STAT 535 Lecture 5

Decomposable graphical models, triangulation and the Junction Tree

©Marina Meilă
mmp@stat.washington.edu

Reading KF Ch 4.5, 10.1.1,10.1.2, 10.4.2; For Min/Max Spanning Tree algorithm KF A3.1 or for more detail CRLS Ch. 22

1 The decomposable models class

The figure below illustrates the relationship between Bayes nets, Markov nets and decomposable models.

A decomposable model can be defined in several equivalent ways:

- a Markov field whose underlying graph is chordal
- a Bayes net with no V-structures
- a Bayes net which has a Markov field perfect map
- a graphical model whose underlying (hyper)graph is a junction tree[1]

The junction tree supports general and optimally efficient algorithms for inference, and therein lies the importance of decomposable models. These algorithms folow from some very interesting and elegant mathematical properties of junction tree graphs, of which we will examine a few (there are simple instances of more powerful properties of these graphs).

In what follows, we will study the properties of decomposable models that make them so special, and will encounter an alternative representation for them, called the junction tree. Before that, we discuss triangulation algorithms.

[1]Note that we will use the term junction tree to denote either: (i) the junction tree structure, or (ii) the structure + parametrization, i.e the decomposable graphical model, or (i) the data structure that implements this graphical model. Usually the meaning will be clear from the context.

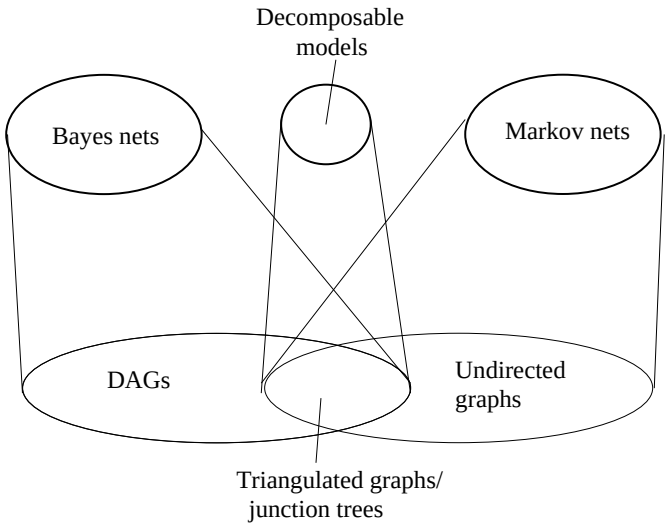
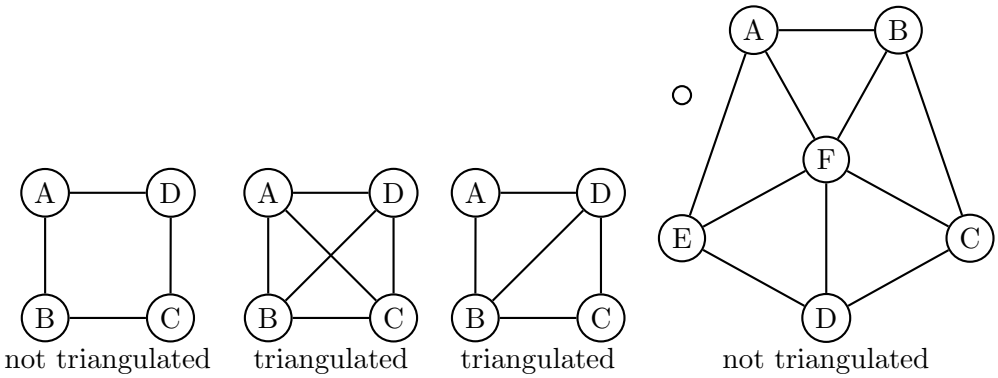


Figure 1:

2 Triangulation algorithms

We say that an undirected graph is **triangulated** (or **chordal**) if every cycle of length greater than 3 has a **chord**.



The following algorithm, called the Tarjan Elimination algorithm, can

- verify that a graph is chordal

- turn a non-chordal graph into a chordal graph by adding chords, that is adding edges to the original graph. This process is called **triangulation**.

TARJAN ELIMINATION **Algorithm**
Input Undirected graph $G = (V, \mathcal{E})$
1. Pick any undeleted node $X \in V$
2. Connect all neighbors of X
3. Mark X as deleted (also mark as deleted all edges incident to X)
Repeat from 1. until no nodes are left
Output The graph and the added edges

Verifying that a graph is chordal. If a graph is already triangulated, then it has at least 2 nodes whose neighbors form a clique. Such a node is called **simplicial**. When a simplicial node is eliminated, the TARJAN ELIMINATION algorithm adds no new edges to the graph. It can also be shown (**Exercise** for you) that if a simplicial node is removed from a chordal graph, the remaining graph is chordal as well.

Hence, we can verify that $\mathcal{G}$ is chordal, by chosing to eliminate a simplicial node at every step of the TARJAN ELIMINATION algoritm. If at any step we cannot find a simplicial node to eliminate, it means that the graph is not chordal. On the other hand, if the algorithm eliminates all the nodes (without adding any edges), this proves that $\mathcal{G}$ is chordal.

Note that the TARJAN ELIMINATION will add new edges to a chordal graph if we eliminate nodes that are not simplicial. Hence, it is essential to choose simplicial node at every step of the algorithm.

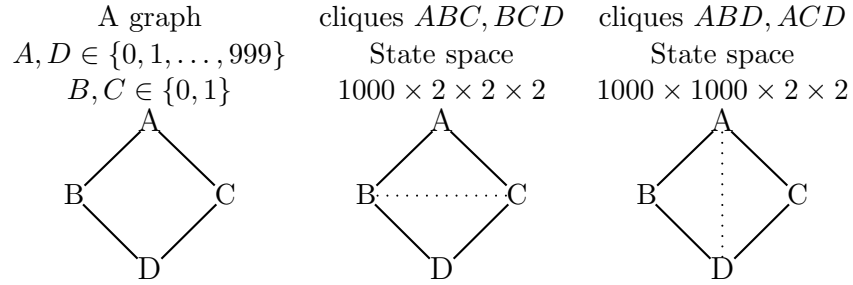
Now we turn to the problem of triangulating a non-chordal graph.

2.1 Criteria for a good triangulation

The Tarjan Elimination algorithm guarantees to create a chordal graph. But the resulting graph depends on the chosen **elimination ordering**. Since computation in the resulting chordal graph (or graphical model) depends on the size of the resulting cliques, we would like to produce either (a) a chordal

graph with a max clique size as small as possible, or (b) a decomposable model which requires as little memory as possible. The second requirement, as we shall see, amounts to minimizing the sum of the state-spaces of all cliques.

These criteria are not equivalent. If all variables have the same **arity** (=number of values), then (a) and (b) are of the same order, i.e exponential in the max clique size. But if some variables take a very large number of values, then the variables' arities matter and one should try to optimize (b). This is the case with speech and language models, where variables can represent phonemes ($\sim 10^2$ values) or words ($\sim 10^3$ values).



2.2 Finding a good elimination ordering

Finding the optimal elimination ordering with respect to either criterion is NP-hard. The proof is by reduction to MAX CLIQUE, which is known not only to be NP-hard, but also hard to approximate within a $1 + \epsilon$ factor.

Hence, the only working triangulation algorithms which attempt to minimize the size of the resulting state space are heuristics. The ones I will present here are greedy, but they can be combined with any amount of search.

1. First, one should look for a simplicial node at every step of the triangulation. This also guarantees that the Tarjan algorithm will not add edges to aa graph that is already chordal.
2. Second, if no simplicial node is found, how to choose the next non-simplicial node to eliminate? Below are two heuristics that have worked better than others in experiments.

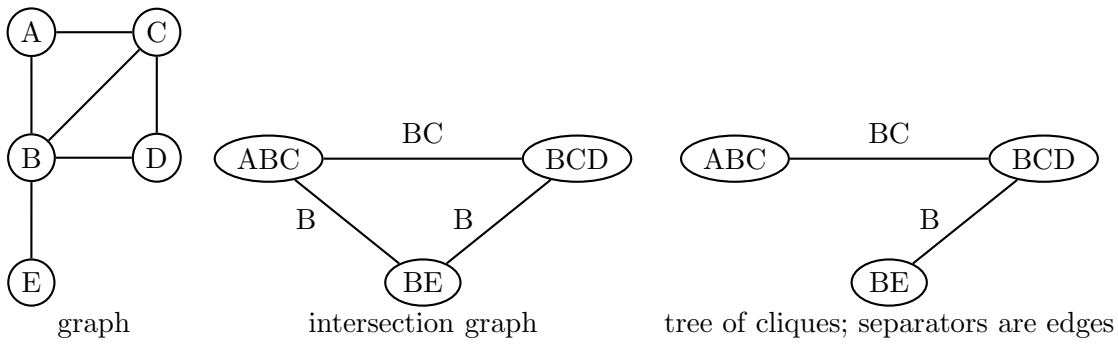


Figure 2:

- Choose the node that creates a clique with the minimum state space.
- Choose the node X that minimizes $\sum_{A,B} r_A r_B$ where the sum is taken over all pairs of neighbors $A, B \in n(X)$ which are not already connected by an edge; r_A, r_B are the numbers of values A, B can take. This heuristic is called the **minimum weight** heuristic.

3 The junction tree - structure

In a triangulated graph, the maximal cliques and their intersections, called **separators**, play an important role. A triangulated graph can be represented as a tree of maximal cliques. Figure 2 shows an example.

To obtain the junction tree of a decomposable graph, one proceeds as follows.

Input a chordal graph $G = (V, \mathcal{E})$

1. Run the Tarjan elimination algorithm to obtain a list of maximal cliques $\mathcal{C}$ of the chordal graph. Note that the list will contain no more than $n - 1$ cliques, where $n = |V|$.
2. For every $C, C' \in \mathcal{C}$ calculate their intersection $S_{CC'} = C \cap C'$.
3. Construct the **intersection graph** $\mathcal{I}$ of $\mathcal{C}$. The nodes of this graphs are the cliques in $\mathcal{C}$; there is an edge between C and C' in $\mathcal{I}$ iff $S_{CC'}$ is not empty. The edges of the graph are labeled with the sets $S_{CC'}$

(and we will say that the edges *are* the intersections). Each edge has a weight $w_{CC'} = |S_{CC'}|$ given by the number of variables in this intersection.

4. Construct a **Maximum Spanning Tree** of this intersection graph. This is the Junction Tree. It will be a graph with node set $\mathcal{C}$ and edge set $\mathcal{S}$, which is a subset of the edges of the intersection graph $\mathcal{I}$. The edges of the junction tree are sets of variables; they are called **separators**. There are exactly $|\mathcal{C}| - 1$ separators in a junction tree.

Output The (maximal) cliques $\mathcal{C}$, the separators $\mathcal{S}$, the tree.

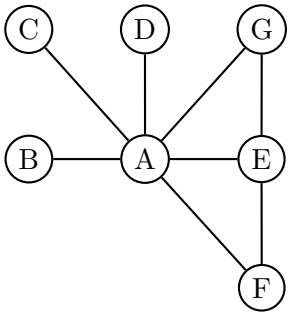
A maximum spanning tree is a tree over V whose sum of edge weights has a maximum value. Here the edge weights are the sizes of the separators.

Remarks:

1. The junction tree structure is not unique. Figures 2 and 5 show two examples of junction tree construction.
2. In a junction tree, there can be several edges that have the same set of variables as labels; i.e $S_{C_1C_2} = S_{C_1'C_2'}$. Therefore, the set of separators $\mathcal{S}$ is a **multiset**, where each separator has a multiplicity. Figure 3 exemplifies this case.
3. The **multiset** of separators $\mathcal{S}$ is the same for all junction trees of the same chordal graph; see Figure 3. Note also from the figure that separators can be subsets of other separators.

Junction trees are defined only for chordal graphs. For any junction tree, the running intersection property holds.

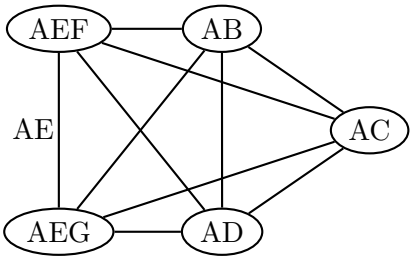
Running intersection property: Let $G = (V, \mathcal{E})$ a chordal graph, and $\mathcal{J}$ a junction tree for it. For any variable X in V, the set of cliques and separators that contain X is a connected subgraph of $\mathcal{J}$.



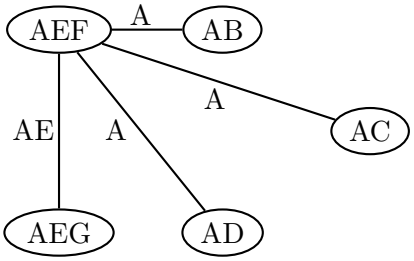
a. chordal graph

AB
AC
AD
AEF
AEG

b. maximal cliques $\mathcal{C}$



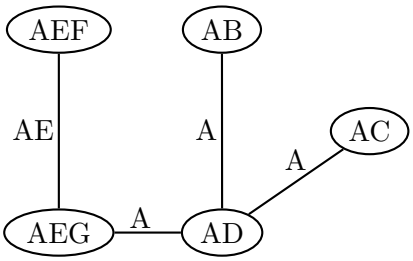
c. intersection graph (all unlabeled edges have label A)



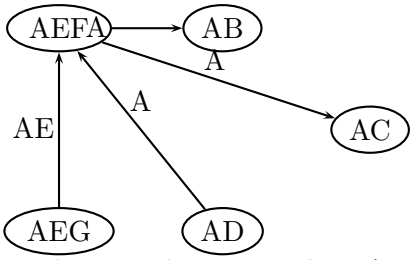
d. a junction tree

AE
A
A
A

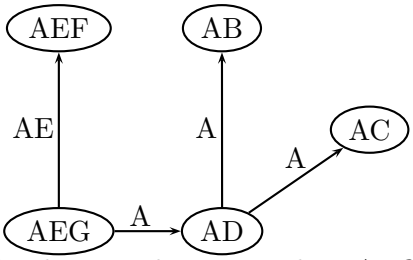
e. separators $\mathcal{S}$



f. another junction tree



g. the tree above rooted at AD



h. the tree above rooted at AEG

Figure 3: Separators can have multiplicities: **(a)** a chordal graph, **(b)** its maximal cliques list $\mathcal{C}$, **(c)** the corresponding intersection graph, **(d)** a junction tree, **(e)** the multiset of separators $\mathcal{S}$, **(f)** another junction tree, which has the same separators $\mathcal{S}$, **(g), (h)** rooted versions of the junction trees in **(d,f)**

.

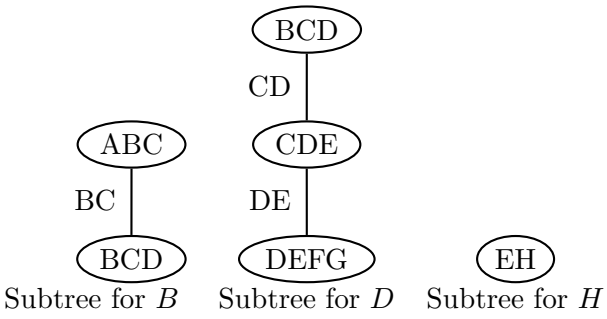


Figure 4: Illustration of the running intersection property for the junction tree in figure 5.

4 The junction tree - parameterization

A joint probability distribution that factors according to a junction tree has the form:

$$P(X_1,\, X_2, \ldots\, X_n) \;=\; \frac{\prod_{\mathcal{C}} P(X_{\mathcal{C}})}{\prod_{\mathcal{S}} P(X_{\mathcal{S}})} \tag{1}$$

where $\mathcal{C}, \mathcal{S}$ are respectively indices over the cliques and separators of the graph G.

For the junction trees in figures 2 and 5, the factorizations are

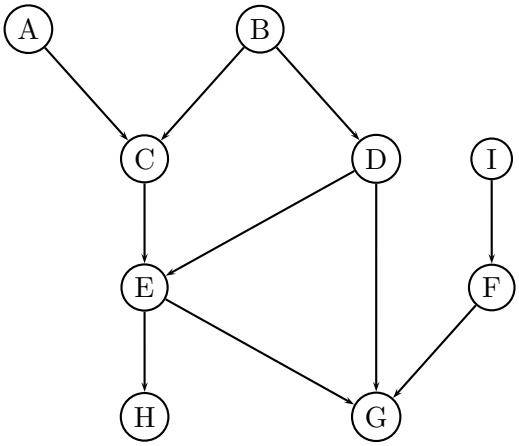
$$P(A,B,C,D,E) \;=\; \frac{P(A,B,C)P(B,C,D)P(B,E)}{P(B,C)P(B)} \tag{2}$$

$$P(A,B,C,D,E,F,G,H,I) \;=\; \frac{P(A,B,C)P(B,C,D)P(C,D,E)P(D,E,F,G)P(E,H)P(F,I)}{P(B,C)P(C,D)P(D,E)P(E)P(F)} \tag{3}$$

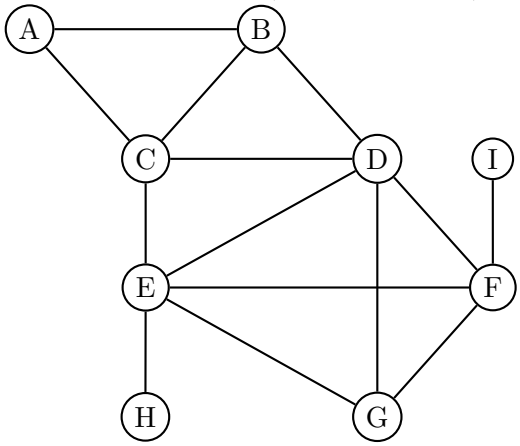
Any junction tree factorization can be easily seen as a Markov net factorization. Obviously, any decomposable model is a Markov net. Therefore, we often refer to $P_{\mathcal{C}}, P_{\mathcal{S}}$ as **clique/separator potentials**. However, in decomposable models the potentials are in a form that exhibits the local probability tables. In this context, **local** means within a clique or a separator. In contrast with Bayes nets, the local probability distributions that build a decomposable model are marginal probabilities.

However, the parametrization (1), can be turned into a parametrization reminiscent of Bayesian networks, called the **conditional probability** parametriza-

A Bayes net



The Moral graph (already chordal)



The elimination, cliques

Eliminate	Clique	
A	ABC	
B	BCD	
C	CDE	
H	EH	
I	FI	
D	$DEFG$	
E	EFG	non-maximal
F	FG	non-maximal
G	G	non-maximal

The junction tree

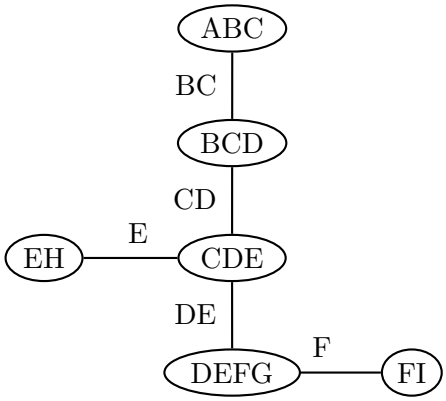


Figure 5: From a Bayes net to a junction tree.

tion of the junction tree.

For this, the junction tree is turned into a **rooted tree**, i.e a directed treee where every node has a single parent. This can be achieved by the following simple method: choose an arbitrary **root node**; then direct all edges outward from the root node, proceeding recursively from the root towards its descendants. When this process is accomplished, every clique C in the junction tree other than the root will have a unique parent $C' = pa(C)$. For the root we set $pa(root) = \emptyset$. Then, the distribution (1) equals

$$P_V(X_1, X_2, \dots X_n) \;=\; \prod_{\mathcal{C}} P(X_{\mathcal{C}\setminus pa(\mathcal{C})|pa(\mathcal{C})}) \tag{4}$$

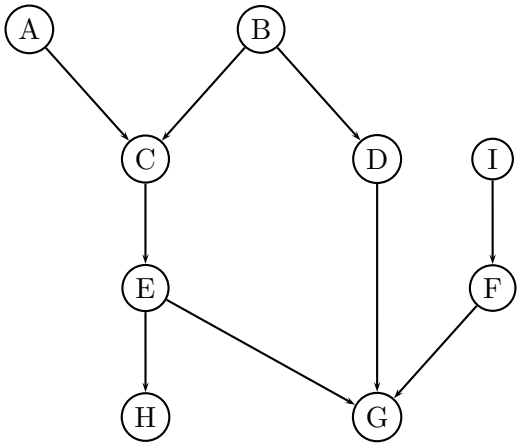
For the junction tree in figure 2, choosing the root at clique ABC, the conditional probability factorizations is

$$P(A, B, C, D, E) \;=\; P(A, B, C)P(D|B, C)P(E|B). \tag{5}$$

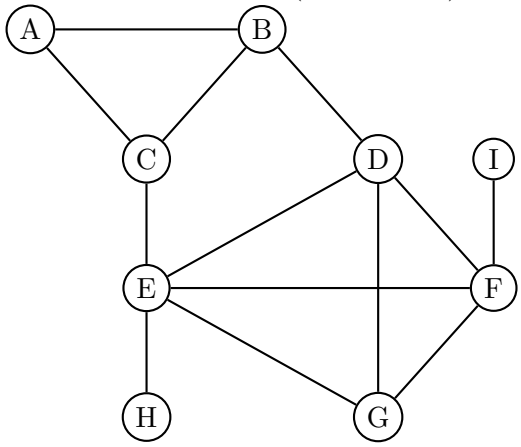
For the junction tree in figure 5, choosing the root at clique $DEFG$, the conditional probability factorizations is

$$P(A, B, C, D, E, F, G, H, I) \;=\; P(D, E, F, G)P(C|D, E)P(B|C, D)P(A|B, C)P(H|E)P(I|F). \tag{6}$$

A Bayes net



The Moral graph (not chordal)

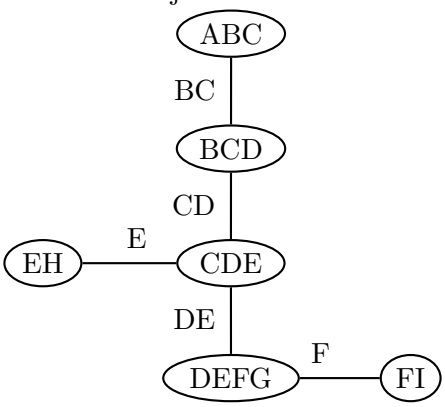


The elimination and triangulation

Eliminate	Clique	New edge
A	ABC	
B	BCD	CD
C	CDE	
H	EH	
I	FI	
D	$DEFG$	
E	EFG	non-maximal
F	FG	non-maximal
G	G	non-maximal

OR Eliminate	Clique	New edge
A	ABC	
C	BCE	BE
B	BDE	
H	EH	
I	FI	
D	$DEFG$	
E	EFG	non-maximal
F	FG	non-maximal
G	G	non-maximal

The junction tree



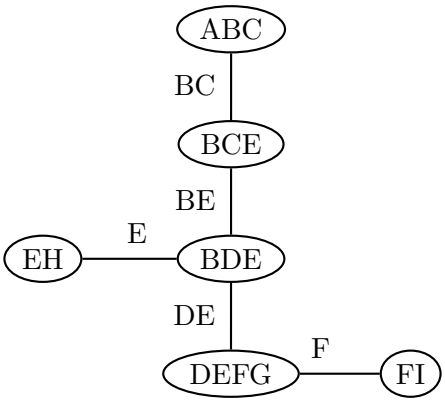


Figure 6: From a Bayes net to a junction tree, with triangulation. Note that two triangulations are possible, which end in different junction trees of similar "size".