Readings:
K&F: 17.3, 17.4, 17.5.1, 8.1, 12.1

Structure Learning

finally

(The Good), The Bad, The Ugly

Inference

Graphical Models – 10708

Carlos Guestrin

Carnegie Mellon University

October 13th, 2008

Decomposable score

- Log data likelihood

$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = m \sum_i \hat{I}(X_i, \mathbf{Pa}_{X_i}) - m \sum_i \hat{H}(X_i)$$

- Decomposable score:
  - Decomposes over families in BN (node and its parents)
  - Will lead to significant computational efficiency!!!
  - Score($G : D$) = $\sum_i$ FamScore(X_i|$\mathbf{Pa}_{X_i}$: D)

for MLE FamScore$(X_i \mid Pa_{X_i} : D) = m\hat{I}(X_i; Pa_{X_i}) - m\hat{H}(X_i)$

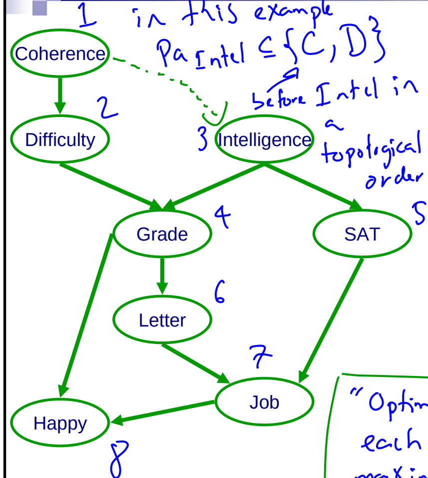
Structure learning for general graphs

- In a tree, a node only has one parent

  Chow-Liu

- **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



in this example $Pa_{Intel} \subseteq \{C, D\}$

before Intel in a topological order

$Score(G:D)$

$= ScoreFam(C:D) + ScoreF(D|C:D) + ScoreF(I:D) + SF(G|D,I:D) + \ldots$

only change: $SF(I|C:D)$

"Optimal" algorithm: each node X_i picks set of parents that maximize $ScoreFam(X_i|Pa_{X_i}:D)$

$\Rightarrow$ but would get a BN with cycles

Fixed variable order 1

max number of parents = d

- Pick a variable order
  - guess
  - 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

$Pa_{X_i} \subseteq \{X_1, \ldots X_{i-1}\} \leftarrow n^d$

OPTIMAL BN, with d parents, consistent with order

Fixed variable order 2

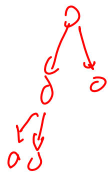
- Fix max number of parents to k
- For each *i* in order
  - Pick $\mathbf{Pa}_{X_i} \subseteq \{X_1,\ldots,X_{i-1}\}$
    - Exhaustively search through all possible subsets
    - $\mathbf{Pa}_{X_i}$ is maximum $\mathbf{U} \subseteq \{X_1,\ldots,X_{i-1}\}$ FamScore(X_i|$\mathbf{U}$: *D*)
- Optimal BN 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

1 2 4 5 3

details in reading

Learn BN structure using local search

Starting from Chow-Liu tree



result of fixed order search with $K = d$

Local search, possible moves:

Only if acyclic!!! ✓

- Add edge
- Delete edge
- Invert edge

⋮

if you are really eager, advanced search techniques like, tabu search, beam, A*

Select using favorite score

BIC

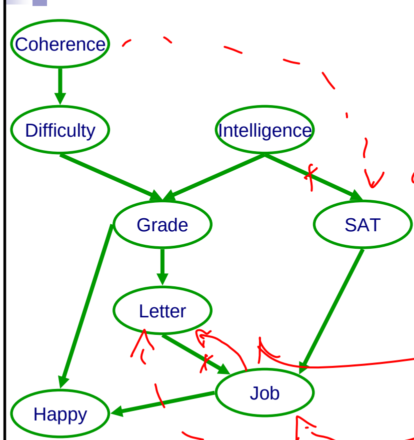
or

Bayesian

or

⋮

Exploit score decomposition in local search

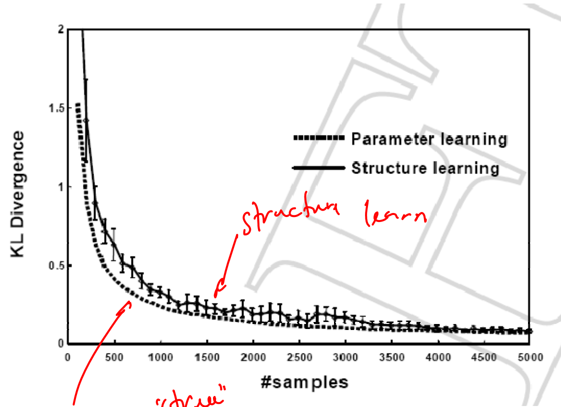


- Add edge and delete edge:
  - Only rescore one family!

only rescore ScoreFam(S|I:D)

- Reverse edge
  - Rescore only two families

Some experiments



Known "true" structure

Alarm network

Order search versus graph search

- Order search advantages
  - For fixed order, optimal BN – more "global" optimization
  - Space of orders much smaller than space of graphs
- Graph search advantages
  - Not restricted to k parents
    - Especially if exploiting CPD structure, such as CSI
  - Cheaper per iteration
  - Finer moves within a graph

e.g.: using decision trees
noisy-OR
:

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

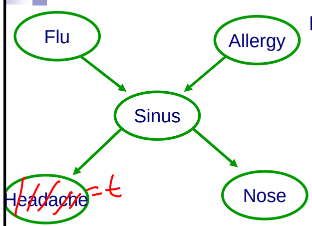
$$\log P(D \mid \mathcal{G}) = \log \int_{\theta_{\mathcal{G}}} P(D \mid \mathcal{G}, \theta_{\mathcal{G}}) P(\theta_{\mathcal{G}} | \mathcal{G}) d\theta_{\mathcal{G}}$$

- 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 $O(N^{k+1})$) 2k+6
- Search techniques
  - Search through orders
  - Search through structures
- Bayesian model averaging

Inference in graphical models: Typical queries 1



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

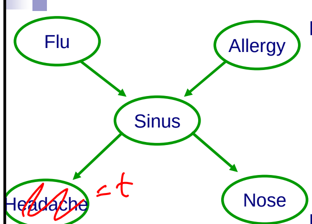
$$P(A=t \mid H=t)$$

$$P(A=t \mid H=t) \propto P(A=t, H=t)$$

$$P(A=t, H=t) = \sum_s \sum_n \sum_f P(A=t, H=t, s, n, f)$$

Inference in graphical models: Typical queries 2 – Maximization

sometimes called maximum a posteriori (MAP)



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

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

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

$$\max_a P(A=a \mid H=t)$$

$$= \max_a \sum_s \sum_f \sum_n P(A=a, s, f, n \mid H=t)$$

Are MPE and MAP Consistent?

Sinus → Nose

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

P(N|S) =

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

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

MPE : S=t , N=t

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

$\max_s P(S=s)$

$\Rightarrow$ MAP(S) = S=f

C++ Library

- Now available, join:
  - http://groups.google.com/group/10708-f08-code/
- The library implements the following functionality:
  - random variables, random processes, and linear algebra
  - factorized distributions, such Gaussians, multinomial distributions, and mixtures
  - graph structures and basic graph algorithms
  - graphical models, including Bayesian networks, Markov networks, andjunction trees
  - basic static and dynamic inference algorithms
  - parameter learning for Gaussian distributions, Chow Liu
- Fairly advanced C++ (not for everyone ☺)

Complexity of conditional probability queries 1

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

$E = \emptyset$

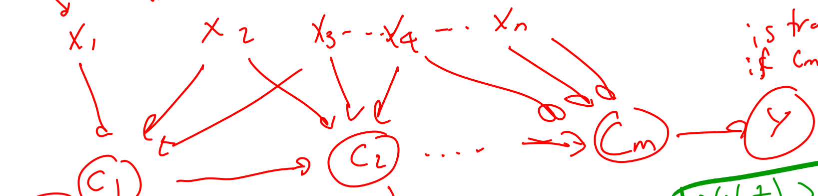
$P(Y=t)$

want a satisfying assignment exists?

Reduction – 3-SAT

$$\underbrace{(\bar{X}_1 \vee X_2 \vee X_3)}_{C_1} \wedge \underbrace{(\bar{X}_2 \vee X_3 \vee X_4)}_{C_2} \wedge \ldots \quad \ldots \; C_m$$

CPT: 50/50 uniform



is true if C_m is true

CPT: C_1 is true w prob 1 ⇔ Clause 1 satisfied

C_2 true w prob 1 ⇔ Clause 2 sat & Clause 1 sat

$P(Y=t) > 0$ ⇔ SAT formula is satisfyable

Complexity of conditional probability queries 2

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

#P-complete problems: e.g., how many satisfying assignments a SAT formula has?

2^n assignments, each has probability $\frac{1}{2^n}$

$$P(Y=t) = \frac{\#\text{ sat assignments}}{2^n}$$

Inference is #P-complete, hopeless?

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

one key property → low treewidth graphs

others, e.g., → CSI
→ associative potentials

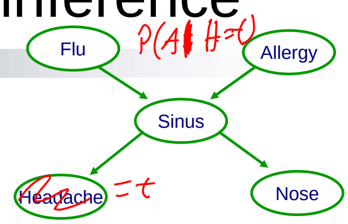
Complexity for other inference questions

- Probabilistic inference
  - general graphs: #P-complete
  - poly-trees and low tree-width: polynomial
- Approximate probabilistic inference
  - Absolute error: $|P(x) - \hat{P}(x)| \leq \varepsilon$ ← NP-hard for any $\varepsilon \leq 0.5$
  - Relative error: $1-\varepsilon \leq \frac{P(x)}{\hat{P}(x)} \leq 1+\varepsilon$ ← NP-hard for any $\varepsilon > 0$
- Most probable explanation (MPE)
  - general graphs: NP-complete
  - poly-trees and low tree-width: polynomial
- 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.:

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

: compute $\forall x$ $\Rightarrow P(X=x, E=e)$

- Normalization:

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

$P(A=t, H=t) = 0.2$

$P(A=f, H=t) = 0.1$

normalize

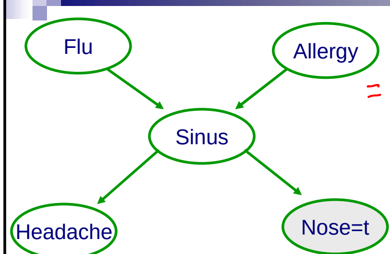
$P(A=t \mid H=t) = 2/3$

Marginalization

$P(F=t, N=t) = \sum_{s} P(F=t, s, N=t)$

$= P(F=t, S=t, N=t) + P(F=t, S=f, N=t$

Probabilistic inference example



$P(A \mid N=t) \propto P(A, N=t)$

$= \sum_{f} \sum_{s} \sum_{h} P(A, f, s, h, N=t)$

$= \sum_{f} \sum_{s} \sum_{h} P(f)\, P(A)\, P(s \mid f, A)\, P(h \mid s)\, P(N=t \mid s)$

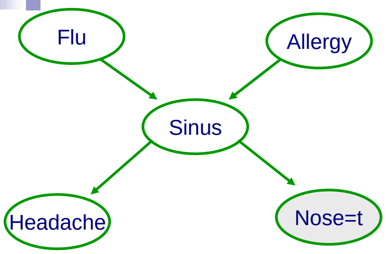
$= \sum_{f} \sum_{s} P(f)\, P(A)\, P(s \mid f, A)\, P(N=t \mid s) \overset{1}{\cancel{\sum_{h} P(h \mid s)}}$

$= \sum_{f} P(f)\, P(A) \underbrace{\sum_{S} P(s \mid f, A)\, P(N=t \mid s)}_{g_1(f,A)}$

$= P(A) \underbrace{\sum_{f} P(f)\, g_1(f, A)}_{g_2(A)} = P(A)\, g_2(A) = P(A, N=t)$

Inference seems exponential in number of variables!

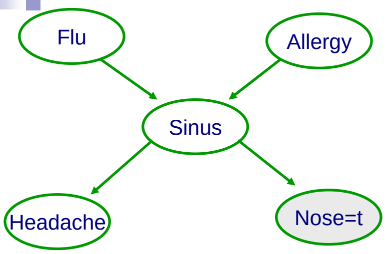
Fast probabilistic inference example – Variable elimination



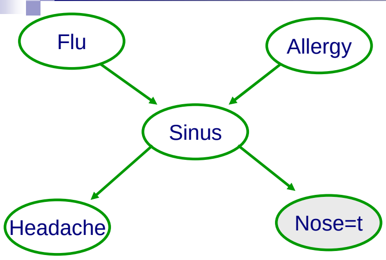
(Potential for) Exponential reduction in computation!

Understanding variable elimination – Exploiting distributivity

Understanding variable elimination – Order can make a HUGE difference

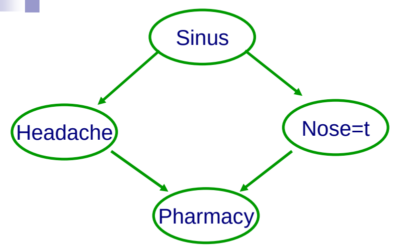


Understanding variable elimination – Intermediate results

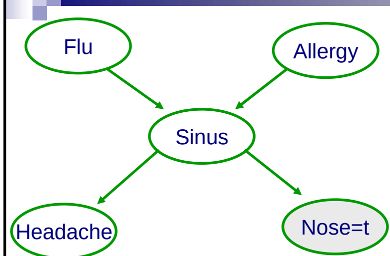


Intermediate results are probability distributions

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