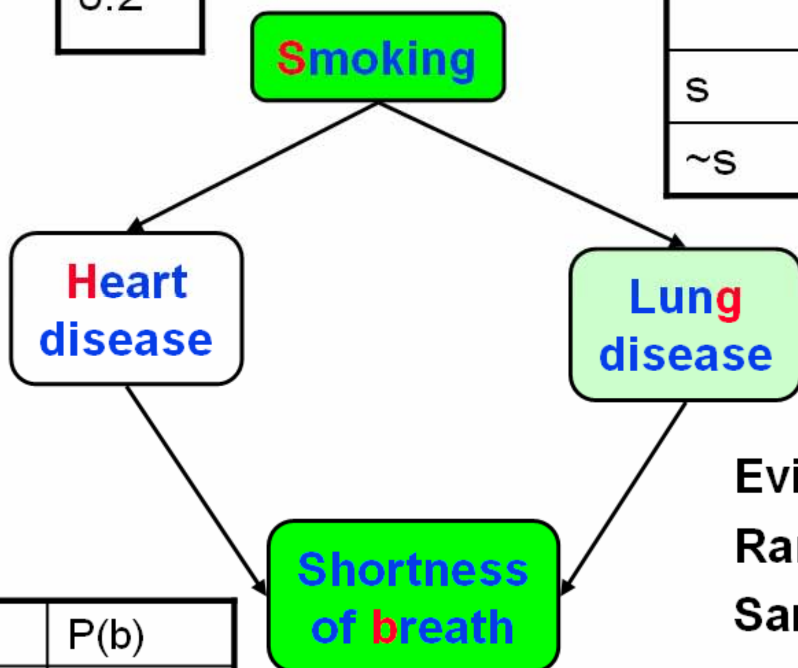
Evidence: s, b

Randomly set: h, b

H	G	P(b)
h	g	0.9
h	~g	0.8
~h	g	0.7
~h	~g	0.1

Example

P(s)
0.2



	P(g)
s	0.8
~s	0.1

	P(s)
s	0.6
~s	0.1

H	G	P(b)
h	g	0.9
h	~g	0.8
~h	g	0.7
~h	~g	0.1

Evidence: s, b

Randomly set: h, g

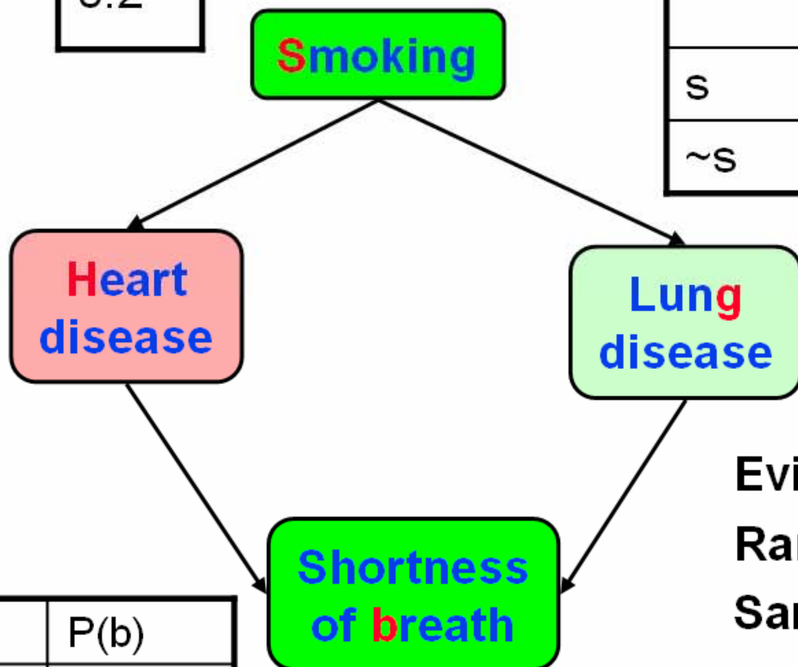
Sample H using P(H|s,g,b)

Example

P(s)
0.2

	P(g)
s	0.8
~s	0.1

	P(s)
s	0.6
~s	0.1



H	G	P(b)
h	g	0.9
h	~g	0.8
~h	g	0.7
~h	~g	0.1

Evidence: s, b

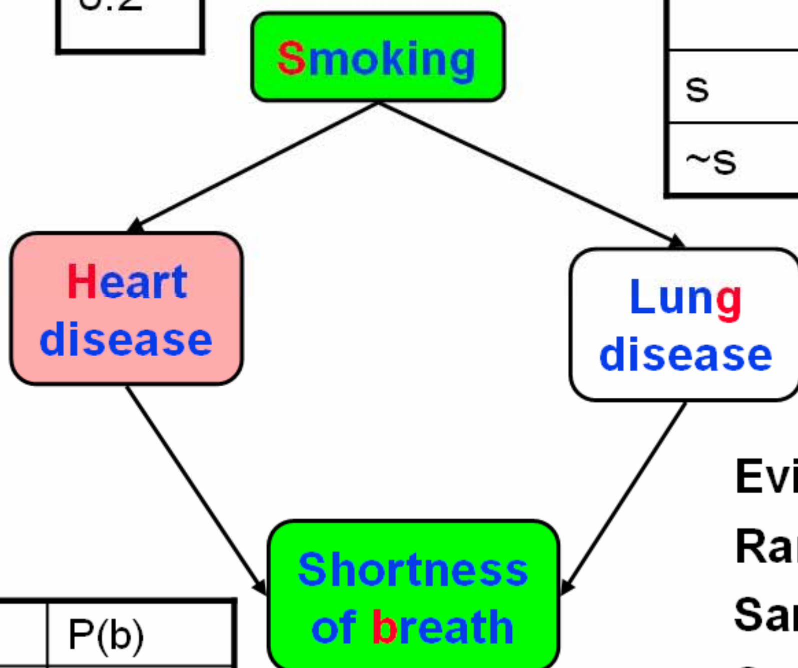
Randomly set: ~h, g

Sample H using P(H|s,g,b)

⇒ Suppose result is ~h

Example

P(s)
0.2



	P(s)
s	0.6
~s	0.1

	P(g)
s	0.8
~s	0.1

H	G	P(b)
h	g	0.9
h	~g	0.8
~h	g	0.7
~h	~g	0.1

Evidence: s, b

Randomly set: ~h, g

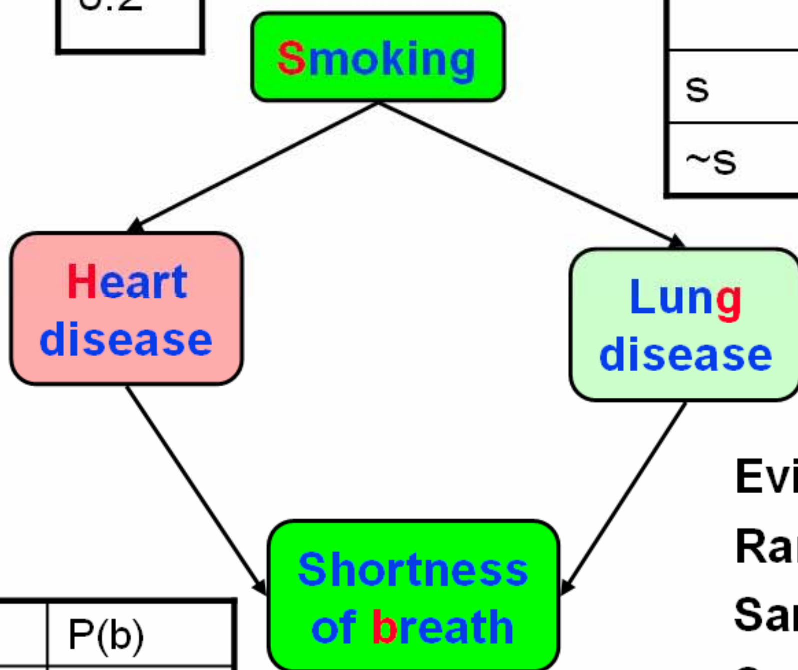
Sample H using P(H|s,g,b)

Suppose result is ~h

Sample G using P(G|s,~h,b)

Example

P(s)
0.2



	P(g)
s	0.8
~s	0.1

	P(s)
s	0.6
~s	0.1

H	G	P(b)
h	g	0.9
h	~g	0.8
~h	g	0.7
~h	~g	0.1

Evidence: s, b

Randomly set: ~h, g

Sample H using P(H|s,g,b)

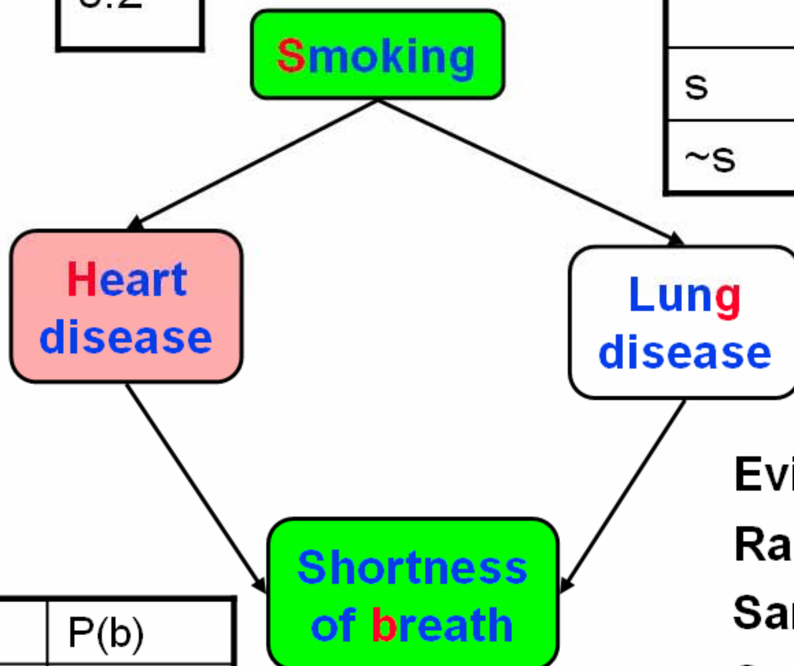
Suppose result is ~h

Sample G using P(G|s,~h,b)

⇒Suppose result is g

Example

P(s)
0.2



	P(s)
s	0.6
~s	0.1

	P(g)
s	0.8
~s	0.1

H	G	P(b)
h	g	0.9
h	~g	0.8
~h	g	0.7
~h	~g	0.1

Evidence: s, b

Randomly set: ~h, g

Sample H using P(H|s,g,b)

Suppose result is ~h

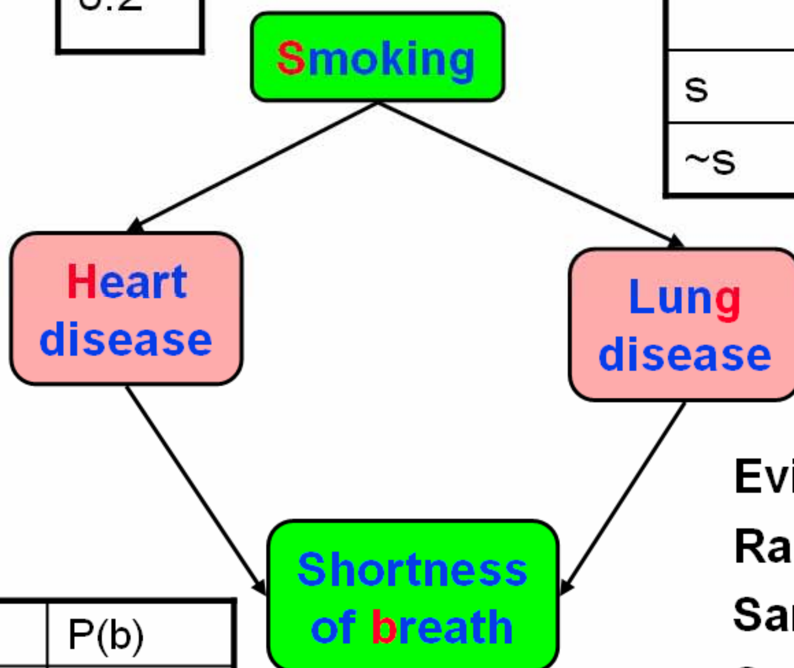
Sample G using P(G|s,~h,b)

⇒Suppose result is g

Sample G using P(G|s,~h,b)

Example

P(s)
0.2



	P(g)
s	0.8
~s	0.1

	P(s)
s	0.6
~s	0.1

H	G	P(b)
h	g	0.9
h	~g	0.8
~h	g	0.7
~h	~g	0.1

Evidence: s, b

Randomly set: ~h, g

Sample H using P(H|s,g,b)

Suppose result is ~h

Sample G using P(G|s,~h,b)

⇒Suppose result is g

Sample G using P(G|s,~h,b)

⇒Suppose result is ~g

Gibbs MCMC Summary

Advantages:

- No samples are discarded
- No problem with samples of low weight
- Can be implemented very efficiently
  - 10K samples @ second

Disadvantages:

- Can get stuck if relationship between two variables is *deterministic*
- Many variations have been devised to make MCMC more robust

Gibbs MCMC Summary

$$P(X|E) = \frac{\textit{number of samples with } X{=}x}{\textit{total number of samples}}$$

Advantages:

- No samples are discarded
- No problem with samples of low weight
- Can be implemented very efficiently
  - 10K samples @ second

Disadvantages:

- Can get stuck if relationship between two variables is *deterministic*
- Many variations have been devised to make MCMC more robust

Other approaches

- Search based techniques
  - search for high-probability instantiations
  - use instantiations to approximate probabilities
- Structural approximation
  - simplify network
    - eliminate edges, nodes
    - abstract node values
    - simplify CPTs
  - do inference in simplified network

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Course Contents

- Concepts in Probability
- Bayesian Networks
- Inference
- Decision making

» Learning networks from data

- Reasoning over time
- Applications

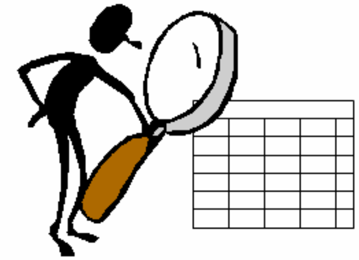
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Learning networks from data

- The learning task
- Parameter learning
  - Fully observable
  - Partially observable
- Structure learning
- Hidden variables

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The learning task

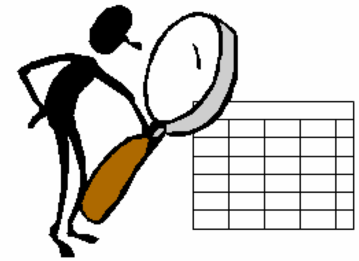


B	E	A	C	N
$\bar{b}$	e	a	c	$\bar{n}$
b	$\bar{e}$	$\bar{a}$	$\bar{c}$	n
		$\vdots$		

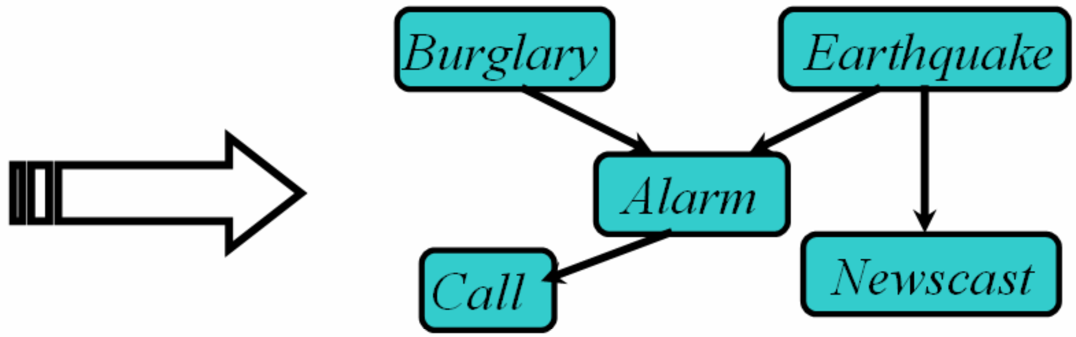
Input: training data

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The learning task



B	*E*	*A*	*C*	*N*
$\bar{b}$	e	a	c	$\bar{n}$
b	$\bar{e}$	$\bar{a}$	$\bar{c}$	n
⋮				

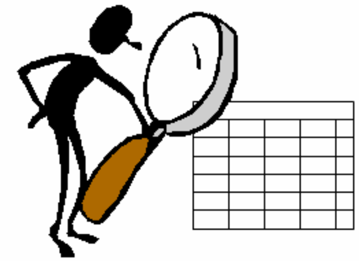


Input: training data

Output: BN modeling data

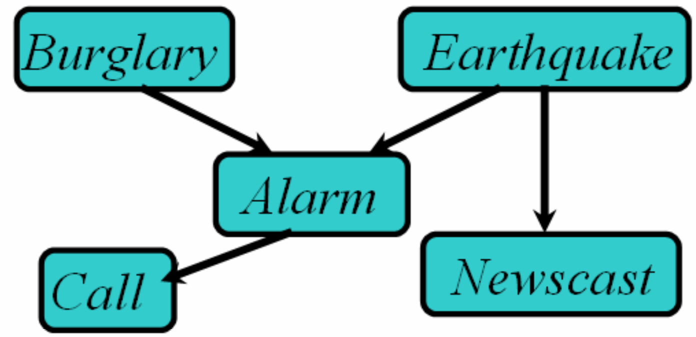
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The learning task



B	E	A	C	N
$\bar{b}$	e	a	c	$\bar{n}$
b	$\bar{e}$	$\bar{a}$	$\bar{c}$	n
⋮				

Input: training data

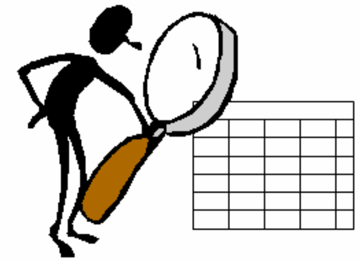


Output: BN modeling data

- Input: fully or partially observable data cases?

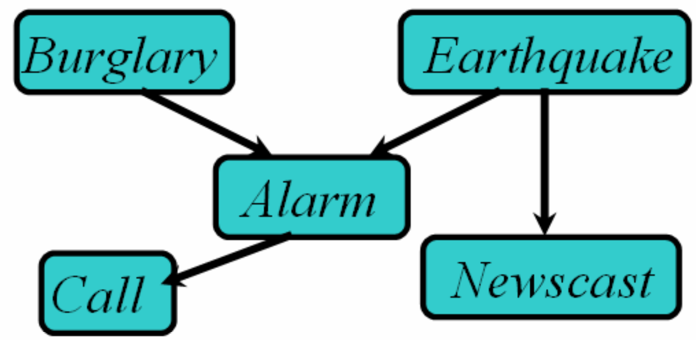
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

The learning task



B	E	A	C	N
$\bar{b}$	e	a	c	$\bar{n}$
b	$\bar{e}$	$\bar{a}$	$\bar{c}$	n
⋮				

Input: training data



Output: BN modeling data

- Input: fully or partially observable data cases?
- Output: parameters or also structure?

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Parameter learning: one variable

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Parameter learning: one variable

- Unfamiliar coin: 

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Parameter learning: one variable

- Unfamiliar coin:
  - Let θ = bias of coin (long-run fraction of heads)

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Parameter learning: one variable

- Unfamiliar coin:
  - Let θ = bias of coin (long-run fraction of heads)
- If θ known (given), then

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Parameter learning: one variable

- Unfamiliar coin:
  - Let θ = bias of coin (long-run fraction of heads)
- If θ known (given), then
  - $P(X = heads \mid \theta) =$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Parameter learning: one variable

- Unfamiliar coin:

  - Let θ = bias of coin (long-run fraction of heads)
- If θ known (given), then
  - $P(X = heads \mid \theta) = \theta$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Parameter learning: one variable

- Unfamiliar coin:
  - Let θ = bias of coin (long-run fraction of heads)
- If θ known (given), then
  - $P(X = heads \mid \theta) = \theta$
- Different coin tosses independent given θ

$\Rightarrow P(X_1, \ldots, X_n \mid \theta) =$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Parameter learning: one variable

- Unfamiliar coin:
  - Let θ = bias of coin (long-run fraction of heads)
- If θ known (given), then
  - $P(X = heads \mid \theta) = \theta$
- Different coin tosses independent given θ

$$\Rightarrow P(\underbrace{X_1, \ldots, X_n}_{h \text{ heads}, \, t \text{ tails}} \mid \theta) =$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Parameter learning: one variable

- Unfamiliar coin:
  - Let θ = bias of coin (long-run fraction of heads)
- If θ known (given), then
  - $P(X = heads \mid \theta) = \theta$
- Different coin tosses independent given θ

$$\Rightarrow P(\underbrace{X_1, \ldots, X_n}_{h \text{ heads, } t \text{ tails}} \mid \theta) = \theta^h (1-\theta)^t$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Maximum likelihood

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Maximum likelihood

- Input: a set of previous coin tosses
  - $X_1, \ldots, X_n$ = {H, T, H, H, H, T, T, H, . . ., H}

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Maximum likelihood

- Input: a set of previous coin tosses
  - $X_1, \ldots, X_n = \{\underbrace{\text{H, T, H, H, H, T, T, H, } \ldots, \text{H}}_{h \text{ heads, } t \text{ tails}}\}$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Maximum likelihood

- Input: a set of previous coin tosses
  - $X_1, \ \ldots, X_n = \{\underbrace{\text{H, T, H, H, H, T, T, H, . . ., H}}_{h \text{ heads, } t \text{ tails}}\}$
- Goal: estimate θ

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Maximum likelihood

- Input: a set of previous coin tosses
  - $X_1, \ldots, X_n = \{\underbrace{\text{H, T, H, H, H, T, T, H, } \ldots\text{, H}}_{h \text{ heads, } t \text{ tails}}\}$
- Goal: estimate θ
- The likelihood $P(X_1, \ldots, X_n \mid \theta) = \theta^h (1-\theta)^t$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Maximum likelihood

- Input: a set of previous coin tosses
  - $X_1, \ldots, X_n = \{\underbrace{\text{H, T, H, H, H, T, T, H, . . ., H}}_{h \text{ heads, } t \text{ tails}}\}$
- Goal: estimate θ
- The likelihood $P(X_1, \ldots, X_n \mid \theta) = \theta^h (1-\theta)^t$
- The maximum likelihood solution is:

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Maximum likelihood

- Input: a set of previous coin tosses
  - $X_1, \ldots, X_n = \{\underbrace{\text{H, T, H, H, H, T, T, H, . . ., H}}_{h \text{ heads, } t \text{ tails}}\}$
- Goal: estimate θ
- The likelihood $P(X_1, \ldots, X_n \mid \theta) = \theta^h (1-\theta)^t$
- The maximum likelihood solution is:

$$\theta^* = \frac{h}{h+t}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

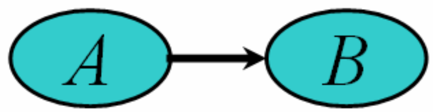
General parameter learning

- A multi-variable BN is composed of several independent parameters ("coins").

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

General parameter learning

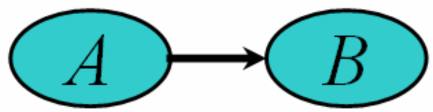
- A multi-variable BN is composed of several independent parameters (“coins”).



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

General parameter learning

- A multi-variable BN is composed of several independent parameters (“coins”).

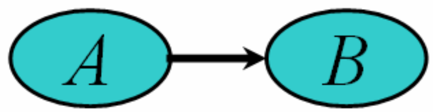


Three parameters:

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

General parameter learning

- A multi-variable BN is composed of several independent parameters ("coins").



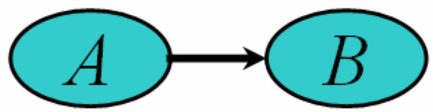
Three parameters:

$$\theta_A,\ \theta_{B|a},\ \theta_{B|\overline{a}}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

General parameter learning

- A multi-variable BN is composed of several independent parameters ("coins").



Three parameters:

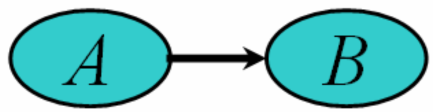
$$\theta_A,\ \theta_{B|a},\ \theta_{B|\overline{a}}$$

- Can use same techniques as one-variable case to learn each one separately

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

General parameter learning

- A multi-variable BN is composed of several independent parameters ("coins").



Three parameters:

$$\theta_A,\ \theta_{B|a},\ \theta_{B|\overline{a}}$$

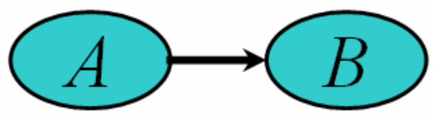
- Can use same techniques as one-variable case to learn each one separately

Max likelihood estimate of $\theta_{B|\overline{a}}$ *would be:*

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

General parameter learning

- A multi-variable BN is composed of several independent parameters ("coins").



Three parameters:

$$\theta_A,\ \theta_{B|a},\ \theta_{B|\overline{a}}$$

- Can use same techniques as one-variable case to learn each one separately

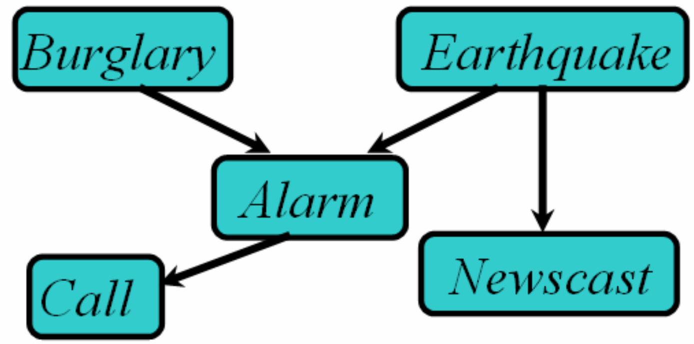
Max likelihood estimate of $\theta_{B|\overline{a}}$ *would be:*

$$\theta^*_{B|\overline{a}} = \frac{\#\text{data cases with } b, \overline{a}}{\#\text{data cases with } \overline{a}}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Partially observable data

B	E	A	C	N
$\bar{b}$	?	a	c	?
b	?	$\bar{a}$	?	n
		$\vdots$		



- Fill in missing data with “expected” value
  - expected = distribution over possible values
  - use “best guess” BN to estimate distribution

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Intuition

- In fully observable case:

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Intuition

- In fully observable case:

$$\theta^*_{n|e} = \frac{\#\text{data cases with } n, e}{\#\text{data cases with } e}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Intuition

- In fully observable case:

$$\theta^*_{n|e} = \frac{\#\text{data cases with } n, e}{\#\text{data cases with } e} = \frac{\sum_j I(n,e \mid d_j)}{\sum_j I(e \mid d_j)}$$

$$I(e \mid d_j) = \begin{cases} 1 & \text{if } E{=}e \text{ in data case } d_j \\ 0 & \text{otherwise} \end{cases}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Intuition

- In fully observable case:

$$\theta^*_{n|e} = \frac{\#\text{data cases with } n, e}{\#\text{data cases with } e} = \frac{\sum_j I(n,e \mid d_j)}{\sum_j I(e \mid d_j)}$$

$$I(e \mid d_j) = \begin{cases} 1 & \text{if } E{=}e \text{ in data case } d_j \\ 0 & \text{otherwise} \end{cases}$$

- In partially observable case *I* is unknown.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Intuition

- In fully observable case:

$$\theta^*_{n|e} = \frac{\#\text{data cases with } n, e}{\#\text{data cases with } e} = \frac{\sum_j I(n,e \mid d_j)}{\sum_j I(e \mid d_j)}$$

$$I(e \mid d_j) = \begin{cases} 1 & \text{if } E{=}e \text{ in data case } d_j \\ 0 & \text{otherwise} \end{cases}$$

- In partially observable case *I* is unknown.

Best estimate for *I* is:

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Intuition

- In fully observable case:

$$\theta^*_{n|e} = \frac{\#\text{data cases with } n, e}{\#\text{data cases with } e} = \frac{\sum_j I(n,e \mid d_j)}{\sum_j I(e \mid d_j)}$$

$$I(e \mid d_j) = \begin{cases} 1 & \text{if } E{=}e \text{ in data case } d_j \\ 0 & \text{otherwise} \end{cases}$$

- In partially observable case *I* is unknown.

Best estimate for *I* is: $\hat{I}(n,e \mid d_j) = P_{\theta^*}(n,e \mid d_j)$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Intuition

- In fully observable case:

$$\theta^*_{n|e} = \frac{\#\text{data cases with } n, e}{\#\text{data cases with } e} = \frac{\sum_j I(n,e \mid d_j)}{\sum_j I(e \mid d_j)}$$

$$I(e \mid d_j) = \begin{cases} 1 & \text{if } E{=}e \text{ in data case } d_j \\ 0 & \text{otherwise} \end{cases}$$

- In partially observable case I is unknown.

Best estimate for I is: $\hat{I}(n,e \mid d_j) = P_{\theta^*}(n,e \mid d_j)$

Problem: θ^* unknown.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Expectation Maximization (EM)

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Expectation Maximization (EM)

- Expectation (E) step
  - Use current parameters θ to estimate filled in data.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Expectation Maximization (EM)

- Expectation (E) step
  - Use current parameters θ to estimate filled in data.

$$\hat{I}(n, e \mid d_j) = P_\theta(n, e \mid d_j)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Expectation Maximization (EM)

- Expectation (E) step
  - Use current parameters θ to estimate filled in data.

$$\hat{I}(n, e \mid d_j) = P_\theta(n, e \mid d_j)$$

- Maximization (M) step
  - Use filled in data to do max likelihood estimation

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Expectation Maximization (EM)

- Expectation (E) step
  - Use current parameters θ to estimate filled in data.

$$\hat{I}(n,e \mid d_j) = P_\theta\,(n,e \mid d_j)$$

- Maximization (M) step
  - Use filled in data to do max likelihood estimation

$$\tilde{\theta}_{n|e} = \frac{\sum_j \hat{I}(n,e \mid d_j)}{\sum_j \hat{I}(e \mid d_j)}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Expectation Maximization (EM)

- Expectation (E) step
  - Use current parameters θ to estimate filled in data.

$$\hat{I}(n,e \mid d_j) = P_\theta\,(n,e \mid d_j)$$

- Maximization (M) step
  - Use filled in data to do max likelihood estimation

$$\tilde{\theta}_{n|e} = \frac{\sum_j \hat{I}(n,e \mid d_j)}{\sum_j \hat{I}(e \mid d_j)}$$

- Set: $\theta := \tilde{\theta}$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Expectation Maximization (EM)

Repeat :

- Expectation (E) step
  - Use current parameters θ to estimate filled in data.

$$\hat{I}(n, e \mid d_j) = P_\theta\,(n, e \mid d_j)$$

- Maximization (M) step
  - Use filled in data to do max likelihood estimation

$$\tilde{\theta}_{n|e} = \frac{\sum_j \hat{I}(n, e \mid d_j)}{\sum_j \hat{I}(e \mid d_j)}$$

- Set: $\theta := \tilde{\theta}$

until convergence.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Structure learning

Goal:

find "good" BN structure (relative to data)

Solution:

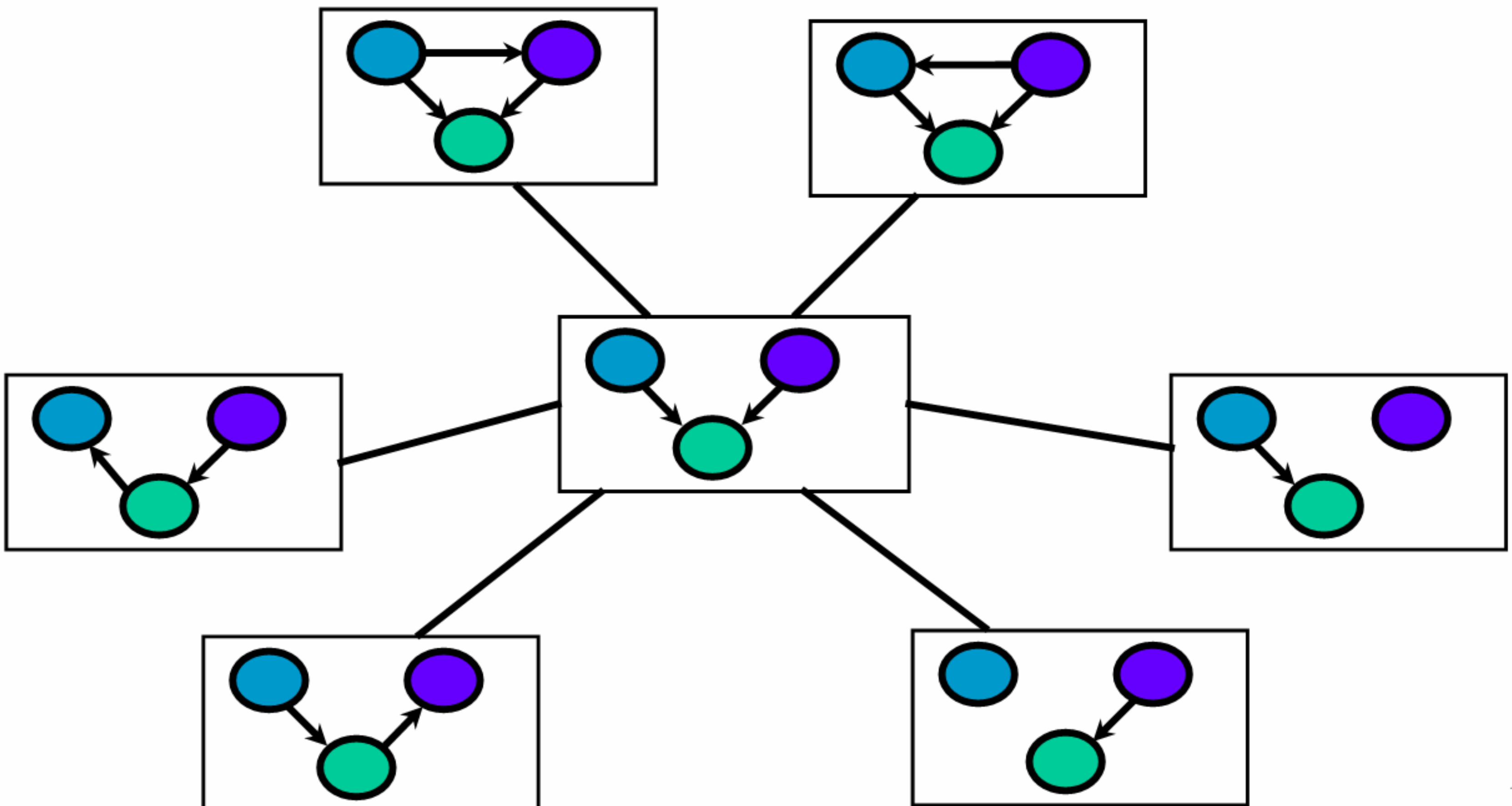
do heuristic search over space of network structures.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Search space

Space = network structures

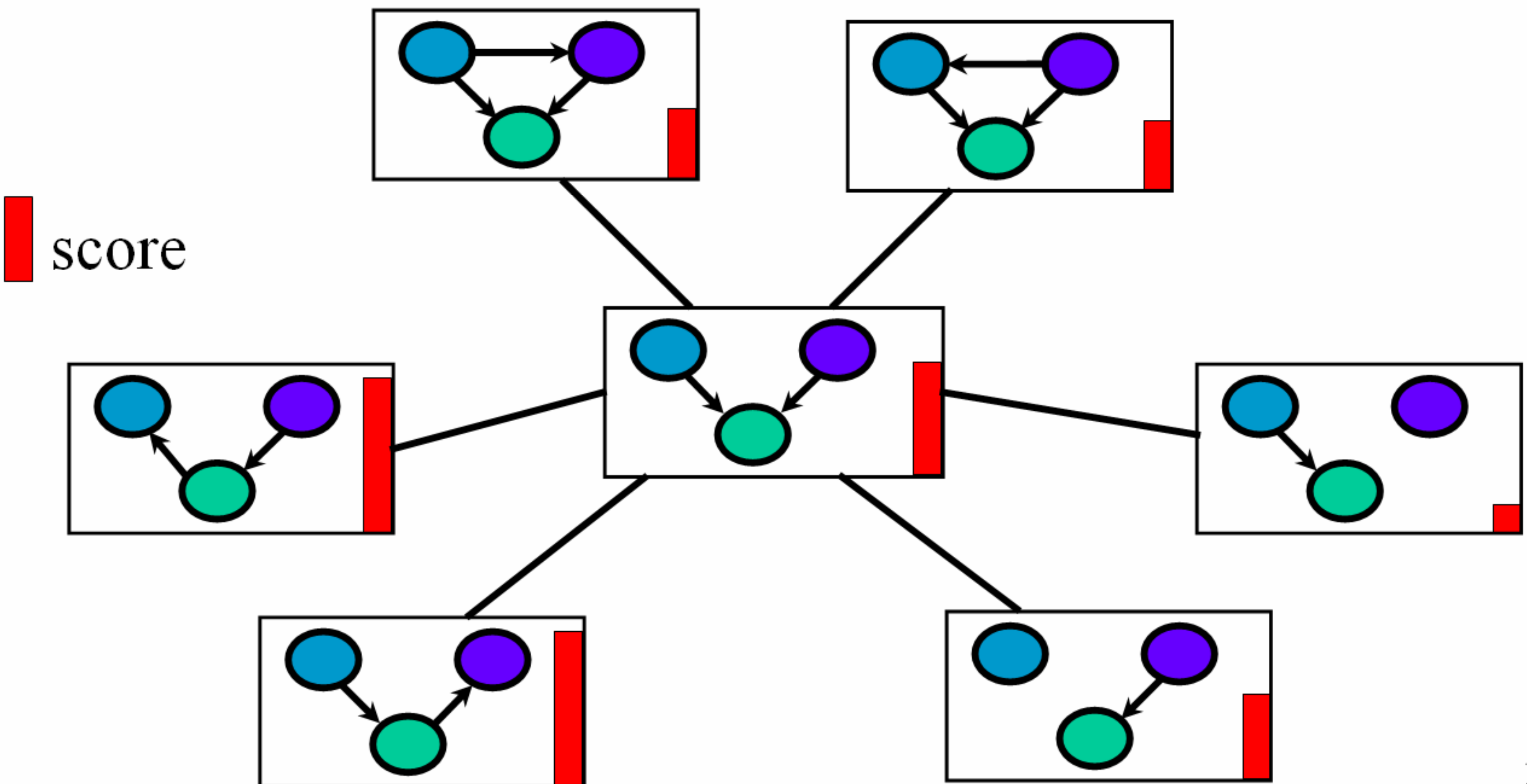
Operators = add/reverse/delete edges



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Heuristic search

Use scoring function to do heuristic search (any algorithm).
Greedy hill-climbing with randomness works pretty well.



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Scoring

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Scoring

- Fill in parameters using previous techniques & score completed networks.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Scoring

- Fill in parameters using previous techniques & score completed networks.
- One possibility for score:

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Scoring

- Fill in parameters using previous techniques & score completed networks.
- One possibility for score:

  likelihood function: $Score(B) = P(data \mid B)$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Scoring

- Fill in parameters using previous techniques & score completed networks.
- One possibility for score:

  likelihood function: $Score(B) = P(data \mid B)$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Scoring

- Fill in parameters using previous techniques & score completed networks.
- One possibility for score:

  likelihood function: $Score(B) = P(data \mid B)$ 

  Example: X, Y independent coin tosses

  typical $data = (27\ h\text{-}h,\ 22\ h\text{-}t,\ 25\ t\text{-}h,\ 26\ t\text{-}t)$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

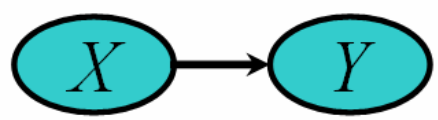
Scoring

- Fill in parameters using previous techniques & score completed networks.
- One possibility for score:

  likelihood function: *Score(B) = P(data | B)*

  Example: *X*, *Y* independent coin tosses
  typical *data = (27 h-h, 22 h-t, 25 t-h, 26 t-t)*

  Maximum likelihood network structure:



Scoring

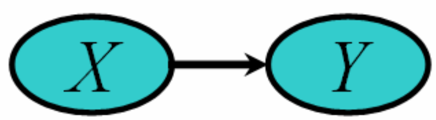
- Fill in parameters using previous techniques & score completed networks.
- One possibility for score:

  likelihood function: $Score(B) = P(data \mid B)$

  Example: X, Y independent coin tosses

  typical $data = (27\ h\text{-}h,\ 22\ h\text{-}t,\ 25\ t\text{-}h,\ 26\ t\text{-}t)$

  Maximum likelihood network structure:



Max. likelihood network typically fully connected

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Scoring

- Fill in parameters using previous techniques & score completed networks.
- One possibility for score:

  likelihood function: *Score(B) = P(data | B)*

  Example: *X*, *Y* independent coin tosses
  typical *data = (27 h-h, 22 h-t, 25 t-h, 26 t-t)*

  Maximum likelihood network structure:

$$X \rightarrow Y$$

Max. likelihood network typically fully connected

This is not surprising: maximum likelihood always overfits…

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Better scoring functions

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Better scoring functions

- MDL formulation: balance fit to data and model complexity (# of parameters)

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Better scoring functions

- MDL formulation: balance fit to data and model complexity (# of parameters)

$$Score(B) = P(data \mid B) - \text{model complexity}$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Better scoring functions

- MDL formulation: balance fit to data and model complexity (# of parameters)

$$Score(B) = P(data \mid B) - model\ complexity$$

- Full Bayesian formulation

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Better scoring functions

- MDL formulation: balance fit to data and model complexity (# of parameters)

$$Score(B) = P(data \mid B) - model\ complexity$$

- Full Bayesian formulation
  - prior on network structures & parameters

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Better scoring functions

- MDL formulation: balance fit to data and model complexity (# of parameters)

$$Score(B) = P(data \mid B) - model\ complexity$$

- Full Bayesian formulation
  - prior on network structures & parameters
  - more parameters $\Rightarrow$ higher dimensional space

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Better scoring functions

- MDL formulation: balance fit to data and model complexity (# of parameters)

$$Score(B) = P(data \mid B) - model\ complexity$$

- Full Bayesian formulation
  - prior on network structures & parameters
  - more parameters $\Rightarrow$ higher dimensional space
  - get balance effect as a byproduct*

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Better scoring functions

- MDL formulation: balance fit to data and model complexity (# of parameters)

$$Score(B) = P(data \mid B) - model\ complexity$$

- Full Bayesian formulation
  - prior on network structures & parameters
  - more parameters $\Rightarrow$ higher dimensional space
  - get balance effect as a byproduct*

* with Dirichlet parameter prior, MDL is an approximation to full Bayesian score.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Course Contents

- Concepts in Probability
- Bayesian Networks
- Inference
- Decision making
- Learning networks from data
- Reasoning over time

» Applications

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

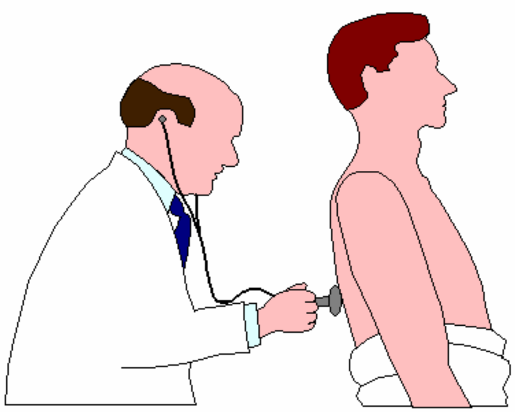
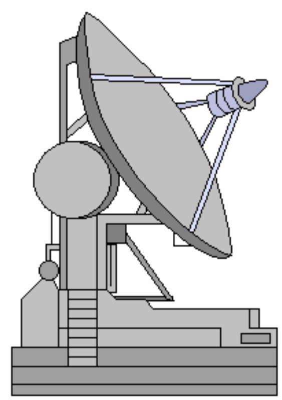
Applications

- Medical expert systems
  - Pathfinder
  - Parenting MSN
- Fault diagnosis
  - Ricoh FIXIT
  - Decision-theoretic troubleshooting
- Vista
- Collaborative filtering

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Why use Bayesian Networks?

- Explicit management of uncertainty/tradeoffs
- Modularity implies maintainability
- Better, flexible, and robust recommendation strategies


© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Pathfinder

- Pathfinder is one of the first BN systems.
- It performs diagnosis of lymph-node diseases.
- It deals with over 60 diseases and 100 findings.
- Commercialized by Intellipath and Chapman Hall publishing and applied to about 20 tissue types.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

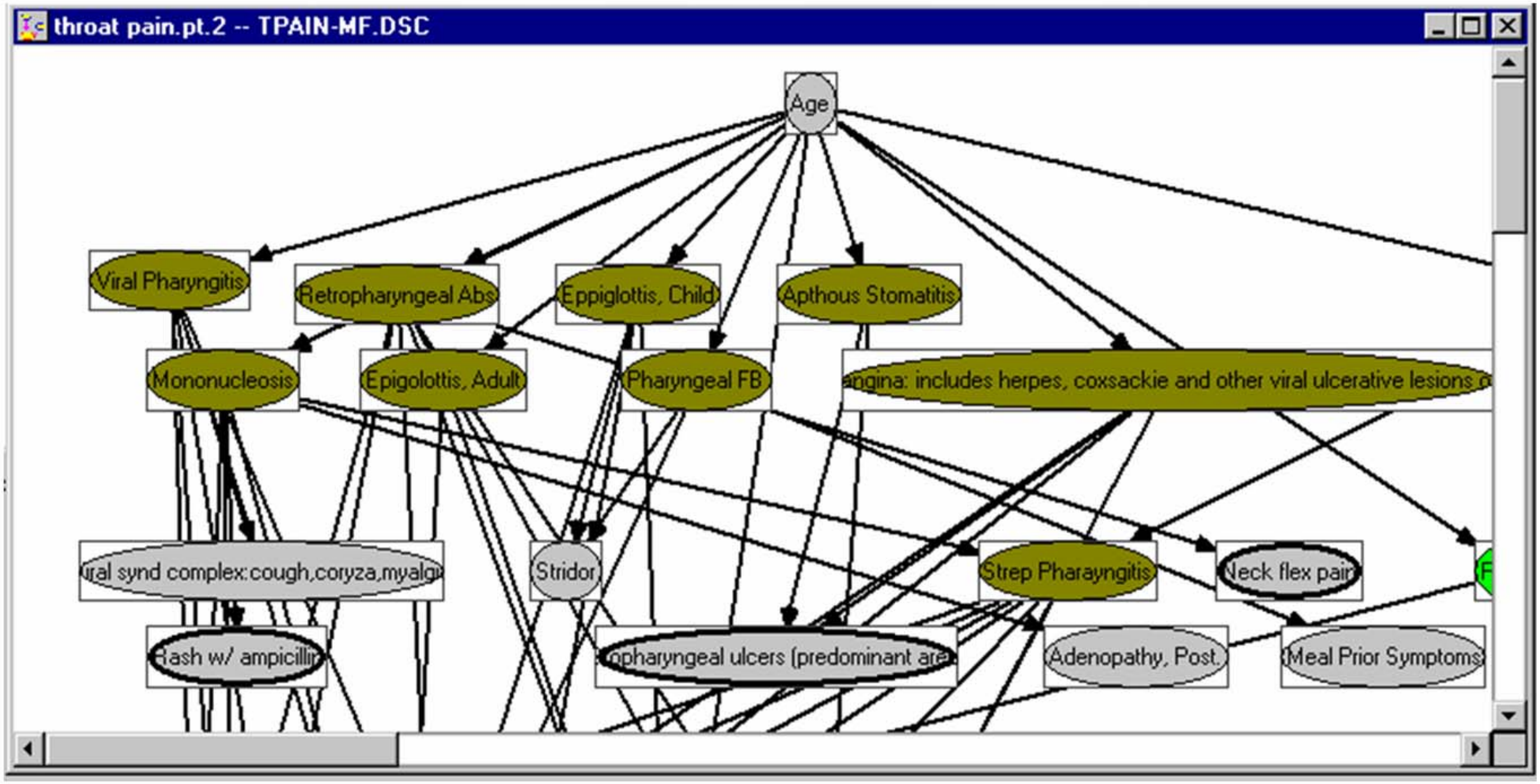
On Parenting: Selecting problem

- Diagnostic indexing for Home Health site on Microsoft Network
- Enter symptoms for pediatric complaints
- Recommends multimedia content

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

On Parenting : MSN

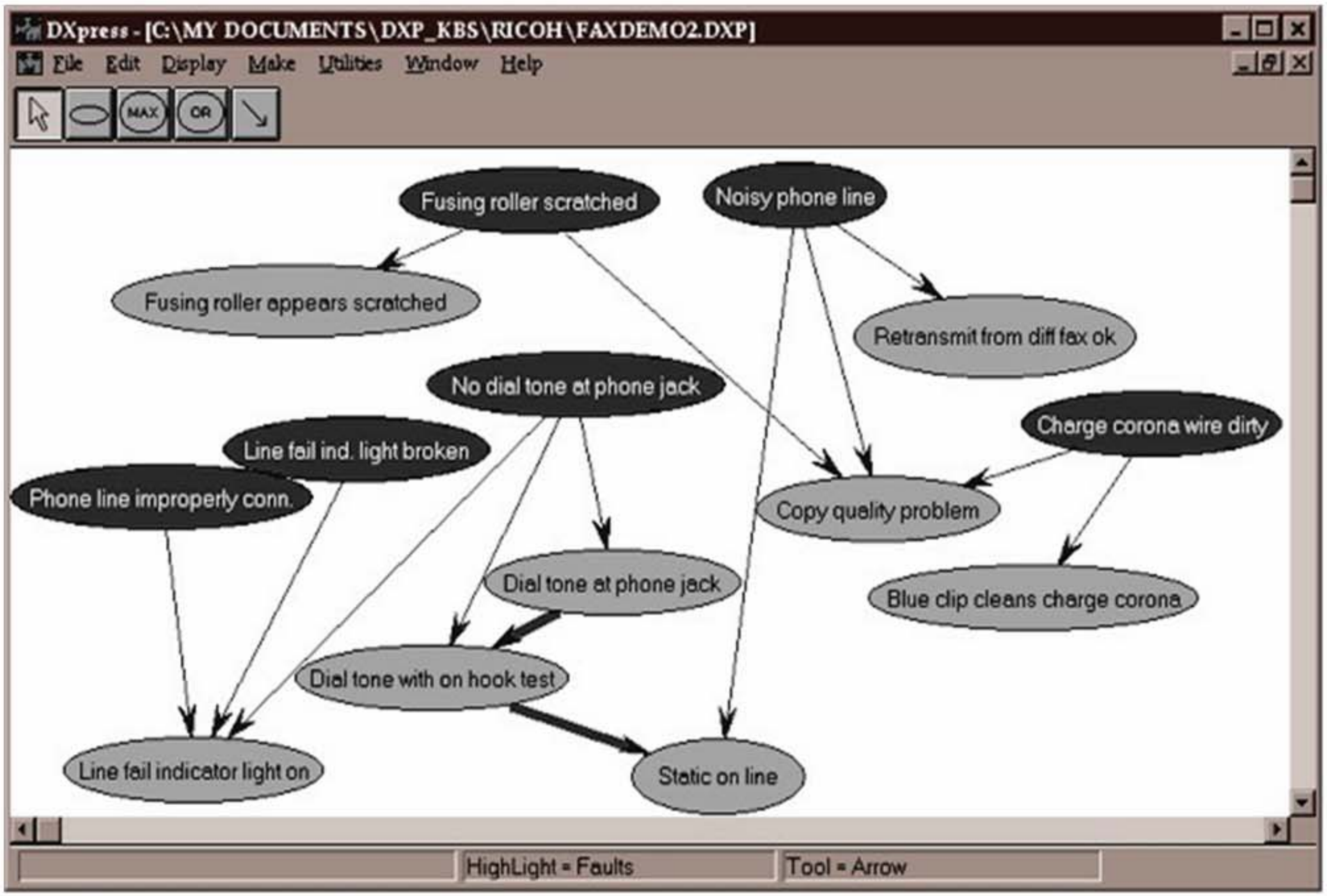
Original Multiple Fault Model



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

RICOH Fixit

- Diagnostics and information retrieval



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What is Collaborative Filtering?

- A way to find cool websites, news stories, music artists etc
- Uses data on the preferences of many users, not descriptions of the content.
- Firefly, Net Perceptions (GroupLens), and others offer this technology.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

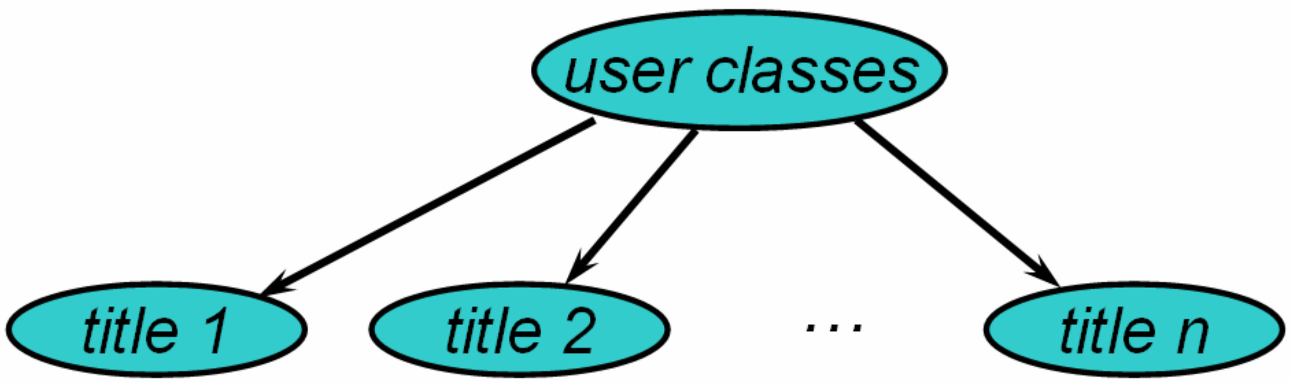
Bayesian Clustering for Collaborative Filtering

- Probabilistic summary of the data
- Reduces the number of parameters to represent a set of preferences
- Provides insight into usage patterns.
- Inference:

$$P(\textit{Like title } i \mid \textit{Like title } j, \textit{Like title } k)$$

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Applying Bayesian clustering



	class1	class2	...
title1	p(like)=0.2	p(like)=0.8	
title2	p(like)=0.7	p(like)=0.1	
title3	p(like)=0.99	p(like)=0.01	
	...		

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

MSNBC Story clusters

<u>Readers of commerce and technology stories (36%):</u>

- E-mail delivery isn't exactly guaranteed
- Should you buy a DVD player?
- Price low, demand high for Nintendo

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

MSNBC Story clusters

Readers of commerce and technology stories (36%):

- E-mail delivery isn't exactly guaranteed
- Should you buy a DVD player?
- Price low, demand high for Nintendo

Readers of top promoted stories (29%):

- 757 Crashes At Sea
- Israel, Palestinians Agree To Direct Talks
- Fuhrman Pleads Innocent To Perjury

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

MSNBC Story clusters

Readers of commerce and technology stories (36%):

- E-mail delivery isn't exactly guaranteed
- Should you buy a DVD player?
- Price low, demand high for Nintendo

Readers of top promoted stories (29%):

- 757 Crashes At Sea
- Israel, Palestinians Agree To Direct Talks
- Fuhrman Pleads Innocent To Perjury

Sports Readers (19%):

- Umps refusing to work is the right thing
- Cowboys are reborn in win over eagles
- Did Orioles spend money wisely?

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

MSNBC Story clusters

Readers of commerce and technology stories (36%):

- E-mail delivery isn't exactly guaranteed
- Should you buy a DVD player?
- Price low, demand high for Nintendo

Readers of top promoted stories (29%):

- 757 Crashes At Sea
- Israel, Palestinians Agree To Direct Talks
- Fuhrman Pleads Innocent To Perjury

Sports Readers (19%):

- Umps refusing to work is the right thing
- Cowboys are reborn in win over eagles
- Did Orioles spend money wisely?

Readers of "Softer" News (12%):

- The truth about what things cost
- Fuhrman Pleads Innocent To Perjury
- Real Astrology

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Top 5 shows by user class

Class 1
- Power rangers
- Animaniacs
- X-men
- Tazmania
- Spider man

Class 2
- Young and restless
- Bold and the beautiful
- As the world turns
- Price is right
- CBS eve news

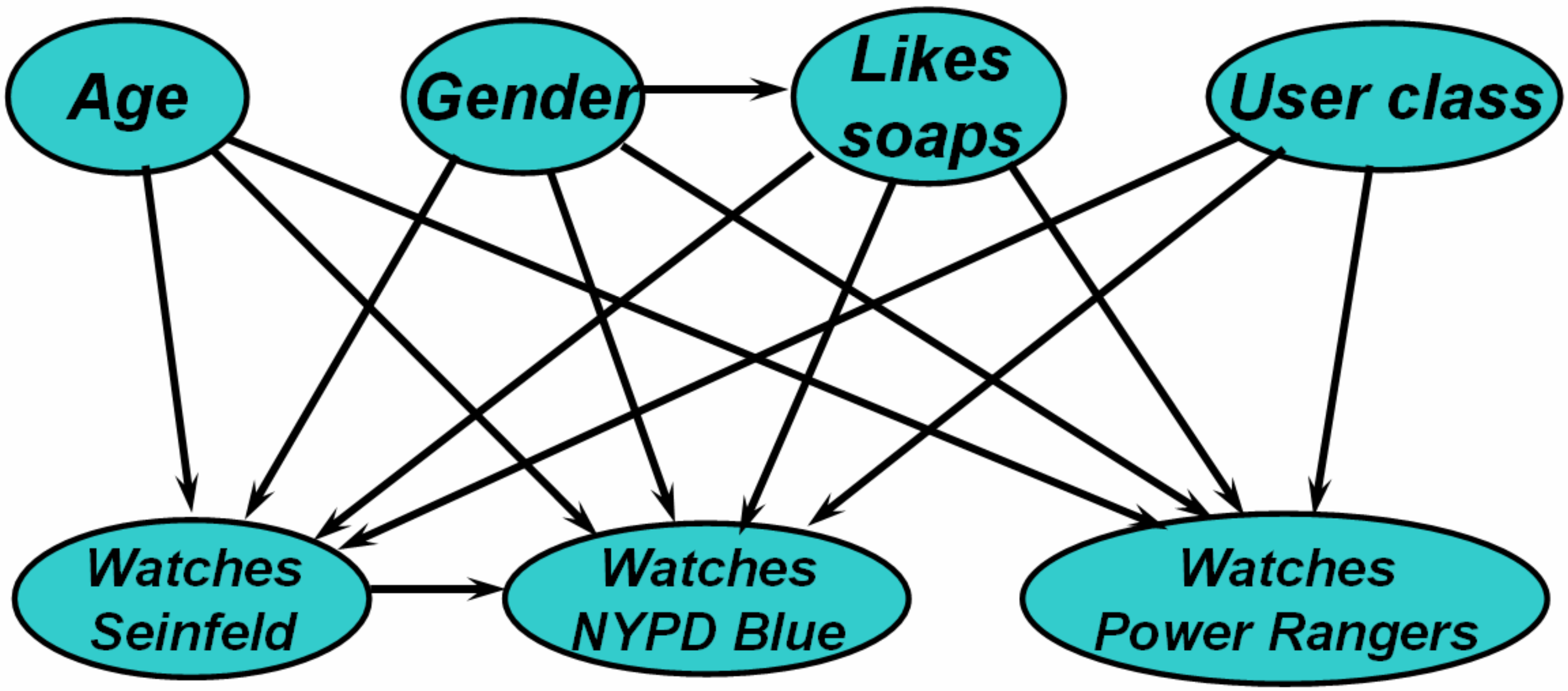
Class 3
- Tonight show
- Conan O'Brien
- NBC nightly news
- Later with Kinnear
- Seinfeld

Class 4
- 60 minutes
- NBC nightly news
- CBS eve news
- Murder she wrote
- Matlock

Class 5
- Seinfeld
- Friends
- Mad about you
- ER
- Frasier

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

Richer model



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's old?

Decision theory & probability theory provide:

- principled models of belief and preference;
- techniques for:
  - integrating evidence (conditioning);
  - optimal decision making (max. expected utility);
  - targeted information gathering (value of info.);
  - parameter estimation from data.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's new?

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

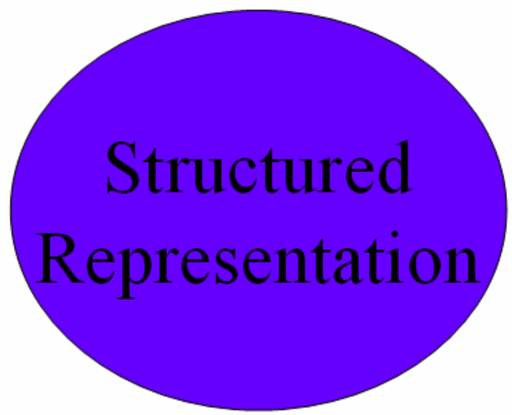
What's new?

Bayesian networks exploit domain structure to allow compact representations of complex models.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's new?

Bayesian networks exploit domain structure to allow compact representations of complex models.

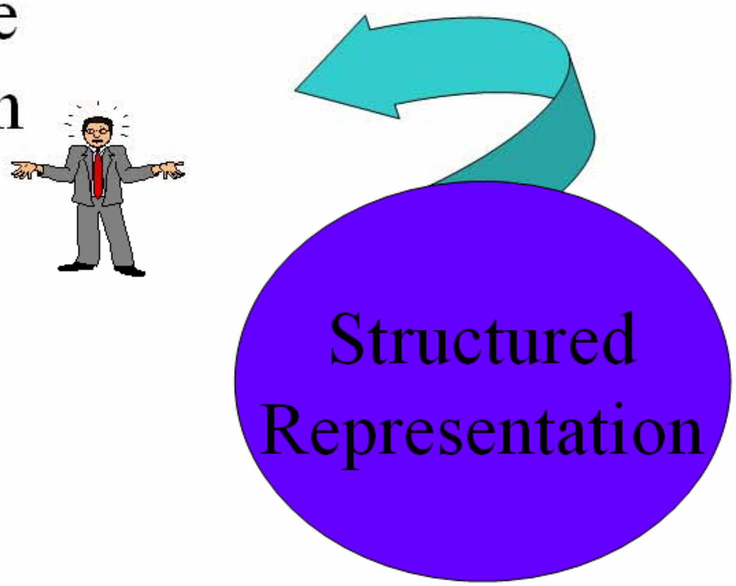


© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's new?

Bayesian networks exploit domain structure to allow compact representations of complex models.

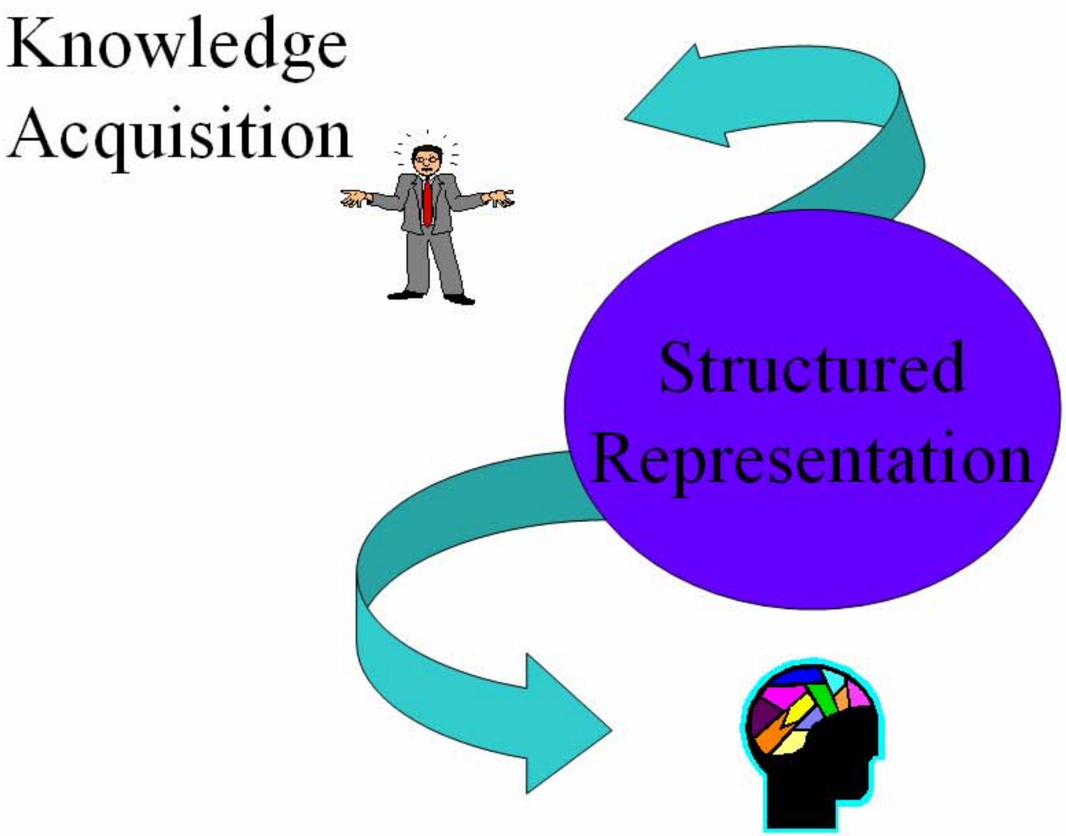
Knowledge Acquisition



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What’s new?

Bayesian networks exploit domain structure to allow compact representations of complex models.

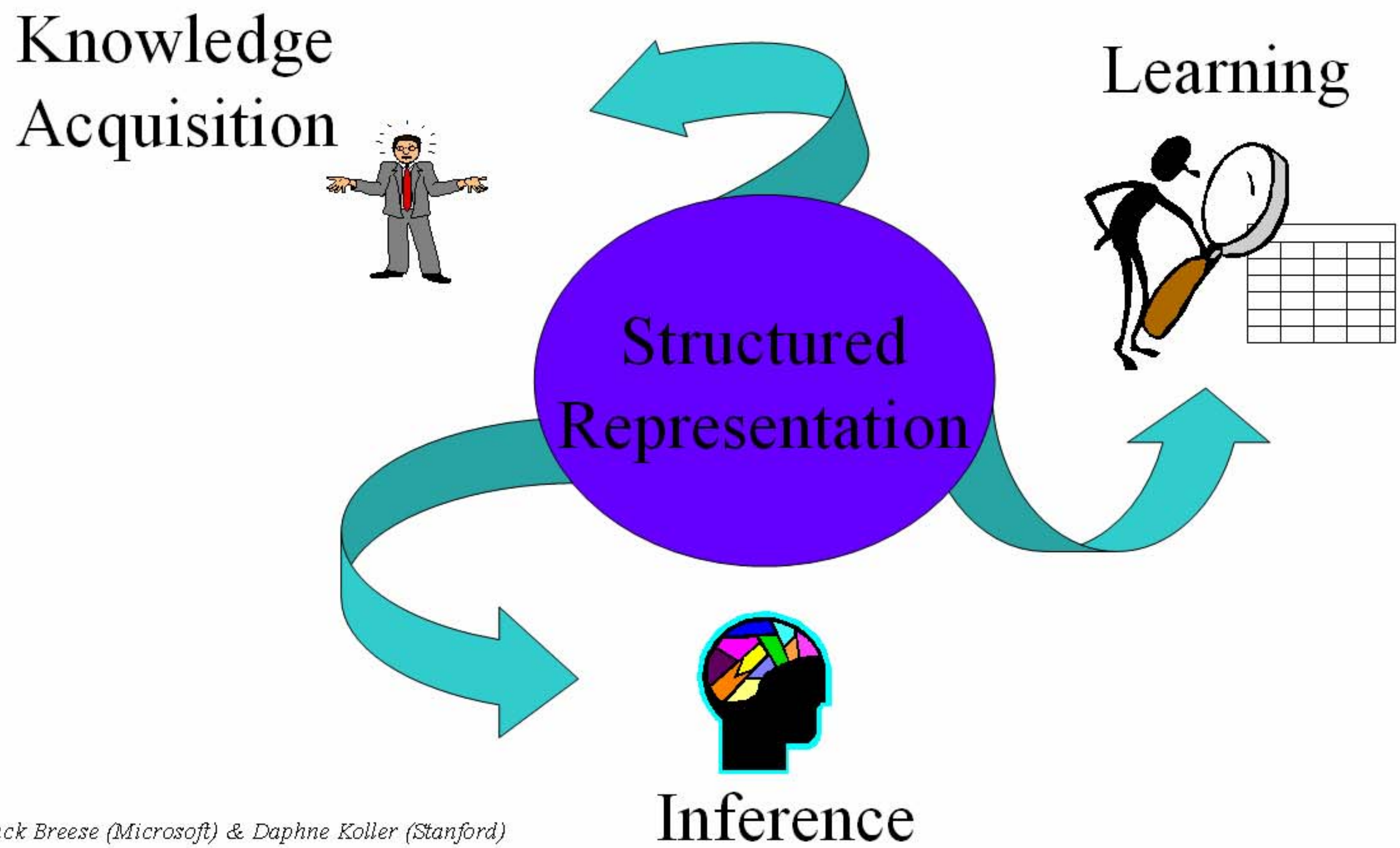


Inference

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's new?

Bayesian networks exploit domain structure to allow compact representations of complex models.



© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's in our future?

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What’s in our future?

- Better models for:

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What’s in our future?

- Better models for:
  - preferences & utilities;

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's in our future?

- Better models for:
  - preferences & utilities;
  - not-so-precise numerical probabilities.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's in our future?

- Better models for:
  - preferences & utilities;
  - not-so-precise numerical probabilities.
- Inferring causality from data.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's in our future?

- Better models for:
  - preferences & utilities;
  - not-so-precise numerical probabilities.
- Inferring causality from data.
- More expressive representation languages:

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's in our future?

- Better models for:
  - preferences & utilities;
  - not-so-precise numerical probabilities.
- Inferring causality from data.
- More expressive representation languages:
  - structured domains with multiple objects;

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's in our future?

- Better models for:
  - preferences & utilities;
  - not-so-precise numerical probabilities.
- Inferring causality from data.
- More expressive representation languages:
  - structured domains with multiple objects;
  - levels of abstraction;

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's in our future?

- Better models for:
  - preferences & utilities;
  - not-so-precise numerical probabilities.
- Inferring causality from data.
- More expressive representation languages:
  - structured domains with multiple objects;
  - levels of abstraction;
  - reasoning about time;

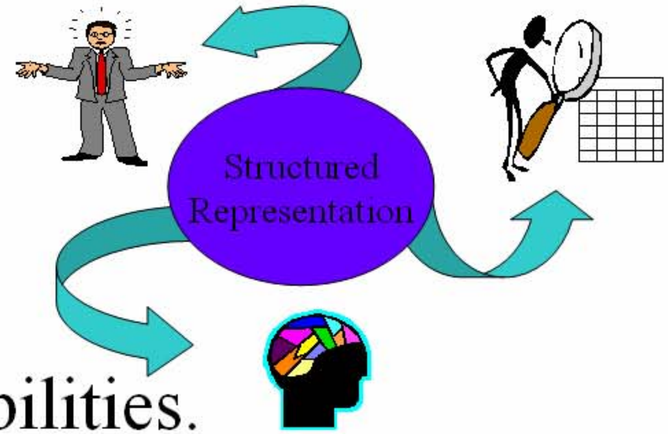
© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's in our future?

- Better models for:
  - preferences & utilities;
  - not-so-precise numerical probabilities.
- Inferring causality from data.
- More expressive representation languages:
  - structured domains with multiple objects;
  - levels of abstraction;
  - reasoning about time;
  - hybrid (continuous/discrete) models.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)

What's in our future?



- Better models for:
  - preferences & utilities;
  - not-so-precise numerical probabilities.
- Inferring causality from data.
- More expressive representation languages:
  - structured domains with multiple objects;
  - levels of abstraction;
  - reasoning about time;
  - hybrid (continuous/discrete) models.

© Jack Breese (Microsoft) & Daphne Koller (Stanford)