Lecture 10: Junction Tree Algorithm
CSCI 780 - Machine Learning

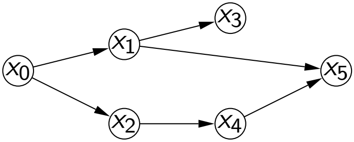
Andrew Rosenberg

March 4, 2010

Today

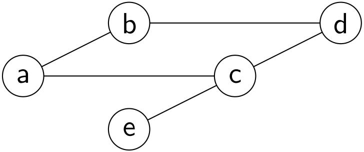
- Junction Tree Algorithm
    - Efficient calculation of marginals in a graphical model

Example of Graphical Model.



$$\theta(x_i, \pi_i) = \frac{m(x_i, \pi_i)}{m(\pi_i)}$$

Efficient Computation of Marginals



- Pass **messages** (small tables) around the graph.
- The **messages** will be small functions that propagate potentials around an undirected graphical model.
- The inference technique is the **Junction Tree Algorithm**

Junction Tree Algorithm

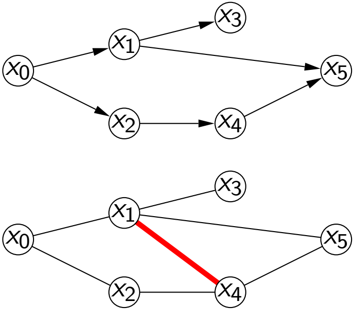
- Moralization
- Introduce Evidence
- Triangulate
- Construct Junction Tree
- Propagate Probabilities

Junction Tree Algorithm

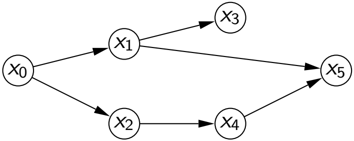
- Moralization
- Introduce Evidence
- Triangulate
- Construct Junction Tree
- Propagate Probabilities

Moralization

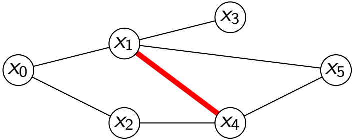
- Converts a directed graph to an undirected graph.
- Moralization "marries" the parents.
  - Insert an undirected edge between two nodes that have a child in common.
  - Replace all directed edges with undirected edges.



Moralization

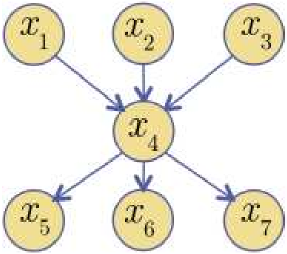


$$p(x_0)p(x_1|x_0)p(x_2|x_0)p(x_3|x_1)p(x_4|x_2)p(x_5|x_1,x_4)$$

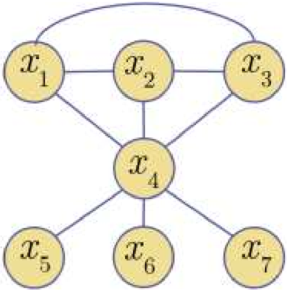


$$\frac{1}{Z}\psi(x_0,x_1)\psi(x_0,x_2)\psi(x_1,x_3)\psi(x_2,x_4)\psi(x_1,x_4,x_5)$$

Another Moralization Example



$\longrightarrow$



Junction Tree Algorithm

- Moralization
- Introduce Evidence
- Triangulate
- Construct Junction Tree
- Propagate Probabilities

Introduce Evidence

- Given a moral graph. Identify what is observed $\mathbf{x}_E$.
- Reduce Probability functions since we know that some values are fixed.
- So only keep probability functions over remaining nodes $\mathbf{x}_F$

$$\begin{aligned} p(X) &= \frac{1}{Z}\psi(x_1, x_2, x_3, x_4)\psi(x_4, x_5)\psi(x_4, x_6)\psi(x_4, x_7) \\ p(X_F|\bar{X}_E) &\propto \frac{1}{Z}\psi(x_1, x_2, x_3 = \bar{x}_3, x_4 = \bar{x}_4)\psi(x_4 = \bar{x}_4, x_5)\psi(x_4 = \bar{x}_4, x_6)\psi(x_4 = \bar{x}_4, x_7) \\ &\propto \frac{1}{Z}\hat{\psi}(x_1, x_2)\hat{\psi}(x_5)\hat{\psi}(x_6)\hat{\psi}(x_7) \end{aligned}$$

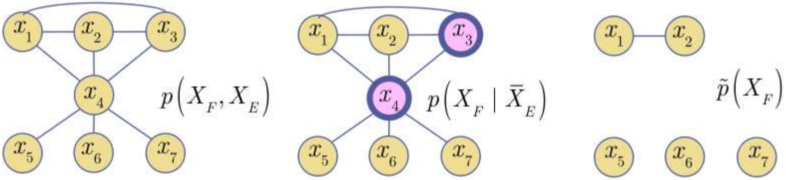
- Replace potential functions with **slices**

.4	.1
.12	.15

- Requires a different normalization term

Introduce Evidence

Observing $\mathbf{x}_E$ separates nodes.



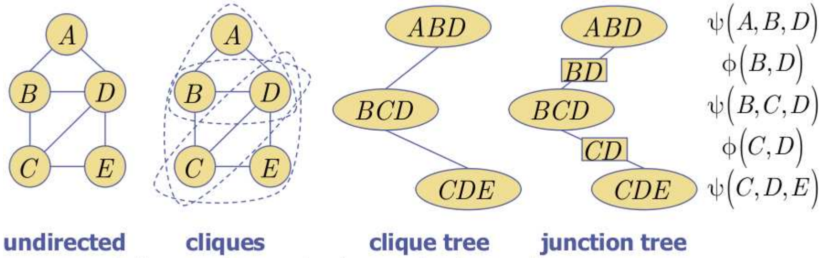
Normalization Calculation

- Avoid it until the end, when we want to calculate an individual marginal.

Junction Tree Algorithm

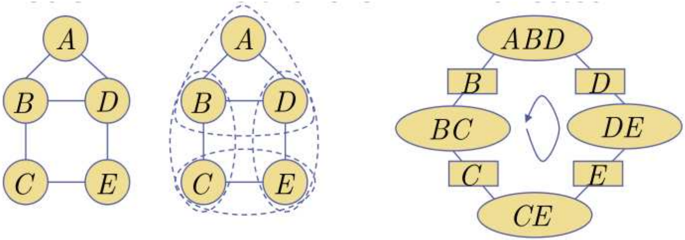
- Moralization
- Introduce Evidence
- Triangulate
- Construct Junction Tree
- Propagate Probabilities

- Ultimately we want to construct **Junction Trees**.
  - Each node is a **clique** of variables in a modal graph.
  - Edges connect cliques
  - There is a unique path from a node to the **root**
  - Between each connected clique node there is a **separator** node.
  - Separators contain intersections of variables



Triangulation

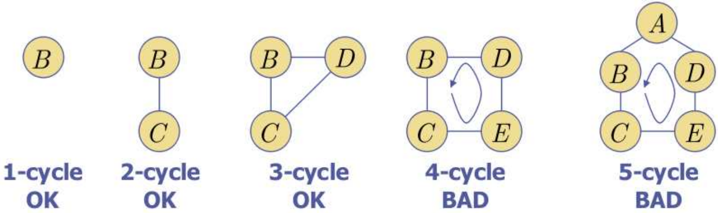
How do we construct a Junction Tree?



- Need to guarantee that a Junction Graph, made up of the cliques and separators of an undirected graph is a **Tree**
- To do this, we make triangles (3 node cliques)
  - I.e. eliminate any (chordless) cycles of 4 or more nodes.

Triangulation

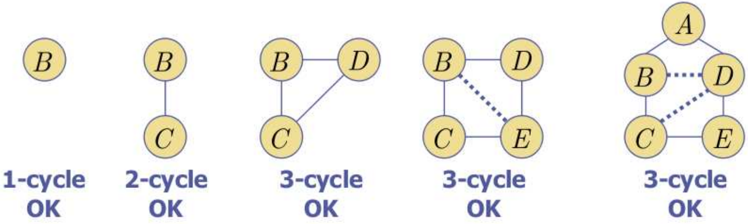
- There are potentially many choices for which edge to add.



- We'd like to keep the largest clique size small – Small ψ tables.
- However, Triangulation that minimizes the largest clique size is NP-complete.
- Suboptimal triangulation is acceptable (poly-time) and generally doesn't introduce too many extra dimensions.

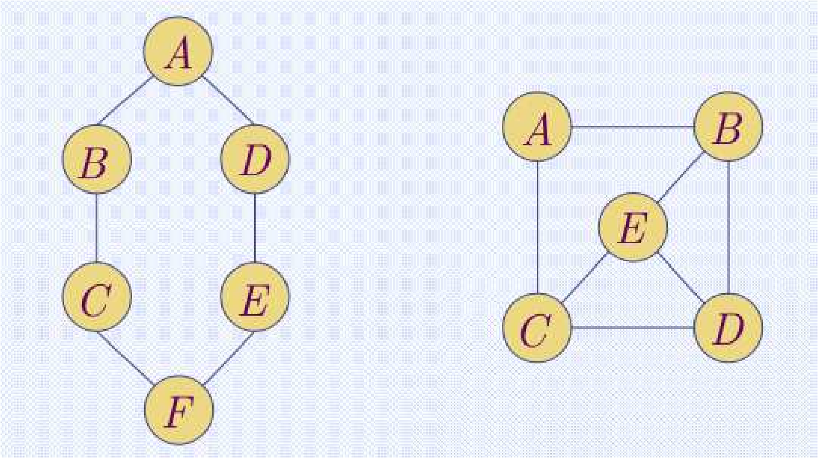
Triangulation

- There are potentially many choices for which edge to add.

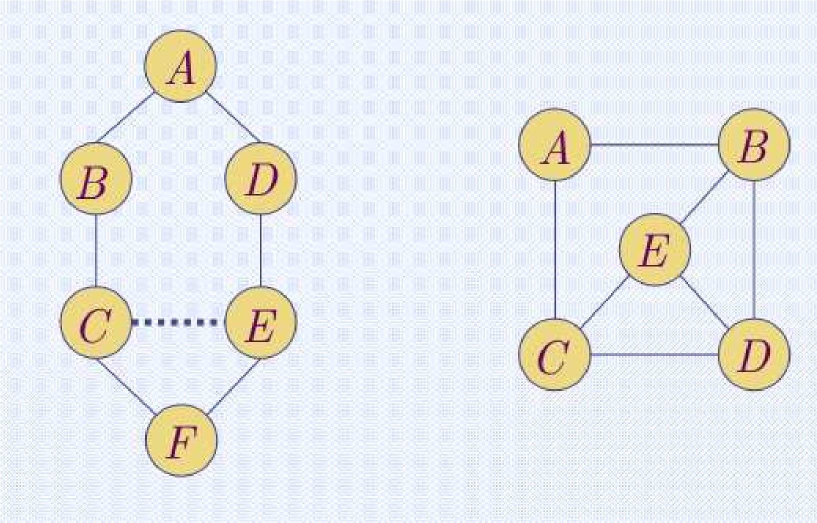


- We'd like to keep the largest clique size small – Small ψ tables.
- However, Triangulation that minimizes the largest clique size is NP-complete.
- Suboptimal triangulation is acceptable (poly-time) and generally doesn't introduce too many extra dimensions.

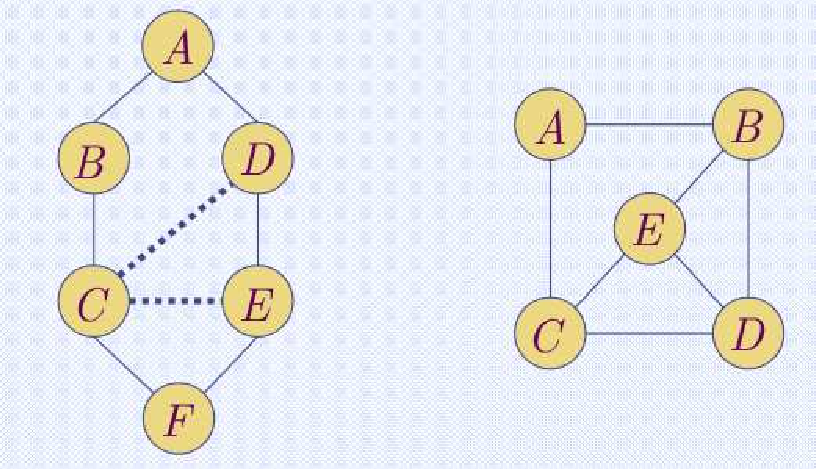
Triangulation Examples



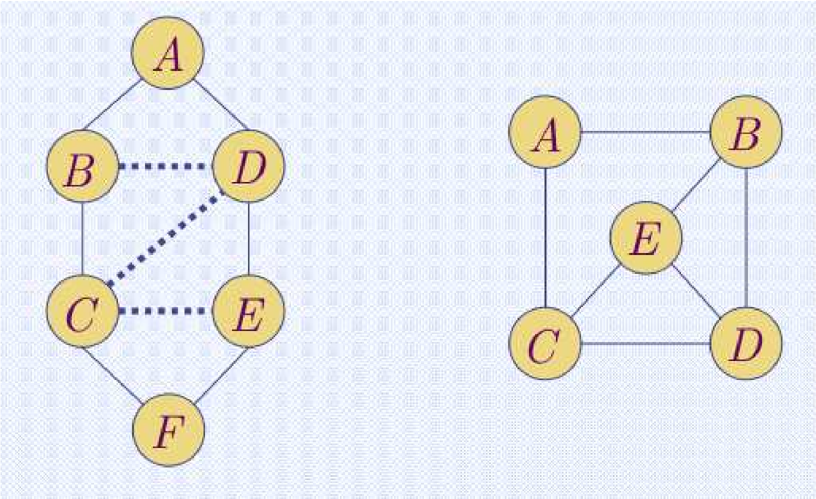
Triangulation Examples



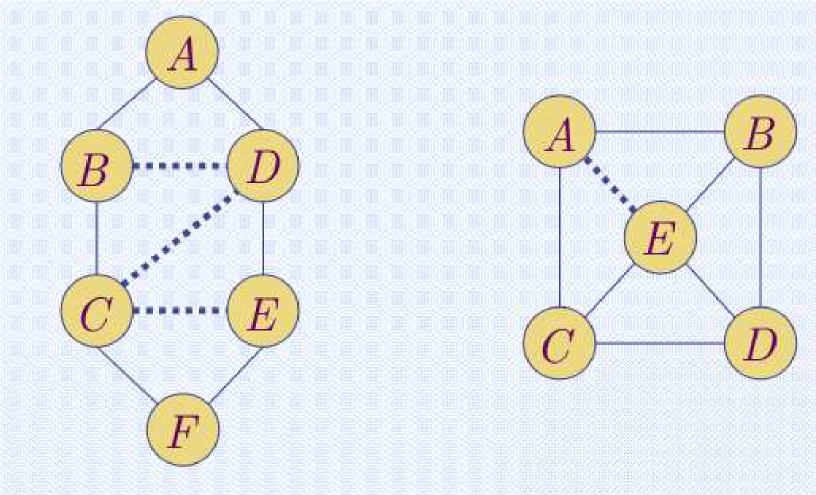
Triangulation Examples



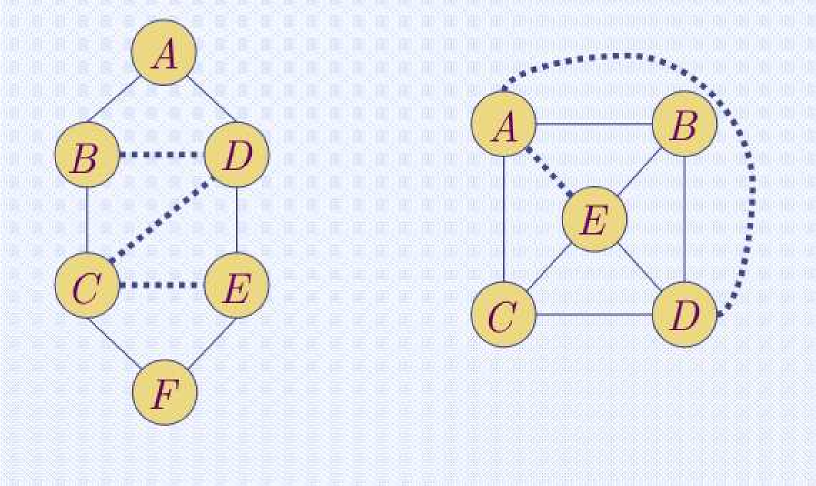
Triangulation Examples



Triangulation Examples



Triangulation Examples

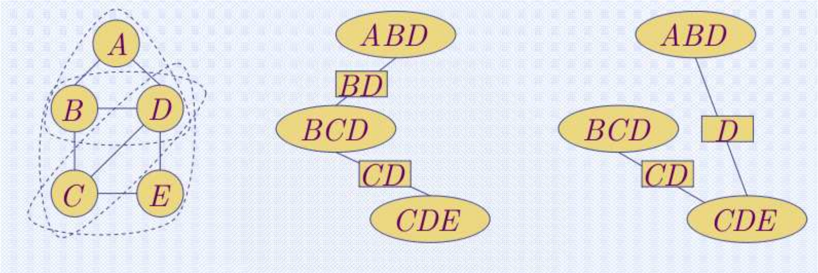


Junction Tree Algorithm

- Moralization
- Introduce Evidence
- Triangulate
- Construct Junction Tree
- Propagate Probabilities

Constructing Junction Trees

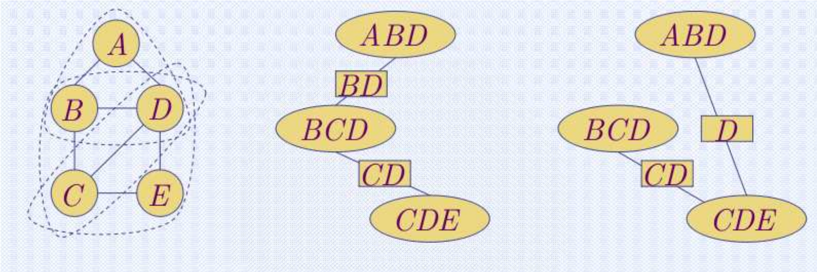
Multiple trees can be constructed from the same graph.



- Junction Trees must satisfy the **Running Intersection Property**
  - On the path connecting clique node V to clique node W, all other clique nodes must include the nodes in $V \cap W$.

Constructing Junction Trees

Multiple trees can be constructed from the same graph.



- Junction Trees must satisfy the **Running Intersection Property**
  - On the path connecting clique node V to clique node W, all other clique nodes must include the nodes in $V \cap W$.

Also: A Junction Tree has the largest total separator cardinality.

$$|\phi(B, C)| + |\phi(C, D)| > |\phi(C, D)| + |\phi(D)|$$

Forming a Junction Tree

- Given a set of cliques, must connect the nodes, such that the Running Intersection Property holds.
  - A valid Junction Tree maximizes the cardinality of the separators

Kruskal's algorithm

1. Initialize a tree with no edges
2. Calculate the size of separators between all pairs
3. Connect two cliques with the largest separator cardinality (that doesn't create a loop
4. Repeat 3 until all nodes are connected.

Junction Tree Algorithm

- Moralization
- Introduce Evidence
- Triangulate
- Construct Junction Tree
- Propagate Probabilities

Now what?

We have a valid Junction Tree!

- What can we do with it. (or...who cares?)

Probabilities in Junction Trees.

$$p(X) = \frac{1}{Z} \prod_C \hat{\psi}(\mathbf{x}_C)$$

$$p(X) = \frac{1}{Z} \frac{\prod_C \psi(\mathbf{x}_C)}{\prod_S \phi(\mathbf{x}_S)}$$

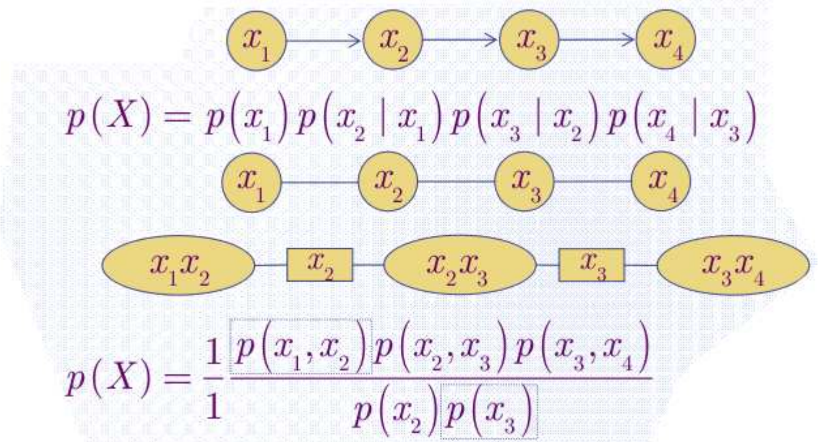
- This is equivalent to de-absorbing smaller cliques from maximal cliques.
- Doesn't change anything, just a less compact description.

Conversion from Directed Graph

Example Conversion.

$$p(X) = \frac{1}{Z} \frac{\prod_C \psi(\mathbf{x}_C)}{\prod_S \phi(\mathbf{x}_S)}$$

We can represent CPTs as clique and separator potential functions (with a normalization term).



Junction Tree Algorithm

Need to make marginals consistent.

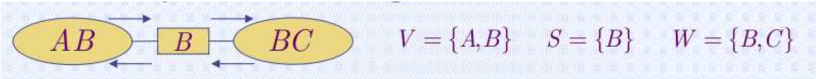
$$\begin{aligned} \psi(A,B,D) &\rightarrow p(A,B,D) \\ \phi(B,D) &\rightarrow p(B,D) \\ \psi(B,C,D) &\rightarrow p(B,C,D) \end{aligned}$$

$$\begin{aligned} \sum_A p(A,B,D) &= p(\hat{B},D) \\ p(B,D) & \\ \sum_D p(B,C,D) &= p(\hat{\hat{B}},D) \end{aligned}$$

The Junction Tree Algorithm sends messages between cliques and separators until consistency is reached.

Junction Tree Algorithm

- Send a message from each clique **to** its separator.
- The message is what the clique thinks the marginal *should be*
- Normalize the clique by each message **from** its separators such that it agrees.



If they agree, finished!

$$\sum_{V \setminus S} \psi_V = \phi_S = p(S) = \phi_S = \sum_{W \setminus S} \psi_W$$

If not....

Junction Tree Algorithm – Message Passing

$$
\begin{aligned}
\phi_S^* &= \sum_{V \setminus S} \psi_V \\
\psi_W^* &= \frac{\phi_S^*}{\phi_S} \psi_W \\
\psi_V^* &= \psi_V
\end{aligned}
$$

$$
\begin{aligned}
\phi_S^{**} &= \sum_{W \setminus S} \psi_W^* \\
\psi_V^{**} &= \frac{\phi_S^{**}}{\phi_S^*} \psi_V^* \\
\psi_W^{**} &= \psi_W^*
\end{aligned}
$$

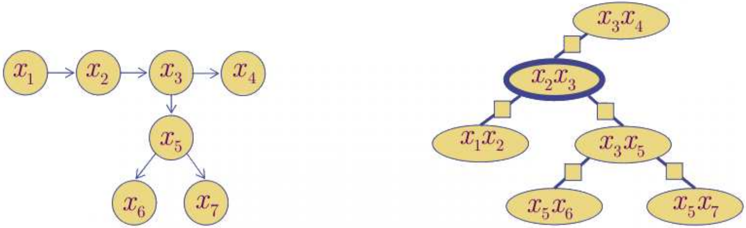
$$
\begin{aligned}
\sum_{V \setminus S} \psi_V^{**} &= \sum_{V \setminus S} \frac{\phi_S^{**}}{\phi_S^*} \psi_V \\
&= \frac{\phi_S^{**}}{\phi_S^*} \sum_{V \setminus S} \psi_V \\
&= \phi_S^{**} = \sum_{W \setminus S} \psi_W^{**}
\end{aligned}
$$

- When convergence is reached – clique potentials are **marginals** and separator potentials are **submarginals**
- $p(\mathbf{x})$ never changes because of this message passing.

$$p(\mathbf{x}) = \frac{1}{Z}\frac{\psi_V^*\psi_W^*}{\phi_S^*} = \frac{1}{Z}\frac{\psi_V\frac{\phi_S^*}{\phi_S}\psi_W}{\phi_S^*} = \frac{1}{Z}\frac{\psi_V\psi_W}{\phi_S}$$

This implies that, so long as $p(\mathbf{x})$ is correctly represented in the potential functions, the junction tree algorithm can be used to make each potential correspond to an appropriate marginal without changing the overall probability function.

Converting From DAG to Junction Tree



- Convert the Directed Graph to the Junction Tree
- Initialize separators to 1, and the clique tables to CPTs.

$$p(X) = p(x_1)p(x_2|x_1)p(x_3|x_2)p(x_4|x_3)p(x_5|x_3)p(x_6|x_5)p(x_7|x_5)$$

$$\begin{aligned} p(X) &= \frac{1}{Z}\frac{\prod_C \psi(X_c)}{\prod_S \phi(X_S)} \\ &= \frac{1}{1}\frac{p(x_1,x_2)p(x_3|x_2)p(x_4|x_3)p(x_5|x_3)p(x_6|x_5)p(x_7|x_5)}{1\cdot 1\cdot 1\cdot 1\cdot 1} \end{aligned}$$

Run JTA to set potential functions to marginals.

Evidence in Junction Tree

- Initialize the same way.

$$\begin{aligned}\psi_{AB} &= p(A,B)\\ \psi_{BC} &= p(C|B)\\ \phi_B &= 1\end{aligned}$$

- Update with a slice instead of the whole table.

$$\begin{aligned}\phi_B^* &= \sum_A \psi_{AB}\delta(A=1) = \sum_A p(A,B)\delta(A=1) = p(A=1,B)\\ \psi_{BC}^* &= \frac{\phi_B^*}{\phi_B}\psi_{BC} = \frac{p(A=1,B)}{1}p(C|B) = p(A=1,B,C)\\ \psi_{AB}^* &= \psi_{AB} = p(A=1,B)\end{aligned}$$

Conditionals

$$p(B,C|A=1) = \frac{\psi_{BC}^*}{\sum_{B,C}\psi_{BC}^*}$$

Efficiency of The Junction Tree Algorithm

All steps are efficient.

1. Construct CPTs
   - Polynomial in # of data points
2. Moralization
   - Polynomial in # of nodes (variables)
3. Introduce Evidence
   - Polynomial in # of nodes (variables)
4. Triangulate
   - Suboptimal=Polynomial, Optimal=NP
5. Construct Junction Tree
   - Polynomial in the number of cliques
   - Identifying Cliques = Polynomial in the number of nodes.
6. Propagate Probabilities
   - Polynomial in number of cliques, Exponential in size of cliques

Bye

- Next
    - Clustering Preview.