CSE 150A-250A AI: Probabilistic Methods

Lecture 5

Fall 2025

Trevor Bonjour
Department of Computer Science and Engineering
University of California, San Diego

Slides adapted from previous versions of the course (Prof. Lawrence, Prof. Alvarado, Prof Berg-Kirkpatrick)

Agenda

Review

Inference

Exact Inference: Variable Elimination

Polytrees

- Node clustering
- Cutset conditioning

Review

d-separation and conditional independence

- **Theorem**

  $P(X, Y|E) = P(X|E)\, P(Y|E)$ if and only if every *path* from a node in X to a node in Y is *blocked* by E.

- **Definition**

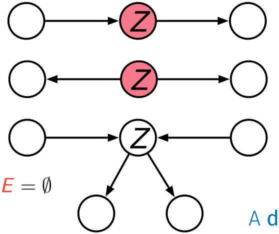
  A path π is **blocked** if there exists a node $Z \in \pi$ for which one of three conditions holds:

(1) $Z \in E$

(2) $Z \in E$

(3) $Z \notin E$

descendants$(Z) \cap E = \emptyset$



causal chain

common cause

unobserved common effect

A **descendant** of Z is *any* node (e.g., child, grandchild) that lies on a directed path from Z.

D-Separation Algorithm

1. Shade all observed nodes $\{Z_1, \ldots, Z_k\}$ in the graph.
2. Enumerate all undirected paths from X to Y.
3. For each path:
    3.1 Decompose the path into triples (segments of 3 nodes).
    3.2 If none of the d-separation blocking conditions apply to any of the triples on the path, then the path is **active** and **d-connects** X and Y. Return $X \not\perp\!\!\!\perp Y \mid \{Z_1, \ldots, Z_k\}$
4. If all paths are blocked , then

$$X \perp\!\!\!\perp Y \mid \{Z_1, \ldots, Z_k\}.$$

Loopy example

A. TRUE or B. FALSE?

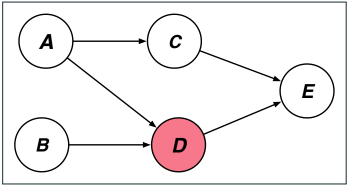
5. $P(B|D, E) \stackrel{?}{=} P(B|D)$

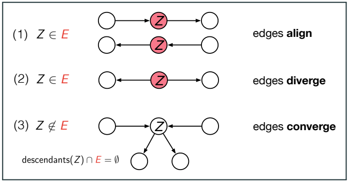
   The evidence is $\{D\}$.
   There are two paths from B to E.

   Path $B \rightarrow D \rightarrow E$
   is blocked by node D,
   satisfying condition (1).

   Path $B \rightarrow D \leftarrow A \rightarrow C \rightarrow E$
   is not blocked by any node.

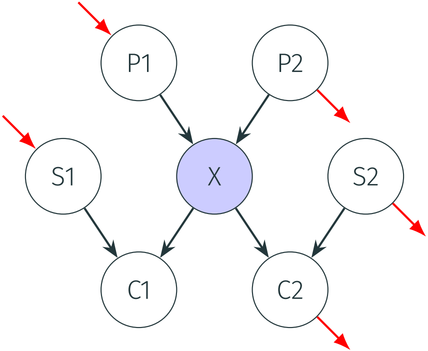
   The statement is   .





Markov Blanket

A Markov Blanket B_X of node X consists of **parents** of X, **children** of X and **”spouses”** (other parents of children of X, but not X) of X.



Every variable is conditionally independent of any other variable given it’s Markov Blanket.

Inference

Inference

- Problem

  Given a set E of evidence nodes, and a set Q of query nodes, how to compute the posterior distribution $P(Q|E)$?

- More precisely

  How to express $P(Q|E)$ in terms of the CPTs $P(X_i|pa(X_i))$ of the BN, which are assumed to be given?

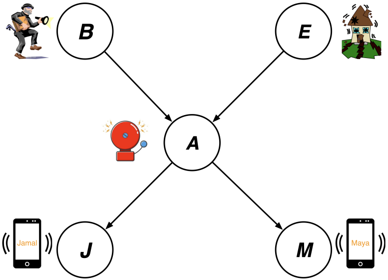
- Tools at our disposal

Bayes rule	marginal independence
marginalization	conditional independence
product rule	

Strategy to compute $P(Q|E)$

Bayes rule	Use to express $P(Q|E)$ in terms of conditional probabilities that respect the order of the DAG.
marginalization	Use to introduce nodes on the left side of the conditioning bar when they need to appear as parents.
product rule	Use to express joint predictions (over multiple variables) in terms of simpler individual predictions.
marginal and conditional independence	Use to remove non-informative variables from the right side of the conditioning bar.

Inference Example



Q. What are the CPTs associated with the DAG shown above?

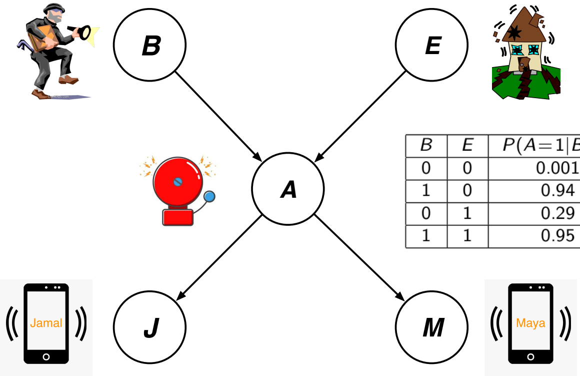
A. P(B|J,M) **B.** P(B) **C.** P(A|B,E)

D. A, B and C **E.** B and C

Inference example

$P(B=1) = 0.001$

$P(E=1) = 0.002$

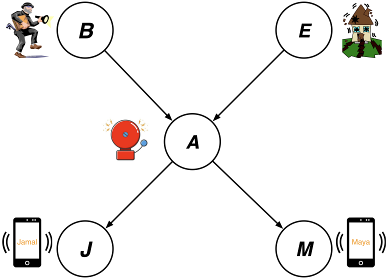


B	E	$P(A=1 \mid B, E)$
0	0	0.001
1	0	0.94
0	1	0.29
1	1	0.95

A	$P(J=1 \mid A)$
0	0.05
1	0.9

A	$P(M=1 \mid A)$
0	0.01
1	0.7

Inference Example



$$P(B|J=1, M=1) = ??$$

Inference Example: Enumeration

$$
\begin{aligned}
P(B|J=1, M=1) &= \frac{P(B, J=1, M=1)}{P(J=1, M=1)} \\
&= \alpha P(B, j, m) \\
&= \alpha \sum_e \sum_a P(B, j, m, E=e, A=a)
\end{aligned}
$$

Inference Example: Enumeration

$$
\begin{aligned}
P(B|j,m) &= \alpha \sum_E \sum_A P(B,j,m,E,A) \\
&= \alpha \sum_E \sum_A P(B)P(E)P(A|B,E)P(j|A)P(m|A) \\
&= \alpha P(B) \sum_E P(E) \sum_A P(A|B,E)P(j|A)P(m|A)
\end{aligned}
$$

Inference Example: Enumeration

$$P(b|j,m) = \alpha P(b) \sum_E P(E) \sum_A P(A|b,E) P(j|A) P(m|A)$$

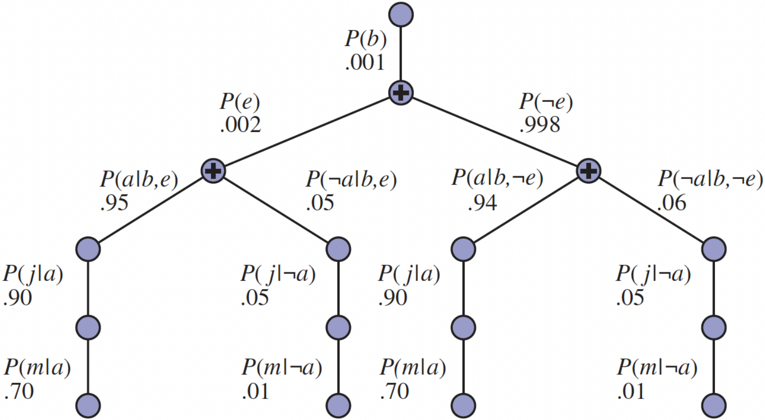


Image Source: *Artificial Intelligence: A Modern Approach* (Russell & Norvig, 2020)

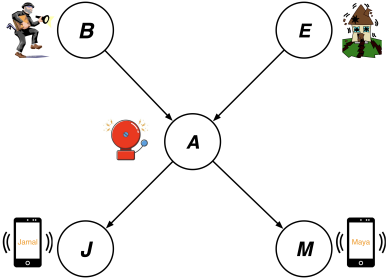
Repeated Computations -> Dynamic Programming

Exact Inference: Variable Elimination

Variable Elimination

- **Idea**: Eliminate redundant calculations by storing intermediate results in ”factors”.
- A **factor** is a function that takes in values of random variables, and produces a number.
- Variable Elimination (VE) works by successively eliminating all non-query, non-evidence variables, one at a time, until only factors involving the query variables remain.
- To eliminate a variable:
  - *join* all factors containing that variable.
  - *sum* out the influence of the variable on the new factor.
  - exploits product form of joint distribution.

VE Example



$P(J) = ??$

VE Example

$$
\begin{aligned}
P(J) &= \sum_{M,A,B,E} P(J, M, A, B, E) \\
&= \sum_{M,A,B,E} P(J|A)P(M|A)P(B)P(A|B, E)P(E) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) \sum_{B} P(B) \sum_{E} P(A|B, E)P(E)
\end{aligned}
$$

VE Example

$$
\begin{aligned}
P(J) &= \sum_{M,A,B,E} P(J, M, A, B, E) \\
&= \sum_{M,A,B,E} P(J|A)P(M|A)P(B)P(A|B, E)P(E) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) \sum_{B} P(B) \sum_{E} P(A|B, E)P(E) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) \sum_{B} (P(B) f1(A, B)
\end{aligned}
$$

VE Example

$$
\begin{aligned}
P(J) &= \sum_{M,A,B,E} P(J, M, A, B, E) \\
&= \sum_{M,A,B,E} P(J|A)P(M|A)P(B)P(A|B, E)P(E) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) \sum_{B} P(B) \sum_{E} P(A|B, E)P(E) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) \sum_{B} (P(B) f1(A, B) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) f2(A)
\end{aligned}
$$

VE Example

$$
\begin{aligned}
P(J) &= \sum_{M,A,B,E} P(J, M, A, B, E) \\
&= \sum_{M,A,B,E} P(J|A)P(M|A)P(B)P(A|B, E)P(E) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) \sum_{B} P(B) \sum_{E} P(A|B, E)P(E) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) \sum_{B} (P(B) f1(A, B) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) f2(A) \\
&= \sum_{A} P(J|A) f3(A)
\end{aligned}
$$

VE Example

$$
\begin{aligned}
P(J) &= \sum_{M,A,B,E} P(J, M, A, B, E) \\
&= \sum_{M,A,B,E} P(J|A)P(M|A)P(B)P(A|B, E)P(E) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) \sum_{B} P(B) \sum_{E} P(A|B, E)P(E) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) \sum_{B} (P(B) f1(A, B)) \\
&= \sum_{A} P(J|A) \sum_{M} P(M|A) f2(A) \\
&= \sum_{A} P(J|A) f3(A) \\
&= f4(J)
\end{aligned}
$$

VE Example

$$P(B|j,m) = \alpha P(B) \sum_E P(E) \sum_A P(A|B,E) P(j|A) P(m|A)$$

A	P(J=1|A)
0	0.05
1	0.9

A	P(M=1|A)
0	0.01
1	0.7

A	$P(j\|A)P(m\|A)$
0	
1	

VE Example

$$P(B|j,m) = \alpha P(B) \sum_E P(E) \sum_A P(A|B,E) P(j|A) P(m|A)$$

A	P(J=1|A)
0	0.05
1	0.9

A	P(M=1|A)
0	0.01
1	0.7

A	$f1(A)$
0	0.05×0.01
1	0.9×0.7

VE Example

$$P(B|j,m) = \alpha P(B) \sum_E P(E) \sum_A P(A|B,E) f1(A)$$

A	P(J=1|A)
0	0.05
1	0.9

A	P(M=1|A)
0	0.01
1	0.7

A	$f1(A)$
0	0.0005
1	0.63

VE Example

$$P(B|j,m) = \alpha P(B)\sum_E P(E)\sum_A P(A|B,E)f1(A)$$

A	$f1(A)$
0	0.0005
1	0.63

B	E	P(A|B,E)
0	0	0.001
1	0	0.94
0	1	0.29
1	1	0.0.95

B	E	$f2(B,E)$
0	0	0.001×0.63+0.999×0.0005
1	0	0.94×0.63+0.06×0.0005
0	1	0.29×0.63+0.71×0.0005
1	1	0.0.95×0.63+0.05×0.0005

VE Example

$$P(B|j,m) = \alpha P(B) \sum_E P(E) f2(B,E)$$

A	$f1(A)$
0	0.05×0.01
1	0.9×0.7

B	E	P(A|B,E)
0	0	0.001
1	0	0.94
0	1	0.29
1	1	0.0.95

B	E	$f2(B,E)$
0	0	0.001
1	0	0.59
0	1	0.18
1	1	0.60

VE Example

$P(B|j,m) = \alpha P(B) \sum_E P(E) f2(B,E)$

$P(B=1) = 0.001$
$P(E=1) = 0.002$

B	E	$f2(B,E)$
0	0	0.001
1	0	0.59
0	1	0.18
1	1	0.60

B	$f3(B)$
0	$0.18 \times 0.002 \times 0.999 + 0.001 \times 0.998 \times 0.999$
1	$0.60 \times 0.002 \times 0.001 + 0.59 \times 0.998 \times 0.001$

VE Example

$$P(B|j,m) = \alpha f3(B)$$

$P(B=1) = 0.001$
$P(E=1) = 0.002$

B	E	$f2(B,E)$
0	0	0.001
1	0	0.59
0	1	0.18
1	1	0.60

B	$f3(B)$
0	0.0013
1	0.0006

VE Example

$P(B|j, m) = \alpha f3(B)$

$P(B = 1) = 0.001$
$P(E = 1) = 0.002$

B	E	$f2(B, E)$
0	0	0.001
1	0	0.59
0	1	0.18
1	1	0.60

B	$f3(B)$
0	0.0013
1	0.0006

VE Example

$\alpha f3(B) \rightarrow P(B|j,m)$

B	$f3(B)$
0	0.0013
1	0.0006

$$N = 0.0013 + 0.0006 = 0.0019$$

B	$P(B|j,m)$
0	0.68
1	0.32

VE: Factors

- Factors are usually represented as a table (therefore an arbitrary function)
- Caution: Factors can look like CPTs, and CPTs can be represented as factors, but factors are **not** necessarily probabilities!
- The values in factors only represent intermediate values in the calculations of some probability - with no real meaning in themselves.

VE: Complexity

Does order of elimination matter?

- In general, yes (but not in the trivial graphs we've been considering)
- Time and space of VE is dominated by the **largest** factor created
- **Heuristic:** Eliminate the variable that will lead to the smallest next factor being created
  - In a **polytree** this leads to **linear** time inference (in size of largest CPT).

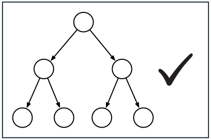
Polytrees

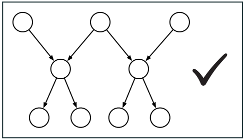
- Definition

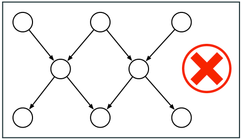
  A **polytree** is a singly connected belief network: between any two nodes there is at most one path.

  Alternatively, a polytree is a belief network without any loops (i.e., undirected cycles).

- Examples





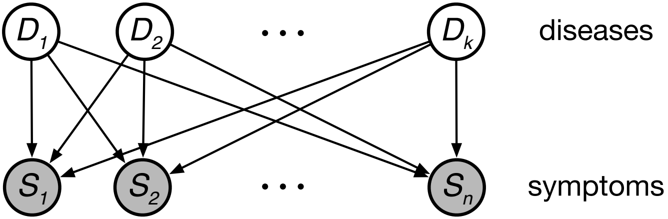


All trees are polytrees.

But not vice versa!

A node in a polytree may have multiple parents.

But many interesting BNs are not polytrees!



How to compute $P(D_i = 1 | S_1, S_2, \ldots, S_n)$?

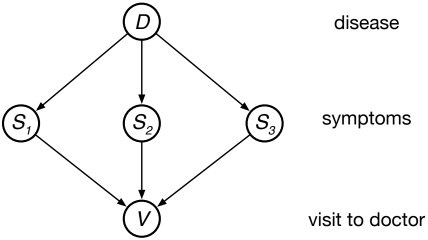
What are general strategies for inference in these BNs?

- Main idea

  Can we transform a loopy BN into a polytree?
  If so, then we can run the exact inference algorithm.

- Example

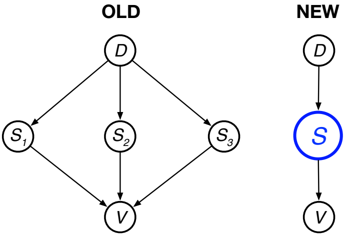
  We'll use a simple BN with binary variables to illustrate two different ways of doing this.



- Key idea

  Merge (well-chosen) nodes in the DAG to remove loops, so that what remains is a polytree.

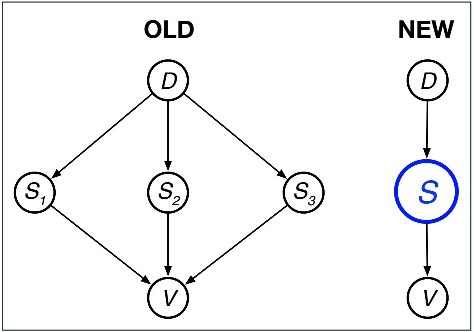
- Example



Cluster nodes $\{S_1, S_2, S_3\}$ into mega-node S.

Merge CPTs at these nodes into mega-CPT $P(S|D)$.

Old versus new nodes



S_3	S_2	S_1	S
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

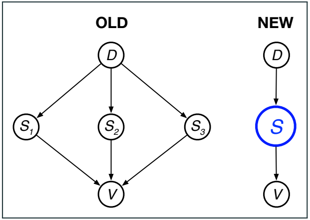
Pro The graph simplifies to a polytree.

Con The node becomes (exponentially) more complex:

$$|S| = |S_1| \cdot |S_2| \cdot |S_3| = 2^3 = 8$$

Old versus new CPTs

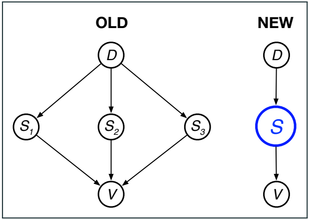
S_3	S_2	S_1	S
0	0	0	**0**
0	0	1	**1**
0	1	0	**2**
0	1	1	**3**
1	0	0	**4**
1	0	1	**5**
1	1	0	**6**
1	1	1	**7**



OLD	NEW
$P(S_1 \| D)$ $P(S_2 \| D)$ $P(S_3 \| D)$	$P(S \| D) = P(S_1, S_2, S_3 \| D) = \prod_{i=1}^{3} P(S_i \| D)$
$P(V \| S_1, S_2, S_3)$	$P(V \| S) = P(V \| S_1, S_2, S_3)$

Worked example

S_3	S_2	S_1	S
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

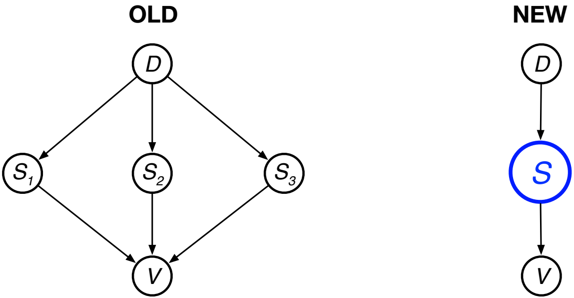


To calculate the new CPTs:

$$
\begin{aligned}
P(S=5|D=0) &= P(S_1=1, S_2=0, S_3=1|D=0) \\
&= P(S_1=1|D=0)\, P(S_2=0|D=0)\, P(S_3=1|D=0)
\end{aligned}
$$

$$
P(V=1|S=5) = P(V|S_1=1, S_2=0, S_3=1)
$$

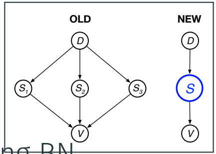
Which nodes to cluster?



In this BN, we can eyeball the right nodes to cluster.

What about in larger BNs?

General case



- **It seems simple enough:**

  Cluster nodes as needed to remove loops.
  Apply exact inference algorithm to the resulting BN.

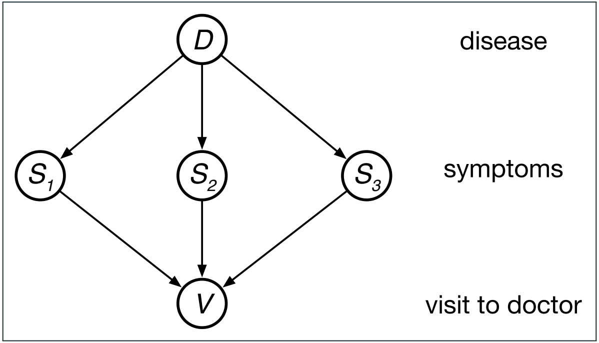
- **But there are tradeoffs:**

  The exact inference algorithm scales linearly in the size of CPTs.
  CPTs grow exponentially when nodes are clustered.

- **Can we optimize this tradeoff?**

  Which clustering leads to maximally efficient inference?
  There is no efficient algorithm to find this!

A different approach?

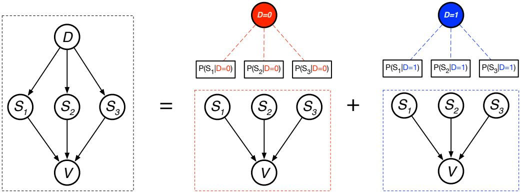


What if, instead of merging nodes, we remove them?

- Key idea

  Remove one or more nodes by instantiating them as evidence.
  Call the exact inference algorithm for each possible instantiation.

- Example

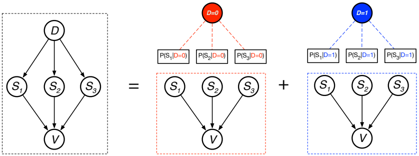


- Definition

  The set of instantiated nodes is called the **cutset**.

Worked example

How to calculate $P(V=1)$?



- Run the exact inference algorithm twice:

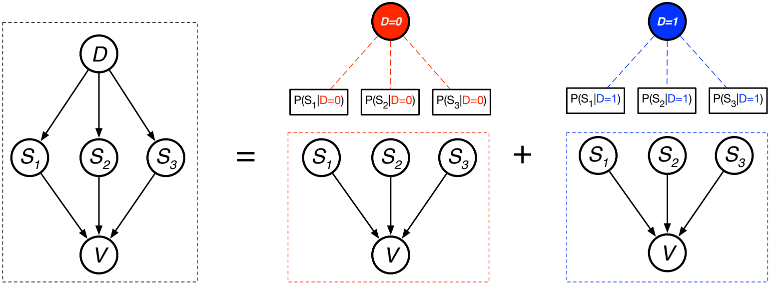
  (1) Compute $P(V=1|D=0)$ from the left polytree.

  (2) Compute $P(V=1|D=1)$ from the right polytree.

- Combine the results:

$$
\begin{aligned}
P(V=1) &= \sum_d P(D=d, V=1) && \boxed{\text{marginalization}} \\
&= \sum_d P(D=d)\, P(V=1|D=d) && \boxed{\text{product rule}} \\
&= P(D=0)P(V=1|D=0) + P(D=1)P(V=1|D=1)
\end{aligned}
$$

How to choose the cutset?



In this BN, we can eyeball the right node to instantiate.

What about in larger BNs?

General case

- **It seems simple enough:**

  Instantiate nodes as needed to remove loops.
  Apply exact inference algorithm to the resulting BNs.

- **But there are tradeoffs:**

  How many times must we run the exact inference algorithm?
  This number grows exponentially with the size of the cutset.

- **Can we optimize this tradeoff?**

  What is the minimal cutset for maximally efficient inference?
  There is no efficient algorithm to compute this!

That's All Folks!