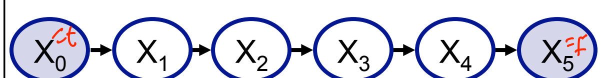
Readings:
K&F: 9.1, 9.2, 9.3, 9.4

Junction Trees 2

Graphical Models – 10708
Carlos Guestrin
Carnegie Mellon University
October 22nd, 2008

What if I want to compute $P(X_i|x_0,x_{n+1})$ for each i?



Compute:

$$P(X_i \mid x_0, x_{n+1})$$

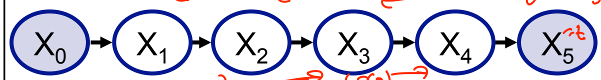
Variable elimination for each i?

$O(n)$

Variable elimination for every i, what's the complexity?

$O(n^2)$

Reusing computation

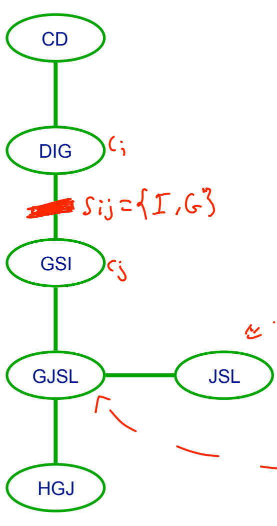


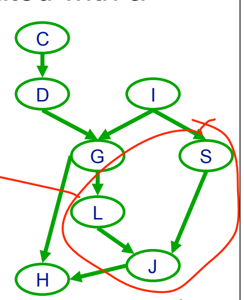
Compute:

$$P(X_i \mid x_0, x_{n+1})$$

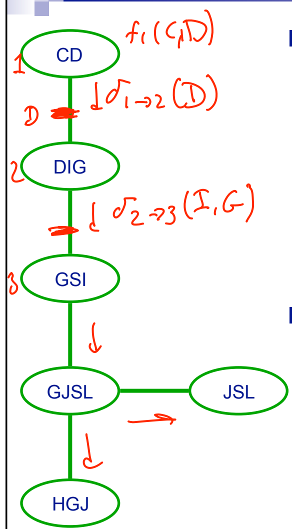
Cluster graph

- **Cluster graph**: For set of factors *F*
  - Undirected graph
  - Each node i associated with a cluster $\mathbf{C}_i$
  - *Family preserving*: for each factor $f_j \in F$, $\exists$ node i such that scope$[f_j] \subseteq \mathbf{C}_i$
  - Each edge i – j is associated with a separator $\mathbf{S}_{ij} = \mathbf{C}_i \cap \mathbf{C}_j$



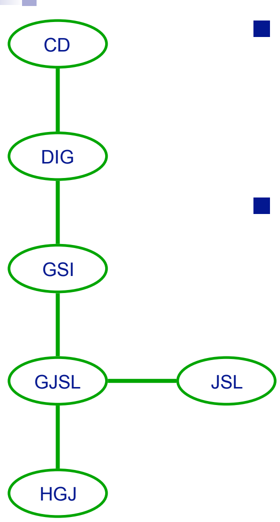


Cluster graph for VE



- **VE generates cluster tree!**
  - One clique for each factor used/generated
  - Edge i – j, if f_i used to generate f_j
  - “Message” from i to j generated when marginalizing a variable from f_i
  - Tree because factors only used once
- **Proposition**:
  - “Message” δ_{ij} from i to j
  - Scope[δ_{ij}] $\subseteq$ $\mathbf{S}_{ij}$

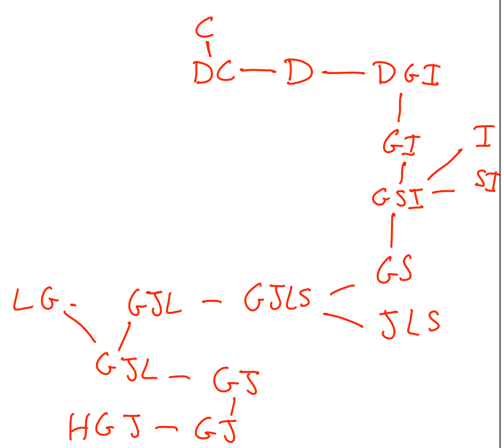
Running intersection property



- **Running intersection property (RIP)**
  - Cluster tree satisfies RIP if whenever $X \in \mathbf{C}_i$ and $X \in \mathbf{C}_j$ then X is in every cluster in the (unique) path from $\mathbf{C}_i$ to $\mathbf{C}_j$
- **Theorem**:
  - Cluster tree generated by VE satisfies RIP

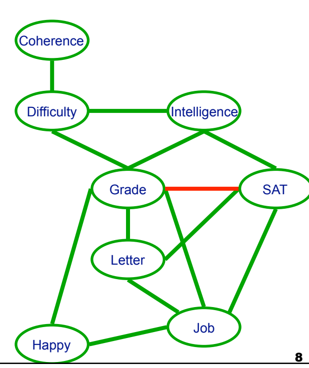
Constructing a clique tree from VE

- Select elimination order ◁
- Connect factors that would be generated if you run VE with order ◁
- Simplify!
  - Eliminate factor that is subset of neighbor

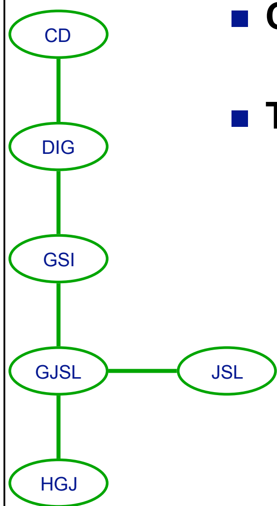


Find clique tree from chordal graph

- Triangulate moralized graph to obtain chordal graph
- Find maximal cliques
  - NP-complete in general
  - Easy for chordal graphs
  - Max-cardinality search
- Maximum spanning tree finds clique tree satisfying RIP!!!
  - Generate weighted graph over cliques
  - Edge weights (i,j) is separator size – $|C_i \cap C_j|$

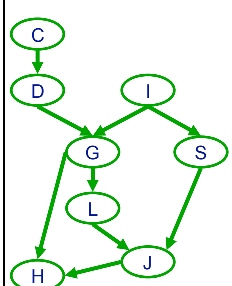


Clique tree & Independencies



- **Clique tree (or Junction tree)**
  - A cluster tree that satisfies the RIP
- **Theorem**:
  - Given some BN with structure *G* and factors *F*
  - For a clique tree *T* for *F* consider $\mathbf{C}_i - \mathbf{C}_j$ with separator $\mathbf{S}_{ij}$:
    - **X** – any set of vars in $\mathbf{C}_i$ side of the tree
    - **Y** – any set of vars in $\mathbf{C}_i$ side of the tree
  - Then, $(\mathbf{X} \perp \mathbf{Y} \mid \mathbf{S}_{ij})$ in BN
  - Furthermore, $\mathrm{I}(T) \subseteq \mathrm{I}(G)$

Variable elimination in a clique tree 1

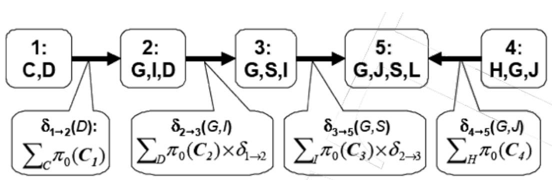


- **Clique tree for a BN**
  - Each CPT assigned to a clique
  - Initial potential $\pi_0(\mathbf{C}_i)$ is product of CPTs

Variable elimination in a clique tree 2

- **VE in clique tree to compute P(X_i)**
  - Pick a root (any node containing X_i)
  - Send messages recursively from leaves to root
    - Multiply incoming messages with initial potential
    - Marginalize vars that are not in separator
  - Clique *ready* if received messages from all neighbors

Belief from message

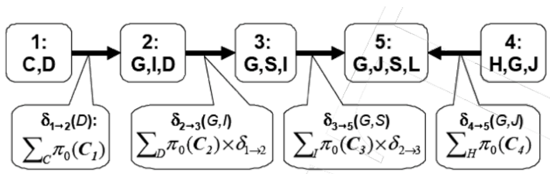


- **Theorem**: When **clique C_i is ready**
  - Received messages from all neighbors
  - Belief $\pi_i(\mathbf{C}_i)$ is product of initial factor with messages:

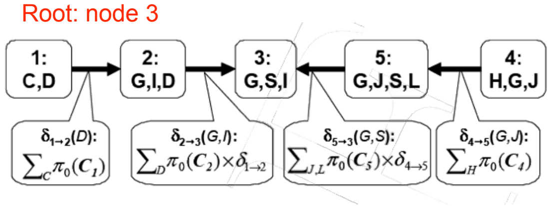
Choice of root

- **Message does not depend on root!!!**

Root: node 5

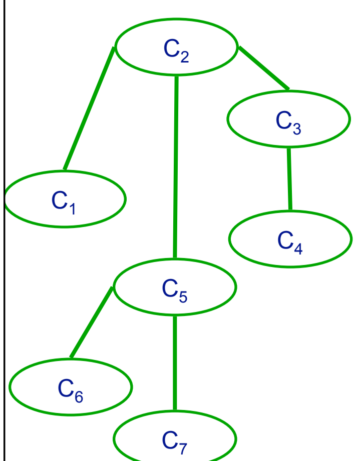


Root: node 3



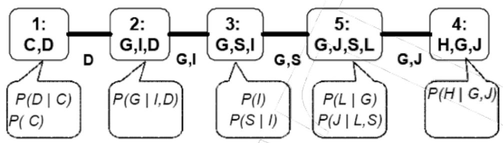
"Cache" computation: Obtain belief for all roots in linear time!!

Shafer-Shenoy Algorithm (a.k.a. VE in clique tree for all roots)



- Clique $\mathbf{C}_i$ *ready to transmit* to neighbor $\mathbf{C}_j$ if received messages from all neighbors but j
  - Leaves are always ready to transmit
- While $\exists$ $\mathbf{C}_i$ ready to transmit to $\mathbf{C}_j$
  - Send message $\delta_{i\rightarrow j}$
- Complexity: Linear in # cliques
  - One message sent each direction in each edge
- **Corollary**: At convergence
  - Every clique has correct belief

Calibrated Clique tree



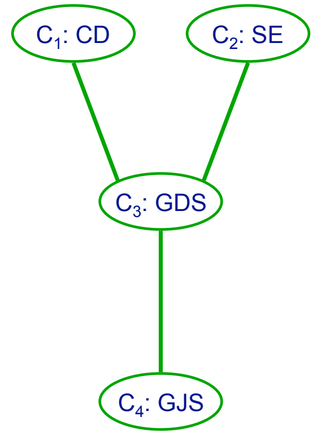
- Initially, neighboring nodes don't agree on "distribution" over separators
- **Calibrated clique tree**:
  - At convergence, tree is *calibrated*
  - Neighboring nodes agree on distribution over separator

Answering queries with clique trees

- Query within clique
- Incremental updates – Observing evidence Z=z
  - Multiply some clique by indicator **1**(Z=z)
- Query outside clique
  - Use variable elimination!

Introducing message passing with division

- Variable elimination (message passing with multiplication)
  - message:
  - belief:
- Message passing with division:
  - message:
  - belief update:



Factor division

- Let **X** and **Y** be disjoint set of variables
- Consider two factors: $\phi_1(\mathbf{X},\mathbf{Y})$ and $\phi_2(\mathbf{Y})$
- Factor $\psi=\phi_1/\phi_2$
  - 0/0=0

a^1	b^1	0.5
a^1	b^2	0.2
a^2	b^1	0
a^2	b^2	0
a^3	b^1	0.3
a^3	b^2	0.45

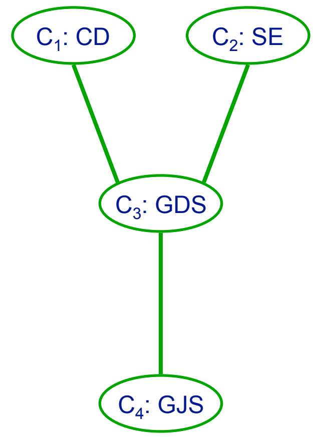
a^1	0.8
a^2	0
a^3	0.6

a^1	b^1	0.625
a^1	b^2	0.25
a^2	b^1	0
a^2	b^2	0
a^3	b^1	0.5
a^3	b^2	0.75

Lauritzen-Spiegelhalter Algorithm (a.k.a. belief propagation)

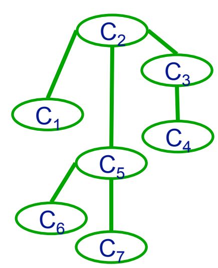
Simplified description see reading for details

- Separator potentials μ_{ij}
  - one per edge (same both directions)
  - holds "last message"
  - initialized to 1
- Message $i \rightarrow j$
  - what does i think the separator potential should be?
    - $\sigma_{i \rightarrow j}$
  - update belief for j:
    - pushing j to what i thinks about separator
  - replace separator potential:



Convergence of Lauritzen -Spiegelhalter Algorithm

- Complexity: Linear in # cliques
  - for the "right" schedule over edges (leaves to root, then root to leaves)
- **Corollary**: At convergence, every clique has correct belief



VE versus BP in clique trees

- VE messages (the one that multiplies)
- BP messages (the one that divides)

Clique tree invariant

- **Clique tree potential**:
  - Product of clique potentials divided by separators potentials
- Clique tree **invariant**:
  - $P(\mathbf{X}) = \pi_T(\mathbf{X})$

Belief propagation and clique tree invariant

- **Theorem**: Invariant is maintained by BP algorithm!

- BP reparameterizes clique potentials and separator potentials
  - At convergence, potentials and messages are marginal distributions

Subtree correctness

- **Informed message** from i to j, if all messages into i (other than from j) are informed
  - Recursive definition (leaves always send informed messages)
- **Informed subtree**:
  - All incoming messages informed
- **Theorem**:
  - Potential of connected informed subtree *T'* is marginal over scope[*T'*]
- **Corollary**:
  - At convergence, clique tree is *calibrated*
    - π_i = P(scope[π_i])
    - μ_{ij} = P(scope[μ_{ij}])

Clique trees versus VE

- Clique tree advantages
  - Multi-query settings
  - Incremental updates
  - Pre-computation makes complexity explicit

- Clique tree disadvantages
  - Space requirements – no factors are "deleted"
  - Slower for single query
  - Local structure in factors may be lost when they are multiplied together into initial clique potential

Clique tree summary

- Solve marginal queries for all variables in only twice the cost of query for one variable
- Cliques correspond to maximal cliques in induced graph
- Two message passing approaches
  - VE (the one that multiplies messages)
  - BP (the one that divides by old message)
- Clique tree invariant
  - Clique tree potential is always the same
  - We are only reparameterizing clique potentials
- Constructing clique tree for a BN
  - from elimination order
  - from triangulated (chordal) graph
- Running time (only) exponential in size of largest clique
  - Solve **exactly** problems with thousands (or millions, or more) of variables, and cliques with tens of nodes (or less)