Bayes Nets

CS 486/686: Introduction to Artificial Intelligence

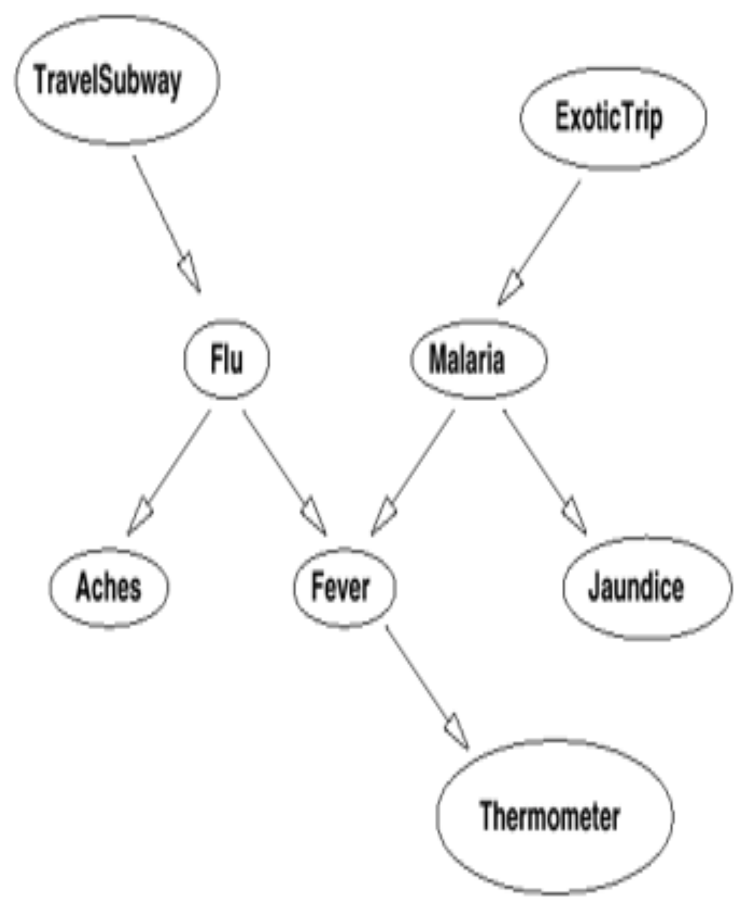
Outline

- Inference in Bayes Nets
- Variable Elimination

Inference in Bayes Nets

- Independence allows us to compute prior and posterior probabilities quite effectively
- We will start with a couple simple examples
  - Networks without loops
    - A loop is a cycle in the underlying undirected graph

Forward Inference



$P(J)=\Sigma_{M,ET}\ P(J|M,ET)P(M,ET)$

(marginalization)

$P(J)=\Sigma_{M,ET}\ P(J|M)P(M|ET)P(ET)$

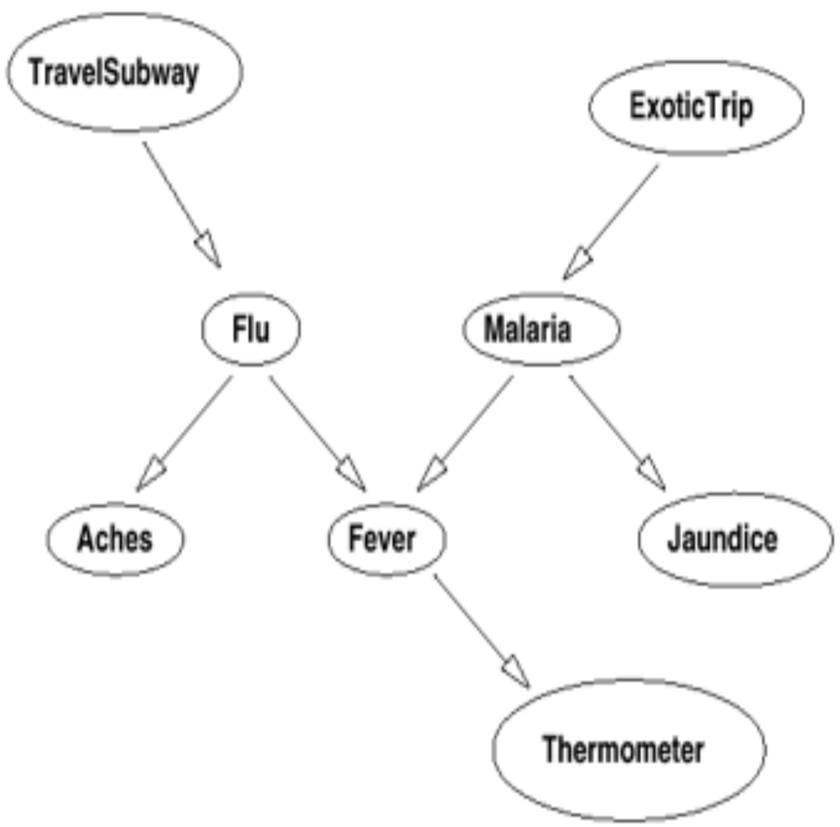
(chain rule and independence)

$P(J)=\Sigma_M P(J|M)\Sigma_{ET} P(M|ET)P(ET)$

(distribution of sum)

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

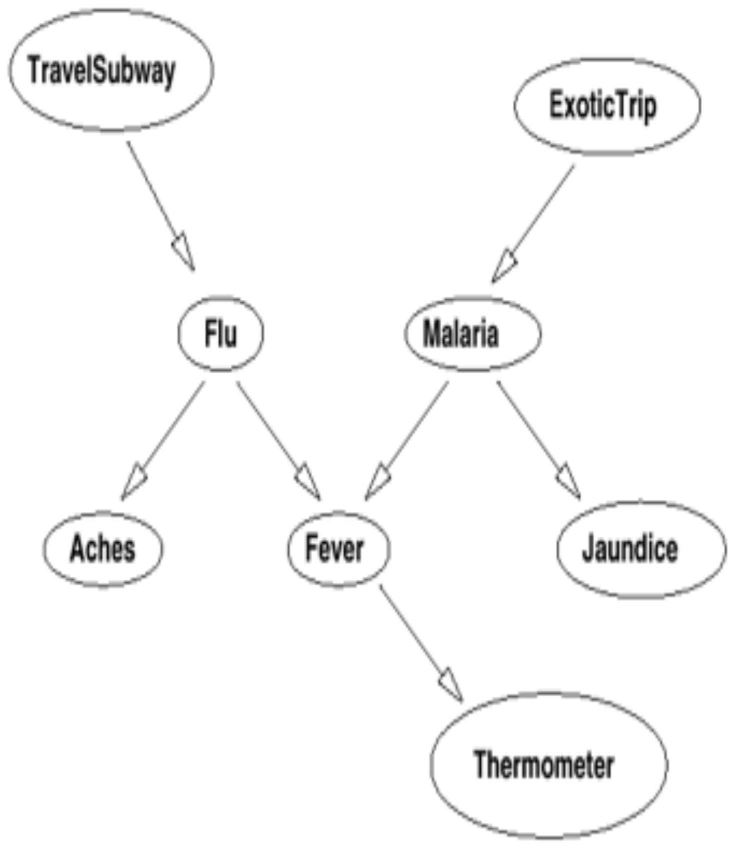
Forward Inference with "Upstream Evidence"



$$P(J|ET) = \Sigma_M P(J|M,ET)\, P(M|ET)$$
$$= \Sigma_M P(J|M)\, P(M|ET)$$

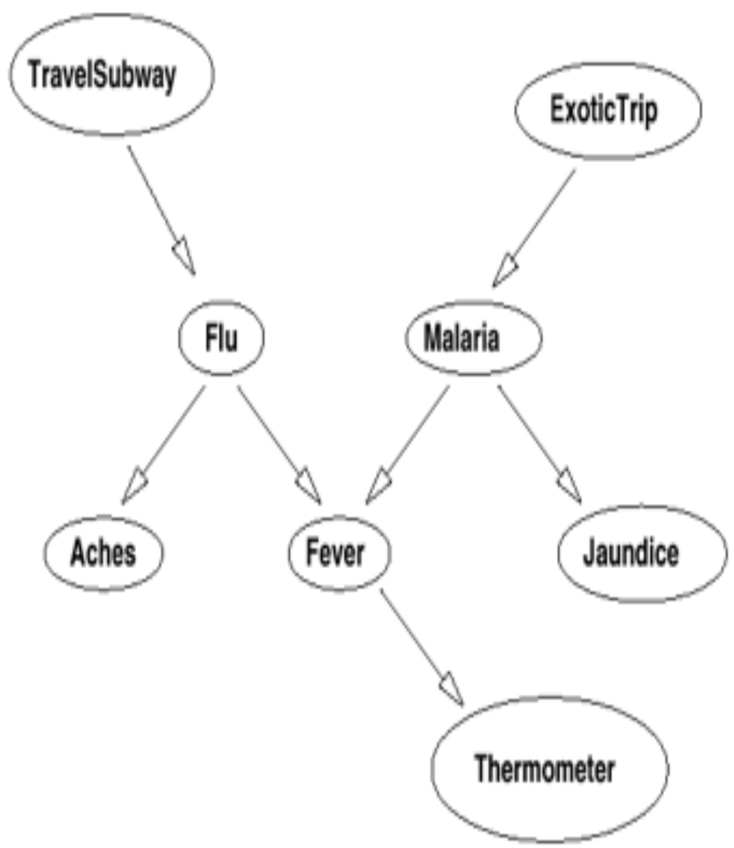
(J is cond independent of ET given M)

Forward Inference with Multiple Parents



$P(Fev)=?$

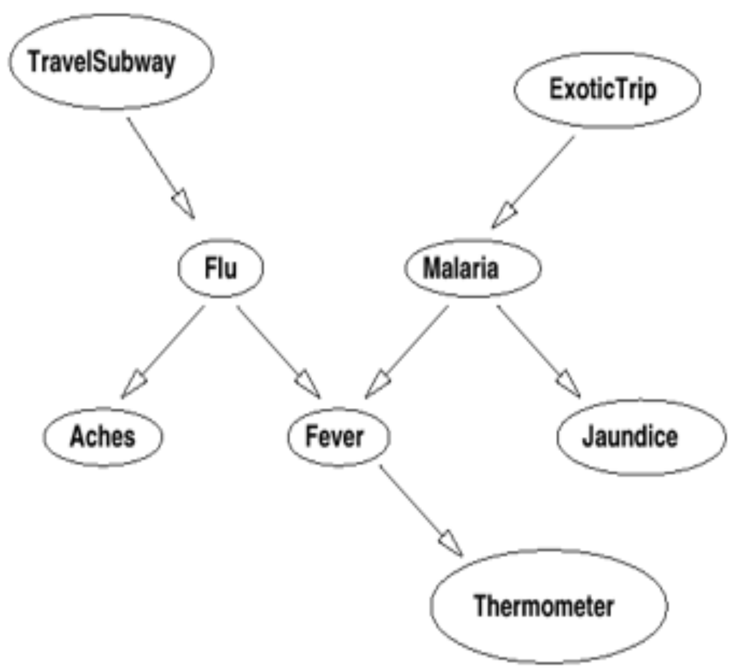
Forward Inference with Evidence



$P(Fev|ts,\sim m)=?$

Simple Backward Inference

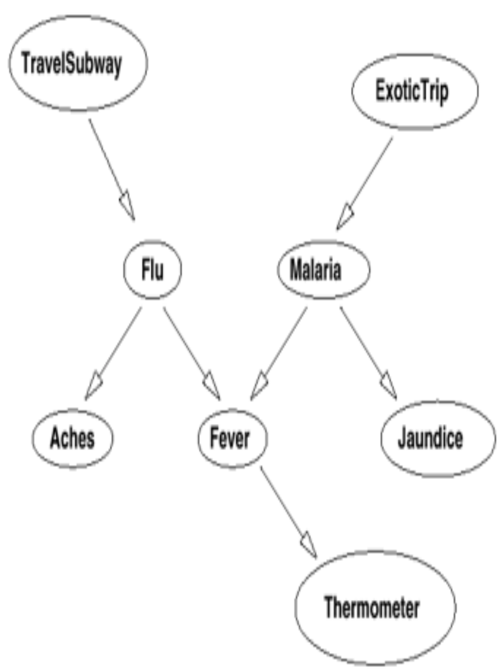
- When evidence is downstream of a query variable, must reason "backwards". This requires Bayes Rule



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

Backward Inference

- Same idea applies when several pieces of evidence lie “downstream”



P(ET|j,fev)=?

Variable Elimination

- Intuitions in previous examples give us a simple inference algorithm for networks without loops:
  - Polytree algorithm

- What about general BN?

Variable Elimination

- Simply applies the summing-out rule (marginalization) repeatedly
- Exploits independence in network and distributes the sum inward
  - Basically doing dynamic programming

Factors

- A function $f(X_1,\ldots,X_k)$ is called a factor
  - View this as a table of numbers, one for each instantiation of the variables
  - Exponential in k
- Each CPT in a BN is a factor
  - $P(C|A,B)$ is a function of 3 variables, A, B, C
    - Represented as $f(A,B,C)$
- Notation: $f(\mathbf{X},\mathbf{Y})$ denotes a factor over variables $\mathbf{X} \cup \mathbf{Y}$
  - **X** and **Y** are sets of variables

Product of Two Factors

- Let f(**X,Y**) and g(**Y,Z**) be two factors with variables **Y** in common
- The product of f and g, denoted by h=fg is
  - h(**X,Y,Z)**=f(**X,Y**) x g(**Y,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

Summing a Variable Out of a Factor

- Let f(X,**Y**) be a factor with variable X and variable set **Y**
- We sum out variable X from f to produce $h=\sum_X f$ where $h(\mathbf{Y})=\sum_{x \in Dom(X)} f(x,\mathbf{Y})$

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		

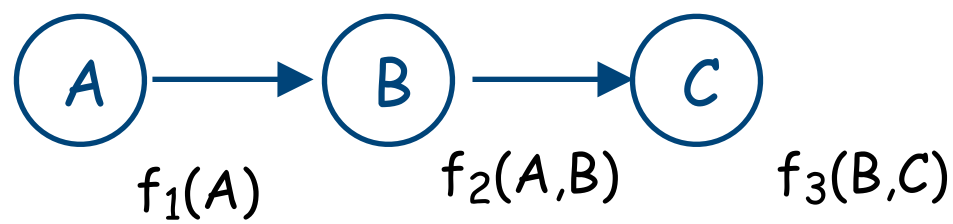
Restricting a Factor

- Let f(X,**Y**) be a factor with variable X
- 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		

Variable Elimination: No Evidence

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

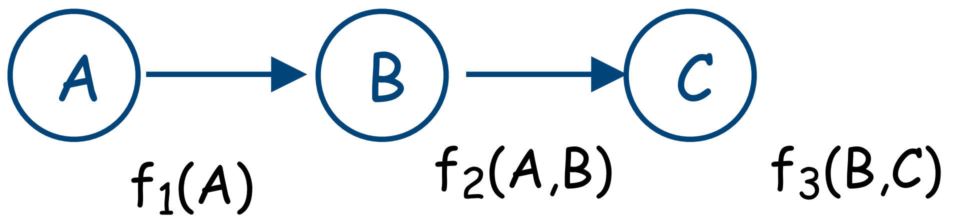


- $P(C) = \Sigma_{A,B}\, P(C|B)\, P(B|A)\, P(A)$

$$= \Sigma_B\, P(C|B)\, \Sigma_A\, P(B|A)\, P(A)$$
$$= \Sigma_B\, f_3(B,C)\, \Sigma_A\, f_2(A,B)\, f_1(A)$$
$$= \Sigma_B\, f_3(B,C)\, f_4(B)$$
$$= f_5(C)$$

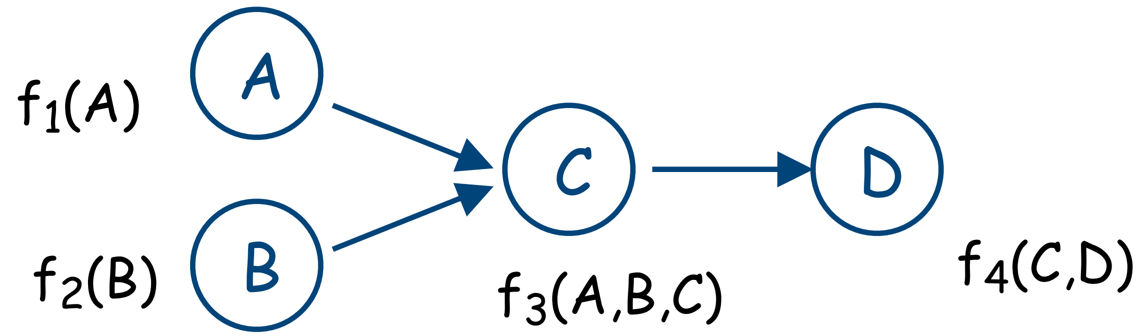
Define new factors: $f_4(B) = \Sigma_A\, f_2(A,B)\, f_1(A)$ and $f_5(C) = \Sigma_B\, f_3(B,C)\, f_4(B)$

Variable Elimination: No Evidence



$f_1(A)$		$f_2(A,B)$		$f_3(B,C)$		$f_4(B)$		$f_5(C)$	
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				

Variable Elimination: No Evidence



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

$$= \Sigma_C\ P(D|C)\ \Sigma_B\ P(B)\ \Sigma_A\ P(C|B,A)\ P(A)$$

$$= \Sigma_C\ f_4(C,D)\ \Sigma_B\ f_2(B)\ \Sigma_A\ f_3(A,B,C)\ f_1(A)$$

$$= \Sigma_C\ f_4(C,D)\ \Sigma_B\ f_2(B)\ f_5(B,C)$$

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

$$= f_7(D)$$

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

Variable Elimination: One View

- Write out desired computation using chain rule, exploiting independence relations in networks
- Arrange terms in convenient fashion
- Distribution each sum (over each variable) in as far as it will go
- Apply operations “inside out”, repeatedly elimination and creating new factors
  - Note that each step eliminates a variable

The Algorithm

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

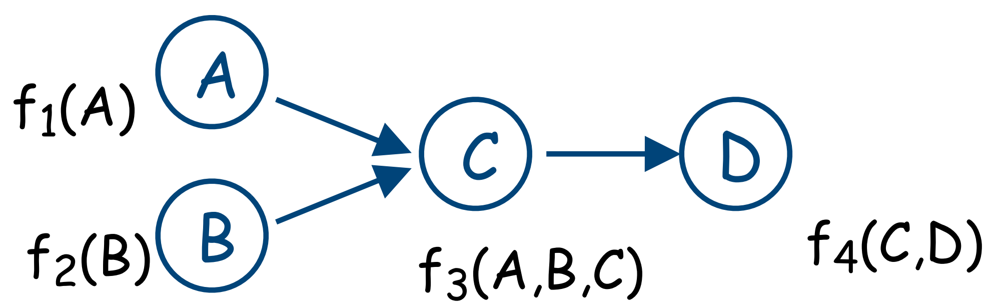
1. Choose an elimination ordering $Z_1, \ldots, Z_n$ of variables in **Z**.
2. For each Z_j -- in the order given -- eliminate $Z_j \in \mathbf{Z}$ as follows:
   (a) Compute new factor $g_j = \Sigma_{Z_j} f_1 \times f_2 \times \ldots \times f_k$, where the f_i are the factors in F that include Z_j
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

Example 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 $f_5(B,C) = \Sigma_A f_3(A,B,C)\, f_1(A)$

Remove: $f_1(A)$, $f_3(A,B,C)$

Step 2: Add $f_6(C) = \Sigma_B f_2(B)\, f_5(B,C)$

Remove: $f_2(B)$, $f_5(B,C)$

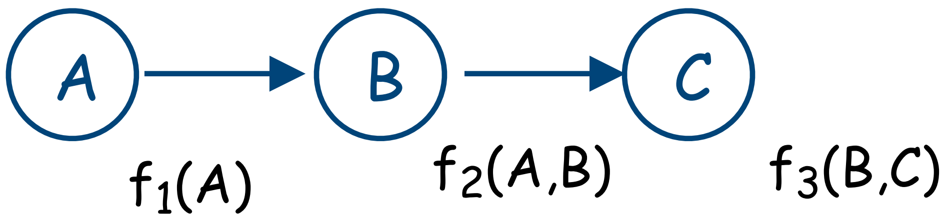
Step 3: Add $f_7(D) = \Sigma_C f_4(C,D)\, f_6(C)$

Remove: $f_4(C,D)$, $f_6(C)$

Last factor $f_7(D)$ is (possibly unnormalized) probability $P(D)$

Variable Elimination: Evidence

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



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

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

$f_6(A)= f_1(A)\, f_5(A)$

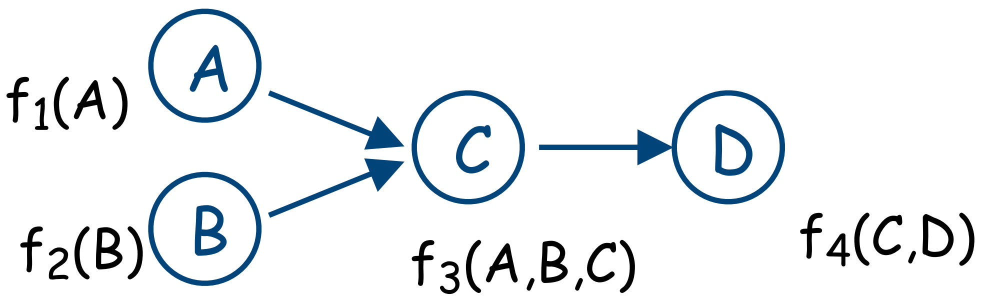
The Algorithm (with Evidence)

- Given query variable Q, evidence variables **E** (observed to be **e**), remaining variables **Z.** Let F be the set of factors corresponding to CPTs for $\{Q\} \cup \mathbf{Z}$.

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

Example

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



Some Notes on VE

- After each iteration j (elimination of Z_j) factors remaining in set F refer only to variables $Z_{j+1},\ldots,Z_n$ and Q
  - No factor mentions an evidence variable after the initial restriction
- Number of iterations is linear in number of variables

Some Notes on VE

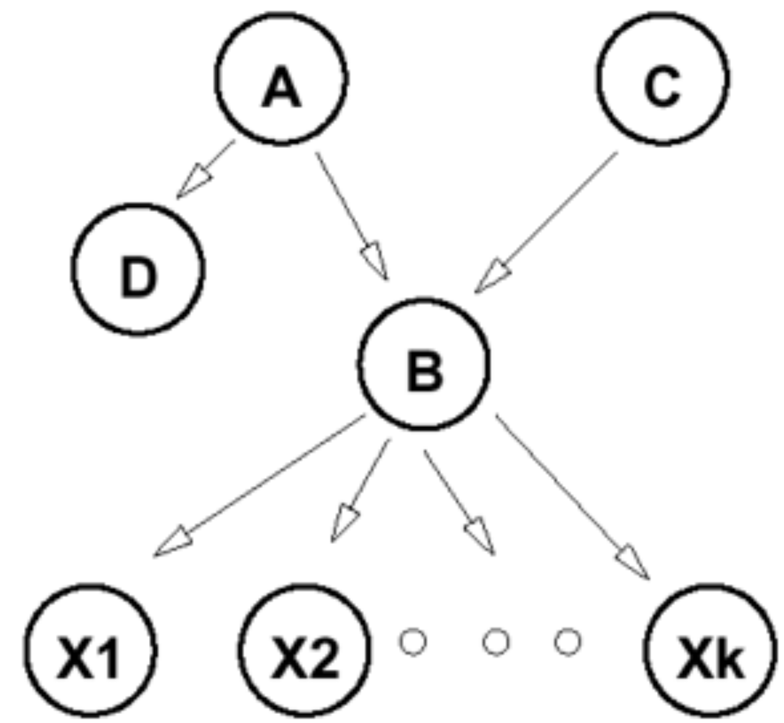
- Complexity is linear in number of variables and exponential in size of the largest factor
  - Recall each factor has exponential size in its number of variables
  - Can't do any better than size of BN (since its original factors are part of the factor set)
  - When we create new factors, we might make a set of variables larger

Some Notes on VE

- Size of resulting factors is determined by elimination ordering
  - For polytrees, easy to find a good ordering
  - For general BN, sometimes good orderings exist and sometimes they don't
    - in which case inference is exponential in number of variables
  - Finding the optimal elimination ordering is NP-hard

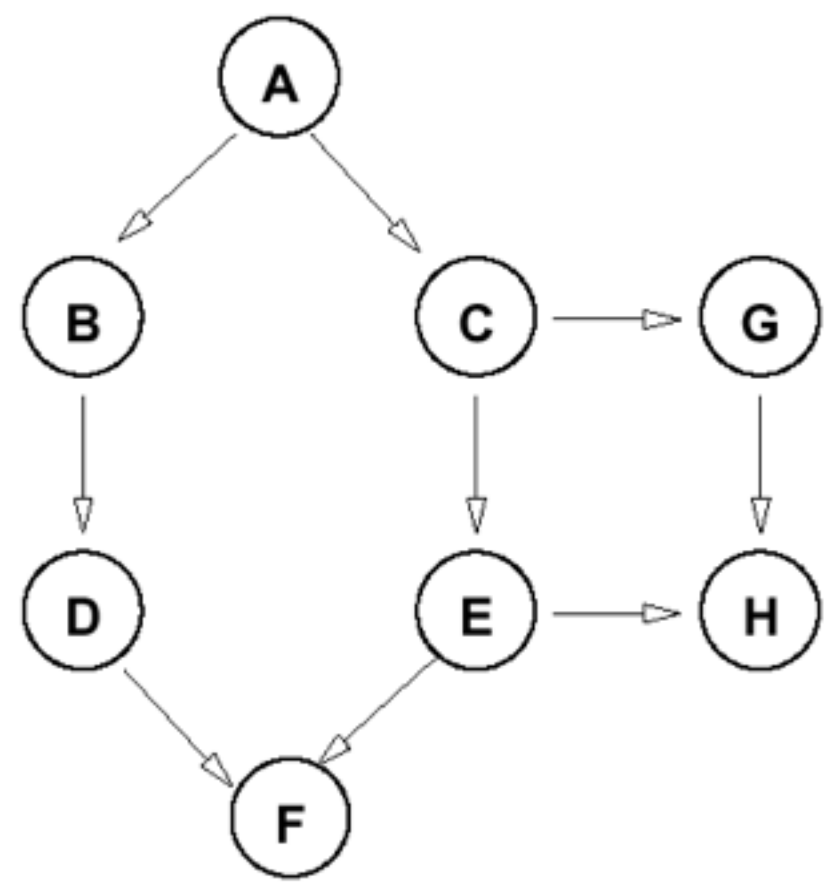
Elimination Ordering: Polytrees

- Inference is linear in size of the network
  - Ordering: eliminate only "singly-connected" nodes
  - Result: no factor ever larger than original CPTs
  - What happens if we eliminate B first?



Effect of Different Orderings

- Suppose query variable is D. Consider different orderings for this network
  - A,F,H,G,B,C,E: Good
  - E,C,A,B,G,H,F: Bad



Relevance

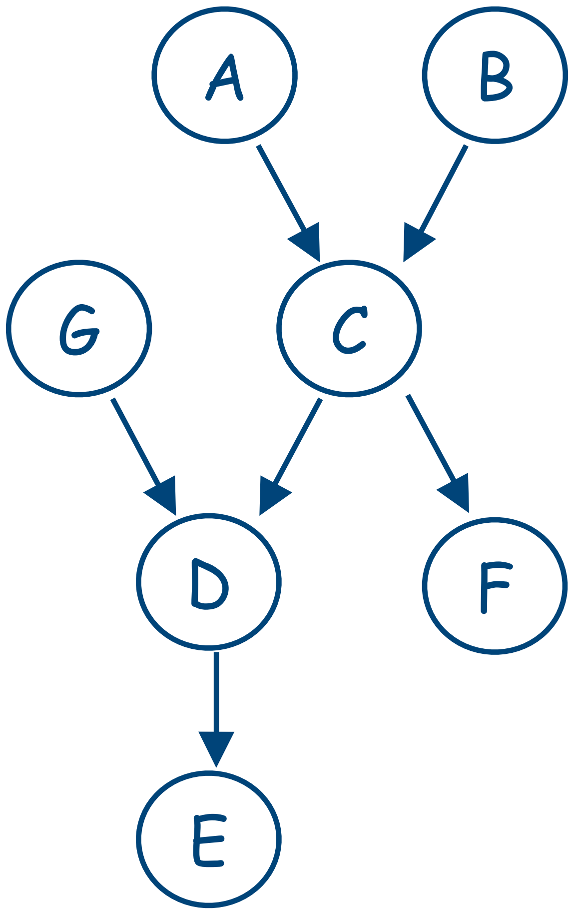
- Certain variables have no impact on the query
  - In ABC network, computing P(A) with no evidence requires elimination of B and C
    - But when you sum out these variables, you compute a trivial factor
    - Eliminating C: $g(C)=\sum_C f(B,C)=\sum_C Pr(C|B)$.
    - Note that $P(c|b)+P(\sim c|b)=1$ and $P(c|\sim b)+P(\sim c|\sim b)=1$

Relevance: A Sound Approximation

- Can restrict our attention to **relevant** variables
- Given query Q, evidence **E**
  - Q is relevant
  - If any node Z is relevant, its parents are relevant
  - If $E \in \mathbf{E}$ is a descendant of a relevant node, then E is relevant

Example

- P(F)
- P(F|E)
- P(F|E,C)



Probabilistic Inference

- Applications of BN in AI are virtually limitless
- Examples
  - mobile robot navigation
  - speech recognition
  - medical diagnosis, patient monitoring
  - fault diagnosis (e.g. car repairs)
  - etc

Where do BNs Come From?

- Often handcrafted
  - Interact with a domain expert to
    - Identify dependencies among variables (causal structure)
    - Quantify local distributions (CPTs)
- Empirical data, human expertise often used as a guide

Where do BNs Come From?

- Recent emphasis on learning BN from data
  - Input: a set of cases (instantiations of variables)
  - Output: network reflecting empirical distribution
  - Issues: identifying causal structure, missing data, discovery of hidden (unobserved) variables, incorporating prior knowledge (bias) about structure