A Tutorial on Inference and Learning in Bayesian Networks

Irina Rish

IBM T.J.Watson Research Center
rish@us.ibm.com
http://www.research.ibm.com/people/r/rish/

Outline

- Motivation: learning probabilistic models from data
- Representation: Bayesian network models
- Probabilistic inference in Bayesian Networks
  - Exact inference
  - Approximate inference
- Learning Bayesian Networks
  - Learning parameters
  - Learning graph structure (model selection)
- Summary

Bayesian Networks

Structured, graphical representation of probabilistic relationships between several random variables

Explicit representation of conditional independencies

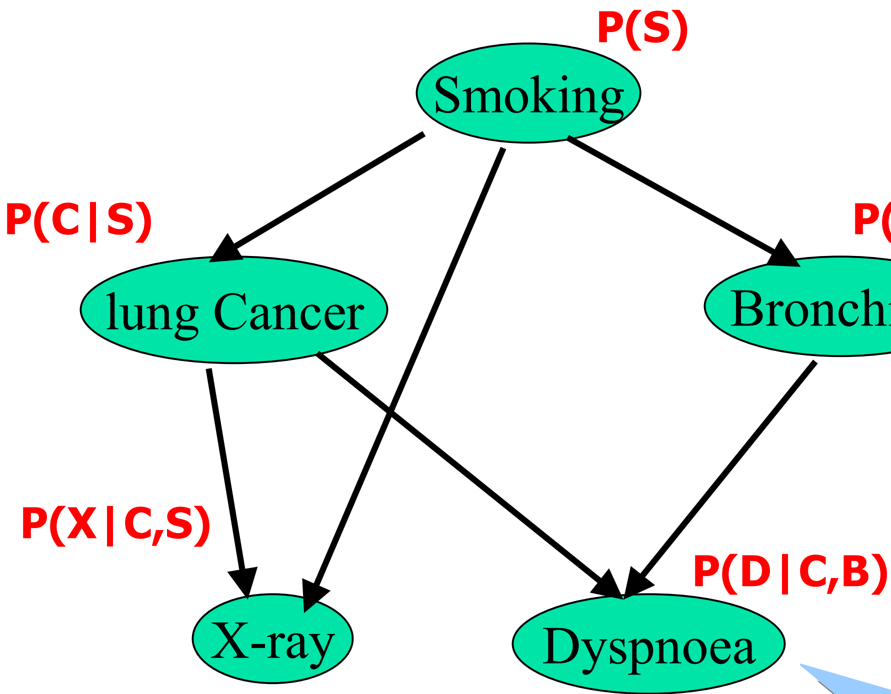
Missing arcs encode conditional independence

Efficient representation of joint PDF $P(X)$

Generative model (not just discriminative): allows arbitrary queries to be answered, e.g.

P (lung cancer=yes | smoking=no, positive X-ray=yes) = ?

Bayesian Network: $\mathbf{BN} = (\mathbf{G}, \boldsymbol{\Theta})$



G - directed acyclic graph (DAG)
nodes – random variables
edges – direct dependencies

$\boldsymbol{\Theta}$ - set of parameters in all conditional probability distributions (CPDs)

CPD:

C	B	D=0	D=1
0	0	0.1	0.9
0	1	0.7	0.3
1	0	0.8	0.2
1	1	0.9	0.1

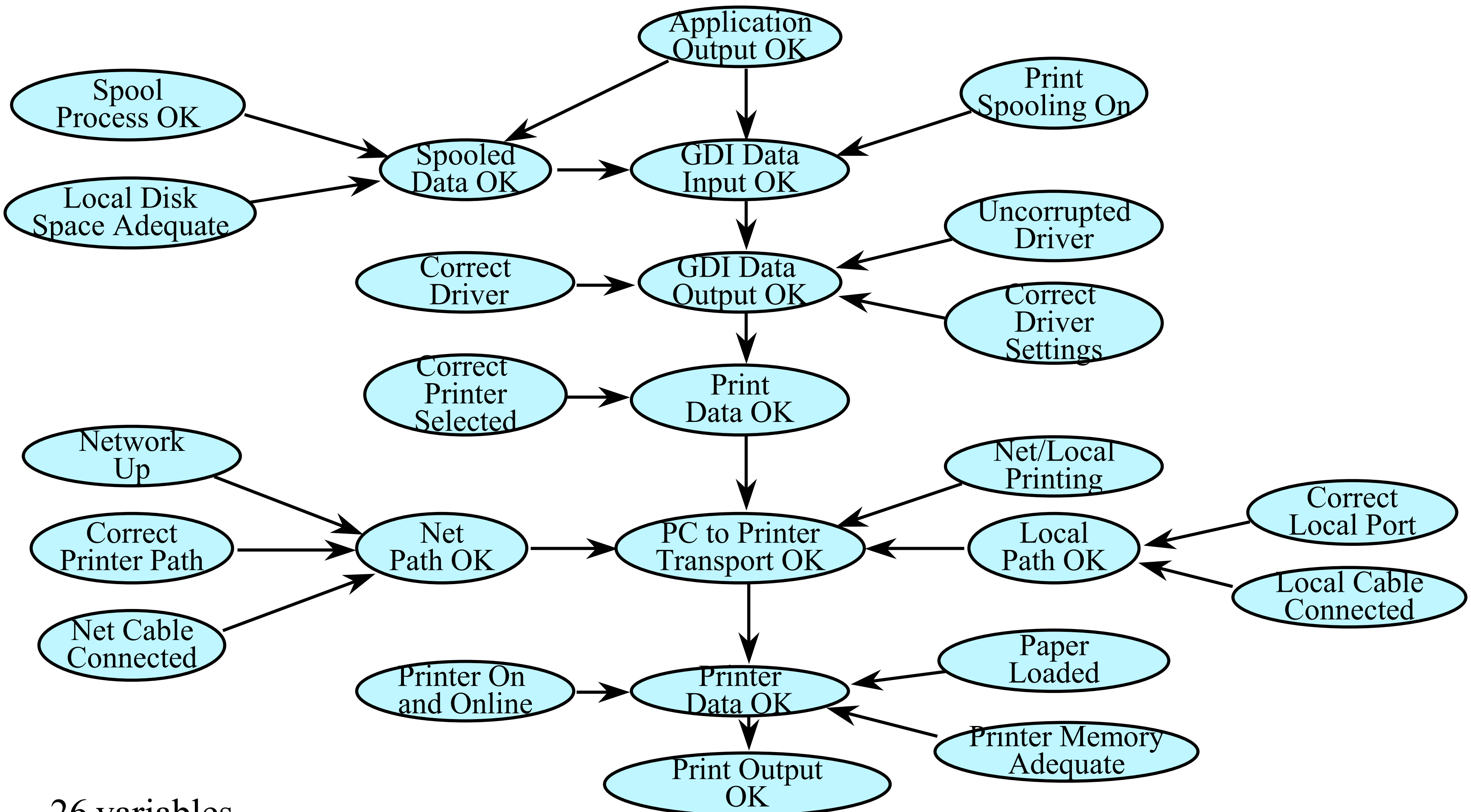
CPD of node X: P(X|parents(X))

Compact representation of joint distribution in a **product form** (chain rule):

$$P(S, C, B, X, D) = P(S)\,P(C|S)\,P(B|S)\,P(X|C,S)\,P(D|C,B)$$

$$\mathbf{1+2+2+4+4=13\ \ parameters\ \ instead\ of\ \ 2^5} = 32$$

Example: Printer Troubleshooting



26 variables

Instead of 2^{26} parameters we get

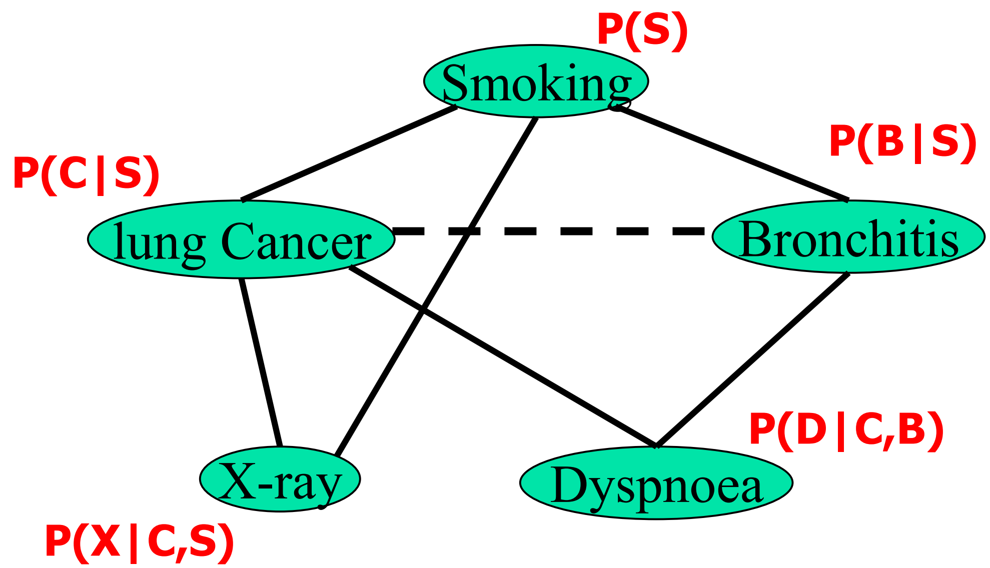
$99 = 17x1 + 1x2^1 + 2x2^2 + 3x2^3 + 3x2^4$

[Heckerman, 95]

“Moral” graph of a BN

Moralization algorithm:

1. Connect (“marry”) parents of each node.
2. Drop the directionality of the edges.



Resulting undirected graph is called the “moral” graph of BN

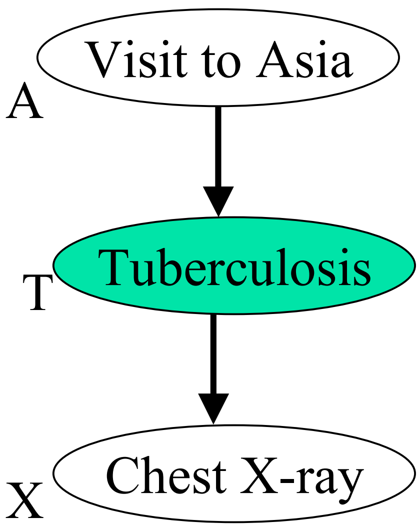
Interpretation:

every pair of nodes that occur together in a CPD is connected by an edge in the moral graph.

CPD for X and its k parents (called “**family**”) is represented by a clique of size **(k+1)** in the moral graph, and contains $d^k(d-1)$ probability parameters where d is the number of values each variable can have (domain size).

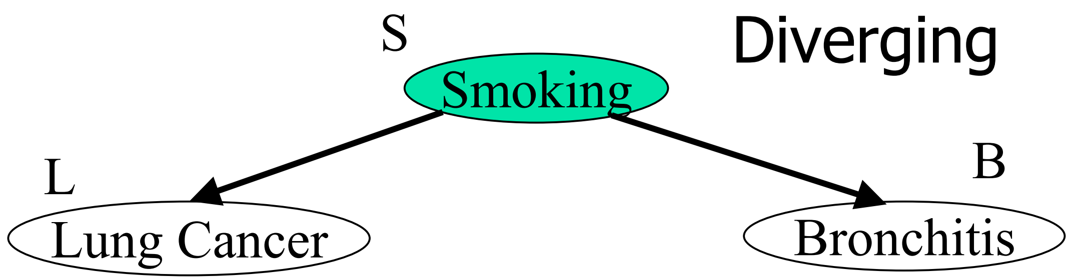
Conditional Independence in BNs: Three types of connections

Serial



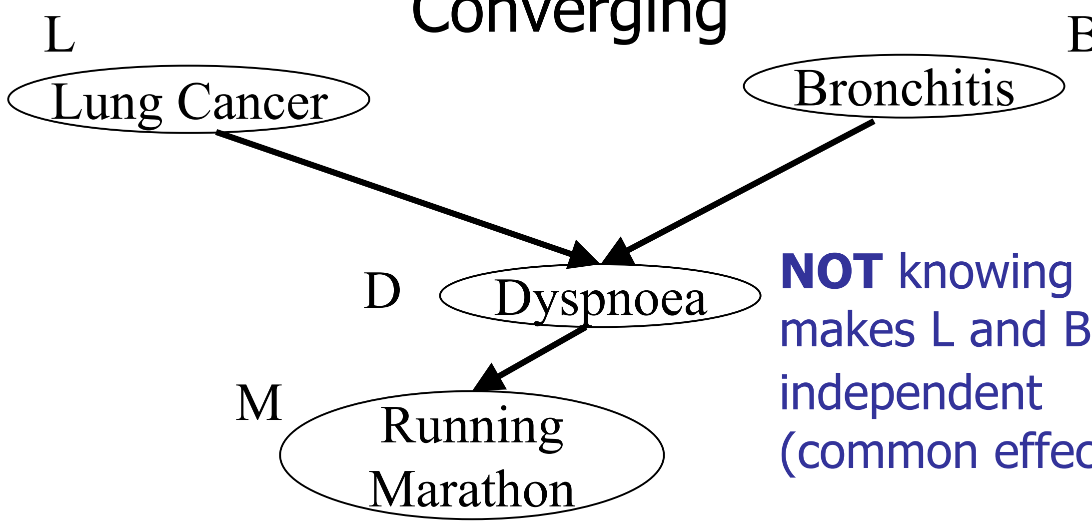
Knowing T makes
A and X independent
(intermediate cause)

Diverging



Knowing S makes L and B independent (common cause)

Converging



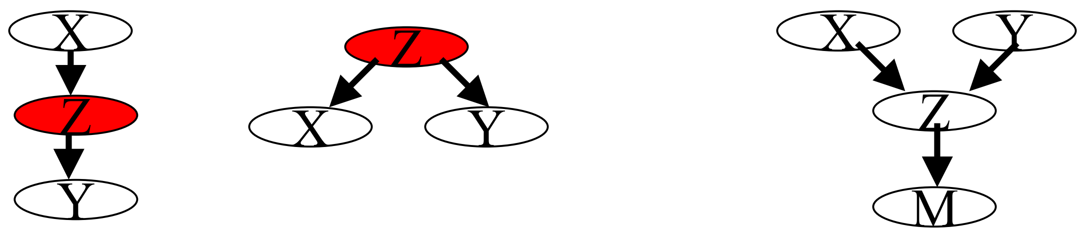
NOT knowing D or M makes L and B independent (common effect)

d-separation

Nodes X and Y are *d-separated* if on *any (undirected) path* between X and Y there is some variable Z such that is either

Z is in a *serial* or *diverging* connection and Z is *known*, or

Z is in a *converging* connection and neither Z nor any of Z's descendants are known

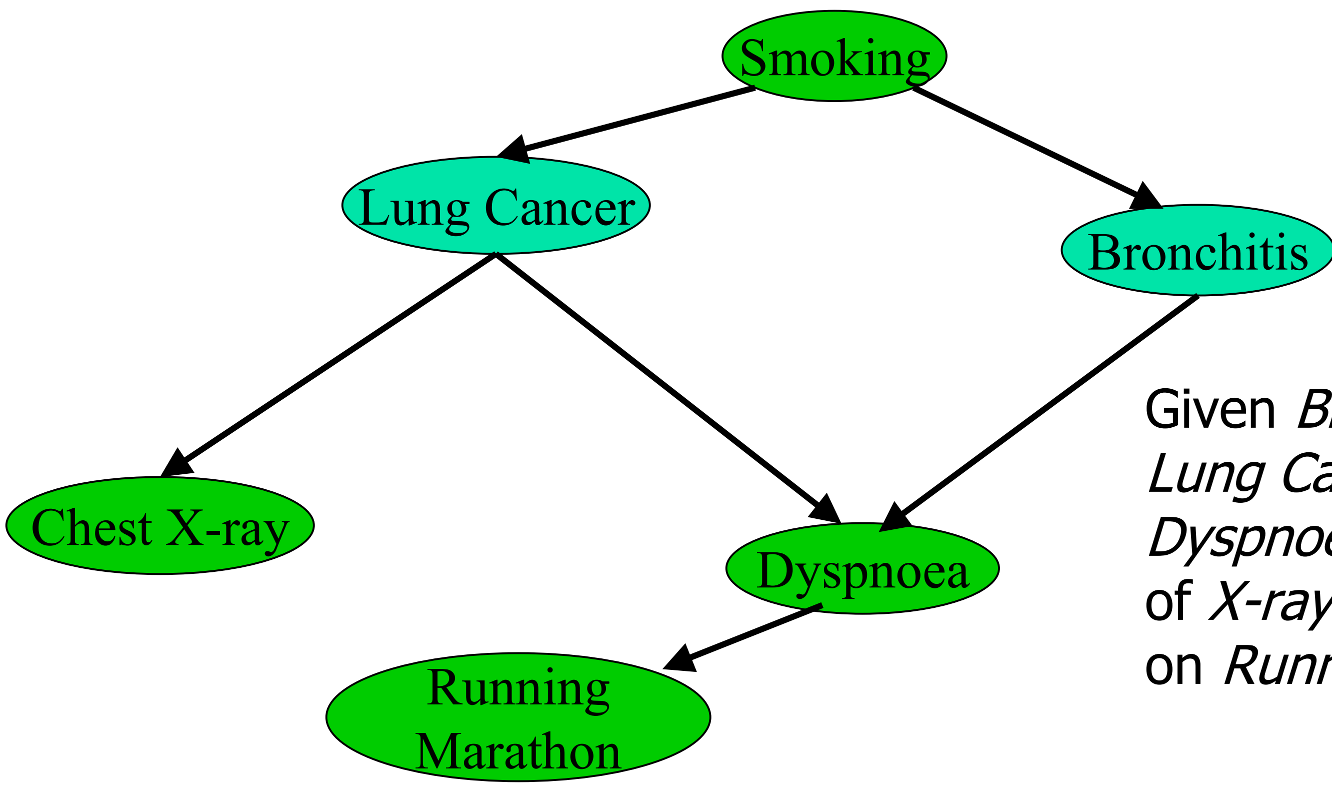


Nodes X and Y are called *d-connected* if they are not d-separated (there exists an undirected path between X and Y not d-separated by any node or a set of nodes)

If nodes X and Y are *d-separated* by Z, then X and Y are *conditionally independent* given Z (see Pearl, 1988)

Independence Relations in BN

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

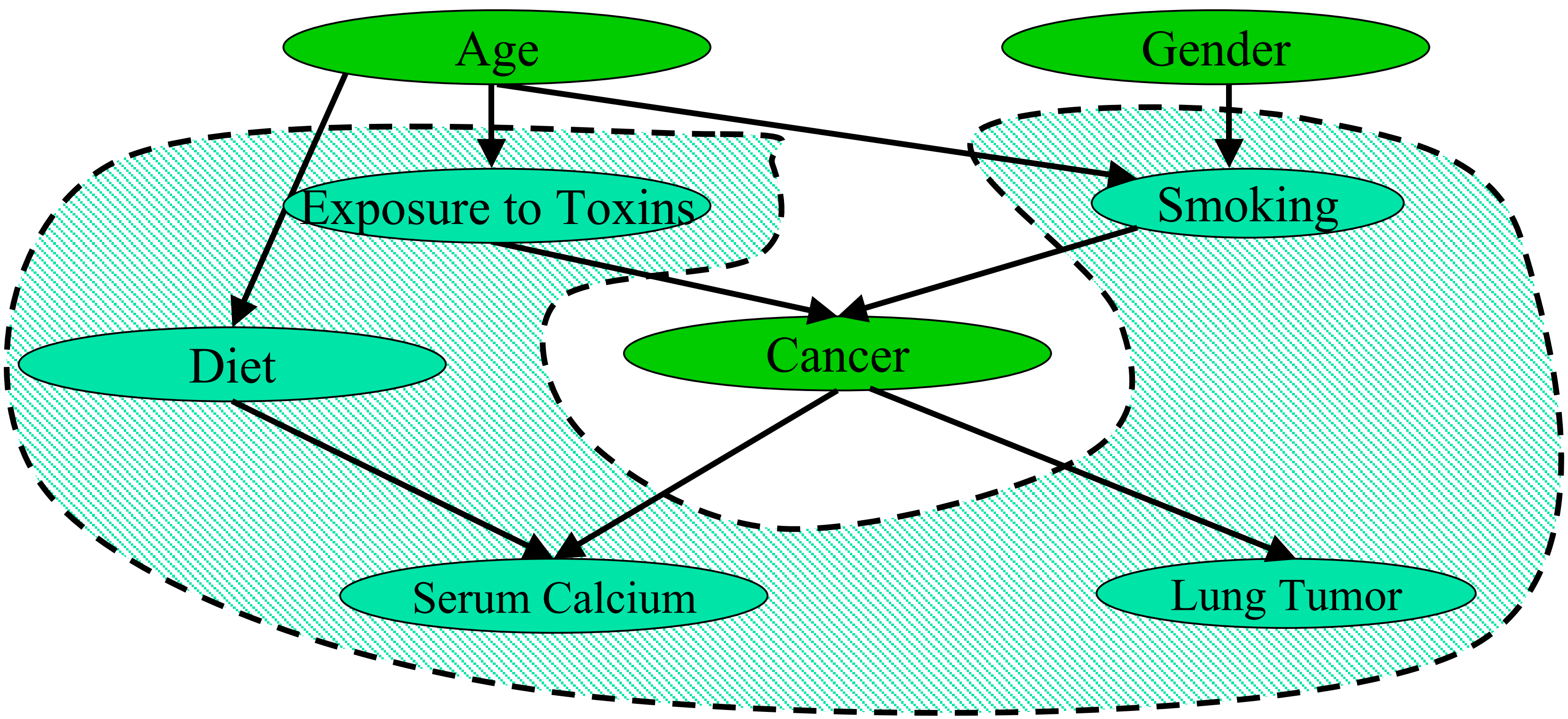


Given *Bronchitis* and *Lung Cancer*, *Dyspnoea* is independent of *X-ray* (but may depend on *Running Marathon*)

Markov Blanket

A node is conditionally independent of ALL other nodes given its *Markov blanket,* i.e. its *parents*, *children*, and "*spouses*" (parents of common children)

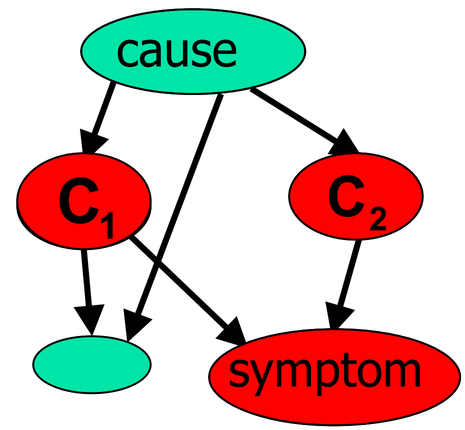
(Proof left as a homework problem ☺)

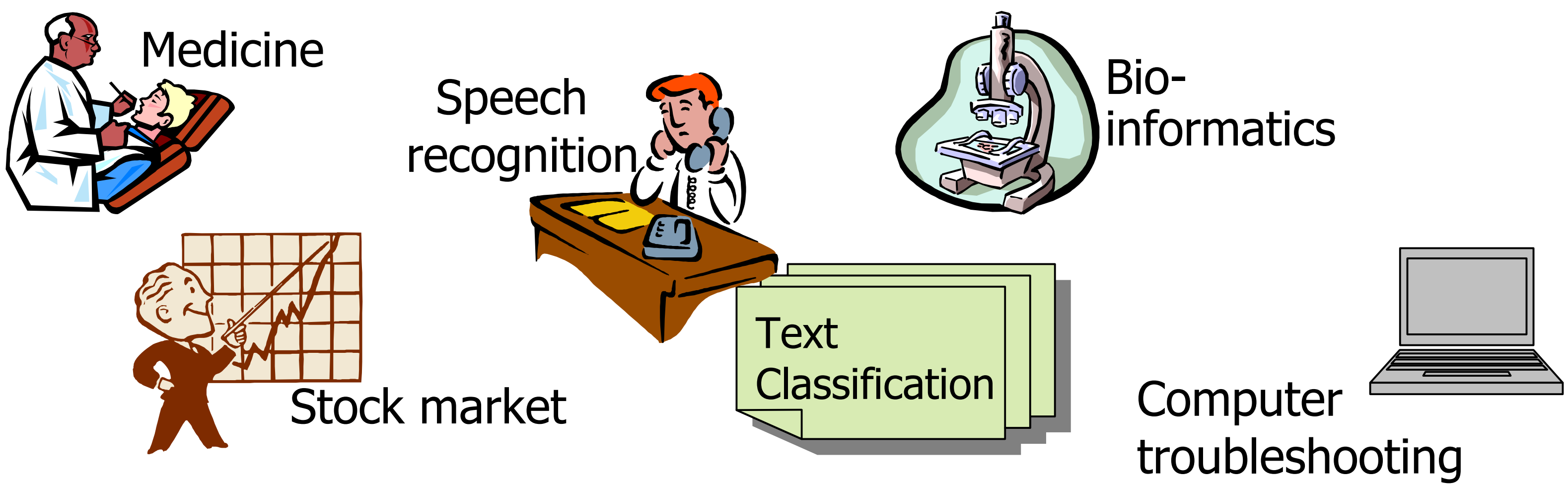


[Breese & Koller, 97]

What are BNs useful for?

- Diagnosis: P(cause|symptom)=?
- Prediction: P(symptom|cause)=?
- Classification: $\max_{class} P(class|data)$
- Decision-making (given a cost function)





Application Examples

APRI system developed at AT&T Bell Labs

- learns & uses Bayesian networks from data to identify customers liable to default on bill payments

NASA Vista system

- predict failures in propulsion systems
- considers time criticality & suggests highest utility action
- dynamically decide what information to show

Application Examples

Office Assistant in MS Office 97/ MS Office 95

- Extension of Answer wizard
- uses naïve Bayesian networks
- help based on past experience (keyboard/mouse use) and task user is doing currently
- This is the “smiley face” you get in your MS Office applications

Microsoft Pregnancy and Child-Care

- Available on MSN in Health section
- Frequently occurring children’s symptoms are linked to expert modules that repeatedly ask parents relevant questions
- Asks next best question based on provided information
- Presents articles that are deemed relevant based on information provided

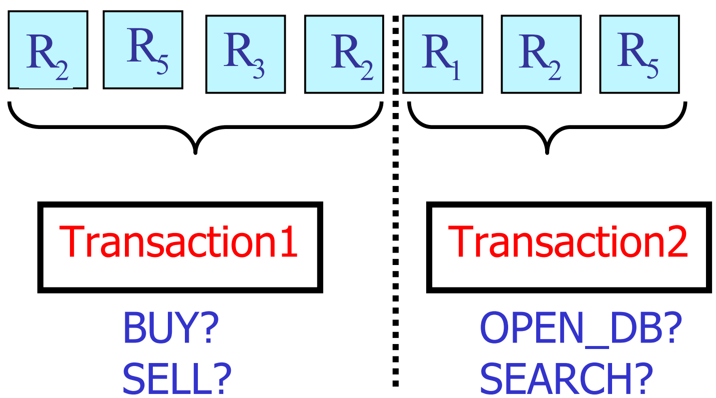
IBM's systems management applications

Machine Learning for Systems @ Watson

(Hellerstein, Jayram, Rish (2000))

End-user transaction recognition

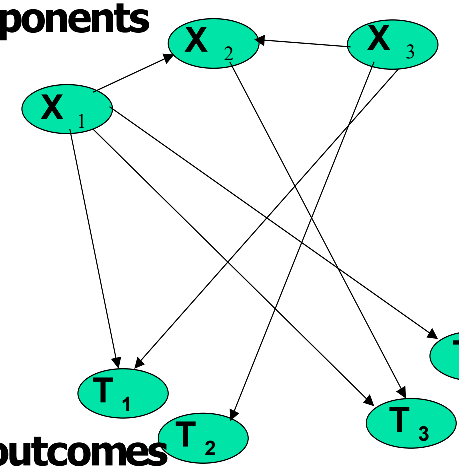
Remote Procedure Calls (RPCs)



(Rish, Brodie, Ma (2001))

Fault diagnosis using probes

Software or hardware components



Probe outcomes

Issues:

- Efficiency (scalability)
- Missing data/noise: sensitivity analysis
- "Adaptive" probing:
  - selecting "most-informative" probes
  - on-line learning/model updates
  - on-line diagnosis

Goal: finding most-likely diagnosis

$$(x_1^*, \ldots x_n^*) = \arg\max_{x_1,\ldots x_n} P(x_1, \ldots x_n \mid t_1, \ldots t_n)$$

Pattern discovery, classification, diagnosis and prediction

Probabilistic Inference Tasks

- Belief updating:

$$\mathrm{BEL}(X_i) = P(X_i = x_i \mid \text{evidence})$$

- Finding most probable explanation (MPE)

$$\bar{x}^* = \arg\max_{\bar{x}} P(\bar{x}, e)$$

- Finding maximum a-posteriory hypothesis

$$(a_1^*, \ldots, a_k^*) = \arg\max_{\bar{a}} \sum_{X/A} P(\bar{x}, e)$$

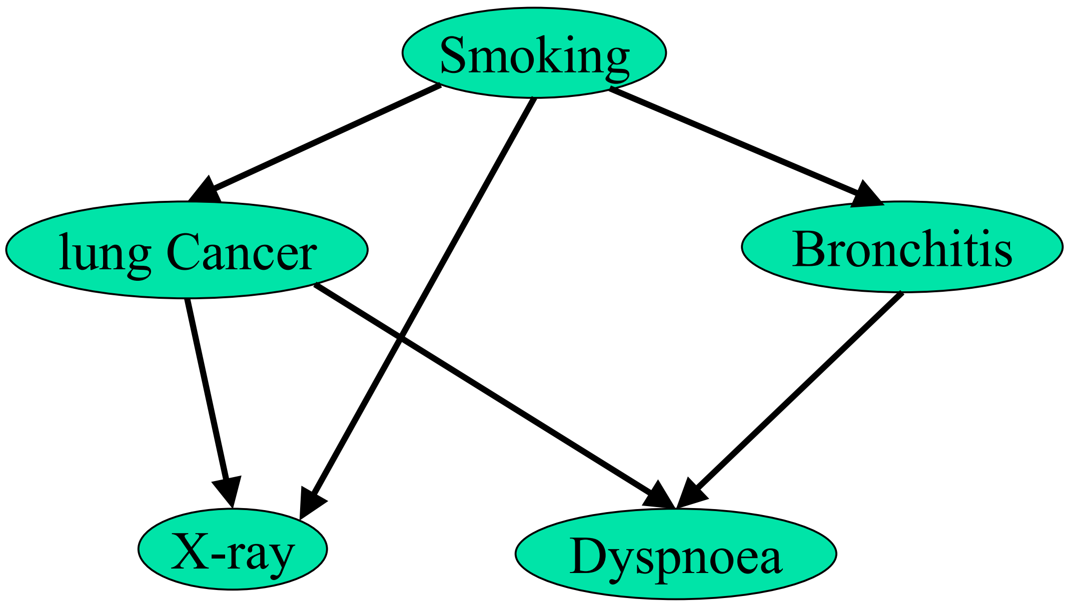
$A \subseteq X$: **hypothesis variables**

- Finding maximum-expected-utility (MEU) decision

$$(d_1^*, \ldots, d_k^*) = \arg\max_{\bar{d}} \sum_{X/D} P(\bar{x}, e) U(\bar{x})$$

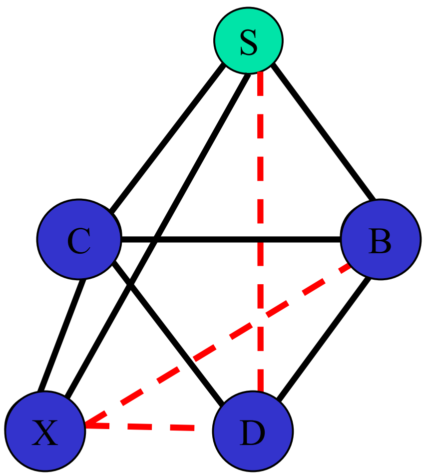
$D \subseteq X$: **decision variables**
$U(\bar{x})$: **utility function**

Belief Updating Task: Example



P (smoking| dyspnoea=yes) = ?

Belief updating: find P(X|evidence)



“Moral” graph

$$P(s|d=1) = \frac{P(s,d=1)}{P(d=1)} \propto P(s,d=1) =$$

$$\sum_{d=1,b,x,c} P(s)\underbrace{P(c|s)}P(b|s)\underbrace{P(x|c,s)P(d|c,b)} =$$

$$P(s)\sum_{d=1}\sum_{b} P(b|s)\sum_{x}\underbrace{\sum_{c} P(c|s)P(x|c,s)P(d|c,b)}_{f(\underbrace{s,d,b,x}_{W^*=4})}$$

Variable Elimination

"induced width"
(max induced clique size)

Complexity: **O(n exp(w*))**

Efficient inference: variable orderings, conditioning, approximations

Variable elimination algorithms

(also called "bucket-elimination")

Belief updating: VE-algorithm *elim-bel* (Dechter 1996)

$\sum_b \prod$ Elimination operator

bucket B: $P(b|a)$ $P(d|b,a)$ $P(e|b,c)$

bucket C: $P(c|a)$ $h^B(a, d, c, e)$

bucket D: $h^C(a, d, e)$

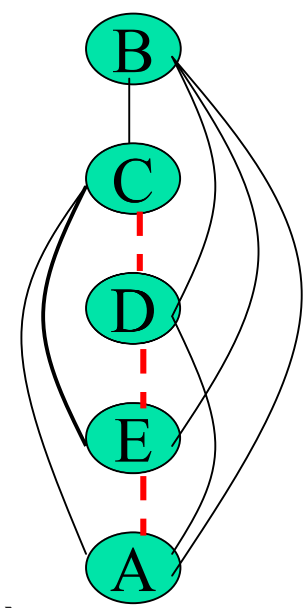
bucket E: $e=0$ $h^D(a, e)$

bucket A: $P(a)$ $h^E(a)$

$P(a|e=0)$

W*=4

"induced width"
(max clique size)



Finding $MPE = \max_{\bar{x}} P(\bar{x})$

VE-algorithm *elim-mpe* (Dechter 1996)

$\sum$ is replaced by ***max*** :

$$MPE = \max_{a,e,d,c,b} P(a)P(c \mid a)P(b \mid a)P(d \mid a,b)P(e \mid b,c)$$

$\max_b \prod$ ← Elimination operator

bucket B: $\overbrace{P(b|a) \quad P(d|b,a) \quad P(e|b,c)}$

bucket C: $P(c|a)$ $h^B(a, \underbrace{d, c, e)}$

bucket D: $h^C(a, d, e)$

bucket E: $e=0$ $h^D(a, e)$

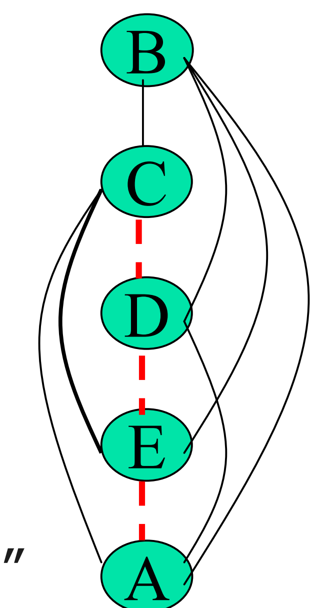
bucket A: $P(a)$ $h^E(a)$

MPE probability

W*=4

"induced width"

(max clique size)



Generating the MPE-solution

5. $b' = \arg\max_b P(b \mid a') \times P(d' \mid b, a') \times P(e' \mid b, c')$

4. $c' = \arg\max_c P(c \mid a') \times h^B(a', d', c, e')$

3. $d' = \arg\max_d h^C(a', d, e')$

2. $e' = 0$

1. $a' = \arg\max_a P(a) \cdot h^E(a)$

B: P(b|a) P(d|b,a) P(e|b,c)

C: P(c|a) $h^B(a, d, c, e)$

D: $h^C(a, d, e)$

E: e=0 $h^D(a, e)$

A: P(a) $h^E(a)$

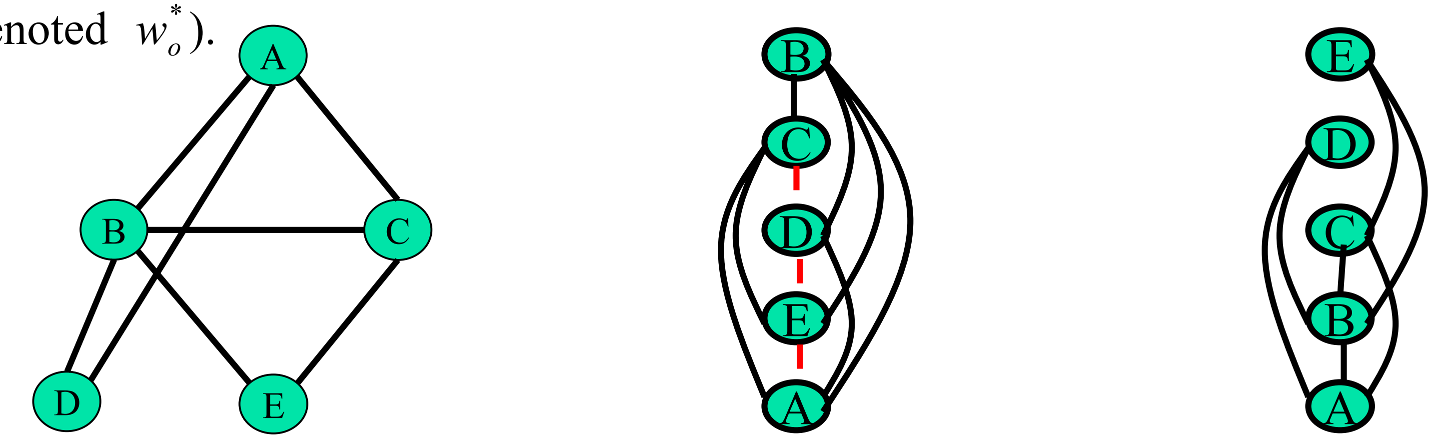
Return (a', b', c', d', e')

Complexity of VE-inference: $O(n\exp(w_o^*))$

w_o^* – the induced width of moral graph along ordering o

Meaning : $w_o^* + 1$ = size of largest clique created during inference

The width $w_o(X)$ of a variable X in graph G along the ordering o is the number of nodes preceding X in the ordering and connected to X (*earlier neighbors*).
The width w_o of a graph is the maximum width $w_o(X)$ among all nodes.
The *induced graph* G' along the ordering o is obtained by recursively connecting earlier neighbors of each node, from last to the first in the ordering.
The width of the induced graph G' is called the *induced width* of the graph G (denoted w_o^*).

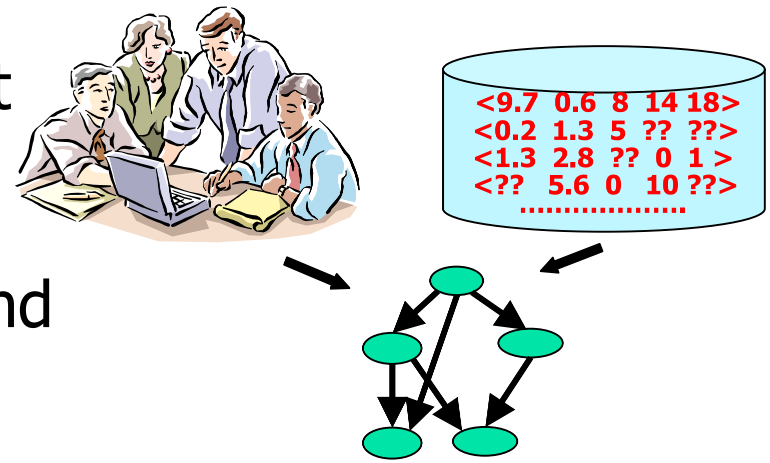


"Moral" graph $\quad w_{o_1}^* = 4 \quad w_{o_2}^* = 2$

Ordering is important! But finding min-w* ordering is NP- hard…
Inference is also NP-hard in general case [Cooper].

Learning Bayesian Networks

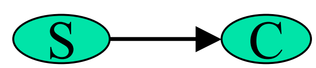
- Combining domain expert knowledge with data



- Efficient representation and inference

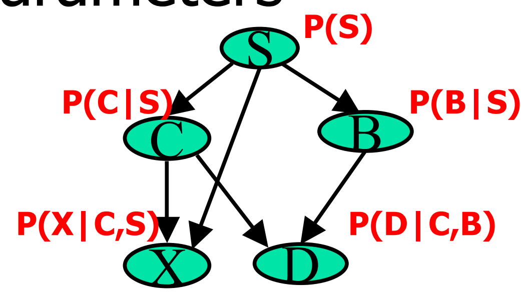
- Incremental learning: P(H) ↗ or ↘

- Handling missing data: **<1.3 2.8 ?? 0 1 >**

- Learning causal relationships: 

Learning tasks: four main cases

- Known graph – learn parameters



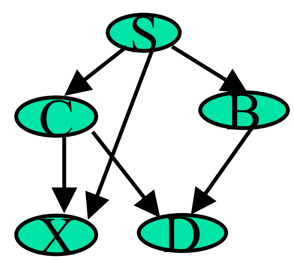
➢ Complete data:
  parameter estimation (ML, MAP)

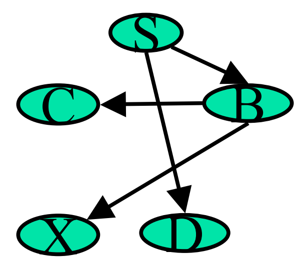
➢ Incomplete data:
  non-linear parametric
  optimization (gradient descent, EM)

- Unknown graph – learn graph and parameters

➢ Complete data:
  optimization (search
  in space of graphs)

➢ Incomplete data:
  structural EM,
  mixture models



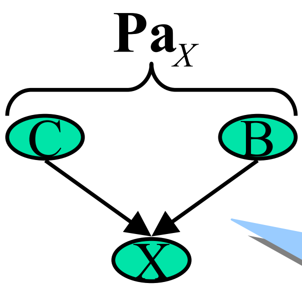


$$\hat{G} = \arg\max_{G} \mathrm{Score}(G)$$

Learning Parameters: complete data

(overview)

- ML-estimate: $\max_{\Theta} \underbrace{\log P(D \mid \Theta)}$ - decomposable!



Multinomial

$$\theta_{x,\mathbf{pa}_X} = P(x \mid \mathbf{pa}_X)$$

counts

$$\mathrm{ML}(\theta_{x,\mathbf{pa}_X}) = \frac{N_{x,\mathbf{pa}_X}}{\sum_x N_{x,\mathbf{pa}_X}}$$

- MAP-estimate (Bayesian statistics) $\max_{\Theta} \underbrace{\log P(D \mid \Theta) P(\Theta)}$

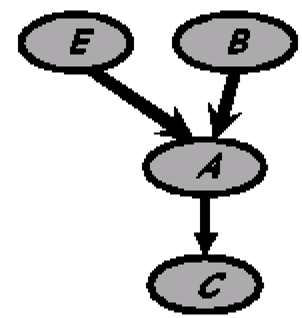
Conjugate priors - **Dirichlet** $Dir(\theta_{\mathbf{pa}_X} \mid \alpha_{1,\mathbf{pa}_X}, \ldots, \alpha_{m,\mathbf{pa}_X})$

$$\mathrm{MAP}(\theta_{x,\mathbf{pa}_X}) = \frac{N_{x,\mathbf{pa}_X} + \alpha_{x,\mathbf{pa}_X}}{\sum_x N_{x,\mathbf{pa}_X} + \underbrace{\sum_x \alpha_{x,\mathbf{pa}_X}}}$$

Equivalent sample size (prior knowledge)

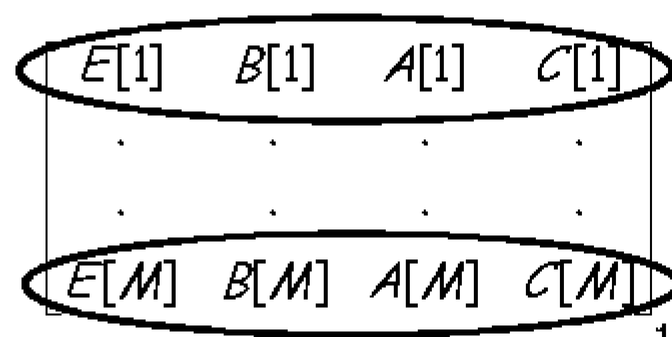
Likelihood Function

- By definition of network, we get



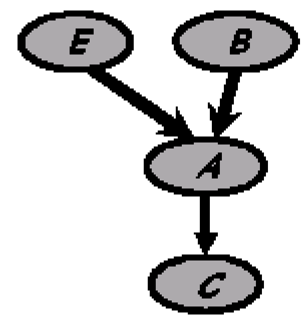
$$L(\Theta : D) = \prod_m P(E[m], B[m], A[m], C[m] : \Theta)$$

$$= \prod_m \left(\begin{array}{l} P(E[m] : \Theta) \\ P(B[m] : \Theta) \\ P(A[m] \mid B[m], E[m] : \Theta) \\ P(C[m] \mid A[m] : \Theta) \end{array} \right)$$



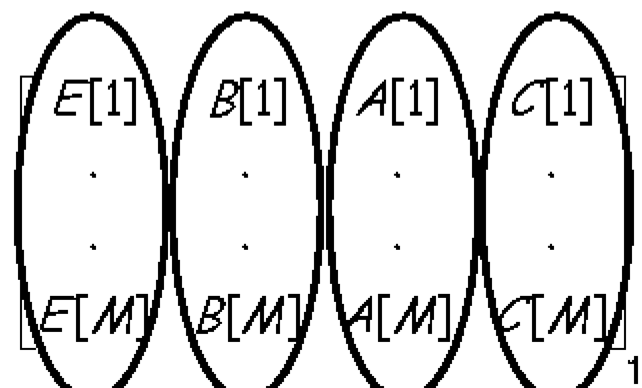
Likelihood Function

- Rewriting terms, we get



$$L(\Theta : D) = \prod_m P(E[m], B[m], A[m], C[m] : \Theta)$$

$$= \prod_m P(E[m] : \Theta) \prod_m P(B[m] : \Theta) \prod_m P(A[m] \mid B[m], E[m] : \Theta) \prod_m P(C[m] \mid A[m] : \Theta)$$



General Bayesian Networks

Generalizing for any Bayesian network:

$$
\begin{aligned}
L(\Theta : D) &= \prod_m P(x_1[m], \ldots, x_n[m] : \Theta) \\
&= \prod_i \prod_m P(x_i[m] \mid Pa_i[m] : \Theta_i) \\
&= \prod_i L_i(\Theta_i : D)
\end{aligned}
$$

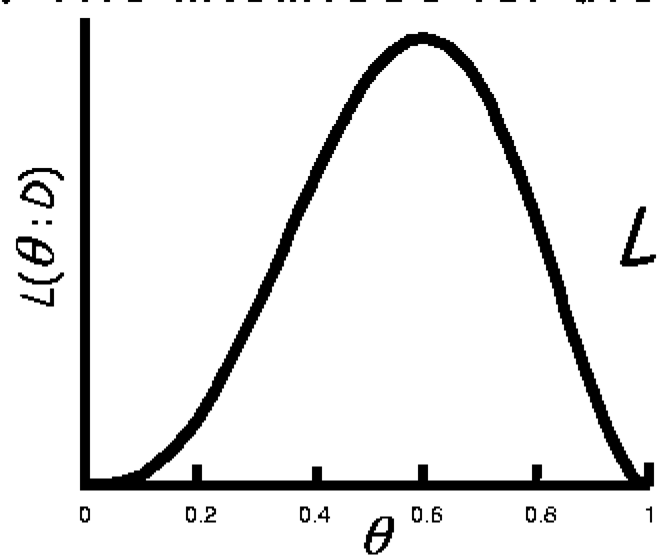
Decomposition

$\Rightarrow$ **Independent estimation problems**

Likelihood Function: Multinomials

$$L(\theta : D) = P(D \mid \theta) = \prod_m P(x[m] \mid \theta)$$

- The likelihood for the sequence H,T, T, H, H is



$$L(\theta : D) = \theta \cdot (1-\theta) \cdot (1-\theta) \cdot \theta \cdot \theta$$

General case: $$L(\Theta : D) = \prod_{k=1}^{K} \theta_k^{N_k}$$

Count of k^th outcome in D

Probability of k^th outcome

Dirichlet Priors

- Recall that the likelihood function is

$$L(\Theta : D) = \prod_{k=1}^{K} \theta_k^{N_k}$$

- **Dirichlet** prior with hyperparameters $\alpha_1, \ldots, \alpha_K$

$$P(\Theta) \propto \prod_{k=1}^{K} \theta_k^{\alpha_k - 1}$$

$\Rightarrow$ the posterior has the same form, with

hyperparameters $\alpha_1 + N_1, \ldots, \alpha_K + N_K$

$$P(\Theta \mid D) \propto P(\Theta) P(D \mid \Theta) \propto \prod_{k=1}^{K} \theta_k^{\alpha_k - 1} \prod_{k=1}^{K} \theta_k^{N_k} = \prod_{k=1}^{K} \theta_k^{\alpha_k + N_k - 1}$$

Learning Parameters: Summary

- Estimation relies on **sufficient statistics**
  - For multinomials: counts $N(x_i, pa_i)$
  - Parameter estimation

$$\hat{\theta}_{x_i|pa_i} = \frac{N(x_i, pa_i)}{N(pa_i)}$$

MLE

$$\tilde{\theta}_{x_i|pa_i} = \frac{\alpha(x_i, pa_i) + N(x_i, pa_i)}{\alpha(pa_i) + N(pa_i)}$$

Bayesian (Dirichlet)

- Both are asymptotically equivalent and consistent
- Both can be implemented in an on-line manner by accumulating sufficient statistics

Learning Parameters: incomplete data

Non-decomposable marginal likelihood (hidden nodes)

Initial parameters

Current model

$(\mathbf{G}, \boldsymbol{\Theta})$

Expectation

Compute EXPECTED
Counts via inference in BN

Data

S	X	D	C	B
<?	0	1	0	1>
<1	1	?	0	1>
<0	0	0	?	?>
<?	?	0	?	1>

………

Expected counts

$$E_{P(X)}[N_{x,\mathbf{pa}_X}] = \sum_{k=1}^{N} p(x, \mathrm{pa}_x \mid y^k, \Theta, G)$$

Maximization

Update parameters
(ML, MAP)

EM-algorithm:
iterate until convergence

Learning graph structure

Find $\hat{\mathbf{G}} = \arg\max_{G} \mathbf{Score(G)}$

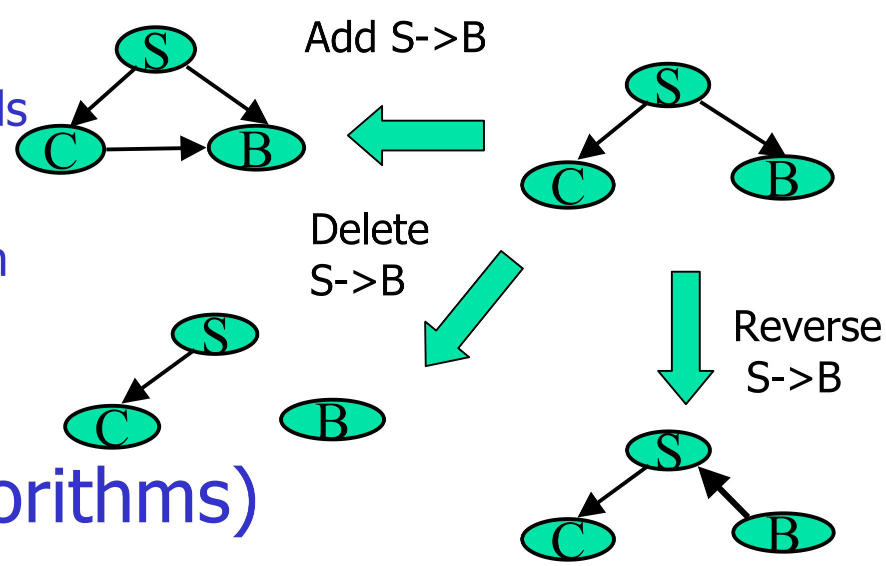
NP-hard optimization

- Heuristic search:

Complete data – local computations

Incomplete data (score non-decomposable):stochastic methods

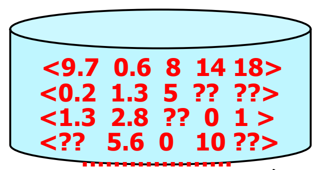
Local greedy search; K2 algorithm

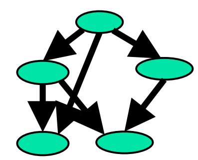


- Constrained-based methods (PC/IC algorithms)
  - Data impose independence relations (constraints) on graph structure

Scoring function:
Minimum Description Length (MDL)

- Learning ⇔ data compression



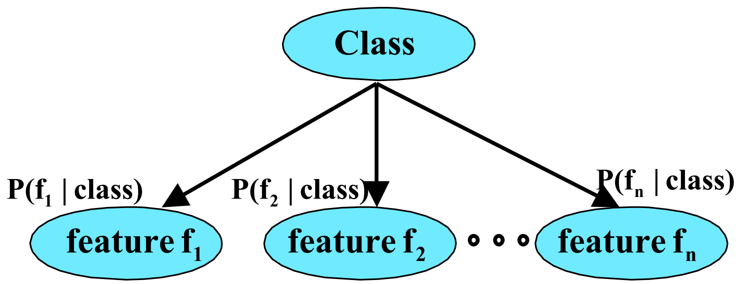


$$MDL(BN \mid D) = \underbrace{-\log P(D \mid \Theta, G)}_{\text{DL(Data|model)}} + \underbrace{\frac{\log N}{2} \mid \Theta \mid}_{\text{DL(Model)}}$$

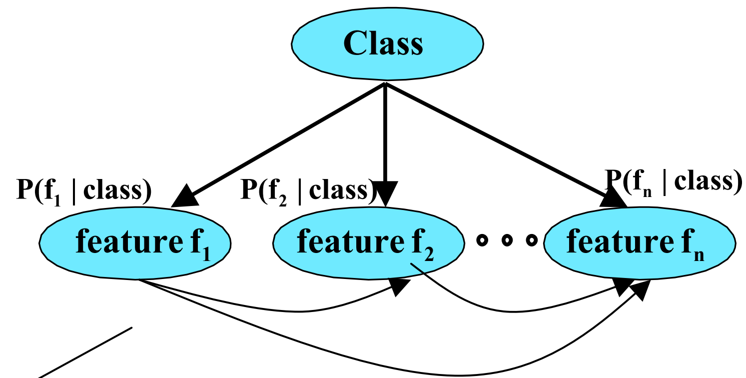
- Other: MDL = -BIC (Bayesian Information Criterion)
- Bayesian score (BDe) - asymptotically equivalent to MDL

Model selection trade-offs

Naïve Bayes – too simple
(less parameters, but bad model)



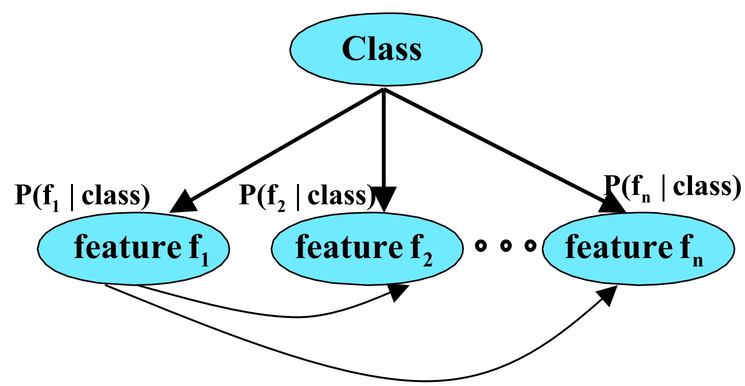
Unrestricted BN – too complex
(possible overfitting + complexity)



Various approximations between the two extremes

TAN:
tree-augmented Naïve Bayes
[Friedman et al. 1997]

Based on Chow-Liu Tree Method
(CL) for learning trees
[Chow-Liu, 1968]

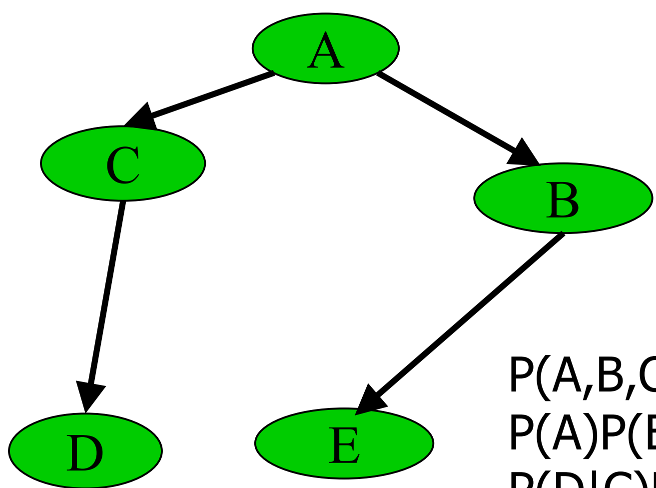


Tree-structured distributions

A joint probability distribution is tree-structured if it can be written as

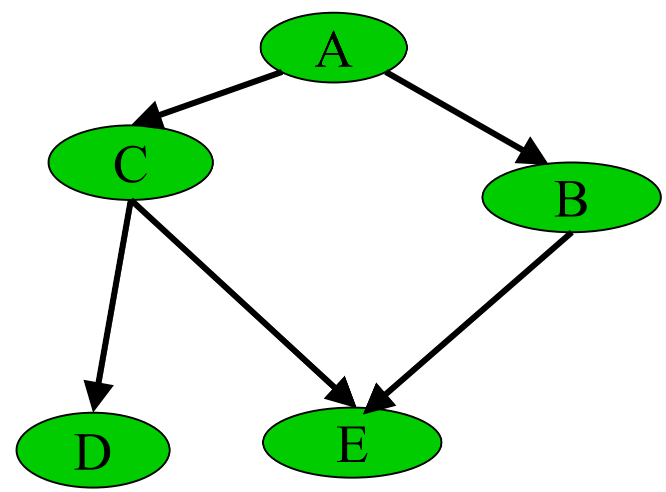
$$P(\mathrm{x}) = \prod_{i=1}^{n} P(x_i \mid x_{j(i)})$$

where $x_{j(i)}$ is the parent of x_i in Bayesian network for P(x) (a directed tree)



P(A,B,C,D,E)=
P(A)P(B|A)P(C|A)
P(D|C)P(E|B)

A tree (with root A)

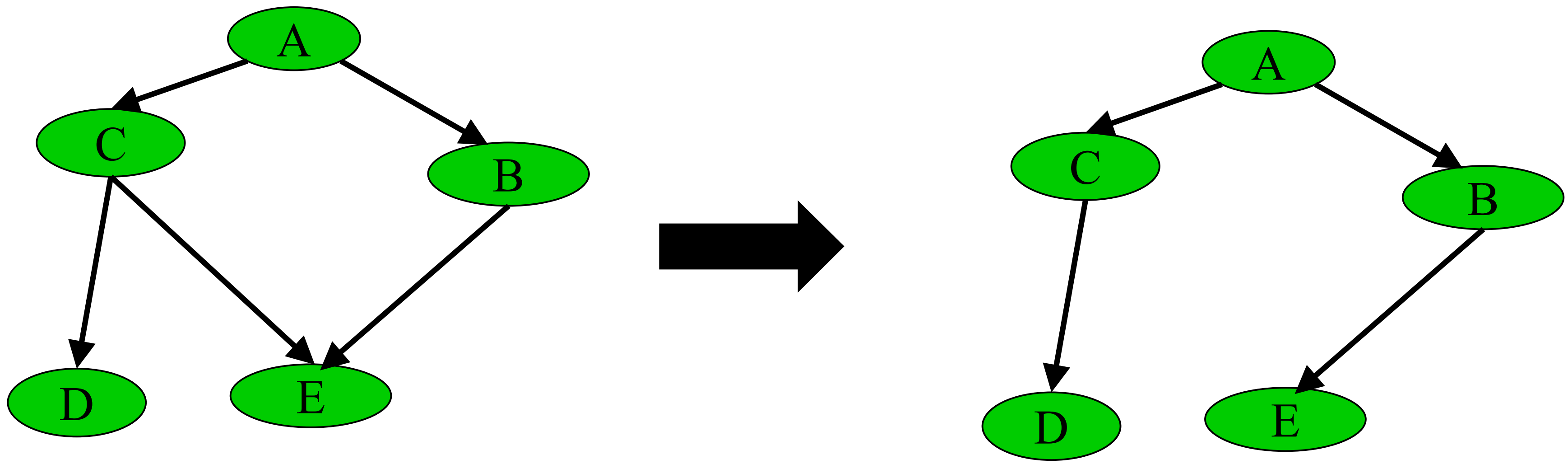


Not a tree – has an (undirected) cycle

A tree requires only **[(d-1) + d(d-1)(n-1)]** parameters, where d is domain size
Moreover, inference in trees is O(n) (linear) since their w*=1

Approximations by trees

True distribution P(X) Tree-approximation P'(X)



How good is approximation? Use cross-entropy (KL-divergence):

$$D(P,P') = P\sum_{\mathrm{x}} P(\mathrm{x})\log\frac{P(\mathrm{x})}{P'(\mathrm{x})}$$

D(P,P') is non-negative, and D(P,P')=0 if and only if P coincides with P' (on a set of measure 1)

How to find the best tree-approximation?

Optimal trees: Chow-Liu result

- **Lemma**

  Given a joint PDF P(x) and a fixed tree structure T, the best approximation P'(x) (i.e., P'(x) that minimizes D(P,P')) satisfies

$$P'(x_i \mid x_{j(i)}) = P(x_i \mid x_{j(i)}) \quad \text{for all } i = 1,\ldots,n$$

  Such P'(x) is called the projection of P(x) on T.

- **Theorem** [Chow and Liu, 1968]

  Given a joint PDF P(x), the KL-divergence D(P,P') is minimized by projecting P(x) on a *maximum-weight spanning tree (MSWT)* over nodes in X, where the weight on the edge (X_i, X_j) is defined by the mutual information measure

$$I(X_i; X_j) = \sum_{x_i, x_j} P(x_i, x_j) \log \frac{P(x_i, x_j)}{P(x_i)P(x_j)}$$

Note, that $I(X;Y) = 0$ when X and Y are independent
and that $I(X;Y) = D(P(x,y), P(x)P(y))$

Proofs

Proof of Lemma :

$$D(P,P') = -\sum_{\mathbf{x}} P(\mathbf{x}) \sum_{i=1}^{n} \log P'(x_i \mid x_{j(i)}) + \sum_{\mathbf{x}} P(\mathbf{x}) \log P(\mathbf{x}) = -\sum_{\mathbf{x}} P(\mathbf{x}) \sum_{i=1}^{n} \log P'(x_i \mid x_{j(i)}) - H(X) =$$

$$= -\sum_{i=1}^{n} \sum_{\mathbf{x}} P(\mathbf{x}) \log P'(x_i \mid x_{j(i)}) - H(X) = -\sum_{i=1}^{n} \sum_{x_i, x_{j(i)}} P(x_i, x_{j(i)}) \log P'(x_i \mid x_{j(i)}) - H(X) = \qquad (1)$$

$$= -\sum_{i=1}^{n} \sum_{x_{j(i)}} P(x_{j(i)}) \sum_{x_i} P(x_i \mid x_{j(i)}) \log P'(x_i \mid x_{j(i)}) - H(X) \qquad (2)$$

A known fact : given P(x), the maximum of $\sum_{x_i} P(x) \log P'(x)$ is achieved by the choice P'(x) = P(x).

Therefore, for any value of *i* and $x_{j(i)}$, the term $\sum_{x_i} P(x_i \mid x_{j(i)}) \log P'(x_i \mid x_{j(i)})$ is maximized by choosing $P'(x_i \mid x_{j(i)}) = P(x_i \mid x_{j(i)})$ (and thus the total $D(P,P')$ is minimized), which proves the Lemma.

Proof of Theorem :

Replacing $P'(x_i \mid x_{j(i)}) = P(x_i \mid x_{j(i)})$ in the expression (1) yields

$$D(P,P') = -\sum_{i=1}^{n} \sum_{x_i, x_{j(i)}} P(x_i, x_{j(i)}) \log[P(x_i x_{j(i)}) / P(x_{j(i)})] - H(X) =$$

$$= -\sum_{i=1}^{n} \sum_{x_i, x_{j(i)}} P(x_i, x_{j(i)}) \left[\log \frac{P(x_i x_{j(i)})}{P(x_{j(i)}) P(x_i)} + \log P(x_i) \right] - H(X) =$$

$$= -\sum_{i=1}^{n} I(X_i, X_{j(i)}) - \sum_{i=1}^{n} \sum_{x_i} P(x_i) \log P(x_i) - H(X).$$

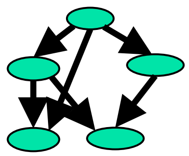
The last two terms are independent of the choice of the tree, and thus $D(P,P')$ is minimized by maximizing the sum of edge weights $I(X_i, X_{j(i)})$.

Chow-Liu algorithm
[As presented in Pearl, 1988]

1. From the given distribution P(x) (or from data generated by P(x)), compute the joint distribution $P(x_i \mid x_j)$ for all $i \neq j$
2. Using the pairwise distributions from step 1, compute the mutual information $I(X_i; X_j)$ for each pair of nodes and assign it as the weight to the corresponding edge (X_i, X_j).
3. Compute the maximum-weight spanning tree (MSWT):
   a. Start from the empty tree over n variables
   b. Insert the two largest-weight edges
   c. Find the next largest-weight edge and add it to the tree if no cycle is formed; otherwise, discard the edge and repeat this step.
   d. Repeat step (c) until n-1 edges have been selected (a tree is constructed).
4. Select an arbitrary root node, and direct the edges outwards from the root.
5. Tree approximation P'(x) can be computed as a projection of P(x) on the resulting directed tree (using the product-form of P'(x)).

Summary: Learning and inference in BNs

- Bayesian Networks – graphical probabilistic models
- Efficient representation and inference
- Expert knowledge + learning from data



- Learning:
  - parameters (parameter estimation, EM)
  - structure (optimization w/ score functions – e.g., MDL)
  - Complexity trade-off:
    - NB, BNs and trees
- There is much more: causality, modeling time (DBNs, HMMs), approximate inference, on-line learning, active learning, etc.

Online/print resources on BNs

Conferences & Journals

- UAI, ICML, AAAI, AISTAT, KDD
- MLJ, DM&KD, JAIR, IEEE KDD, IJAR, IEEE PAMI

Books and Papers

- Bayesian Networks without Tears by Eugene Charniak. AI Magazine: Winter 1991.
- Probabilistic Reasoning in Intelligent Systems by Judea Pearl. Morgan Kaufmann: 1998.
- Probabilistic Reasoning in Expert Systems by Richard Neapolitan. Wiley: 1990.
- CACM special issue on Real-world applications of BNs, March 1995

Online/Print Resources on BNs

AUAI online: www.auai.org. Links to:

- Electronic proceedings for UAI conferences
- Other sites with information on BNs and reasoning under uncertainty
- Several tutorials and important articles
- Research groups & companies working in this area
- Other societies, mailing lists and conferences

Publicly available s/w for BNs

List of BN software maintained by Russell Almond at bayes.stat.washington.edu/almond/belief.html

- several free packages: generally research only
- commercial packages: most powerful (& expensive) is HUGIN; others include Netica and Dxpress
- we are working on developing a Java based BN toolkit here at Watson