

Bayes Nets (cont)

CS 486/686
University of Waterloo
Oct 8, 2009

Outline

- Inference in Bayes Nets
- Variable Elimination

Inference in Bayes Nets

- The independence sanctioned by D-separation (and other methods) allows us to compute prior and posterior probabilities quite effectively.
- We'll look at a couple simple examples to illustrate. We'll focus on networks without *loops*. (A loop is a cycle in the underlying *undirected* graph. Recall the directed graph has no cycles.)

Simple Forward Inference (Chain)

- Computing marginal requires simple forward "propagation" of probabilities

$P(J) = \sum_{M,ET} P(J, M, ET)$
 (marginalization)
 $P(J) = \sum_{M,ET} P(J|M, ET) P(M|ET) P(ET)$
 (chain rule)
 $P(J) = \sum_{M,ET} P(J|M) P(M|ET) P(ET)$
 (conditional independence)
 $P(J) = \sum_M P(J|M) \sum_{ET} P(M|ET) P(ET)$
 (distribution of sum)

Note: all (final) terms are CPTs in the BN
Note: only ancestors of J considered

Simple Forward Inference (Chain)

- Same idea applies when we have "upstream" evidence

$P(J|ET) = \sum_M P(J, M|ET)$
 (marginalisation)
 $P(J|ET) = \sum_M P(J|M, ET) P(M|ET)$
 (chain rule)
 $P(J|ET) = \sum_M P(J|M) P(M|ET)$
 (conditional independence)

Simple Forward Inference (Pooling)

- Same idea applies with multiple parents

$$\begin{aligned}
 P(\text{Fev}) &= \sum_{\text{Flu}, M, \text{TS}, \text{ET}} P(\text{Fev}, \text{Flu}, M, \text{TS}, \text{ET}) \\
 &= \sum_{\text{Flu}, M, \text{TS}, \text{ET}} P(\text{Fev} | \text{Flu}, M, \text{TS}, \text{ET}) P(\text{Flu} | M, \text{TS}, \text{ET}) \\
 &\quad P(M | \text{TS}, \text{ET}) P(\text{TS} | \text{ET}) P(\text{ET}) \\
 &= \sum_{\text{Flu}, M, \text{TS}, \text{ET}} P(\text{Fev} | \text{Flu}, M) P(\text{Flu} | \text{TS}) P(M | \text{ET}) P(\text{TS}) P(\text{ET}) \\
 &= \sum_{\text{Flu}, M} P(\text{Fev} | \text{Flu}, M) [\sum_{\text{TS}} P(\text{Flu} | \text{TS}) P(\text{TS})] \\
 &\quad [\sum_{\text{ET}} P(M | \text{ET}) P(\text{ET})]
 \end{aligned}$$

- (1) by marginalisation; (2) by the chain rule; (3) by conditional independence; (4) by distribution
- note: all terms are CPTs in the Bayes net

Simple Forward Inference (Pooling)

- Same idea applies with evidence

$$\begin{aligned}
 P(\text{Fev}|\text{ts},\sim m) &= \sum_{\text{Flu}} P(\text{Fev},\text{Flu}|\text{ts},\sim m) \\
 &= \sum_{\text{Flu}} P(\text{Fev}|\text{Flu},\text{ts},\sim m) P(\text{Flu}|\text{ts},\sim m) \\
 &= \sum_{\text{Flu}} P(\text{Fev}|\text{Flu},\sim m) P(\text{Flu}|\text{ts})
 \end{aligned}$$



7

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

Simple Backward Inference

- When evidence is downstream of query variable, we must reason "backwards." This requires the use of Bayes rule:

$$\begin{aligned}
 P(\text{ET} | j) &= \alpha P(j | \text{ET}) P(\text{ET}) \\
 &= \alpha \sum_M P(j, M | \text{ET}) P(\text{ET}) \\
 &= \alpha \sum_M P(j | M, \text{ET}) P(M | \text{ET}) P(\text{ET}) \\
 &= \alpha \sum_M P(j | M) P(M | \text{ET}) P(\text{ET})
 \end{aligned}$$

- First step is just Bayes rule
 - normalizing constant α is $1/P(j)$; but we needn't compute it explicitly if we compute $P(\text{ET} | j)$ for each value of ET: we just add up terms $P(j | \text{ET}) P(\text{ET})$ for all values of ET (they sum to $P(j)$)

8

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

Backward Inference (Pooling)

- Same ideas when several pieces of evidence lie "downstream"

$$\begin{aligned}
 P(\text{ET} | j, \text{fev}) &= \alpha P(j, \text{fev} | \text{ET}) P(\text{ET}) \\
 &= \alpha \sum_{M, \text{FI}, \text{TS}} P(j, \text{fev}, M, \text{FI}, \text{TS} | \text{ET}) P(\text{ET}) \\
 &= \alpha \sum_{M, \text{FI}, \text{TS}} P(j | \text{fev}, M, \text{FI}, \text{TS}, \text{ET}) P(\text{fev} | M, \text{FI}, \text{TS}, \text{ET}) \\
 &\quad P(M | \text{FI}, \text{TS}, \text{ET}) P(\text{FI} | \text{TS}, \text{ET}) P(\text{TS} | \text{ET}) P(\text{ET}) \\
 &= \alpha P(\text{ET}) \sum_M P(j | M) \sum_{\text{FI}} P(\text{fev} | M, \text{FI}) \sum_{\text{TS}} P(\text{FI} | \text{TS}) P(\text{TS})
 \end{aligned}$$

- Same steps as before; but now we compute prob of both pieces of evidence given hypothesis ET and combine them. Note: they are independent given M; but not given ET.

9

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

Variable Elimination

- The intuitions in the above examples give us a simple inference algorithm for networks without loops: the *polytree* algorithm.
- Instead we'll look at a more general algorithm that works for general BNs; but the polytree algorithm will be a special case.
- The algorithm, *variable elimination*, simply applies the summing out rule repeatedly.
 - To keep computation simple, it exploits the independence in the network and the ability to distribute sums inward

10

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

Factors

- A function $f(X_1, X_2, \dots, X_k)$ is also called a *factor*. We can view this as a table of numbers, one for each instantiation of the variables X_1, X_2, \dots, X_k .
 - A tabular rep'n of a factor is exponential in k
- Each CPT in a Bayes net is a factor:
 - e.g., $\text{Pr}(C|A,B)$ is a function of three variables, A, B, C
- Notation: $f(\mathbf{X}, \mathbf{Y})$ denotes a factor over the variables $\mathbf{X} \cup \mathbf{Y}$. (Here \mathbf{X}, \mathbf{Y} are sets of variables.)

11

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

The Product of Two Factors

- Let $f(\mathbf{X}, \mathbf{Y})$ & $g(\mathbf{Y}, \mathbf{Z})$ be two factors with variables \mathbf{Y} in common
- The *product* of f and g , denoted $h = f \times g$ (or sometimes just $h = fg$), is defined:

$$h(\mathbf{X}, \mathbf{Y}, \mathbf{Z}) = f(\mathbf{X}, \mathbf{Y}) \times g(\mathbf{Y}, \mathbf{Z})$$

f(A,B)		g(B,C)		h(A,B,C)			
ab	0.9	bc	0.7	abc	0.63	ab~c	0.27
a~b	0.1	b~c	0.3	a~bc	0.08	a~b~c	0.02
~ab	0.4	~bc	0.8	~abc	0.28	~ab~c	0.12
~a~b	0.6	~b~c	0.2	~a~bc	0.48	~a~b~c	0.12

12

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

Summing a Variable Out of a Factor

- Let $f(X,Y)$ be a factor with variable X (Y is a set)
- We *sum out* variable X from f to produce a new factor $h = \sum_X f$, which is defined:

$$h(Y) = \sum_{x \in \text{Dom}(X)} f(x,Y)$$

f(A,B)		h(B)	
ab	0.9	b	1.3
a~b	0.1	~b	0.7
~ab	0.4		
~a~b	0.6		

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

13

Restricting a Factor

- Let $f(X,Y)$ be a factor with variable X (Y is a set)
- We *restrict* factor f to $X=x$ by setting X to the value x and "deleting". Define $h = f_{X=x}$ as: $h(Y) = f(x,Y)$

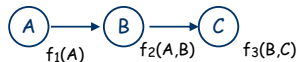
f(A,B)		h(B) = f_{A=a}	
ab	0.9	b	0.9
a~b	0.1	~b	0.1
~ab	0.4		
~a~b	0.6		

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

14

Variable Elimination: No Evidence

- Computing prior probability of query var X can be seen as applying these operations on factors



- $$P(C) = \sum_{A,B} P(C|B) P(B|A) P(A)$$

$$= \sum_B P(C|B) \sum_A P(B|A) P(A)$$

$$= \sum_B f_3(B,C) \sum_A f_2(A,B) f_1(A)$$

$$= \sum_B f_3(B,C) f_4(B) = f_5(C)$$

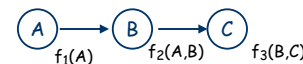
Define new factors: $f_4(B) = \sum_A f_2(A,B) f_1(A)$ and $f_5(C) = \sum_B f_3(B,C) f_4(B)$

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

15

Variable Elimination: No Evidence

- Here's the example with some numbers

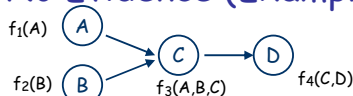


f ₁ (A)		f ₂ (A,B)		f ₃ (B,C)		f ₄ (B)		f ₅ (C)	
a	0.9	ab	0.9	bc	0.7	b	0.85	c	0.625
~a	0.1	a~b	0.1	b~c	0.3	~b	0.15	~c	0.375
		~ab	0.4	~bc	0.2				
		~a~b	0.6	~b~c	0.8				

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

16

VE: No Evidence (Example 2)



- $$P(D) = \sum_{A,B,C} P(D|C) P(C|B,A) P(B) P(A)$$

$$= \sum_C P(D|C) \sum_B P(B) \sum_A P(C|B,A) P(A)$$

$$= \sum_C f_4(C,D) \sum_B f_2(B) \sum_A f_3(A,B,C) f_1(A)$$

$$= \sum_C f_4(C,D) \sum_B f_2(B) f_5(B,C)$$

$$= \sum_C f_4(C,D) f_6(C)$$

$$= f_7(D)$$

Define new factors: $f_5(B,C)$, $f_6(C)$, $f_7(D)$, in the obvious way

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

17

Variable Elimination: One View

- One way to think of variable elimination:
 - write out desired computation using the chain rule, exploiting the independence relations in the network
 - arrange the terms in a convenient fashion
 - distribute each sum (over each variable) in as far as it will go
 - i.e., the sum over variable X can be "pushed in" as far as the "first" factor mentioning X
 - apply operations "inside out", repeatedly eliminating and creating new factors (note that each step/removal of a sum eliminates one variable)

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

18

Variable Elimination Algorithm

- Given query var Q , remaining vars Z . Let F be the set of factors corresponding to CPTs for $\{Q\} \cup Z$.

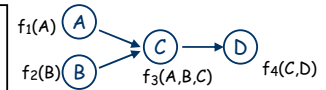
- Choose an elimination ordering Z_1, \dots, Z_n of variables in Z .
- For each Z_j -- in the order given -- eliminate $Z_j \in Z$ as follows:
 - Compute new factor $g_j = \sum_{Z_j} f_1 \times f_2 \times \dots \times f_k$, where the f_i are the factors in F that include Z_j
 - Remove the factors f_i (that mention Z_j) from F and add new factor g_j to F
- The remaining factors refer only to the query variable Q . Take their product and normalize to produce $P(Q)$

CS486/686 Lecture Slides (j) 2009 C. Boutilier, P. Poupart & K. Loran

19

VE: Example 2 again

Factors: $f_1(A)$ $f_2(B)$
 $f_3(A,B,C)$ $f_4(C,D)$
Query: $P(D)?$
Elim. Order: A, B, C



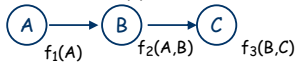
- Step 1: Add $f_5(B,C) = \sum_A f_3(A,B,C) f_1(A)$
 Remove: $f_1(A), f_3(A,B,C)$
- Step 2: Add $f_6(C) = \sum_B f_2(B) f_5(B,C)$
 Remove: $f_2(B), f_5(B,C)$
- Step 3: Add $f_7(D) = \sum_C f_4(C,D) f_6(C)$
 Remove: $f_4(C,D), f_6(C)$
- Last factor $f_7(D)$ is (possibly unnormalized) probability $P(D)$

CS486/686 Lecture Slides (j) 2009 C. Boutilier, P. Poupart & K. Loran

20

Variable Elimination: Evidence

- Computing posterior of query variable given evidence is similar: suppose we observe $C=c$:



$$\begin{aligned}
 P(A|c) &= \alpha P(A) P(c|A) \\
 &= \alpha P(A) \sum_B P(c|B) P(B|A) \\
 &= \alpha f_1(A) \sum_B f_3(B,c) f_2(A,B) \\
 &= \alpha f_1(A) \sum_B f_4(B) f_2(A,B) \\
 &= \alpha f_1(A) f_5(A) \\
 &= \alpha f_6(A)
 \end{aligned}$$

New factors: $f_4(B) = f_3(B,c)$; $f_5(A) = \sum_B f_2(A,B) f_4(B)$;
 $f_6(A) = f_1(A) f_5(A)$

CS486/686 Lecture Slides (j) 2009 C. Boutilier, P. Poupart & K. Loran

21

Variable Elimination with Evidence

- Given query var Q , evidence vars E (observed to be e), remaining vars Z . Let F be set of factors involving CPTs for $\{Q\} \cup Z$.

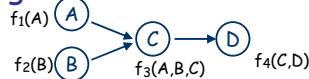
- Replace each factor $f \in F$ that mentions a variable(s) in E with its restriction $f_{E=e}$ (somewhat abusing notation)
- Choose an elimination ordering Z_1, \dots, Z_n of variables in Z .
- Run variable elimination as above.
- The remaining factors refer only to the query variable Q . Take their product and normalize to produce $P(Q)$

CS486/686 Lecture Slides (j) 2009 C. Boutilier, P. Poupart & K. Loran

22

VE: Example 2 again with Evidence

Factors: $f_1(A)$ $f_2(B)$
 $f_3(A,B,C)$ $f_4(C,D)$
Query: $P(A)?$
Evidence: $D = d$
Elim. Order: C, B



Restriction: replace $f_4(C,D)$ with $f_5(C) = f_4(C,d)$

Step 1: Add $f_6(A,B) = \sum_C f_5(C) f_3(A,B,C)$
 Remove: $f_3(A,B,C), f_5(C)$

Step 2: Add $f_7(A) = \sum_B f_6(A,B) f_2(B)$
 Remove: $f_6(A,B), f_2(B)$

Last factors: $f_7(A), f_1(A)$. The product $f_1(A) \times f_7(A)$ is (possibly unnormalized) posterior. So... $P(A|d) = \alpha f_1(A) \times f_7(A)$.

CS486/686 Lecture Slides (j) 2009 C. Boutilier, P. Poupart & K. Loran

23

Some Notes on the VE Algorithm

- After iteration j (elimination of Z_j), factors remaining in set F refer only to variables X_{j+1}, \dots, Z_n and Q . No factor mentions an evidence variable E after the initial restriction.
- Number of iterations: linear in number of variables
- Complexity is linear in number of vars and exponential in size of the largest factor.
 - Recall each factor has exponential size in its number of variables
 - Can't do any better than size of BN (since its original factors are part of the factor set)
 - When we create new factors, we might make a set of variables larger.

CS486/686 Lecture Slides (j) 2009 C. Boutilier, P. Poupart & K. Loran

24

Some Notes on the VE Algorithm

- The size of the resulting factors is determined by elimination ordering! (We'll see this in detail)
- For *polytrees*, easy to find good ordering (e.g., work outside in).
- For general BNs, sometimes good orderings exist, sometimes they don't (then inference is exponential in number of vars).
 - Simply *finding* the optimal elimination ordering for general BNs is NP-hard.
 - Inference in general is NP-hard in general BNs

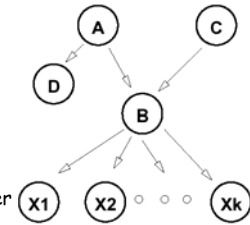
25

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

Elimination Ordering: Polytrees

- Inference is linear in size of network

- ordering: eliminate only "singly-connected" nodes
- e.g., in this network, eliminate D, A, C, X1,...; or eliminate X1,... Xk, D, A, C; or mix up...
- result: no factor ever larger than original CPTs
- eliminating B before these gives factors that include all of A,C, X1,... Xk !!!



26

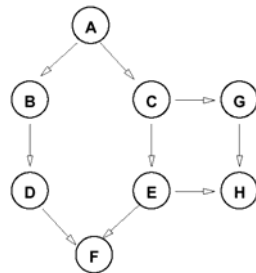
CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

Effect of Different Orderings

- Suppose query variable is D. Consider different orderings for this network

- A,F,H,G,B,C,E:
 - good: why?
- E,C,A,B,G,H,F:
 - bad: why?

- Which ordering creates smallest factors?
 - either max size or total
- which creates largest factors?



27

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

Relevance



- Certain variables have no impact on the query.
 - In ABC network, computing $\Pr(A)$ with no evidence requires elimination of B and C.
 - But when you sum out these vars, you compute a trivial factor (whose value are all ones); for example:
 - eliminating C: $f_4(B) = \sum_C f_3(B,C) = \sum_C \Pr(C|B)$
 - 1 for any value of B (e.g., $\Pr(c|b) + \Pr(\sim c|b) = 1$)
- No need to think about B or C for this query

28

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

Relevance: A Sound Approximation

- Can restrict attention to *relevant* variables. Given query Q, evidence E:
 - Q is relevant
 - if any node Z is relevant, its parents are relevant
 - if $E \in E$ is a descendent of a relevant node, then E is relevant
- We can restrict our attention to the *subnetwork comprising only relevant variables* when evaluating a query Q

29

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson

Next Class

- Decision making
 - Utility Theory
 - Decision Trees
- Russell & Norvig: Chapter 16

30

CS486/686 Lecture Slides (c) 2009 C. Boutilier, P. Poupart & K. Larson