Probabilistic Reasoning

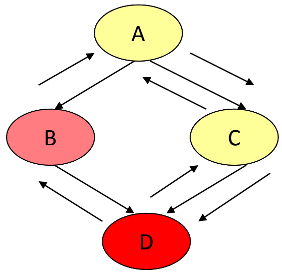
Unit # 5

Intractability of Inference in BN (Laskey)

- **Belief propagation is NP-hard (Cooper, 1988)**
  - **NP-hard problems are problems which are at least as computationally complex as NP-complete problems**
  - **NP-complete is a class of decision problems that are computationally complex**
  - **There is a theorem which says that if there is an algorithm which solves one NP-Complete problem in polynomial time, then any NP-complete problem can be solved in polynomial time**
  - **No such algorithm has yet been found, and most people believe there is none**
- **Although the general problem appears to be very hard, tractable algorithms are available for certain types of networks**
  - **Singly connected networks**
  - **Some sparsely connected networks**
  - **Networks amenable to various approximation algorithms**

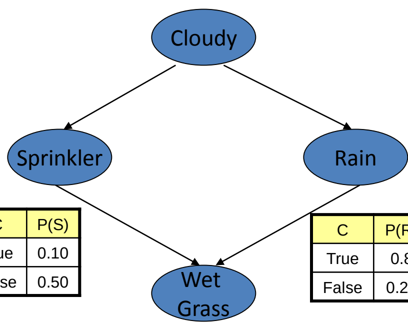
Inference In Multiply Connected Networks

- We get evidence on node D.
- As a result we need to update the marginal probabilities of the rest of the variables
- How to do it?



Exact Inference: Clustering

P(C)=0.5

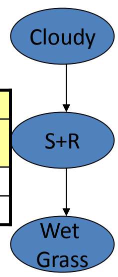


C	P(S)
True	0.10
False	0.50

C	P(R)
True	0.8
False	0.20

S	R	P(W)
True	True	0.9
True	False	0.8
False	True	0.4
False	False	0.1

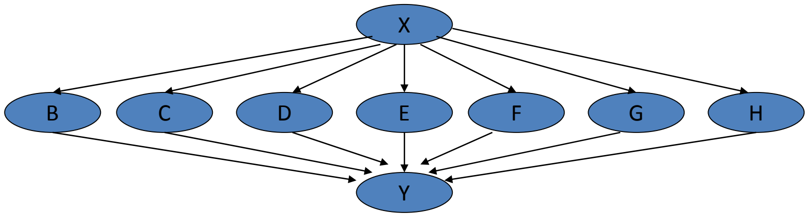
P(C)=0.5



C	S+R			
	True True	True False	False True	False False
True	0.08	0.02	0.72	0.18
False	0.10	0.40	0.10	0.40

S+R	P(W)
True True	0.99
True False	0.90
False True	0.90
False False	0.00

Problem with the Clustering Approach



- We are back to the same number of computations as was in the enumeration approach

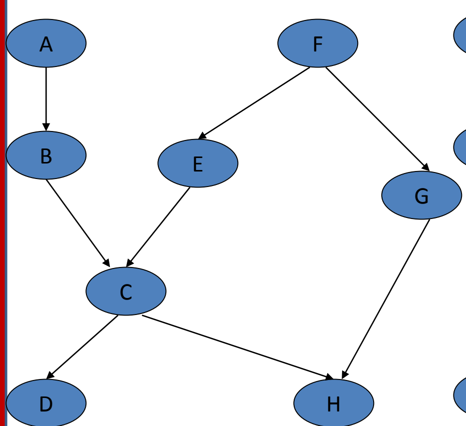
Overview of Junction/Clique Tree Algorithm (Laskey)

- The basic idea is to transform the graph into clusters of nodes so that the graph of clusters is singly connected and has the *junction tree property* (this property ensures that evidence propagates correctly)
- The junction tree becomes a permanent part of the knowledge representation, and changes only if the graph changes
- Constructing a junction tree from a Bayesian network is inherently non-modular, but the junction tree itself is a modular representation
- Beliefs are propagated in the junction tree using a local message-passing algorithm

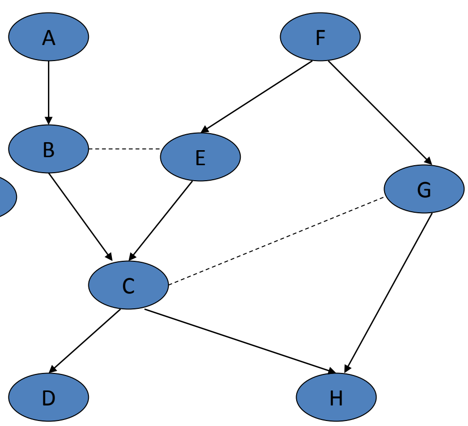
How to Construct a JT

- Moralize the graph by adding links between parents of common children.
- Convert the given BN into an undirected graph by removing the arrowheads.
- Triangulate the graph.
- Order the nodes by maximum cardinality search.
- Find the cliques of the triangulated graph.
- Arrange as a junction tree.

Example I (Laskey)

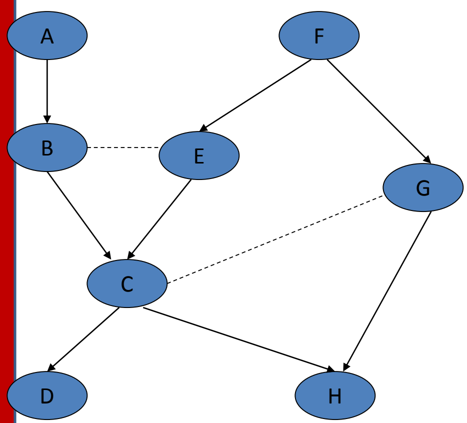


Example Bayesian Network

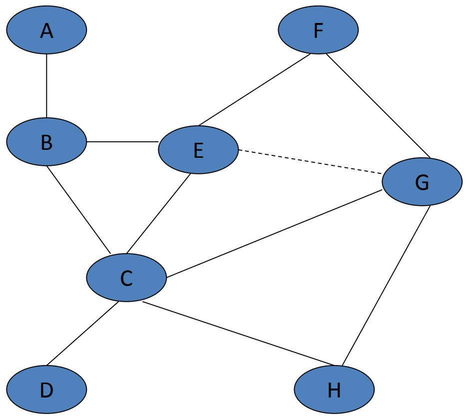


Moralize the graph

Example I (Cont’d)

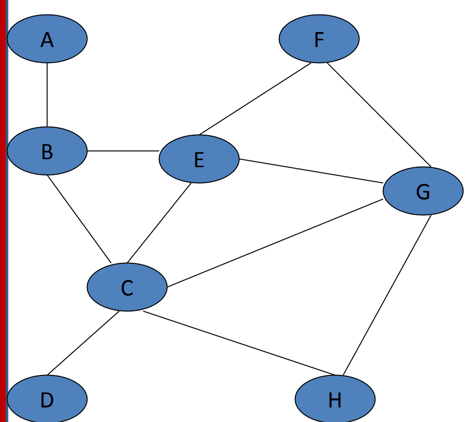


Moralize the graph



Triangulate the graph

Example I (Cont’d)

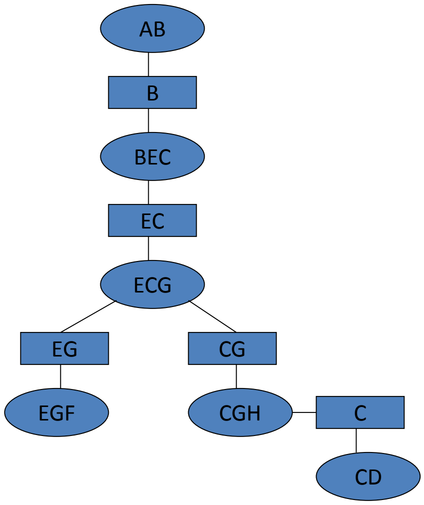


Order Nodes in the Graph

Clique	Max. Vertex
AB	2
BEC	4
ECG	5
EGF	6
CGH	7
CD	8

Example I(Cont'd)

Clique	Max. Vertex
AB	2
BEC	4
ECG	5
EGF	6
CGH	7
CD	8

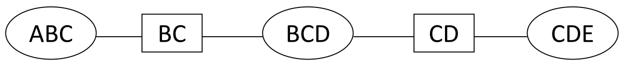


Properties of Clique Tree

- A clique tree is an undirected tree, where each node represents a set of variables, which is called a clique.
- A clique separator is the intersection between its two neighbouring cliques.
- Note that not every clique tree is a junction tree.
- A clique tree is a junction tree if and only if it has the *running intersection property or the junction tree property.*

Properties of a Junction Tree (Zhang)

- The junction tree has the following characteristics:
  - it is an undirected tree, its nodes are clusters of variables
  - given two clusters, C_1 and C_2, every node on the path between them contains their intersection $C_1 \cap C_2$
  - a Separator, S, is associated with each edge and contains the variables in the intersection between neighbouring nodes



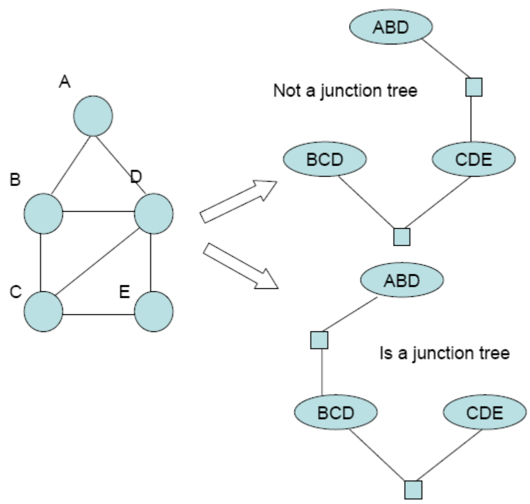
Junction Tree Property (Williams)

- A clique tree is a junction tree if it has the following junction tree property:
  - if a node appears in two cliques, it appears everywhere on the path between the cliques.
- *Mathematically, a clique tree possesses the junction tree property if for every pair of cliques V* and *W, all cliques on the (unique) path between V and W contain* $V \cap W$.
- For every triangulated graph there exists a clique tree which obeys the junction tree property
- Not all clique trees are junction trees

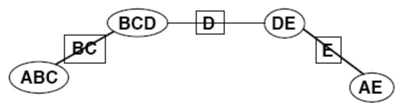
How to Construct a JT

- Moralize the graph by adding links between parents of common children.
- Convert the given BN into an undirected graph by removing the arrowheads.
- Triangulate the graph.
  - If there is a cycle of length > 3 with no chord, add a chord between two non-adjacent vertices in the cycle.
- Order the nodes by maximum cardinality search.
  - Choose any node in the graph and label it 1.
  - For i=2 to n
    - Choose the node with the most labeled neighbors and label it i.
    - If any two labeled neighbors of i are not adjacent to each other
      - Add a link between those neighbors
      - Restart the algorithm.
- Find the cliques of the triangulated graph.
- Arrange as a junction tree.

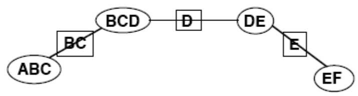
Junction Tree Examples (Jordan)



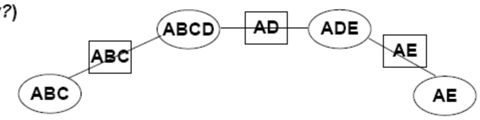
Junction Tree Examples (Laskey)



A cluster tree that is not a junction tree
(*Why not?*)

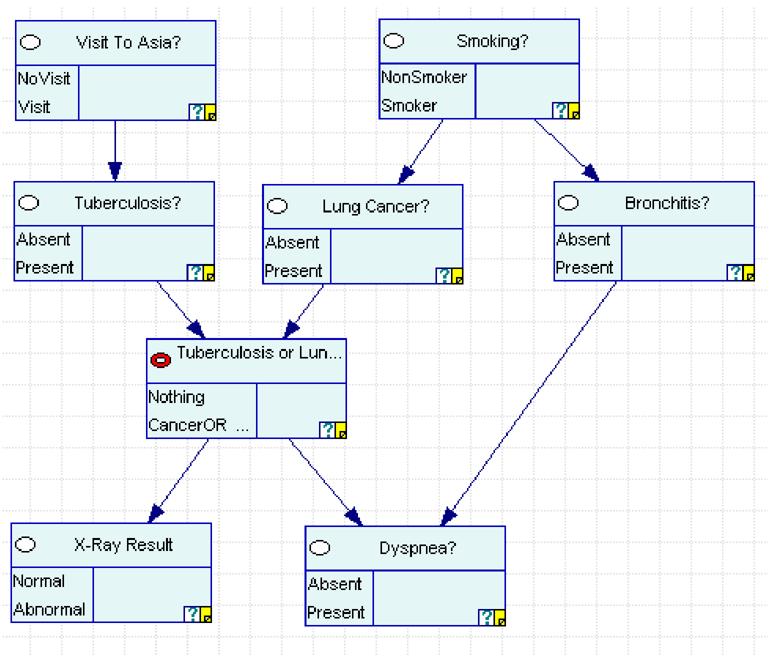


A junction tree (*Why?*)



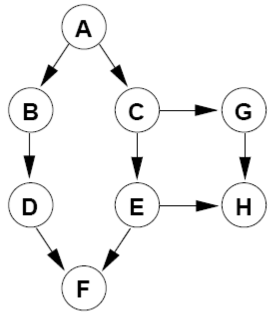
Also a junction tree (*Why?*)

•



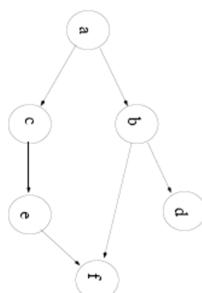
Example III
(Huang & Darwiche)

- Consider this BN and transform it into a junction tree by performing moralization, triangulation and maximum cardinality search.

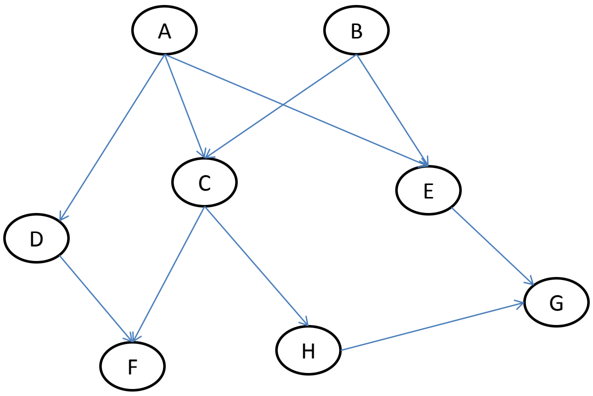


Example IV

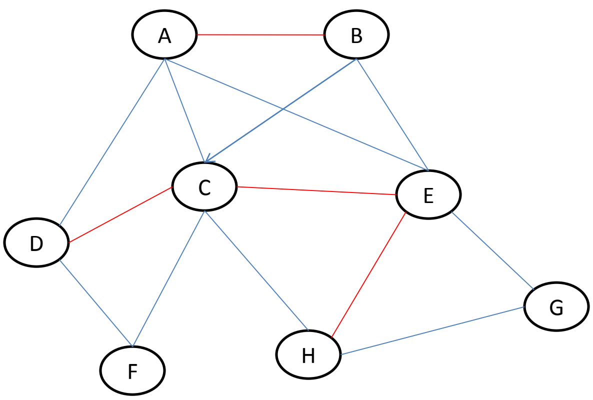
- Consider this BN and transform it into a junction tree by performing moralization, triangulation and maximum cardinality search.



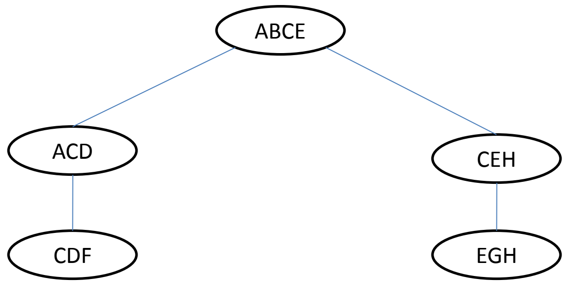
Example V
(Source: Laskey)



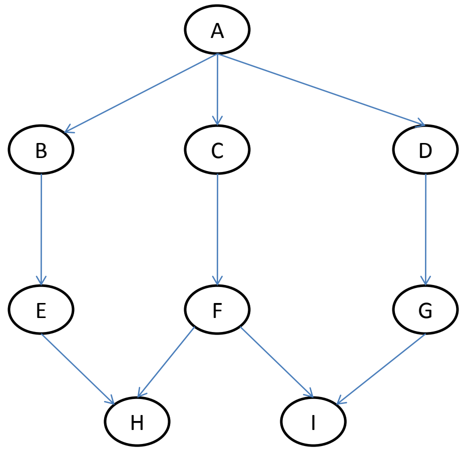
Example V (Cont'd)



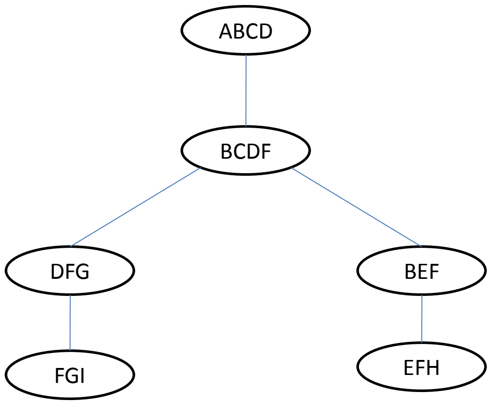
Example V (Cont'd)



Example VI (Source: Korb & Nicholson)



Example VI (Cont'd)



Approximate Inference Mechanisms

- Simulation Based Schemes
  - Logic Sampling
  - Likelihood Weighting
- Approximate Algorithms
  - Loopy Belief Propagation

Logic Sampling (Laskey)

- Logic sampling takes random draws from the joint distribution of all variables in the network
- Each node keeps a running count of how many times each of its states is drawn
- To take one random draw (one observation on all variables):
  - Sample the nodes in order (so parents of a node are sampled before the node)
  - For root node(s), select state with probability P(state)
  - For non-root node(s) select state with probability P(state|sampled states of parents)
  - After all nodes have been sampled
    - Check whether sampled values of evidence values match observed states
    - If they don't match, throw away the draw
    - If they match, increment the count for sampled state of each non-evidence node
- Estimate P(Xt|Xe) by observed frequency among the retained samples

Algorithm (From Jensen and Nielsen)

1. Let $(X_1, \ldots, X_n)$ be a topological ordering of the variables.
2. For $j = 1$ to N:
   a) For $i = 1$ to n:
      - Sample a state x_i for X_i using $P(X_i \mid \text{pa}(X_i) = \pi)$, where π is the configuration already sampled for $\text{pa}(X_i)$.

   b) If $\mathbf{x} = (x_1, \ldots, x_n)$ is consistent with $\mathbf{e}$, then

   $$N(X_k = x_k) := N(X_k = x_k) + 1,$$

   where x_k is the state that was sampled for X_k.
3. Return:

$$P(X_k = x_k \mid \mathbf{e}) \approx \frac{N(X_k = x_k)}{\sum_{x \in \text{sp}(X_k)} N(X_k = x)}.$$

Likelihood Weighting (Laskey)

- Logic sampling can be very inefficient because it throws out so many observations.
- It can be modified to avoid throwing out observations.
- To take one random draw (one observation on all variables):
  - The evidence variables are fixed to their observed states.
  - Generate random samples for the non-evidence variables.
  - Weight the observation by a "sampling weight" that depends upon the evidence.
- Estimate P(Xt|Xe) as the weighted average.

Algorithm (From Jensen and Nielsen)

1. Let $(X_1, \ldots, X_n)$ be a topological ordering of the variables.
2. For $j = 1$ to N:
   a) w:=1.
   b) For $i = 1$ to n:
      - Let $\mathbf{x}'$ be the configuration of $(X_1, \ldots, X_{i-1})$ specified by $\mathbf{e}$ and the previous samples.
      - If $X_i \notin \mathcal{E}$, then:
        - Sample a state x_i for X_i using $P(X_i \mid \text{pa}(X_i) = \pi)$, where $\text{pa}(X_i) = \pi$ is consistent with $\mathbf{x}'$.
      - else
        $w := w \cdot P(X_i = e_i \mid \text{pa}(X_i) = \pi)$, where $\text{pa}(X_i) = \pi$ is consistent with $\mathbf{x}'$.
   c) $N(X_k = x_k) := N(X_k = x_k) + w$, where x_k is the sampled state for X_k.
3. Return:

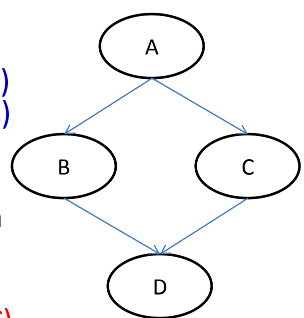
$$P(X_k = x_k \mid \mathbf{e}) \approx \frac{N(X_k = x_k)}{\sum_{x \in \text{sp}(X_k)} N(X_k = x)}.$$

Loopy Belief Propagation

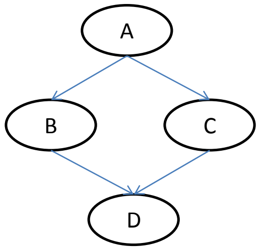
- Originally the message-passing belief propagation was developed for polytrees.
- In loopy belief propagation, we make certain independence assumption (assuming that no loop exists when one exist)

Concept Behind Loopy Belief Propagation

- $P(B) = P(B|A)P(A)+P(B|\sim A)P(\sim A)$
- $P(C) = P(C|A)P(A)+P(C|\sim A)P(\sim A)$
- $P(D) = P(D|B,C)P(B,C) + P(D|B,\sim C)P(B,\sim C) + P(D|\sim B,C)P(\sim B,C) + P(D|\sim B,\sim C)P(\sim B,\sim C)$
- If we assume that B and C are independent (when they are not as obvious from the structure) then we can compute the above equation as
- $P(D) = P(D|B,C)P(B)P(C) + P(D|B,\sim C)P(B)P(\sim C) + P(D|\sim B,C)P(\sim B)P(C) + P(D|\sim B,\sim C)P(\sim B)P\sim C)$



Examples



SET I

P(A) = 0.3
P(B|A)=0.7, P(B|~A)=0.05
P(C|A)=0.2, P(C|~A)=0.9
P(D|B,C) =0.9, P(D|B,~C) =0.6
P(D|~B,C)=0.8, P(D|~B,~C)=0.02

SET II

P(A) = 0.3
P(B|A)=0.9, P(B|~A)=0.01
P(C|A)=0.02, P(C|~A)=0.95
P(D|B,C) =0.99, P(D|B,~C) =0.95
P(D|~B,C)=0.90, P(D|~B,~C)=0.02

- Using Set I and II, compute the probability of D, P(D), using loopy belief propagation and compare the results with the ones obtained through GeNIe (exact, logic sampling and likelihood weighting). What do you observe?