

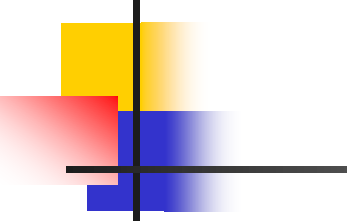


Capturing Independence Graphically; Directed Graphs

COMPSCI 276, Spring 2011
Set 3: Rina Dechter

(Reading: Pearl chapters 3, Darwiche chapter 4)

BAYESIAN NETWORKS

- 
- Represent induced and non-transitive dependencies
 - Employ meaningful parameters
 - *d*-separation: paths that traverse converging arrows are *blocked* until the consequence variable (or any of its descendants) is instantiated.

coin 1 → *bell* ← *coin 2*

- *d*-separation is the usual cutset criterion whenever the arrows are diverging (*height* ← *age* → *reading ability*) or cascaded (*weather* → *wheat crop* → *wheat price*).
1. Given a probability distribution P , can we construct an edge-minimal DAG D that is an I -map of P ?
 2. Given a pair (P, D) can we test whether D is a (minimal) I -map of P ?
 3. Given a DAG D , can we construct a probability distribution P such that D is a perfect map of P ?



d-separation

- To test whether **X** and **Y** are **d-separated** by **Z** in dag G , we need to consider every path between a node in **X** and a node in **Y**, and then ensure that the path is blocked by **Z**.
- A path is blocked by **Z** if **at least** one valve (node) on the path is 'closed' given **Z**.
- A divergent valve or a sequential valve is closed if it is in **Z**
- A convergent valve is closed if it is not on **Z** nor any of its descendants are in **Z**.

DEPENDENCE SEMANTICS FOR BAYESIAN NETWORKS

DEFINITION: If X, Y , and Z are three disjoint subsets of nodes in a DAG D , then Z is said to *d-separate* X from Y , denoted $\langle X \mid Z \mid Y \rangle_D$, if there is no path between a node in X and a node in Y along which the following two conditions hold: (1) every node with converging arrows is in Z or has a descendent in Z and (2) every other node is outside Z .

- If a path satisfies the condition above, it is said to be *active*; otherwise, it is said to be *blocked* by Z .

$$\langle 2 \mid 1 \mid 3 \rangle_D, \neg \langle 2 \mid 15 \mid 3 \rangle_D$$

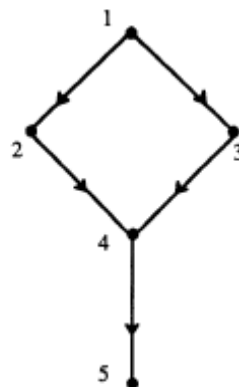


Figure 3.10. A DAG depicting *d-separation*; node 1 blocks the path 2-1-3, while node 5 activates the path 2-4-3.

No path
Is active =
Every path is
blocked



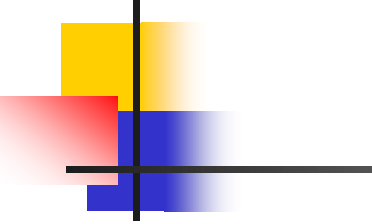
BAYESIAN NETWORKS AS I-MAPS

DEFINITION: A DAG D is said to be an *I-map* of a dependency model M if every d -separation condition displayed in D corresponds to a valid conditional independence relationship in M , i.e., if for every three disjoint sets of vertices X , Y , and Z we have

$$\langle X|Z|Y \rangle_D \Rightarrow I(X, Z, Y)_M.$$

- A DAG is a *minimal I-map* of M if none of its arrows can be deleted without destroying its *I-mapness*.

DEFINITION: Given a probability distribution P on a set of variables U , a DAG $D = (U, \vec{E})$ is called a *Bayesian network* of P iff D is a minimal *I-map* of P .



CONSTRUCTING A BAYESIAN NETWORK FOR ANY GIVEN DISTRIBUTION P

DEFINITION: Let M be a dependency model defined on a set $U = \{X_1, X_2, \dots, X_n\}$ of elements, and let d be an ordering $(X_1, X_2, \dots, X_i, \dots)$ of the elements of U .

- The **boundary strata** of M relative to d is an ordered set of subsets of U , $(B_1, B_2, \dots, B_i, \dots)$, such that each B_i is a Markov boundary of X_i with respect to the set $U_{(i)} = \{X_1, X_2, \dots, X_{i-1}\}$, i.e., B_i is a minimal set satisfying $B_i \subseteq U_{(i)}$ and $I(X_i, B_i, U_{(i)} - B_i)$.
- The DAG created by designating each B_i as parents of vertex X_i is called a *boundary DAG* of M relative to d .

THEOREM 9: [Verma 1986]: Let M be any semi-graphoid (i.e., any dependency model satisfying the axioms of Eqs. (3.6a) through (3.6d)). If D is a boundary DAG of M relative to any ordering d , then D is a minimal *I*-map of M .

Constructing a Bayesian Network for any distribution P

COROLLARY 3: Given a probability distribution $P(x_1, x_2, \dots, x_n)$ and any ordering d of the variables, the DAG created by designating as parents of X_i any minimal set Π_{X_i} of predecessors satisfying

$$P(x_i | \Pi_{X_i}) = P(x_i | x_1, \dots, x_{i-1}), \quad \Pi_{X_i} \subseteq \{X_1, X_2, \dots, X_{i-1}\} \quad (3.27)$$

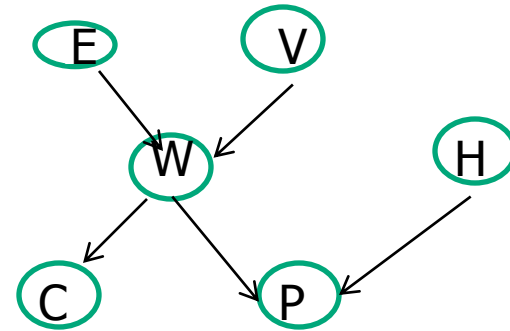
is a Bayesian network of P .

- If P is strictly positive, then all of the parent sets are unique (see Theorem 4) and the Bayesian network is unique (given d).

COROLLARY 4: Given a DAG D and a probability distribution P , a necessary and sufficient condition for D to be a Bayesian network of P is that each variable X be conditionally independent of all its non-descendants, given its parents Π_X , and that no proper subset of Π_X satisfy this condition.

Bayesian networks as i-maps

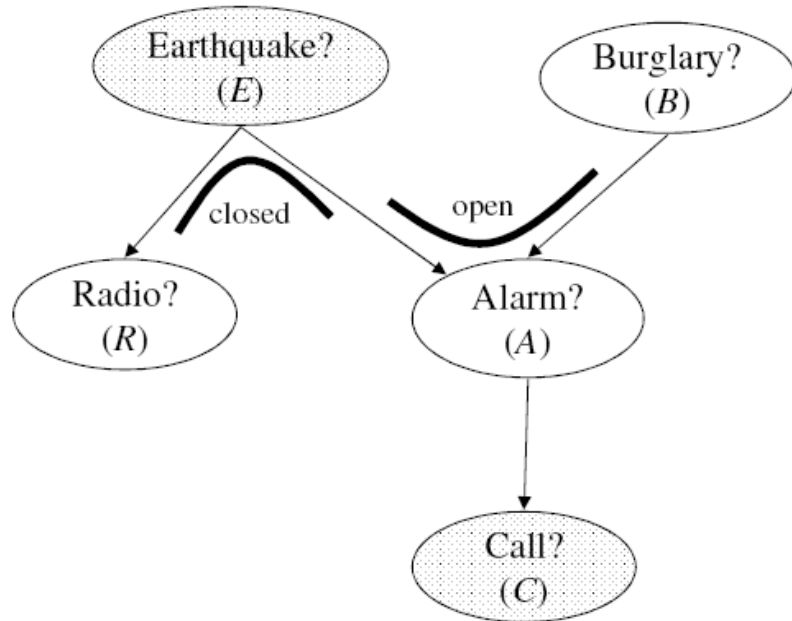
- E: Employment
- V: Investment
- H: Health
- W: Wealth
- C: Charitable contributions
- P: Happiness



Are C and V d-separated give E and P?
Are C and H d-separated?

d-separation

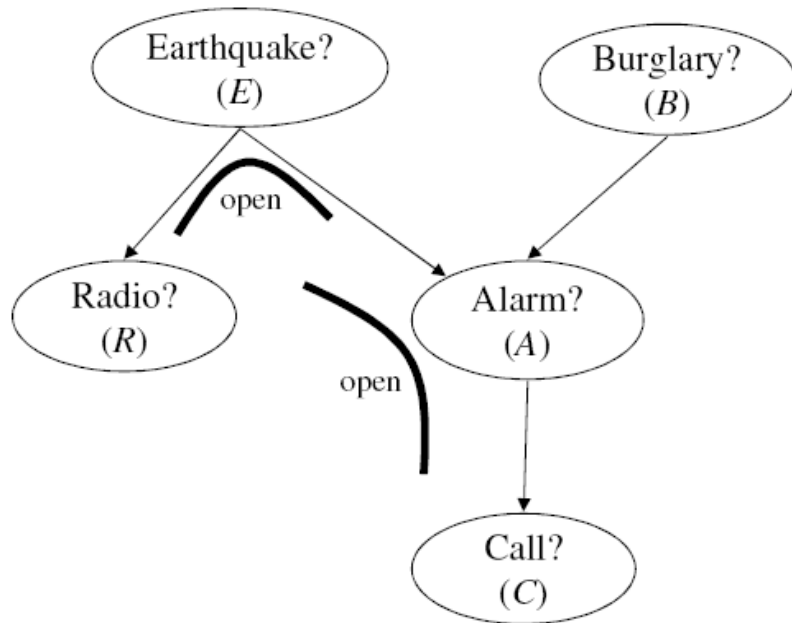
$I_{dsep}(R, EC, B)?$



Example

R and B are d-separated by E and C . The closure of only one valve is sufficient to block the path, therefore, establishing d-separation.

d-separation

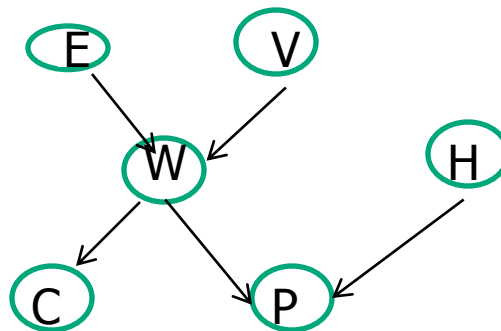


Example

R and *C* are not d-separated since both valves are open. Hence, the path is not blocked and d-separation does not hold.

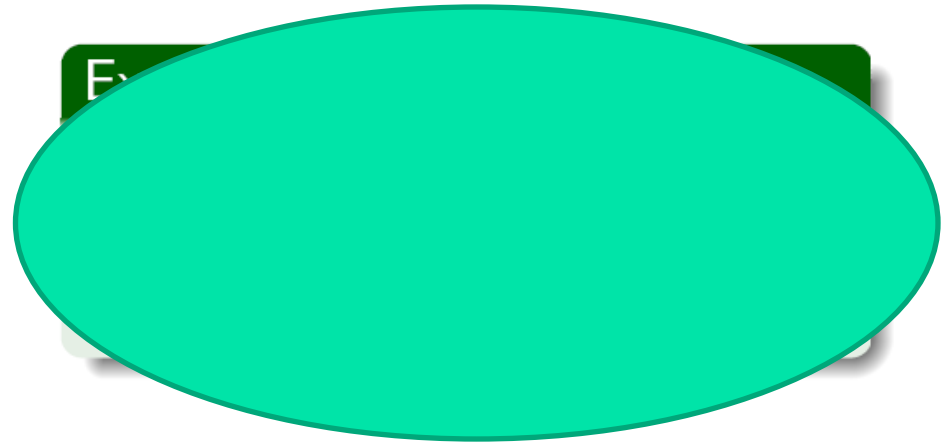
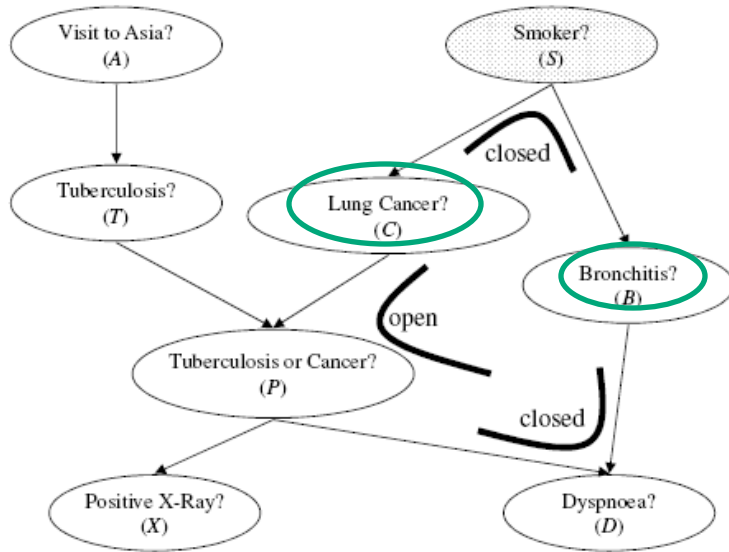
D-separation using ancestral graph

- X is d-separated from Y given Z ($\langle X, Z, Y \rangle_d$) iff:
- Take the the ancestral graph that contains $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ and their ancestral subsets.
- Moralized the obtained subgraph
- Apply regular undirected graph separation
- Check: $(E, \{\}, V), (E, P, H), (C, EW, P), (C, E, HP)$?



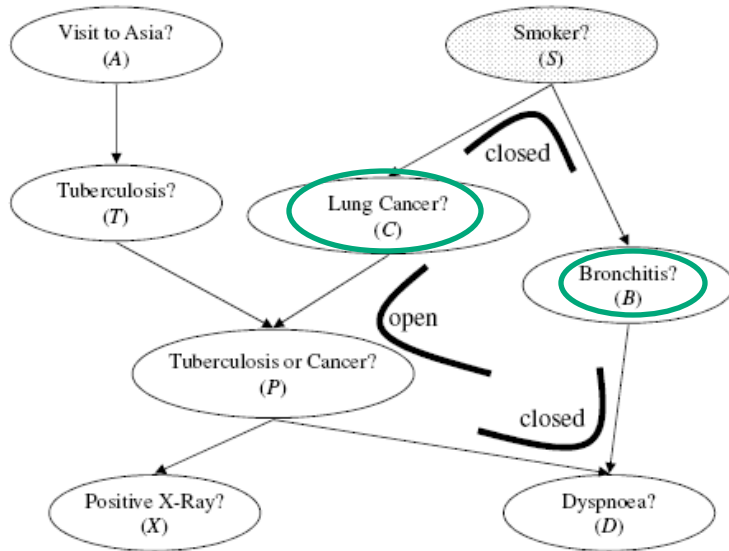
d-separation

$I_{dsep}(\mathbf{C}, \mathbf{S}, \mathbf{B}) = ?$



d-separation

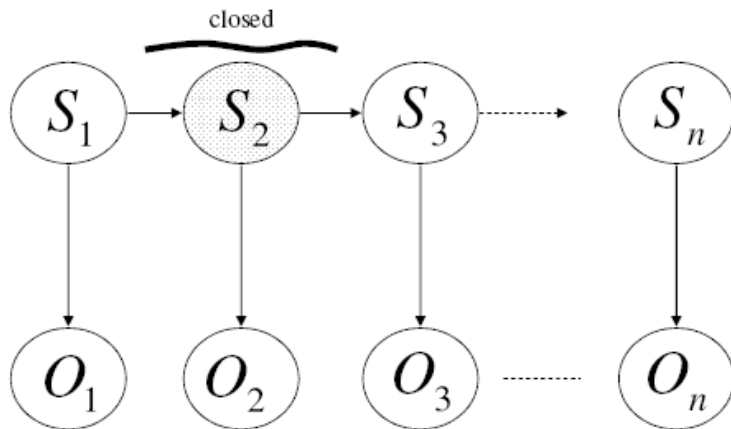
$I_{dsep}(C, S, B)$



Example

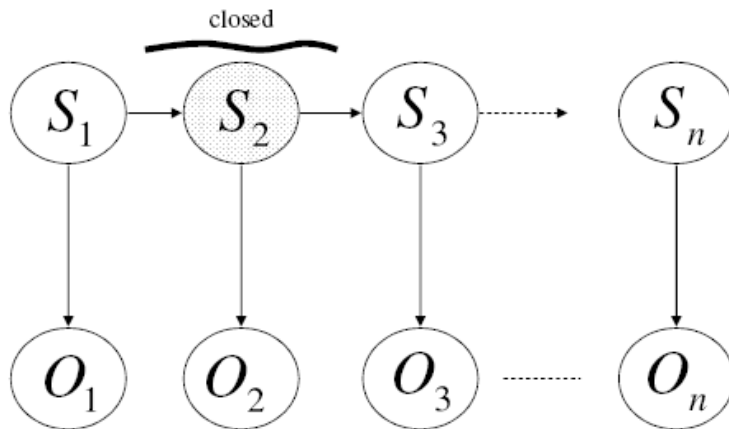
C and B are d-separated by S since both paths between them are blocked by S .

d-separation



$I_{\text{Pr}}(S_1, S_2, \{S_3, S_4\})$ for any probability distribution Pr which is induced by the DAG.

d-separation



Example

Any path between S_1 and $\{S_3, S_4\}$ must have the valve $S_1 \rightarrow S_2 \rightarrow S_3$ on it, which is closed given S_2 . Hence, every path from S_1 to $\{S_3, S_4\}$ is blocked by S_2 , and we have $\text{dsep}_G(S_1, S_2, \{S_3, S_4\})$, which leads to $I_{\text{Pr}}(S_1, S_2, \{S_3, S_4\})$.

$I_{\text{Pr}}(S_1, S_2, \{S_3, S_4\})$ for any probability distribution Pr which is induced by the DAG.

Capturing Independence Graphically

These examples of independence are all implied by a formal interpretation of each DAG as a set of conditional independence statements.

Given a variable V in a DAG G :

$\text{Parents}(V)$ are the parents of V in DAG G , that is, the set of variables N with an edge from N to V .

$\text{Descendants}(V)$ are the descendants of V in DAG G , that is, the set of variables N with a directed path from V to N (we also say that V is an ancestor of N in this case).

$\text{Non_Descendants}(V)$ are all variables in DAG G other than V , $\text{Parents}(V)$ and $\text{Descendants}(V)$. We will call these variables the non-descendants of V in DAG G .

Capturing Independence Graphically

We will formally interpret each DAG G as a compact representation of the following independence statements (**Markovian assumptions**):

$$I(V, \text{Parents}(V), \text{Non_Descendants}(V)),$$

for all variables V in DAG G .

- If we view the DAG as a causal structure, then $\text{Parents}(V)$ denotes the **direct causes** of V and $\text{Descendants}(V)$ denotes the **effects** of V .
- Given the direct causes of a variable, our beliefs in that variable will no longer be influenced by any other variable except possibly by its effects.

Completeness of d-separation

It is not a d-map

d-separation is **not complete** in the following sense:

- Consider a network with three binary variables $X \rightarrow Y \rightarrow Z$.
- Z is not d-separated from X .
- Z can be independent of X in a probability distribution induced by this network.

Example

Choose the CPT for variable Y so that $\theta_{y|x} = \theta_{y|\bar{x}}$.

Y independent of X since

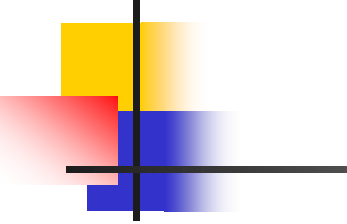
- $\Pr(y) = \Pr(y|x) = \Pr(y|\bar{x})$ and
- $\Pr(\bar{y}) = \Pr(\bar{y}|x) = \Pr(\bar{y}|\bar{x})$.

Z is also independent of X .



Perfect Maps for Dags

- Theorem 10 [Geiger and Pearl 1988]: For any dag D there exists a P such that D is a perfect map of P relative to d-separation.
- Corollary 7: d-separation identifies any implied independency that follows logically from the set of independencies characterized by its dag.



GENERALIZATION OF CELEBRATED MARKOV CHAIN PROPERTY

- If in a sequence of n trials X_1, X_2, \dots, X_n the outcome of any trial X_k (where $2 \leq k \leq n$) depends only on the outcome of the directly preceding trial X_{k-1} , then, given the entire sequence, the outcome of X_k depends only on its predecessor and successor, X_{k-1} and X_{k+1} .

$$I(X_k, X_{k-1}, X_1 \cdots X_{k-2}) \Rightarrow I(X_k, X_{k-1}, X_{k+1}, X_1 \cdots X_{k-2}, X_{k+2} \cdots X_n).$$

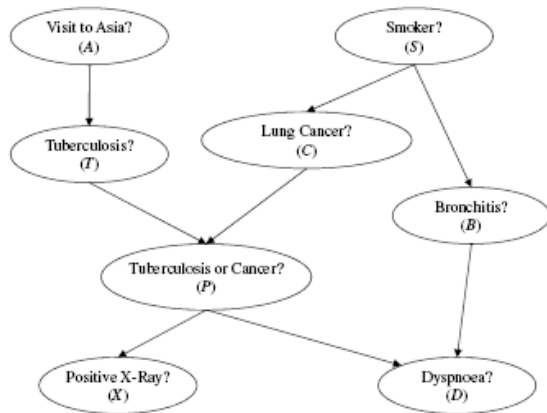
- Theorem 9 generalizes the Markov chain property to non-probabilistic dependencies and to structures that are not chains.

COROLLARY 6: In any Bayesian network, the union of the following three types of neighbors is sufficient for forming a Markov blanket of a node X : the direct parents of X , the direct successors of X , and all direct parents of X 's direct successors.

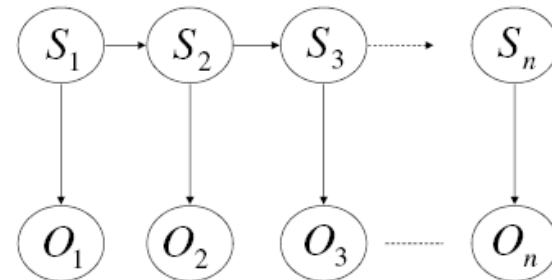
The ancestral undirected graph G of a directed graph D is
An i-ma of D . Is it a Markov network of D ?

Blanket Examples

If \Pr is induced by DAG G , then a Markov blanket for variable X with respect to \Pr can be constructed using its parents, children, and spouses in DAG G . Here, variable Y is a spouse of X if the two variables have a common child in DAG G .

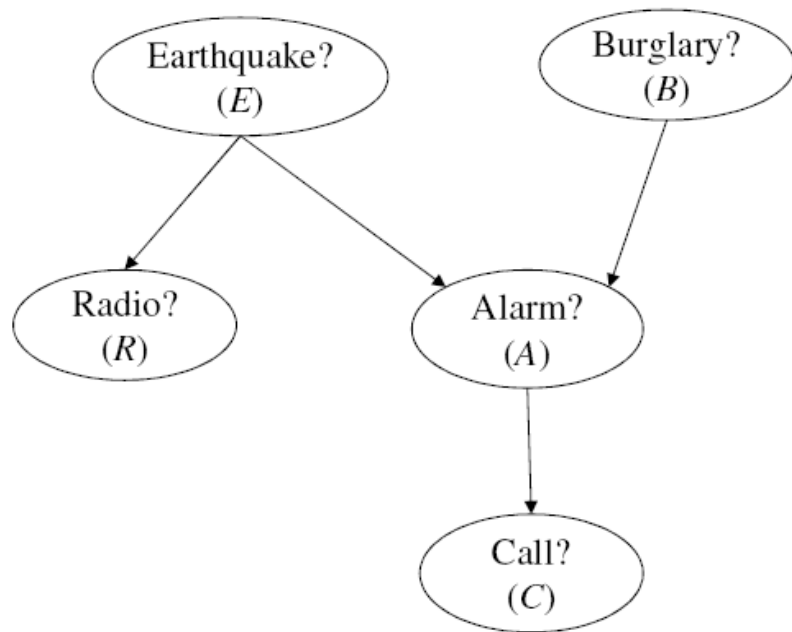


$\{S, P, T\}$ is a Markov blanket for variable C



$\{S_{t-1}, S_{t+1}, O_t\}$ is a Markov blanket for every variable S_t , where $t > 1$

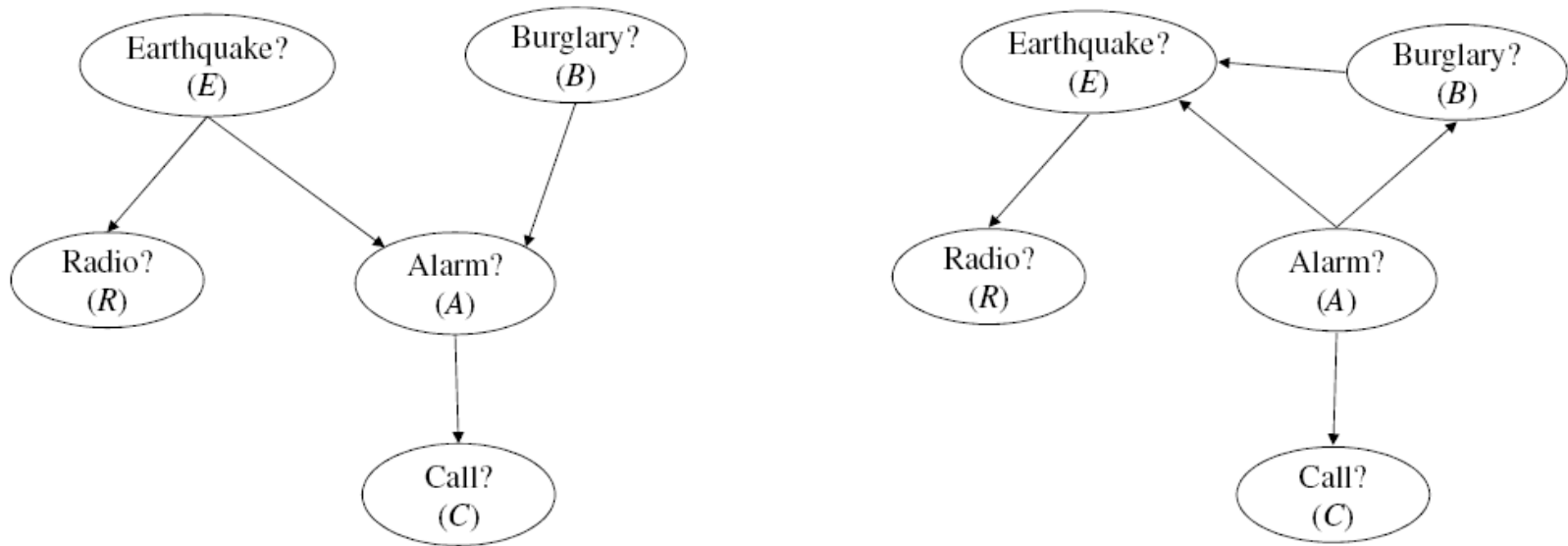
Capturing Independence Graphically



$I(C, A, \{B, E, R\})$
 $I(R, E, \{A, B, C\})$
 $I(A, \{B, E\}, R)$
 $I(B, \emptyset, \{E, R\})$
 $I(E, \emptyset, B)$

Note that variables B and E have no parents, hence, they are marginally independent of their non-descendants.

Capturing Independence Graphically



Every independence which is declared (or implied) by the second DAG is also declared (or implied) by the first one. Hence, if we accept the first DAG, then we must also accept the second.



Bayesian networks as Knowledge-bases

- Given any distribution, P , and an ordering we can construct a minimal i-map.
- The conditional probabilities of x given its parents is all we need.
- In practice we go in the opposite direction: the parents must be identified by human expert... they can be viewed as direct causes, or direct influences.

BAYESIAN NETWORK AS A KNOWLEDGE BASE

STRUCTURING THE NETWORK

- Given any joint distribution $P(x_1, \dots, x_n)$ and an ordering d of the variables in U , Corollary 4 prescribes a simple recursive procedure for constructing a Bayesian network.
- Choose X_1 as a root and assign to it the marginal probability $P(x_1)$ dictated by $P(x_1, \dots, x_n)$.
- If X_2 is dependent on X_1 , a link from X_1 to X_2 is established and quantified by $P(x_2|x_1)$. Otherwise, we leave X_1 and X_2 unconnected and assign the prior probability $P(x_2)$ to node X_2 .
- At the i -th stage, we form the node X_i , draw a group of directed links to X_i from a parent set Π_{X_i} defined by Eq. (3.27), and quantify this group of links by the conditional probability $P(x_i | \pi_{X_i})$.
- The result is a directed acyclic graph that represents all the independencies that follow from the definitions of the parent sets.

- Conversely, the conditional probabilities $P(x_i | \pi_{X_i})$ on the links of the DAG contains all the information necessary for reconstructing the original distribution function.

$$\begin{aligned}
 P(x_1, x_2, \dots, x_n) &= P(x_n | x_{n-1}, \dots, x_1) P(x_{n-1} | x_{n-2}, \dots, x_1) \\
 &\quad \cdots P(x_3 | x_2, x_1) P(x_2 | x_1) P(x_1) \\
 &= \prod_i P(x_i | \pi_{X_i}). \tag{3.28}
 \end{aligned}$$

For example, the distribution corresponding to the DAG of Figure 3.11 can be written by inspection:

$$\begin{aligned}
 &P(x_1, x_2, x_3, x_4, x_5, x_6) \tag{3.29} \\
 &= P(x_6 | x_5) P(x_5 | x_2, x_3) P(x_4 | x_1, x_2) \cdot P(x_3 | x_1) P(x_2 | x_1) P(x_1).
 \end{aligned}$$

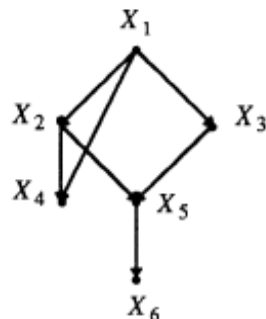


Figure 3.11. A Bayesian network representing the distribution

$$\begin{aligned}
 &P(x_6 | x_5) P(x_5 | x_2, x_3) P(x_4 | x_3, x_2) \\
 &P(x_3 | x_1) P(x_2 | x_1) P(x_1).
 \end{aligned}$$

- In practice, $P(x_1, \dots, x_n)$ is not available.
- The parent sets Π_{X_i} must be identified by human judgment.
- To specify the strengths of influences, assess the conditional probabilities $P(x_i | \pi_{X_i})$ by some functions $F_i(x_i, \pi_{X_i})$ and make sure these assessments satisfy

$$\sum_{x_i} F_i(x_i, \pi_{X_i}) = 1, \quad (3.30)$$

where $0 \leq F_i(x_i, \pi_{X_i}) \leq 1$

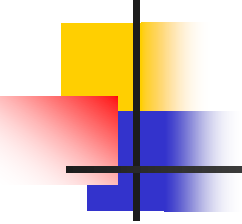
- This specification is complete and consistent because the product form

$$P_a(x_1, \dots, x_n) = \prod_i F_i(x_i, \pi_{X_i}) \quad (3.31)$$

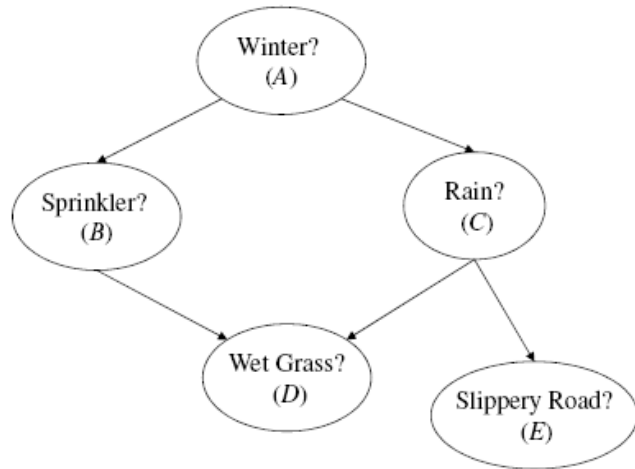
constitutes a joint probability distribution that supports the assessed quantities.

$$P_a(x_i | \pi_{X_i}) = \frac{P_a(x_i, \pi_{X_i})}{P_a(\pi_{X_i})} = \frac{\sum_{x_j \notin (x_i \cup \Pi_{X_i})} P_a(x_1, \dots, x_n)}{\sum_{x_j \in \Pi_{X_i}} P_a(x_1, \dots, x_n)} = F_i(x_i, \pi_{X_i}) \quad (3.32)$$

- DAGs constructed by this method will be called *Bayesian belief networks* or *causal networks* interchangeably.

- 
-
- Assessing the parameters of $P(x_i | \Pi_{X_i})$ requires estimating the likelihood that the event $X_i = x_i$ will occur, given every instantiation of the variables in Π_{X_i}
 - If the number of parents k is large, estimating $P(x | \Pi_{X_i})$ may require canonical models, i.e., prototypical clusters of variables; each requiring about k parameters.
 - Common examples of such structures are noisy OR-gates (i.e., any variable is likely to trigger the effect), noisy AND-gates, and various enabling mechanisms (i.e., variables having no influence of their own except that they enable other influences to take effect).

Parameterizing the Independence Structure



A	B	$\Theta_{B A}$
true	true	.2
true	false	.8
false	true	.75
false	false	.25

A	C	$\Theta_{C A}$
true	true	.8
true	false	.2
false	true	.1
false	false	.9

A	Θ_A
true	.6
false	.4

B	C	D	$\Theta_{D B,C}$
true	true	true	.95
true	true	false	.05
true	false	true	.9
true	false	false	.1
false	true	true	.8
false	true	false	.2
false	false	true	0
false	false	false	1

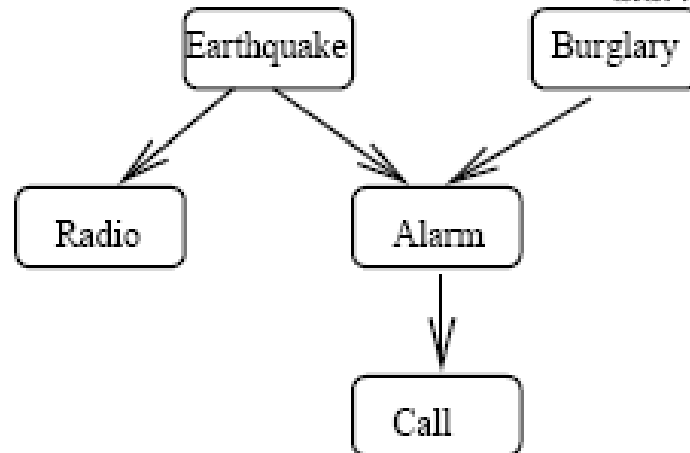
C	E	$\Theta_{E C}$
true	true	.7
true	false	.3
false	true	0
false	false	1

Parameterizing the Independence Structure

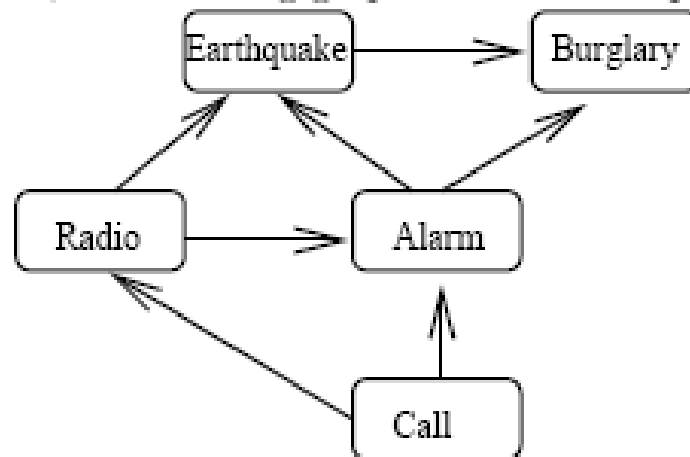
- The CPT $\Theta_{X|\mathbf{U}}$ is exponential in the number of parents \mathbf{U} .
- If every variable can take up to d values, and has at most k parents, the size of any CPT is bounded by $O(d^{k+1})$.
- If we have n network variables, the total number of Bayesian network parameters is bounded by $O(n \cdot d^{k+1})$.
- This number is quite reasonable as long as the number of parents per variable is relatively small.

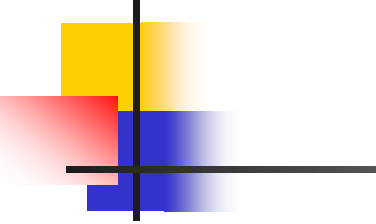
The role of causality

t, C . In this case, we get our original network G_{alarm} :



t, B . In this case, the resulting graph G_1 is not as sparse:





THE ROLE OF CAUSALITY

- The topology of a Bayesian network can be extremely sensitive to the node ordering d .
e.g., a tree in one ordering might become a complete graph if that ordering is reversed.
- Why do people agree on whether two propositions are directly or indirectly related?
Social convention of adopting a standard ordering of events that conforms to the flow of time and causation.
- Why, do we use temporal ordering to organize our memory?
Information about temporal precedence is more readily available than other indexing information?
Networks constructed with temporal ordering are inherently more parsimonious?

DECOMPOSABLE MODELS

- Consider a distribution P having a Markov network in the form of a chain

$$X_1 \text{---} X_2 \text{---} X_3 \text{---} X_4.$$

Expand P in the order dictated by the chain,

$$P(x_1, x_2, x_3, x_4) = P(x_1) P(x_2|x_1) P(x_3|x_1, x_2) P(x_4|x_1, x_2, x_3),$$

Using the conditional independencies encoded in the chain, we obtain

$$P(x_1, x_2, x_3, x_4) = P(x_1) P(x_2|x_1) P(x_3|x_2) P(x_4|x_3).$$

- P is a product of three functions, pairwise conditional probabilities.

- Expand P in the order (X_3, X_2, X_4, X_1) , we get

$$\begin{aligned} P(x_3, x_2, x_4, x_1) &= P(x_3) P(x_2|x_3) P(x_4|x_3, x_2) P(x_1|x_3, x_2, x_4) \\ &= P(x_3) P(x_2|x_3) P(x_4|x_3) P(x_1|x_2), \end{aligned}$$

- As we order the variables from left to right, every variable except the first should have at least one of its graph neighbors to its left.
- (X_1, X_4, X_2, X_3) would not yield the desired product form because X_4 is positioned to the left of its only neighbor, X_3 .

Product form over Markov trees

- Two ways to find product-form distribution for Markov Trees: *directed trees* and *product division*.

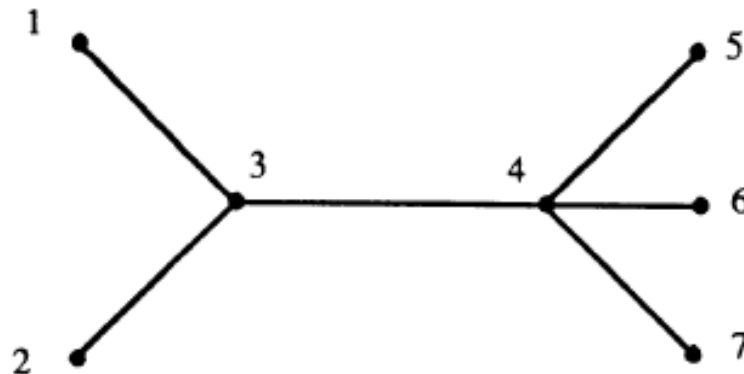


Figure 3.7. An undirected tree of seven variables.

- Choose node 3 as a root and assign to the links arrows pointing away from the root. Write the product distribution by inspection, going from parents to children:

$$P(1, \dots, 7) = P(3) P(1|3) P(2|3) P(4|3) P(5|4) P(6|4) P(7|4). \quad (3.20)$$

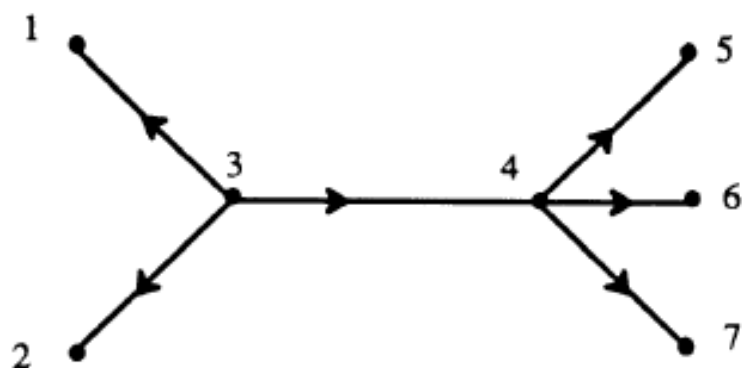


Figure 3.8. A directed tree with root 3.



- The second method: divide the product of the marginal distributions on the edges (i.e., cliques) by the product of the distributions of the intermediate nodes (i.e., the intersections of the cliques).

$$P(1, \dots, 7) = \frac{P(1, 3) P(2, 3) P(3, 4) P(4, 5) P(4, 6) P(4, 7)}{P(3) P(3) P(4) P(4) P(4)}, \quad (3.21)$$

- Each variable in the denominator appears one more time than it appears in the numerator.

Trees are not the only distributions that have product meaningful forms. They can generalize to **join-trees**

JOIN TREES

$$\begin{aligned}
 P(a, b, c, d, e) &= P(a) P(b | a) P(c | a, b) P(d | a, b, c) P(e | a, b, c, d) \\
 &= P(a) P(b | a) P(c | a, b) P(d | b, c) P(e | c) \\
 &= \frac{P(a, b, c) P(b, c, d)}{P(b, c)} \frac{P(c, e)}{P(c)}. \tag{3.22}
 \end{aligned}$$

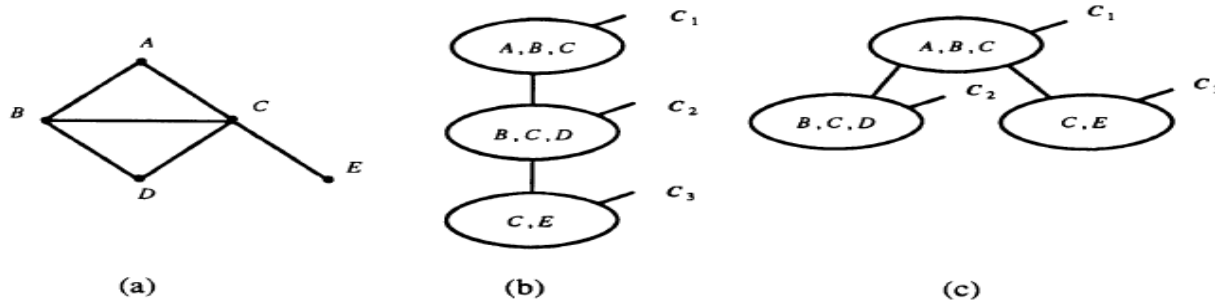



Figure 3.9. Two join trees, (b) and (c), constructed from the cliques of the graph in (a).

- The numerator is a product of the distributions of the cliques, and the denominator is a product of the distributions of their intersections.
- $P(c)$ appears only once in the denominator.
- The unique feature of the graph in Figure 3.9a: there is a tree that is an I -map of P , with vertices corresponding to the cliques of G .



DEFINITION: An undirected graph $G = (V, E)$ is said to be *chordal* if every cycle of length four or more has a chord, i.e., an edge joining two nonconsecutive vertices.

THEOREM 7: Let G be an undirected graph $G = (V, E)$. The following four conditions are equivalent:

1. G is chordal.
2. The edges of G can be directed acyclically so that every pair of converging arrows emanates from two adjacent vertices.
3. All vertices of G can be deleted by arranging them in separate piles, one for each clique, and then repeatedly applying the following two operations:
 - Delete a vertex that occurs in only one pile.
 - Delete a pile if all its vertices appear in another pile.
4. There is a tree T (called a *join tree*) with the cliques of G as vertices, such that for every vertex v of G , if we remove from T all cliques not containing v , the remaining subtree stays connected. In other words, any two cliques containing v are either adjacent in T or connected by a path made entirely of cliques that contain v .

Any induced graph is chordal

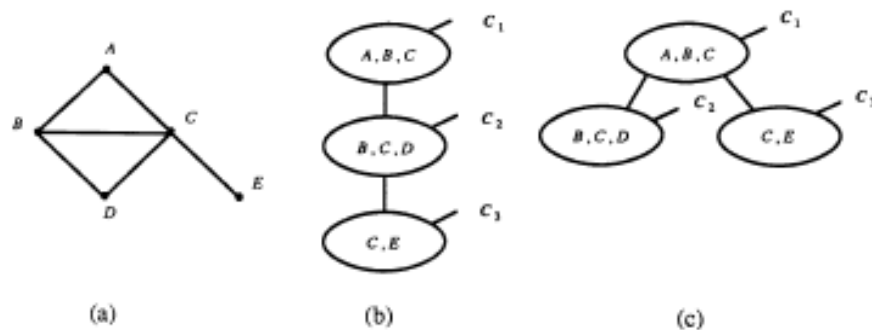
The running intersection property

GRAPH TRIANGULATION (FILL-IN) ALGORITHM: Tarjan and Yannakakis [1984]

1. Compute an ordering for the nodes, using a *maximum cardinality search*, i.e., number vertices from 1 to $|V|$, in increasing order, always assigning the next number to the vertex having the largest set of previously numbered neighbors (breaking ties arbitrarily).
 2. From $n = |V|$ to $n = 1$, recursively fill in edges between any two nonadjacent parents of n , i.e., neighbors of n having lower ranks than n (including neighbors linked to n in previous steps). If no edges are added the graph is chordal; otherwise, the new filled graph is chordal.
- Given a graph $G = (V, E)$ we can construct a join tree using the following procedure.

ASSEMBLING A JOIN TREE

1. Use the fill-in algorithm to generate a chordal graph G' (if G is chordal, $G = G'$).
2. Identify all cliques in G' . Since any vertex and its parent set (lower ranked nodes connected to it) form a clique in G' , the maximum number of cliques is $|V|$.
3. Order the cliques C_1, C_2, \dots, C_t by rank of the highest vertex in each clique.
4. Form the join tree by connecting each C_i to a predecessor C_j ($j < i$) sharing the highest number of vertices with C_i .



EXAMPLE: Consider the graph in Figure 3.9a. One maximum cardinality ordering is (A, B, C, D, E) .

- Every vertex in this ordering has its preceding neighbors already connected, hence the graph is chordal and no edges need be added.
- The cliques are ranked C_1 , C_2 , and C_3 as shown in Figure 3.9b.
- $C_3 = \{C, E\}$ shares only vertex C with its predecessors C_2 and C_1 , so either one can be chosen as the parent of C_3 .
- These two choices yield the join trees of Figures 3.9b and 3.9c.
- Now suppose we wish to assemble a join tree for the same graph with the edge (B, C) missing.
- The ordering (A, B, C, D, E) is still a maximum cardinality ordering, but now when we discover that the preceding neighbors of node D (i.e., B and C) are nonadjacent, we should fill in edge (B, C) .
- This renders the graph chordal, and the rest of the procedure yields the same join trees as in Figures 3.9b and 3.9c.

DECOMPOSABLE DISTRIBUTIONS

DEFINITION: A probability model P is said to be **decomposable** if it has a minimal I -map that is chordal. P is said to be **decomposable relative to a graph G** if the following two conditions are met:

- i. G is an I -map of P .
- ii. G is chordal.

LEMMA 1: If P is decomposable relative to G , then any join tree T of the cliques of G is an I -map relative to P . In other words, if C_X , C_Y , and C_Z are three disjoint sets of vertices in T , and X , Y , and Z are their corresponding sets of variables in G , then $I(X, Z, Y)_P$ whenever C_Z separates C_X from C_Y in T (written $\langle C_X | C_Z | C_Y \rangle_T$).

THEOREM 8: If P is decomposable relative to G , then the joint distribution of P can be written as a product of the distributions of the cliques of G divided by a product of the distributions of their intersections.

Proof: Let T be the join tree of the cliques of G , and let $(C_1, C_2, \dots, C_l, \dots)$ be an ordering of the cliques that is consistent with T

$$P(x_1, x_2, \dots, x_n) = \prod_i P(c_i | c_1, \dots, c_{i-1}) = \prod_i P(c_i | c_{j(i)}) \quad (3.24)$$

$$= \prod_i P(c_i | c_i \cap c_{j(i)}) \quad (3.25)$$

$$= \prod_i \frac{P(c_i)}{P(c_i \cap c_{j(i)})} \quad (3.26)$$

•Decomposable models have a probability distribution expressible in product form

•To make P decomposable relative to some chordal graph G , it is enough to triangulate its Markov network (which originally may not be chordal).

•Lemma 1 is important because we have a tree of clusters that is an i -map of the original distribution and allows the product form.

•As we will see: this tree of clusters, allows message propagation for query processing along the tree of clusters.

HOW EXPRESSIVE ARE BAYESIAN NETWORKS?

1. Can all dependencies that are representable by a Markov network also be represented by a Bayesian network?
2. How well can Bayesian networks represent the type of dependencies induced by probabilistic models?

- A diamond-shaped Markov network asserts $I(A, BC, D)$ and $I(B, AD, C)$.

No Bayesian network can express these two relationships simultaneously and exclusively.

- Every chordal graph can be oriented so that the tails of every pair of converging arrows are adjacent.

Hence, every dependency model that is isomorphic to a chordal graph is also isomorphic to a DAG.

