# Efficient Inference with Junction Trees

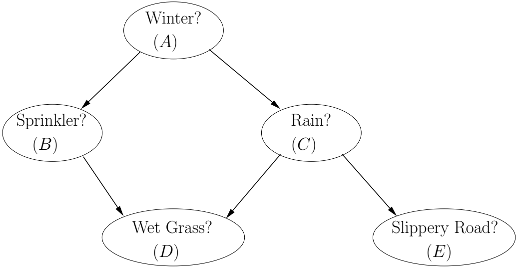
Brandon Malone

Much of this material is adapted from Chapters 7 and 9 in Darwiche's book
Many of the images were taken from the Internet

February 4, 2014

# Efficient Inference with Jointrees

Suppose we have a general Bayesian network.



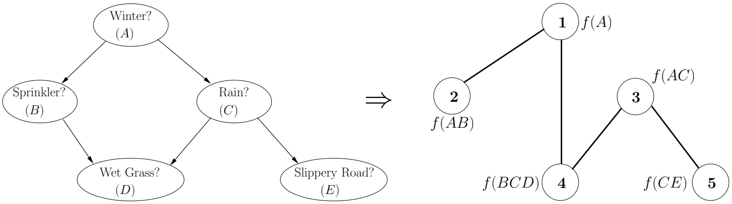

We can answer probabilistic queries with elimination trees.

How do we construct (good) elimination trees?

1. Jointrees
2. Constructing Jointrees
3. Wrap-up

# Elimination Trees

A Bayesian network is just a set of factors.




We can use an **elimination tree**, $\mathcal{T}$ to perform inference.

1. Each factor is assigned to exactly one node.
2. The factors for node $i$ are multiplied to give $\phi_i$.

The **width** of the tree gives its efficiency.

So which factors go where?

# Jointrees

A **jointree** for DAG $G$ is an elimination tree whose nodes are called **clusters** which satisfies:

- Each cluster, $\mathbf{C}_i$ is a set of variables from $G$.
- Each **family** (variable and its parents) appear together in some cluster.
- All nodes on all paths between every cluster which contains $X$ also contain $X$. (Running intersection)

The **separator** of edge $i - j$ is $\mathbf{C}_i \cap \mathbf{C}_j$.

The **width** of a jointree is the size of its largest cluster minus one.

Why would we define width that way?

# Constructing jointrees

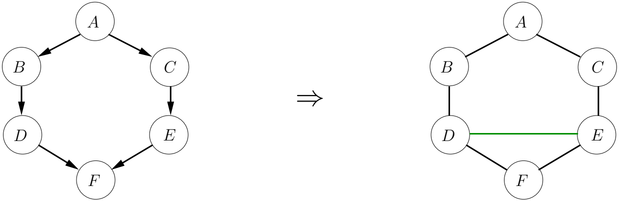
Constructing a jointree for a DAG $G$ consists of four steps.

- Moralizing $G$ to create $M_G$
- Triangulating $M_G$ to create $T_G$
- Extracting maximal cliques from $T_G$
- Assembling the cliques into a a join tree $J_T$

# Moral graphs

The **moral graph** for a DAG $G$ is an undirected graph constructed as follows:

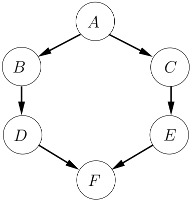
- Add an undirected edge between every pair of variables that have a common child.
- Make all edges undirected.



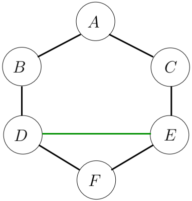
# Moral graphs

The **moral graph** for a DAG $G$ is an undirected graph constructed as follows:

- Add an undirected edge between every pair of variables that have a common child.
- Make all edges undirected.



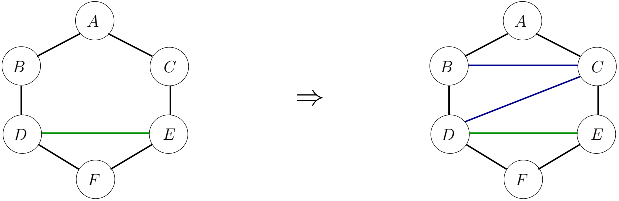

$\Rightarrow$




How does this relate to the Markov blanket of $X$?

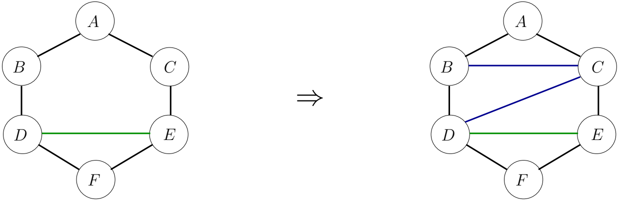
# Triangularization

Add a **chord** (edge) to each cycle of length greater than 3.



# Triangularization

Add a **chord** (edge) to each cycle of length greater than 3.

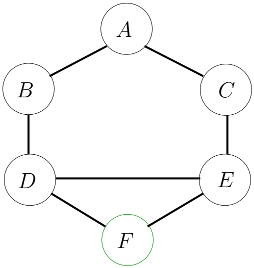



Does it matter which edges we add?

# Elimination orders

**Elimination order** for a (moral) graph: a total ordering over its nodes

We **eliminate** a node by adding **edges** to its non-adjacent neighbors.

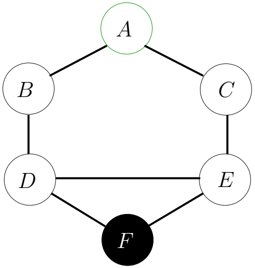



Elimination order: $F$, $A$, $B$, $C$, $D$, $E$

# Elimination orders

**Elimination order** for a (moral) graph: a total ordering over its nodes

We **eliminate** a node by adding **edges** to its non-adjacent neighbors.

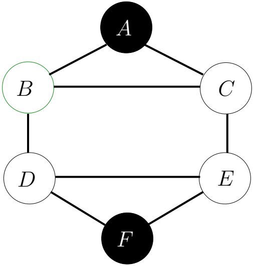



Elimination order: $F$, $A$, $B$, $C$, $D$, $E$

# Elimination orders

**Elimination order** for a (moral) graph: a total ordering over its nodes

We **eliminate** a node by adding **edges** to its non-adjacent neighbors.

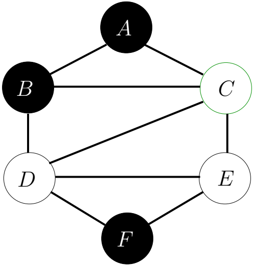



Elimination order: $F$, $A$, $B$, $C$, $D$, $E$

# Elimination orders

**Elimination order** for a (moral) graph: a total ordering over its nodes

We **eliminate** a node by adding **edges** to its non-adjacent neighbors.

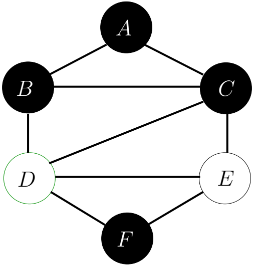



Elimination order: $F$, $A$, $B$, $C$, $D$, $E$

# Elimination orders

**Elimination order** for a (moral) graph: a total ordering over its nodes

We **eliminate** a node by adding **edges** to its non-adjacent neighbors.

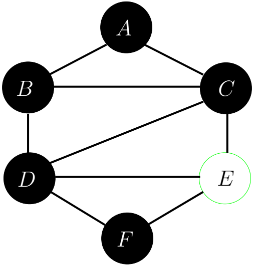



Elimination order: $F$, $A$, $B$, $C$, $D$, $E$

# Elimination orders

**Elimination order** for a (moral) graph: a total ordering over its nodes

We **eliminate** a node by adding **edges** to its non-adjacent neighbors.

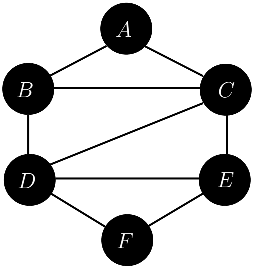



Elimination order: $F$, $A$, $B$, $C$, $D$, $E$

# Elimination orders

**Elimination order** for a (moral) graph: a total ordering over its nodes

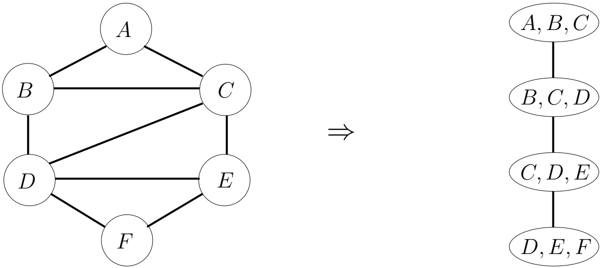
We **eliminate** a node by adding **edges** to its non-adjacent neighbors.




Elimination order: $F$, $A$, $B$, $C$, $D$, $E$

# Triangulation to elimination trees

The (maximal) cliques in the triangulated graph are the clusters in the jointree.



# Inference in Bayesian networks

**procedure** $\textsc{BuildJoinTree}$(DAG $G$, triangulation order $\pi$)

  $M_G \leftarrow$ moralize $G$

  $T_G \leftarrow$ triangulate $M_G$ according to $\pi$

  $\mathcal{S} \leftarrow$ set of maximal cliques in $T_G$

  **return** jointree $J_G$ implied by $\mathcal{S}$ and running intersection

**end procedure**

# Inference in Bayesian networks

---

**procedure** $\textsc{BuildJoinTree}$(DAG $G$, triangulation order $\pi$)
 $M_G \leftarrow$ moralize $G$
 $T_G \leftarrow$ triangulate $M_G$ according to $\pi$
 $\mathcal{S} \leftarrow$ set of maximal cliques in $T_G$
 **return** jointree $J_G$ implied by $\mathcal{S}$ and running intersection
**end procedure**

---

- Evidence?
- Joint for variables not in the same cluster?

# Inference in Bayesian networks

**procedure** BUILDJOINTREE(DAG $G$, triangulation order $\pi$)
  $M_G \leftarrow$ moralize $G$
  $T_G \leftarrow$ triangulate $M_G$ according to $\pi$
  $\mathcal{S} \leftarrow$ set of maximal cliques in $T_G$
  **return** jointree $J_G$ implied by $\mathcal{S}$ and running intersection
**end procedure**

- Evidence? *Indicator factors*
- Joint for variables not in the same cluster? *Add redundant clusters respecting running intersection*

# Recap

During this part of the course, we have discussed:

- Concepts of probability theory
- Justifications for probability (e.g., the Dutch book argument)
- Bayesian networks as a compact, parametric representation of a probability distribution
- Equivalence among Bayesian networks
- Probabilistic inference in NBCs, HMMs, and general Bayesian networks
- Learning NBCs (and HMMs) from (complete) data

# Next in probabilistic models

During the rest of the course, we will cover:

- Parameter learning
  - General structures, complete data
  - Fixed structures, incomplete data (EM)
- Structure learning
  - (Penalized) Likelihood and overfitting
  - Model selection algorithms