Readings:

K&F: 15.1, 15.2, 15.3, 15.4, 15.5

K&F: 7 (overview of inference)

K&F: 8.1, 8.2 (Variable Elimination)

Structure Learning in BNs 3:

(the good, the bad,) and, finally, the ugly

Inference

we now get to use the BNs!!

Graphical Models – 10708

Carlos Guestrin

Carnegie Mellon University

October 6th, 2006

Bayesian, a decomposable score

$$\log P(D \mid \mathcal{G}) = \log \int_{\theta_{\mathcal{G}}} P(D \mid \mathcal{G}, \theta_{\mathcal{G}}) P(\theta_{\mathcal{G}} \mid \mathcal{G}) d\theta_{\mathcal{G}}$$

- As with last lecture, assume:
  - Local and global parameter independence

$\theta_{Y|X} \perp \theta_X$

- Also, prior satisfies **parameter modularity**:
  - If X_i has same parents in G and G', then parameters have same prior

$\theta_{X_i|Pa_{X_i}}$

- Finally, structure prior P(*G*) satisfies **structure modularity**
  - Product of terms over families

$P(G) \propto \prod_i P(E(X_i, Pa_{X_i}))$

  - E.g., P(*G*) $\propto$ c$^{|G|}$

$|G|$ = number edges

$c < 1$

like fewer edges

- Bayesian score decomposes along families!

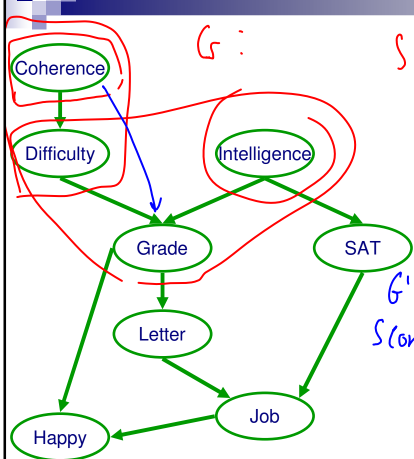
$\log P(G \mid D) \propto \sum_i \text{ScoreFam}(X_i \mid Pa_{X_i,G} : D)$

Structure learning for general graphs

- In a tree, a node only has one parent
- **Theorem**:
  - The problem of learning a BN structure with at most *d* parents is NP-hard for any (fixed) $d \geq 2$
- Most structure learning approaches use heuristics
  - Exploit score decomposition
  - (Quickly) Describe two heuristics that exploit decomposition in different ways

Understanding score decomposition

Score Fam = SF



G :

$$Score(G:D) = \sum_i ScoreFam(X_i \mid Pa_{X_i,G} : D)$$

$$= SF(C \mid \emptyset : D) + SF(D \mid C : D) + SF(I \mid \emptyset : D) + SF(G \mid D, I : D) + \ldots$$

G' Suppose C becomes parent of G

$$Score(G:D) - Score(G':D)$$

$$= SF(G \mid D, I : D) - SF(G \mid D, I, C : D)$$

Fixed variable order 1

F, A, S, N, H

- Pick a variable order $\prec$
  - e.g., $X_1,\ldots,X_n$
- X_i can only pick parents in $\{X_1,\ldots,X_{i-1}\}$
  - Any subset
  - Acyclicity guaranteed!
- Total score = sum score of each node

G: X_i picked parents Pa_{X_i}

$Score(G:D) = \sum_i SF(X_i \mid Pa_{X_i} : D)$

DAG: 1…i-1

$X_1 \to X_2$, X_3, X_4

selecting parents of X_5 any subset of $X_1 \ldots X_4$ ⇒ still have a DAG

max choice parents foreach X_i $Score(G:D)$ ⇒ indep. choice for each node is OPTIMAL!! :)

Fixed variable order 2

FASHN good order

HFANS bad order

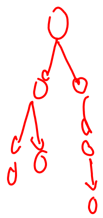
- Fix max number of parents to k $2^{k-1} \le \sum_{j=1}^{k} \binom{i-1}{j} \le 2^{i-1}-1$
- For each *i* in order $\prec$
  - Pick $\mathbf{Pa}_{X_i} \subseteq \{X_1,\ldots,X_{i-1}\}$ complexity =
    - Exhaustively search through all possible subsets
    - $\mathbf{Pa}_{X_i}$ is maximum $\mathbf{U} \subseteq \{X_1,\ldots,X_{i-1}\}$ FamScore(X_i|$\mathbf{U}$: D)
- Optimal BN for each order!!!
- Greedy search through space of orders:
  - E.g., try switching pairs of variables in order
  - If neighboring vars in order are switched, only need to recompute score for this pair
    - O(n) speed up per iteration

FASHN

FSAHN

Learn BN structure using local search

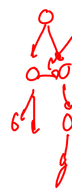
Starting from Chow-Liu tree



Local search,
possible moves:
Only if acyclic!!!
- Add edge
- Delete edge
- Invert edge

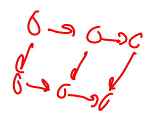
Computed locally / efficiently because of score decomposition Fam Score

Select using favorite score

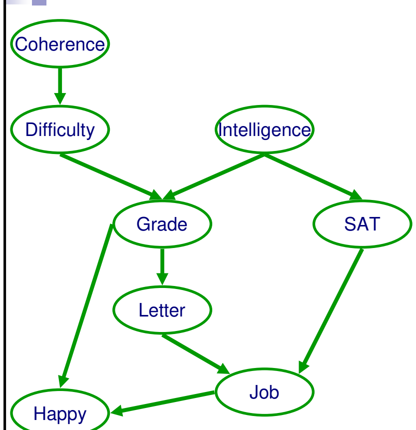


add edge

delete
add
invert



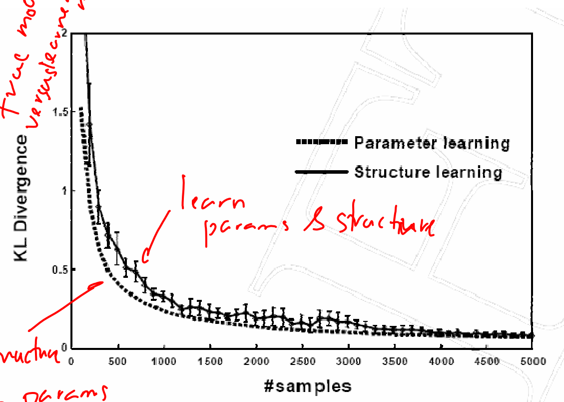
Exploit score decomposition in local search



- Add edge and delete edge:
  - Only rescore one family!
- Reverse edge
  - Rescore only two families

Some experiments

true model versus learned model



learn params & structure

true structure learn params

Alarm network → generated samples

Order search versus graph search

- Order search advantages ~~random~~
  - For fixed order, optimal BN – more "global" optimization
  - Space of orders much smaller than space of graphs

    $n!$ $\Omega(2^{n^2})$
- Graph search advantages
  - Not restricted to k parents
    - Especially if exploiting CPD structure, such as CSI
  - Cheaper per iteration
  - Finer moves within a graph

Bayesian model averaging

- So far, we have selected a single structure
- But, if you are really Bayesian, must average over structures
  - Similar to averaging over parameters

$$\log P(D \mid \mathcal{G}) = \log \int_{\theta_{\mathcal{G}}} P(D \mid \mathcal{G}, \theta_{\mathcal{G}}) P(\theta_{\mathcal{G}} | \mathcal{G}) d\theta_{\mathcal{G}}$$

$P(G|D) \rightarrow$ probability for each graph

- Inference for structure averaging is very hard!!!
  - Clever tricks in reading

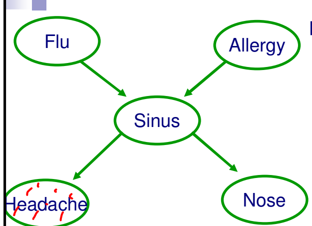
What you need to know about learning BN structures

- Decomposable scores
  - Data likelihood
  - Information theoretic interpretation
  - Bayesian
  - BIC approximation
- Priors
  - Structure and parameter assumptions
  - BDe if and only if score equivalence
- Best tree (Chow-Liu)
- Best TAN
- Nearly best k-treewidth (in $O(N^{k+1})$)
- Search techniques
  - Search through orders
  - Search through structures
- Bayesian model averaging

Announcements

- Don't forget project proposals due this Wednesday
- Special recitation on advanced topic:
  - Ajit Singh on Optimal Structure Learning
  - On Monday Oct 9, 5:30-7:00pm in Wean Hall 4615A

Inference in graphical models: Typical queries 1

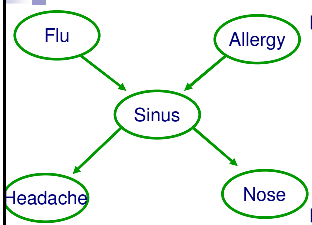


- Conditional probabilities
  - Distribution of some var(s). given evidence

evidence H=t

$P(F \mid e) = P(F \mid H=t)$

Inference in graphical models: Typical queries 2 – Maximization



- Most probable explanation (MPE)
  - Most likely assignment to all hidden vars given evidence $H=t$

lots of people call MPE & MAP query

$$\arg\max_{f,a,s,n} P(F=f, A=a, S=s, N=n \mid H=t)$$

- Maximum a posteriori (MAP)
  - Most likely assignment to some var(s) given evidence

$$\arg\max_{f} P(F=f \mid H=t)$$

Are MPE and MAP Consistent?

P(S=t)=0.4
P(S=f)=0.6

P(N|S) =

S \ N	t	f
t	.9	.1
f	.5	.5

- Most probable explanation (MPE)
  - Most likely assignment to all hidden vars given evidence

$$\arg\max_{s,n} P(S=s, N=n) = (S=t, N=t)$$

$$P(S=s, N=n) = P(S=s) \cdot P(N=n \mid S=s)$$

don't agree

- Maximum a posteriori (MAP)
  - Most likely assignment to some var(s) given evidence

$$\arg\max_{s} P(s) = (S=f)$$

Complexity of conditional probability queries 1

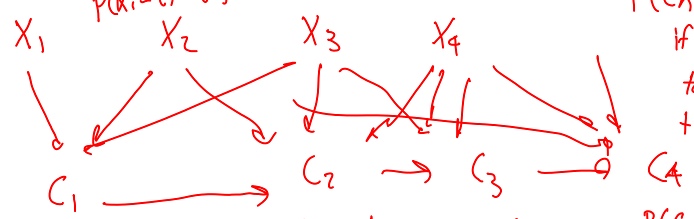
- How hard is it to compute P(X|**E**=**e**)? NP-hard

Reduction – 3-SAT

C_1 C_2 n-clauses

$$(\bar{X}_1 \vee X_2 \vee X_3) \wedge (\bar{X}_2 \vee X_3 \vee X_4) \wedge \ldots$$

$p(x_i = t) = 0.5$



$P(C_n = t) > 0$ if $\exists$ assignment to $X_1 \ldots X_n$ that satisfies all clause

$P(C_n = t) = 0$ if $\nexists$ assignments

$P(C_1 = t \mid X_1, X_2, X_3)$ zero if x_1, x_2, x_3 don't satisfy clause

$P(C_2 = t \mid X_2, X_3, X_4, C_1)$ zero if $X_2 \ldots X_4$ don't satisfy clause or C_1 not true

Complexity of conditional probability queries 2

- How hard is it to compute P(X|**E**=**e**)? #P complete
  - At least NP-hard, but even harder!

$P(C_n = t) = \alpha$

have m literals $X_1 \ldots X_m$

$\text{prob}(X_1 = t, X_2 = f, \ldots X_m = t) = \frac{1}{2^m}$

each satisfying assignment gives $C_n = t$ $\frac{1}{2^m}$ prob

if $P(C_n = t) = \alpha$ $\Rightarrow$ 3-SAT has $\alpha 2^m$ satisfying assignments

$\Rightarrow$ count # sat. assignments

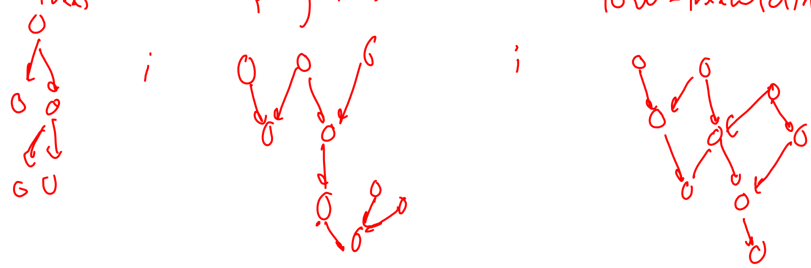
$\Rightarrow$ #P-hard

Inference is #P-~~hard~~ Complete, hopeless?

- Exploit structure!
- Inference is hard in general, but easy for many (real-world relevant) BN structures

structure "simple" ⇒ inference "easy"

trees ; poly-trees ; low-treewidth graphs



Complexity for other inference questions

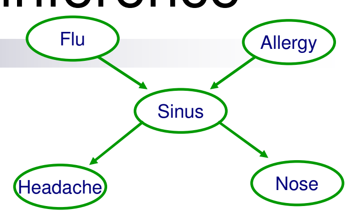
- Probabilistic inference
  - general graphs: #P-complete
  - poly-trees and low tree-width: easy
- Approximate probabilistic inference within $\varepsilon > 0$
  - Absolute error: $|\hat{P} - P| \leq \varepsilon$
  - Relative error: $\frac{|\hat{P} - P|}{|P|} \leq \varepsilon$

  } NP-hard ← $\varepsilon < 0.5$; ← any $\varepsilon > 0$
- Most probable explanation (MPE)
  - general graphs: NP-complete
  - poly-trees and low tree-width: easy
- Maximum a posteriori (MAP)
  - general graphs: NP^{PP}-complete
  - poly-trees and low tree-width: NP-hard

Inference in BNs hopeless?

- In general, yes!
  - Even approximate!
- In practice
  - Exploit structure
  - Many effective approximation algorithms (some with guarantees)
- For now, we'll talk about exact inference
  - Approximate inference later this semester

General probabilistic inference



- Query: $P(X \mid e)$
- Using def. of cond. prob.:

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

- Normalization:

$$P(X \mid e) \propto P(X, e)$$

$P(X=t, e) = 0.4$

$P(X=f, e) = 0.1$

normalize

$P(X=t \mid e) = \frac{0.4}{0.5} = 0.8$

Marginalization

I Know

$P(F,S,N) = P(F) \cdot P(S|F) \cdot P(N|S)$

$P(S=t \mid N=t)$

first focus on $P(S=t, N=t)$, $P(S=f, N=t)$

$P(S, N=t)$

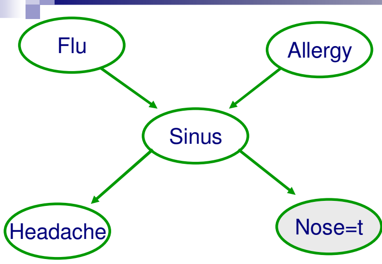
computing table for each value of S

marginalization

$$P(S, N=t) = \sum_f P(F=f, S, N=t)$$

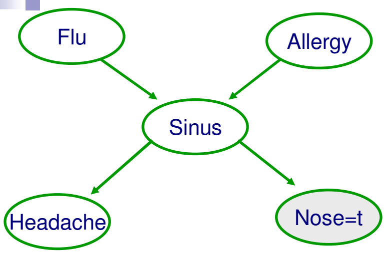
$$= \sum_f P(F=f) \cdot P(S \mid F=f) \cdot P(N=t \mid S)$$

Probabilistic inference example



Inference seems exponential in number of variables!

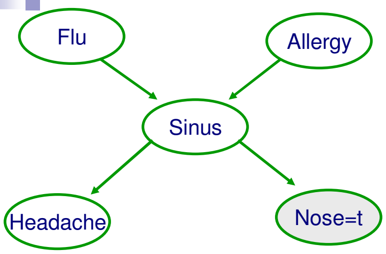
Fast probabilistic inference example – Variable elimination



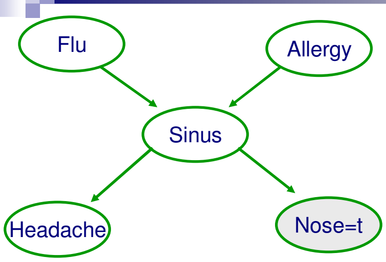
(Potential for) Exponential reduction in computation!

Understanding variable elimination – Exploiting distributivity

Understanding variable elimination – Order can make a HUGE difference

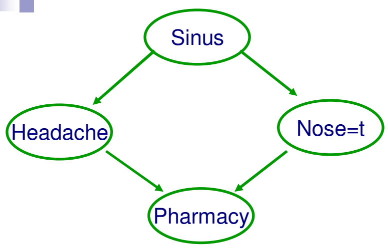


Understanding variable elimination – Intermediate results

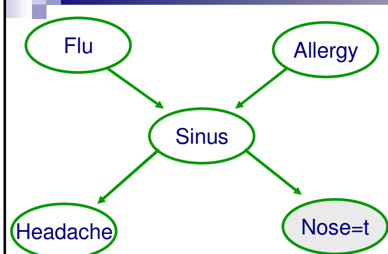


Intermediate results are probability distributions

Understanding variable elimination – Another example



Pruning irrelevant variables



Prune all non-ancestors of query variables
More generally: Prune all nodes not on active trail between evidence and query vars

Variable elimination algorithm

- Given a BN and a query $P(X|e) \propto P(X,e)$
- Instantiate evidence **e**
- Prune non-active vars for {X,**e**}

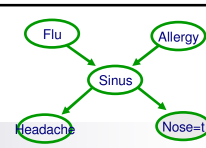
IMPORTANT!!!

- Choose an ordering on variables, e.g., $X_1, \ldots, X_n$
- Initial *factors* $\{f_1,\ldots,f_n\}$: $f_i = P(X_i|\mathbf{Pa}_{X_i})$ (CPT for X_i)
- For i = 1 to n, If $X_i \notin \{X,\mathbf{E}\}$
  - Collect factors $f_1,\ldots,f_k$ that include X_i
  - Generate a new factor by eliminating X_i from these factors

$$g = \sum_{X_i} \prod_{j=1}^{k} f_j$$

  - Variable X_i has been eliminated!
- Normalize P(X,**e**) to obtain P(X|**e**)

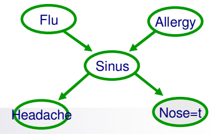
Operations on factors



$$g = \sum_{X_i} \prod_{j=1}^{k} f_j$$

Multiplication:

Operations on factors



$$g = \sum_{X_i} \prod_{j=1}^{k} f_j$$

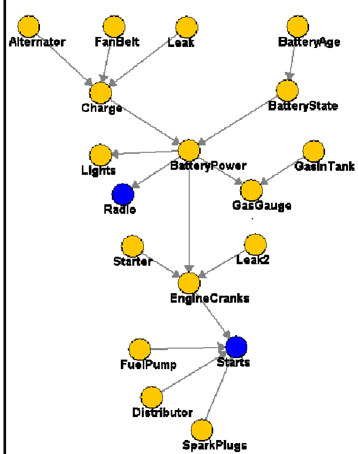
Marginalization:

Complexity of VE – First analysis

- Number of multiplications:

- Number of additions:

Complexity of variable elimination – (Poly)-tree graphs



Variable elimination order:
Start from "leaves" inwards:
- Start from skeleton!
- Choose a "root", any node
- Find topological order for root
- Eliminate variables in reverse order

Linear in CPT sizes!!! (versus exponential)

What you need to know about inference thus far

- Types of queries
  - probabilistic inference
  - most probable explanation (MPE)
  - maximum a posteriori (MAP)
    - MPE and MAP are truly different (don't give the same answer)
- Hardness of inference
  - Exact and approximate inference are NP-hard
  - MPE is NP-complete
  - MAP is much harder (NP^{PP}-complete)
- Variable elimination algorithm
  - Eliminate a variable:
    - Combine factors that include this var into single factor
    - Marginalize var from new factor
  - Efficient algorithm ("only" exponential in induced-width, not number of variables)
    - If you hear: "Exact inference only efficient in tree graphical models"
    - You say: "No!!! Any graph with low induced width"
    - And then you say: "And even some with very large induced-width" (next week with context-specific independence)
- Elimination order is important!
  - NP-complete problem
  - Many good heuristics