Lecture 7: Bayesian Networks (Continued)
CS486/686 Intro to Artificial Intelligence

2026-1-27

Pascal Poupart
David R. Cheriton School of Computer Science

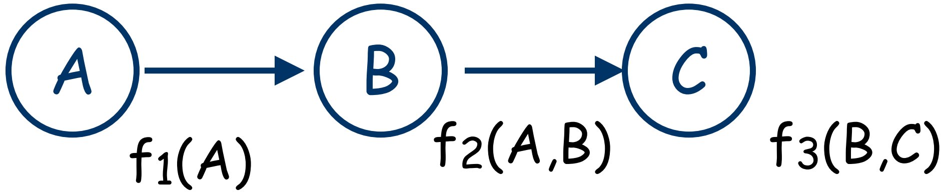
UNIVERSITY OF WATERLOO

Outline

- Variable Elimination with Evidence
- Elimination orders
- Reducing computation to relevant factors

Variable Elimination: Evidence

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



$$P(A|c) = \alpha\, P(A)\, P(c|A)$$
$$= \alpha\, P(A)\, \Sigma_B\, P(c|B)\, P(B|A)$$
$$= \alpha\, f_1(A)\, \Sigma_B\, f_3(B,c)\, f_2(A,B)$$
$$= \alpha\, f_1(A)\, \Sigma_B\, f_4(B)\, f_2(A,B)$$
$$= \alpha\, f_1(A)\, f_5(A)$$
$$= \alpha\, f_6(A)$$

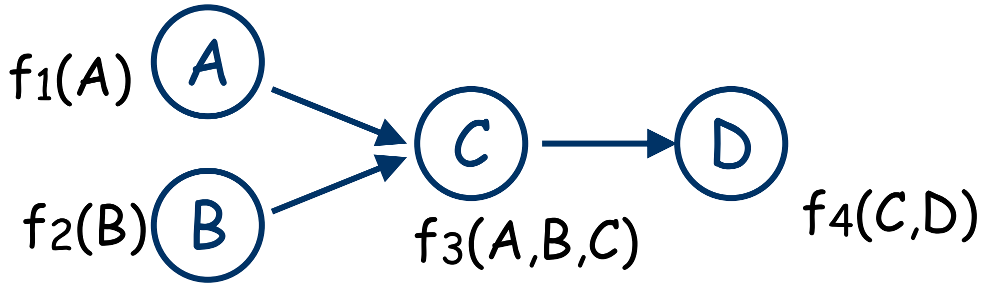
New factors: $f_4(B)= f_3(B,c)$; $f_5(A)= \Sigma_B\, f_2(A,B)\, f_4(B)$; $f_6(A)= f_1(A)\, f_5(A)$

Variable Elimination with Evidence

Given query var Q, evidence vars **E** (observed to be **e**), remaining vars **Z**. Let F be the set of factors involving CPTs for {Q} ∪ **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. For each Z_j -- in the order given -- eliminate $Z_j \in$ **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
4. The remaining factors refer only to the query variable Q.
   Take their product and normalize to produce P(Q)

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 $f_6(A,B)= \Sigma_C f_5(C)\ f_3(A,B,C)$
Remove: $f_3(A,B,C)$, $f_5(C)$
Step 2: Add $f_7(A) = \Sigma_B f_6(A,B)\ f_2(B)$
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… $P(A|d) = \alpha\ f_1(A) \times f_7(A)$.

Some Notes on the VE Algorithm

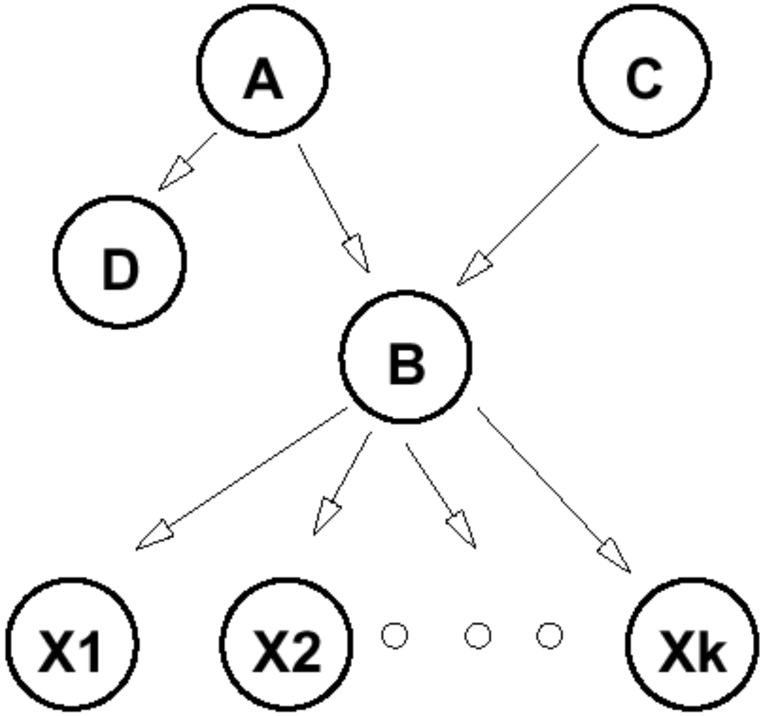
- After iteration j (elimination of Z_j), factors remaining in set F refer only to variables $X_{j+1}, \ldots Z_n$ and Q. No factor mentions an evidence variable E after the initial restriction.
- Number of iterations: linear in number of variables
- Complexity is exponential in the number of variables.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

Elimination Ordering: Polytrees

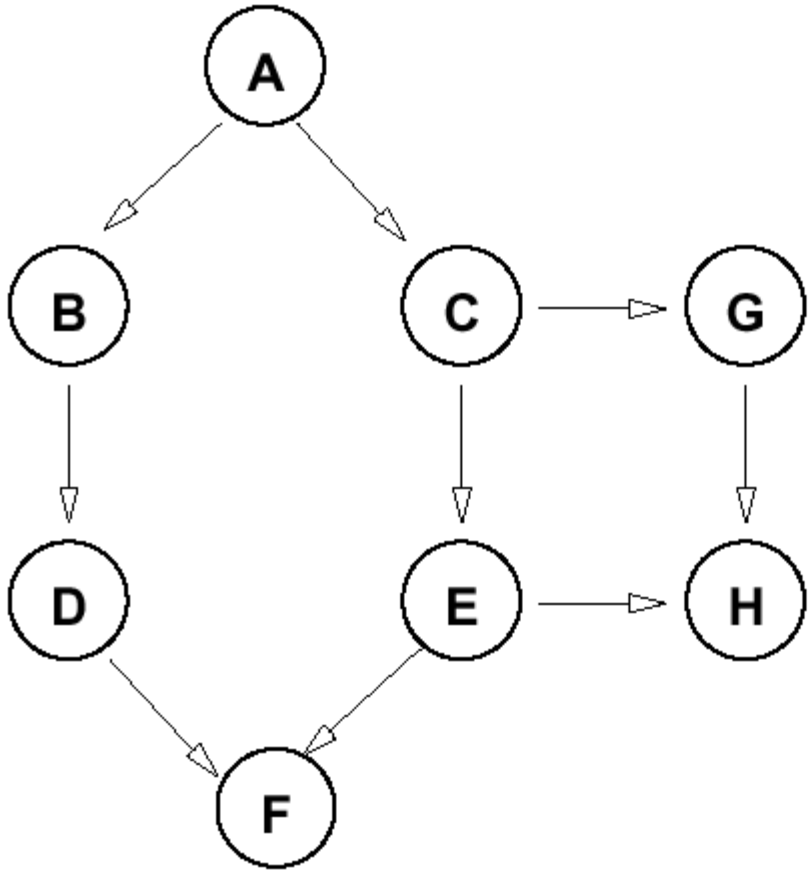
- Inference is linear in size of network
  - ordering: eliminate only "singly-connected" nodes
  - e.g., in this network, eliminate D, A, C, X1,...; or eliminate X1,... Xk, D, A, C; or mix up...
  - result: no factor ever larger than original CPTs
  - eliminating B before these gives factors that include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



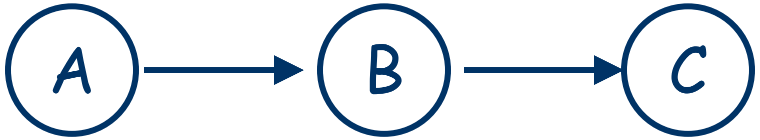
UNIVERSITY OF WATERLOO

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?



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: $f_4(B) = \Sigma_C f_3(B,C) = \Sigma_C \Pr(C|B)$
    - 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

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∈**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

Relevance: Examples

- Query: $P(F)$
  - Relevant: F, C, B, A
- Query: $P(F|E)$
  - Relevant: F, C, B, A
  - Also: E, hence D, G
  - Intuitively, we need to compute $P(C|E) = \alpha P(C)P(E|C)$ to accurately compute $P(F|E)$
- Query: $P(F|E, C)$
  - Algorithm says all variables relevant; but really none except C, F since C cuts off all influence of others)
  - Algorithm is overestimating relevant set