Learning BN tutorial:
ftp://ftp.research.microsoft.com/pub/tr/tr-95-06.pdf
TAN paper:
http://www.cs.huji.ac.il/~nir/Abstracts/FrGG1.html

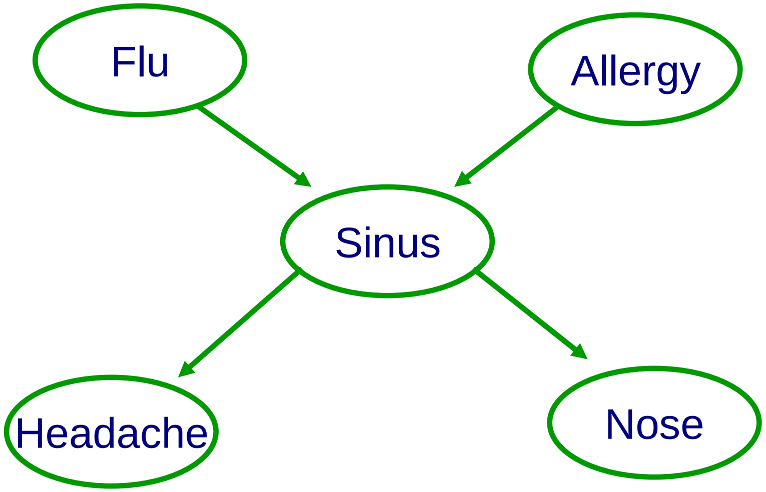
Bayesian Networks – Inference (continued) Learning

Machine Learning – 10701/15781
Carlos Guestrin
Carnegie Mellon University

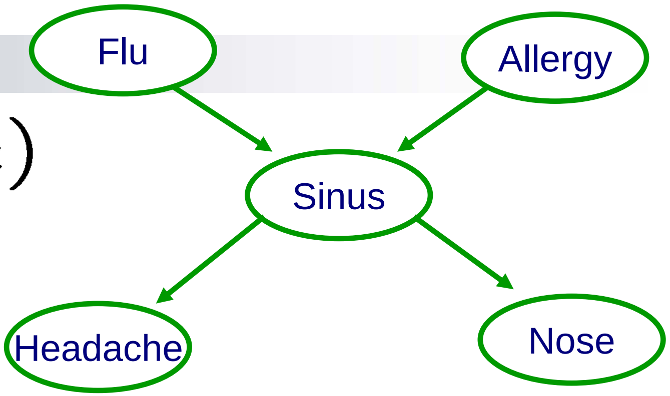
March 23rd, 2005

Review

- Bayesian Networks
  - Compact representation for probability distributions
  - Exponential reduction in number of parameters
- Fast probabilistic inference using variable elimination
  - Compute P(X|e) $P(F|N=t)$
  - Time exponential in tree-width, not number of variables
- Today
  - Finding most likely explanation
  - Using sampling for approximate inference
  - Learn BN structure



Most likely explanation (MLE)



- Query: $\underset{x_1,\ldots,x_n}{\text{argmax}}\, P(x_1,\ldots,x_n \mid e)$

  P(F, S, H, A | N=t)

- Using Bayes rule:

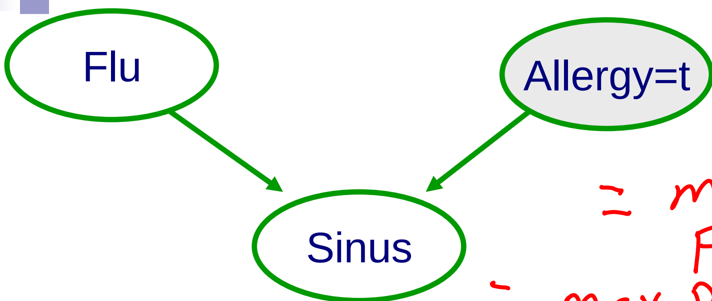
$$\underset{x_1,\ldots,x_n}{\text{argmax}}\, P(x_1,\ldots,x_n \mid e) = \underset{x_1,\ldots,x_n}{\text{argmax}} \frac{P(x_1,\ldots,x_n, e)}{P(e)}$$

- Normalization irrelevant:

$$\underset{x_1,\ldots,x_n}{\text{argmax}}\, P(x_1,\ldots,x_n \mid e) = \underset{x_1,\ldots,x_n}{\text{argmax}}\, P(x_1,\ldots,x_n, e)$$

P(F,S,H,A, N=t)

Max-marginalization



Forward pass:

$$\max_{F,S} P(F,S,A=t)$$

$$= \max_F \max_S P(F) \cdot P(A=t) \cdot P(S|F,A=t)$$

$$= \max_F P(F)\, P(A=t) \underbrace{\max_S P(S|F,A=t)}_{f_1(F)}$$

$$= \max_F P(F)\, P(A=t) \cdot f_1(F)$$

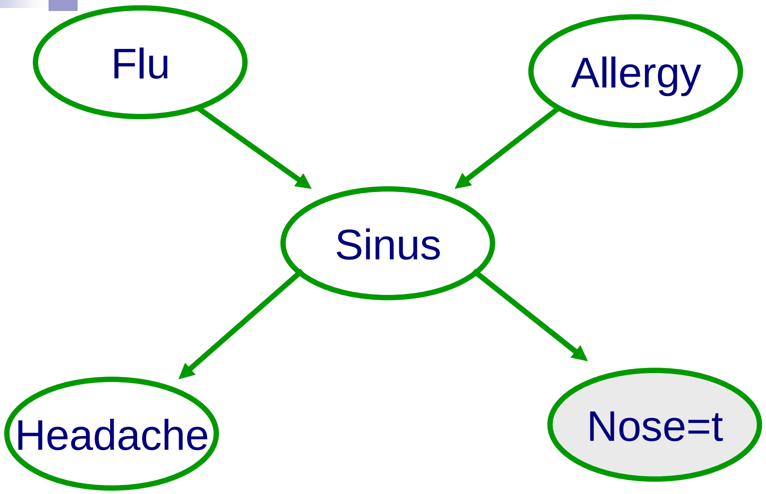
$= f_2() \leftarrow$ Single number – max prob. $P(F,S,A=t)$

Backwards pass:

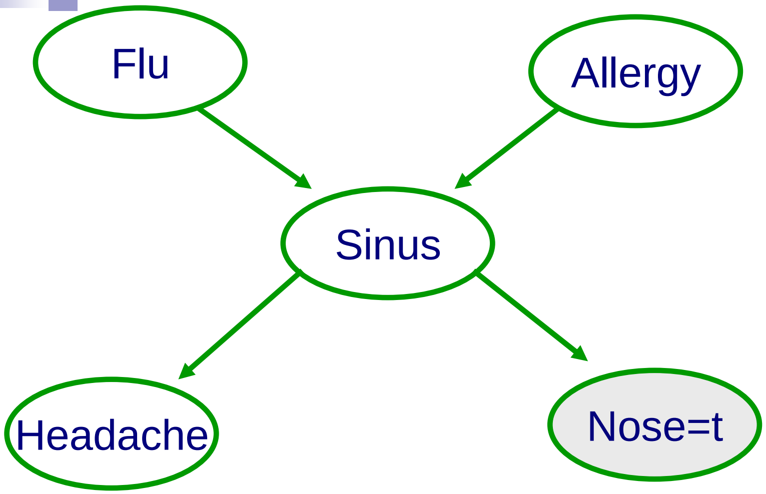
$$F^* = \arg\max_F P(F) \cdot P(A=t) \cdot f_1(F)$$

$$S^* = \arg\max_S P(S|F=F^*, A=t)$$

Example of variable elimination for MLE – Forward pass



Example of variable elimination for MLE – Backward pass



MLE Variable elimination algorithm – Forward pass

- Given a BN and a MLE query $\max_{x_1,\ldots,x_n} P(x_1,\ldots,x_n,e)$
- Instantiate evidence e
- Choose an ordering on variables, e.g., $X_1, \ldots, X_n$
- For i = 1 to n, If $X_i \notin \{e\}$
  - Collect factors $f_1,\ldots,f_k$ that include X_i
  - Generate a new factor by eliminating X_i from these factors

$$g = \max_{x_i} \prod_{j=1}^{k} f_j$$

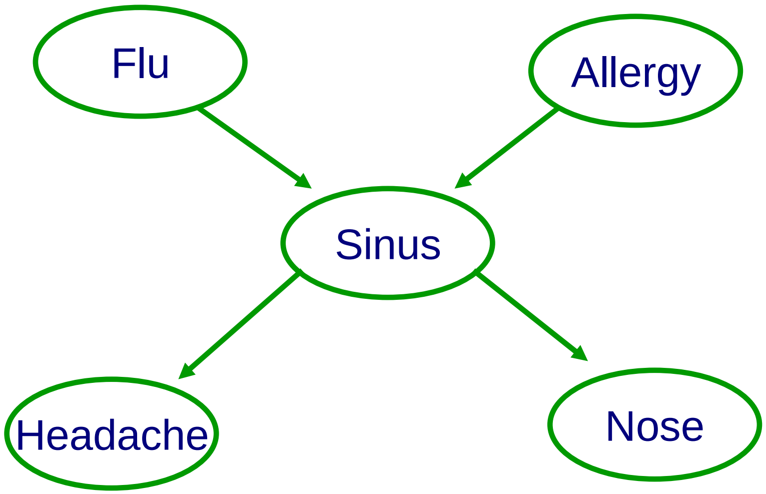
  - Variable X_i has been eliminated!

MLE Variable elimination algorithm – Backward pass

- $\{x_1^*, \ldots, x_n^*\}$ will store maximizing assignment
- For i = n to 1, If $X_i \notin \{e\}$
  opposite
  - Take factors $f_1, \ldots, f_k$ used when X_i was eliminated
  - Instantiate $f_1, \ldots, f_k$, with $\{x_{i+1}^*, \ldots, x_n^*\}$
    - Now each f_j depends only on X_i
  - Generate maximizing assignment for X_i:

$$x_i^* \in \underset{x_i}{\operatorname{argmax}} \prod_{j=1}^{k} f_j$$

Stochastic simulation – Obtaining a sample from the joint distribution



$P(\Theta)\begin{cases} H = 0.7 \\ T = 0.3 \end{cases}$ | Uniform sample $s \sim [0,1]$

$s > 0.3 \rightarrow H$

$s \leq 0.3 \rightarrow T$

$$P(F,A,S,H,N) = P(F).P(A)P(S|FA)\,P(H|S)\,P(N|S)$$

Start F, sample $\bar{F} \sim P(F)$

A, sample $\bar{A} \sim P(A)$

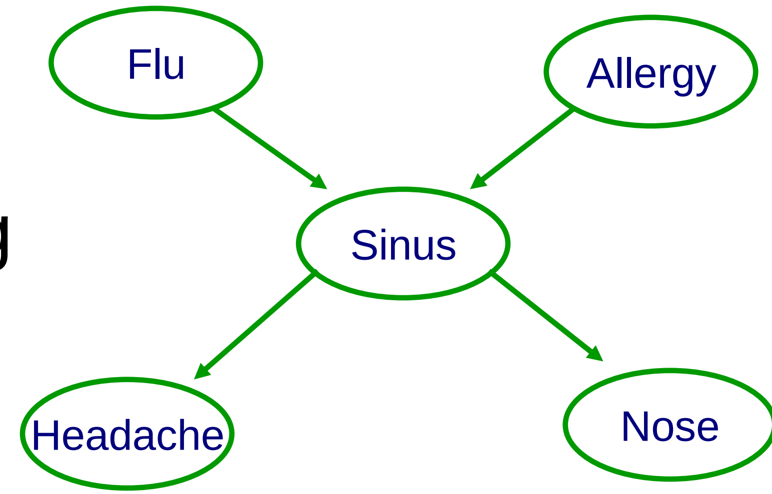
S, sample $\bar{S} \sim P(S | F=\bar{F}, A=\bar{A})$

H, sample $\bar{A} \sim P(H | S=\bar{S})$

N, sample $\bar{N} \sim P(N | N=\bar{S})$

Using stochastic simulation (sampling) to compute P(X)

- Given a BN, a query P(X), and number of samples *m*
- Choose a **topological** ordering on variables, e.g., $X_1, \ldots, X_n$
- For j = 1 to m
  - $\{x_1^j, \ldots, x_n^j\}$ will be j^{th} sample
  - For i = 1 to n
    - Sample x_i^j from the distribution $P(X_i|\mathbf{Pa}_{X_i})$, where parents are instantiated to $\{x_1^j, \ldots, x_{i-1}^j\}$
  - Add $\{x_1^j, \ldots, x_n^j\}$ to "dataset"
- Use counts to compute P(X)

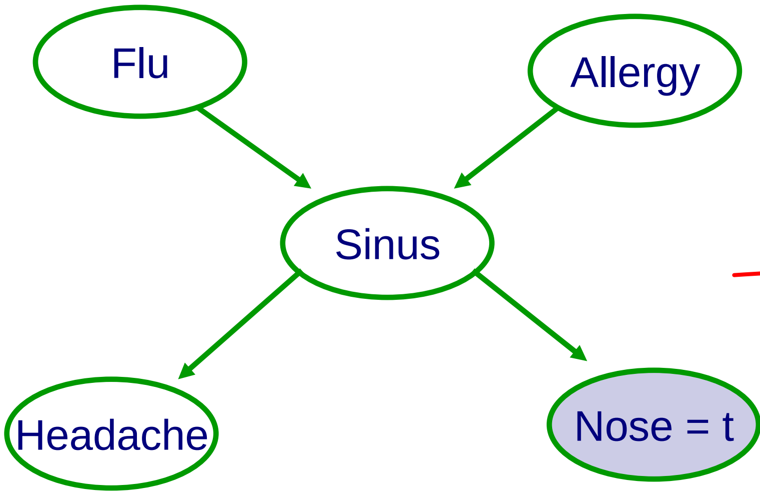


Complexity:

$m \;\; n \cdot C_{RAND}$

samples

Example of using rejection sampling to compute P(X|e)



$P(F \mid N=t)$

Many samples

$\langle F=t,\ A=t,\ S=f,\ H=t,\ N=t \rangle$

~~$\langle F=t,\ A=f,\ S=t,\ H=t,\ N=f \rangle$~~

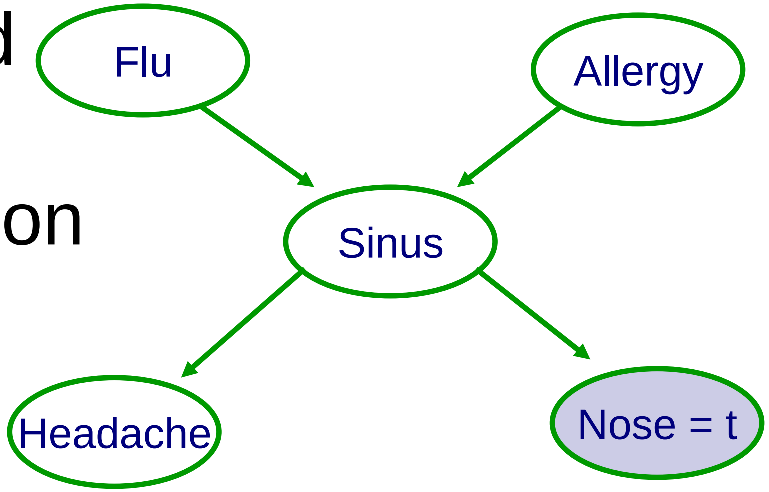
$\vdots$

$\langle F=f,\ A=t,\ S=f,\ H=t,\ N=t$

Reject samples that are not consistent with e

Using rejection sampling to compute P(X|e)

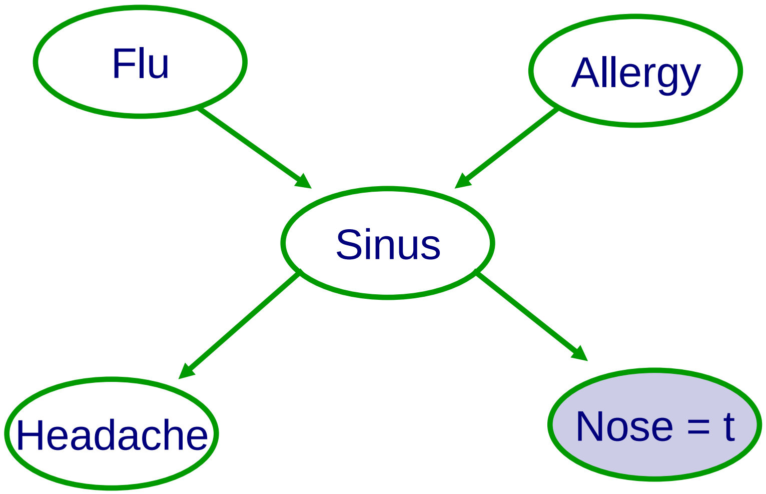
- Given a BN, a query P(X|e), and number of samples *m*
- Choose a **topological** ordering on variables, e.g., $X_1, \ldots, X_n$
- j = 0
- While j < m
  - $\{x_1^j, \ldots, x_n^j\}$ will be j^{th} sample
  - For i = 1 to n
    - Sample x_i^j from the distribution $P(X_i|\mathbf{Pa}_{X_i})$, where parents are instantiated to $\{x_1^j, \ldots, x_{i-1}^j\}$
  - If $\{x_1^j, \ldots, x_n^j\}$ consistent with evidence, add it to "dataset" and j = j +1
- Use counts to compute P(X|e)



Running time in expectation

$\frac{1}{P(e)} m \cdot n \; C_{RAND}$

Example of using importance sampling to compute P(X|e)



$\bar{F} \sim P(F)$

$\bar{A} \sim P(A)$

$\bar{S} \sim P(S|F=\bar{F}, A=\bar{A})$

$\bar{H} \sim P(H|S=\bar{S})$

when get to N

I don't sample!

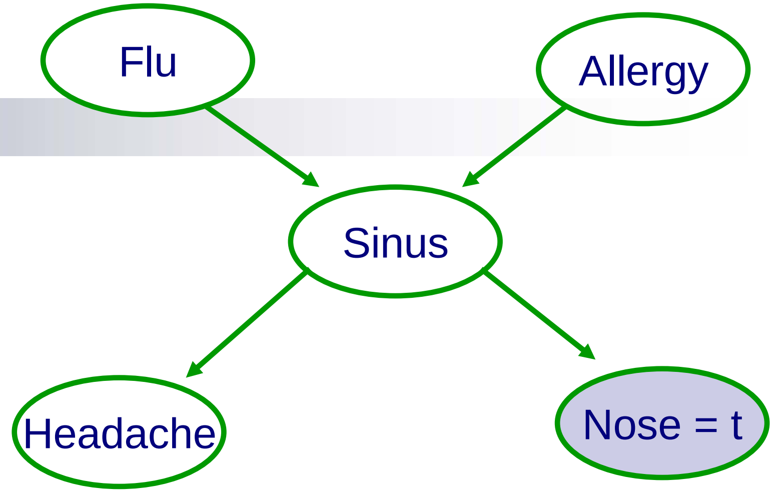
$\bar{N}=t$, but not sampled

$P(N|S=\bar{S})$

weigh this sample

$P(N=t|S=\bar{S})$

Using importance sampling to compute P(X|e)



- For j = 1 to m
  - $\{x_1^j, \ldots, x_n^j\}$ will be j^{th} sample
  - Initialize weight of sample $w^j = 1$
  - For i = 1 to n
    - If $X_i \notin \{e\}$ ← not evidence – Sample
      - Sample x_i^j from the distribution $P(X_i|\mathbf{Pa}_{X_i})$, where parents are instantiated to $\{x_1^j, \ldots, x_{i-1}^j\}$
    - else ← is evidence – don't sample
      - Set x_i^j to assignment in evidence e
      - Multiply weight w^j by $P(x_i^j|\mathbf{Pa}_{X_i})$, where parents are instantiated to $\{x_1^j, \ldots, x_{i-1}^j\}$ $P(X_i = e_i \mid Pa_{X_i})$
  - Add $\{x_1^j, \ldots, x_n^j\}$ to "dataset" with weight w^j
- Use weighted counts to compute P(X|e)

What you need to know about inference

- Bayesian networks
  - A useful compact **representation** for large probability distributions
- Inference to compute
  - Probability of X given evidence e
  - Most likely explanation (MLE) given evidence e
  - Inference is NP-hard
- Variable elimination algorithm
  - Efficient algorithm ("only" exponential in tree-width, not number of variables)
  - Elimination order is important!
  - Approximate inference necessary when tree-width to large
  - Only difference between probabilistic inference and MLE is "sum" versus "max"
- Sampling – Example of approximate inference There are alternatives
  - Simulate from model
  - Likelihood weighting for inference
  - Can be very slow , High variance

Where are we?

- Bayesian networks
  - Represent exponentially-large probability distributions compactly
- Inference in BNs
  - Exact inference very fast for problems with low tree-width
  - Many approximate inference algorithms, e.g., sampling
- Learning BNs
  - Given structure, estimate parameters
    - Using counts (maximum likelihood), MAP is also possible
  - What about learning structure?

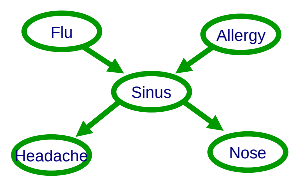
Maximum likelihood (ML) for learning BN structure

θ_{S_i}: parameters of S_i

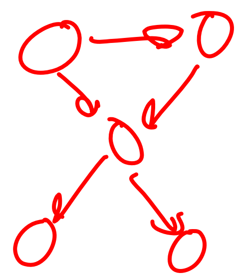
Possible structures

(Acyclic)

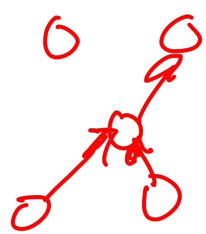
S_1

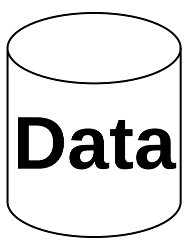


S_2



S_3





<x_1^{(1)},…,x_n^{(1)}>

…

<x_1^{(m)},…,x_n^{(m)}>

Learn parameters using ML

$P(\text{Data} \mid \text{model})$

$P(D \mid S_i, \theta_{S_i})$

$\overset{\text{i.id}}{=} \prod_j P(x^{(j)} \mid S_i, \theta_{S_i})$

lookup each prob. table & multiply

Score structure

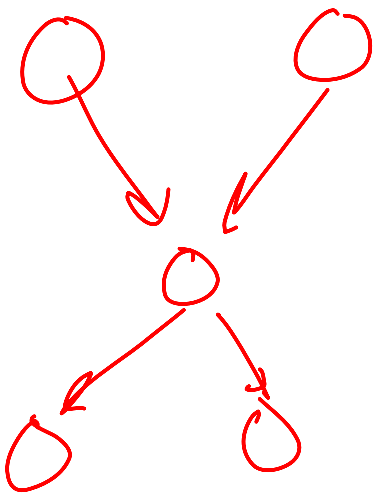
$P(D \mid S_1, \theta_{S_1})$

$P(D \mid S_2, \theta_{S_2})$

$P(D \mid S_3, \theta_{S_3})$

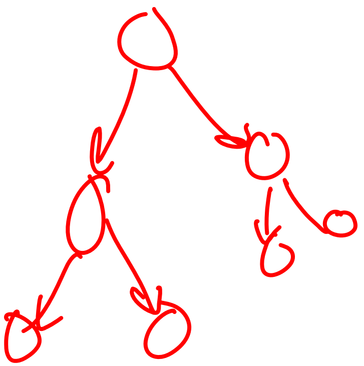
How many graphs are there?

doblely exponential!!



How many trees are there?

(~~doubly~~) exponential $n^n \Rightarrow$ maybe

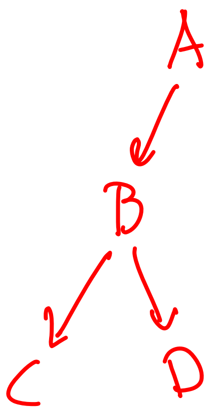


Nonetheless – Efficient optimal algorithm finds best tree

Scoring a tree

Entropy : $H(X) = \cancel{\sum \log}\ - \sum_{x} P(x) \log P(x)$

$$I(X,Y) = \sum_{xy} P(x,y) \log \frac{P(x,y)}{P(x).P(y)}$$



one data point : data $\langle a, b, c, d \rangle$

score : $\log P(a) . P(b|a) . P(c|b) . P(d|b)$

$= \log P(a) + \cancel{\log(P(b))} + \log P(b|a) +$

$\log P(c|b) + \log P(d|b)$

Data set: $\log \prod_{\langle a,b,c,d\rangle \in D} P(a) . P(b|a) P(c|b) P(d|b)$

$$\text{Score} = \sum_{\langle a,b,c,d \rangle \in D} \log P(a) + \log P(b|a) + \log P(c|b) + \log P(d|b)$$

$$= \sum_{i \in (1..n)} \sum_{\langle x_1, \dots x_n \rangle \in D} \log P(x_i | Pa_{x_i})$$

$$\frac{\text{Count}(x_i, Pa_{x_i})}{\text{Count}(Pa_{x_i})}$$

$$= \sum_{i=1}^{n} \sum_{x_i, Pa_{x_i}} \text{Count}(x_i, Pa_{x_i}) \log \underbrace{P(x_i | Pa_{x_i})}$$

Chow-Liu tree learning algorithm

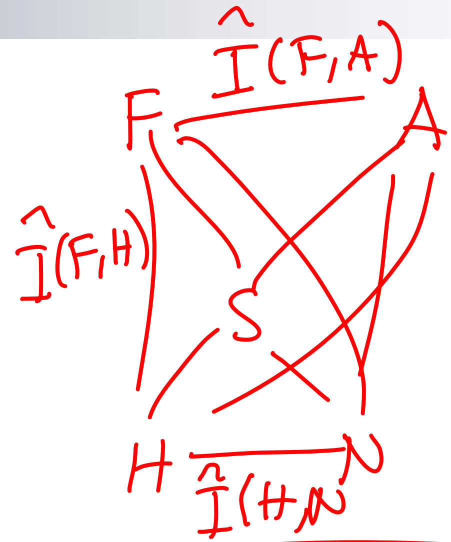
- For each pair of variables X_i, X_j
  - Compute empirical distribution:

$$\widehat{P}(x_i, x_j) = \frac{\text{Count}(x_i, x_j)}{m}$$

  - Compute mutual information:

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

- Define a graph
  - Nodes $X_1, \ldots, X_n$
  - Edge (i,j) gets weight $\widehat{I}(X_i, X_j)$
- Optimal tree BN
  - Compute maximal spanning tree MST
  - * Directions in BN: pick any node as root, breadth-first-search defines directions



MST : undirected tree

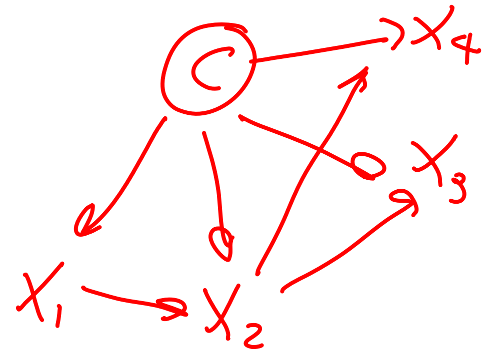
* pick any node as root

$A - B - C$ → $A \to B \to C$, $A \leftarrow B \to C$, $A \leftarrow B \leftarrow C$

Can we extend Chow-Liu

- Tree augmented naïve Bayes (TAN)
  - [Friedman et al. '97]
  - Same as Chow-Liu, but score edges with:

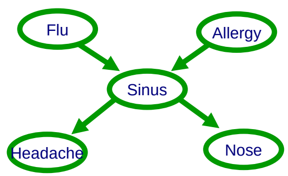
$$\hat{I}(X_i, X_j \mid C) = \sum_{c,x_i,x_j} \hat{P}(c, x_i, x_j) \log \frac{\hat{P}(x_i, x_j \mid c)}{\hat{P}(x_i \mid c)\hat{P}(x_j \mid c)}$$

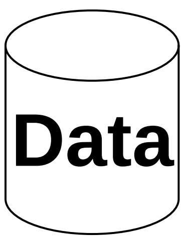


- (Approximate learning) models with tree-width up to *k*
  - [Narasimhan & Bilmes '04]
  - But, $O(n^{k+1})$…

Scoring general graphical models – Model selection problem

What's the best structure?





<x_1^{(1)},…,x_n^{(1)}>

…

<x_m^{(m)},…,x_n^{(m)}>

Max Likelihood Structure: Fully-connected

The more edges, the fewer independence assumptions, the higher the likelihood of the data, but will overfit…

Bayesian score

- Given a structure, distribution over parameters

$$\log P(D \mid S) = \log \int_{\theta_S} P(D \mid S, \theta_S) P(\theta_S | S) d\theta_S$$

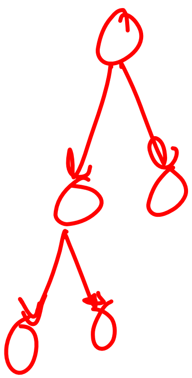
- Difficult integral, use Bayes information criterion (BIC) approximation

$$\underbrace{\log P(D \mid S)}_{\text{true score}} \approx \underbrace{\log P(D \mid S, \theta_S)}_{\text{log likelihood}} - \underbrace{\frac{\text{NumberParameters}(S)}{2} \log m}_{\text{penalty}}$$

- Note: regularize with MDL score
- Best BN under BIC still NP-hard

Learn BN structure using local search

Starting from Chow-Liu tree



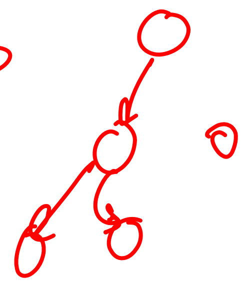
Local search, possible moves:

- Add edge
- Delete edge
- Invert edge

Score using BIC

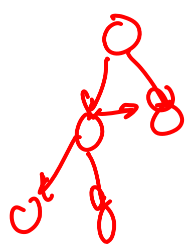
log likelihood − penalty complexity model

delete



test for acyclicity

add edge



iterate

local minimum

What you need to know about learning BNs

- Bayesian networks
  - A useful compact **representation** for large probability distributions
- Inference
  - Variable elimination algorithm
  - Sampling – Example of approximate inference
- Learning BNs
  - Maximum likelihood or MAP learns parameters
  - Best tree (Chow-Liu)
  - Best TAN
  - Other BNs, usually local search with BIC score

Acknowledgements

- JavaBayes applet
  - http://www.pmr.poli.usp.br/ltd/Software/javabayes/Home/index.html