Probabilistic Graphical Models

Lecture 8 – Junction Trees

CS/CNS/EE 155

Andreas Krause

Announcements

- Homework 2 due next Wednesday (Nov 4) in class
  - Start early!!!
- Project milestones due Monday (Nov 9)
  - 4 pages of writeup, NIPS format
  - http://nips.cc/PaperInformation/StyleFiles

Best project award!!

Key questions

- How do we specify distributions that satisfy particular independence properties?

  ➔ **Representation**

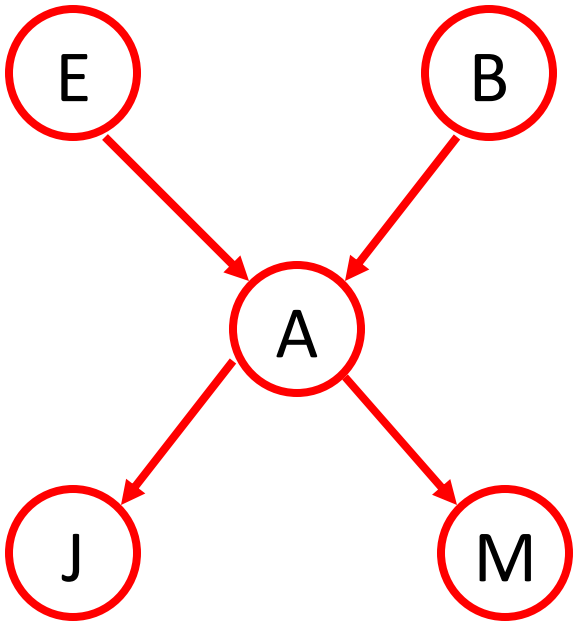
- How can we identify independence properties present in data?

  ➔ **Learning**

- How can we exploit independence properties for efficient computation?

  ➔ **Inference**

Typical queries: Conditional distribution



- Compute distribution of some variables given values for others

Observe $M=T$

Compute $P(E=T \mid M=T)$

$$P(E=T \mid M=T) = \frac{P(E=T, M=T)}{P(M=T)}$$

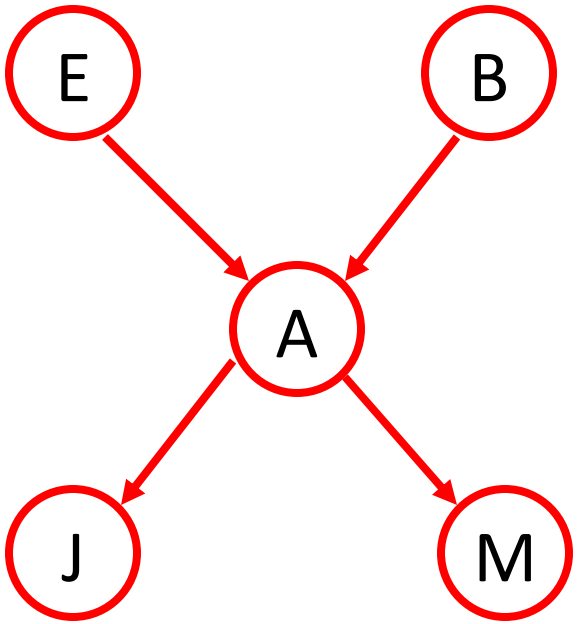
$$P(E=T, M=T) = \sum_b \sum_a \sum_j P(E=T, M=T, B=b, A=a, J=j)$$

$P(E)P(B)P(A|EB)P(J|A)P(M|A)$

2^3 terms

Naive approach exponential in # vars...

Typical queries: Maxizimization



- MPE (Most probable explanation):

  Given values for some vars, compute most likely assignment to all remaining vars

  Given $J=F, M=T$, find

  $(e^*, b^*, a^*) = \underset{e,b,a}{\operatorname{argmax}}\ P(J=F, M=T, e, b, a)$

- MAP (Maximum a posteriori):

  Compute most likely assignment to some variables

  $e^* = \underset{e}{\operatorname{argmax}}\ P(J=F, M=T, E=e) =$

  $= \underset{e}{\operatorname{argmax}} \sum_{a,b} P(J=F, M=T, e, a, b)$

MPE and MAP don't necessarily give same answers...

Hardness of inference for general BNs

- Computing conditional distributions:
  - Exact solution: #P-complete
  - Approximate solution: NP-hard:

    Absolute approx: Finding $|P(x) - \hat{P}(x)| < \varepsilon$ NP-hard even for $\varepsilon = \frac{1}{2}$

    Relative approx: $1-\varepsilon < \frac{\hat{P}(x)}{P(x)} < 1+\varepsilon$ NP hard for $\varepsilon > 0$
- Maximization:
  - MPE: NP-complete
  - MAP: NP^{PP}-complete

    $\max_{x_1 \ldots x_m} \sum_{x_{m+1} \ldots x_n}$
- Inference in general BNs is really hard ☹
- Is all hope lost?

Inference

- Can exploit structure (conditional independence) to efficiently perform **exact inference** in many practical situations

- For BNs where exact inference is not possible, can use algorithms for **approximate inference** (later this term)

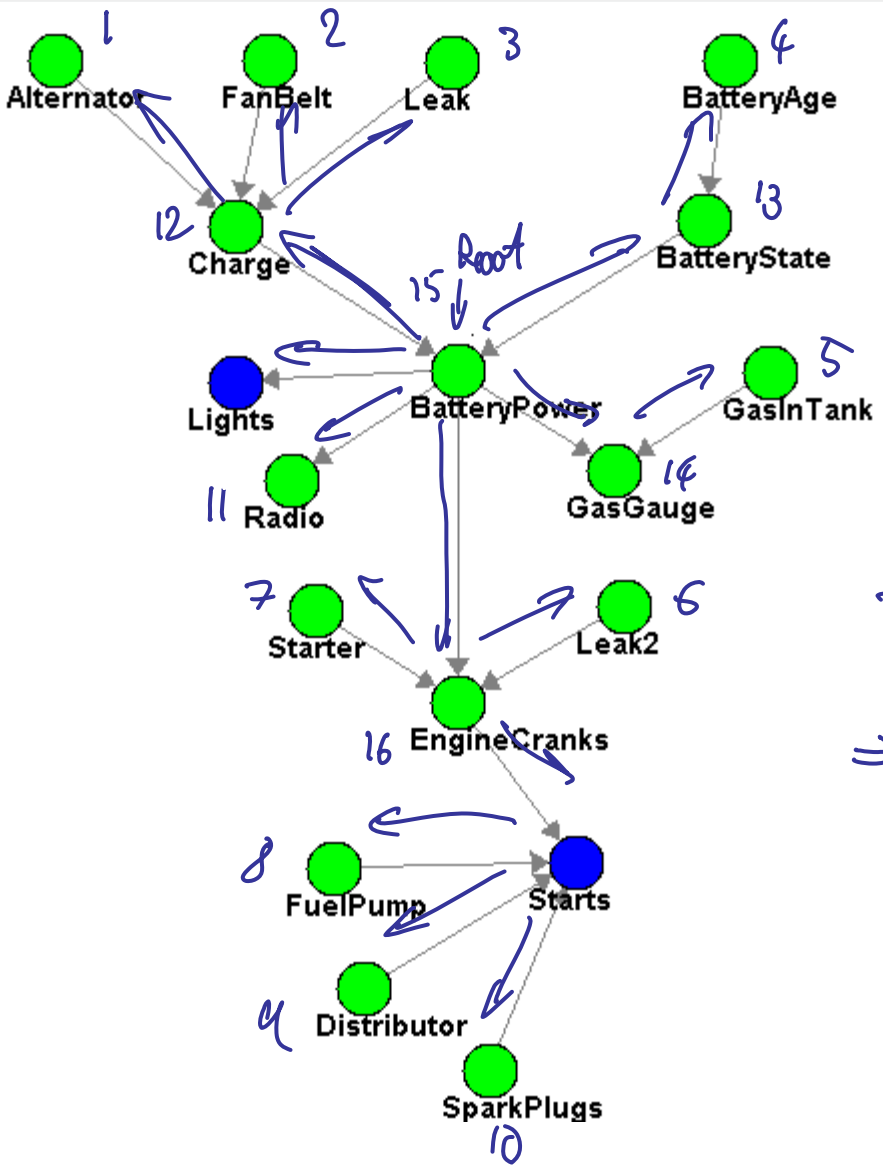
Variable elimination algorithm

- Given BN and Query P(X | **E**=**e**)
- Remove irrelevant variables for {X,**e**}
- Choose an ordering of $X_1,\ldots,X_n$
- Set up initial factors: $f_i = P(X_i \mid \mathbf{Pa}_i)$
- For i =1:n, $X_i \notin \{X,\mathbf{E}\}$
  - Collect all factors f that include X_i
  - Generate new factor by marginalizing out X_i

$$g = \sum_{x_i} \prod_j f_j$$

  - Add g to set of factors
- Renormalize P(x,**e**) to get P(x | **e**)

Variable elimination for polytrees



BNG Polytree if skeleton(G) is tree

Drop edge directions
Pick some root in skeleton
Direct edges outward from root

Eliminate vars in inverse topological order of resulting directed tree (not in query)

Factor sizes do not increase

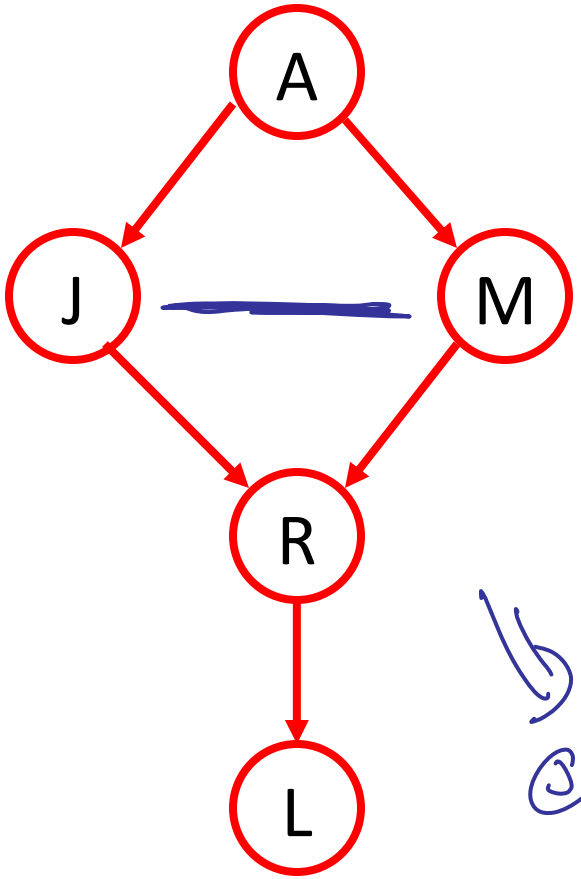
⇒ Inference possible in $O(n)$ steps

Complexity of variable elimination

- Tree graphical models
  - Using correct elimination order, factor sizes do not increase!
  - Inference in linear time!!

- General graphical models
  - Ultimately NP-hard..
  - Need to understand what happens if there are loops

Variable elimination with loops



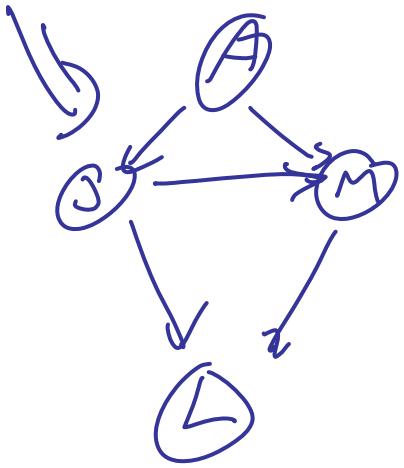
$P(A, L)$

Elim. order R, J, M

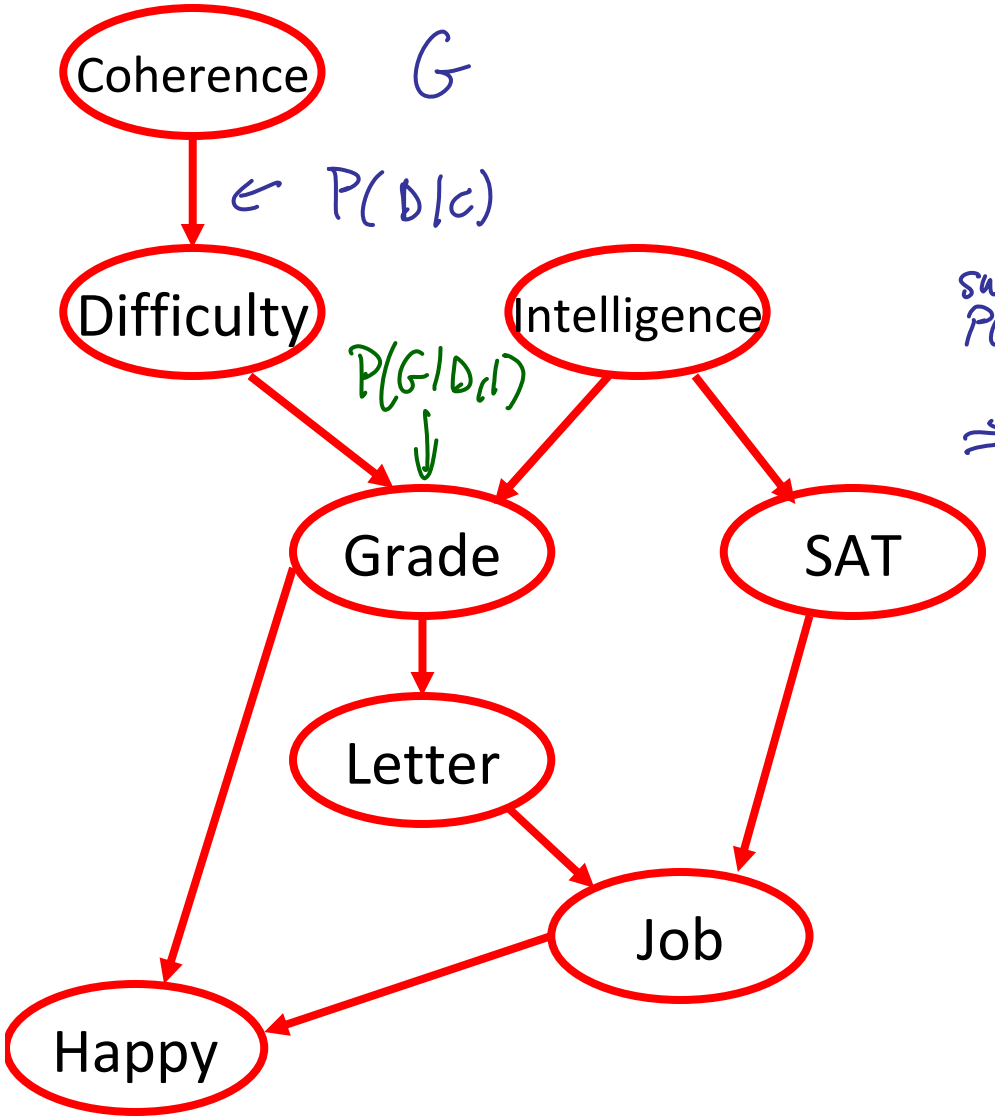
$$P(A,L) = \sum_{rjm} P(A)\, P(J|A)\, P(M|A)\, P(R|JM)\, P(L|R)$$

$$= \sum_{jm} P(A)\, P(J|A)\, P(M|A) \underbrace{\sum_{r} P(r|jm)\, P(L|r)}_{g}$$

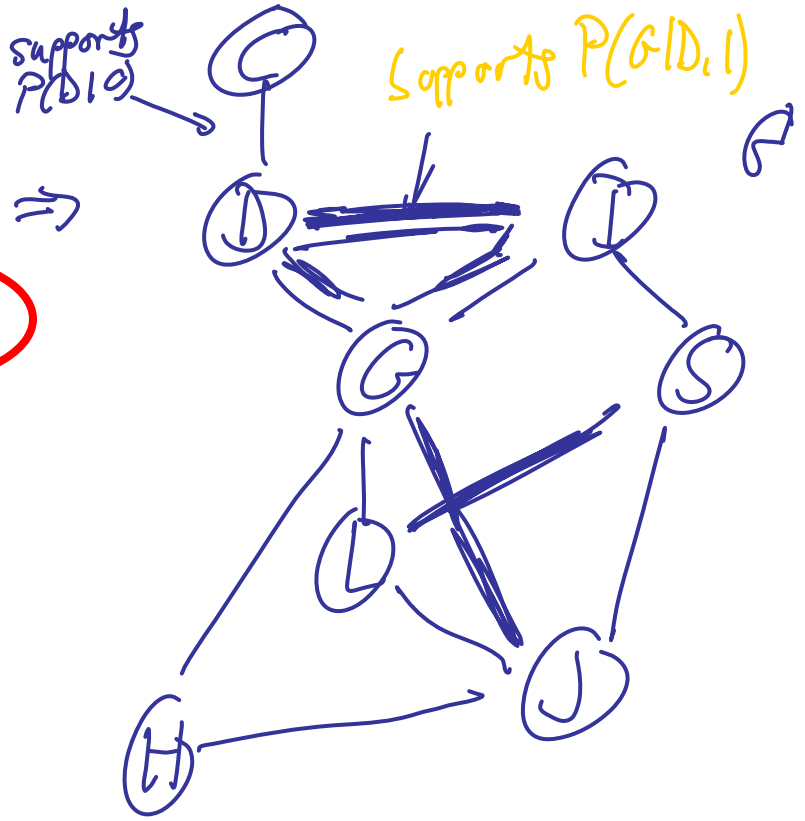
Scope $(g) = \{L, J, M\}$



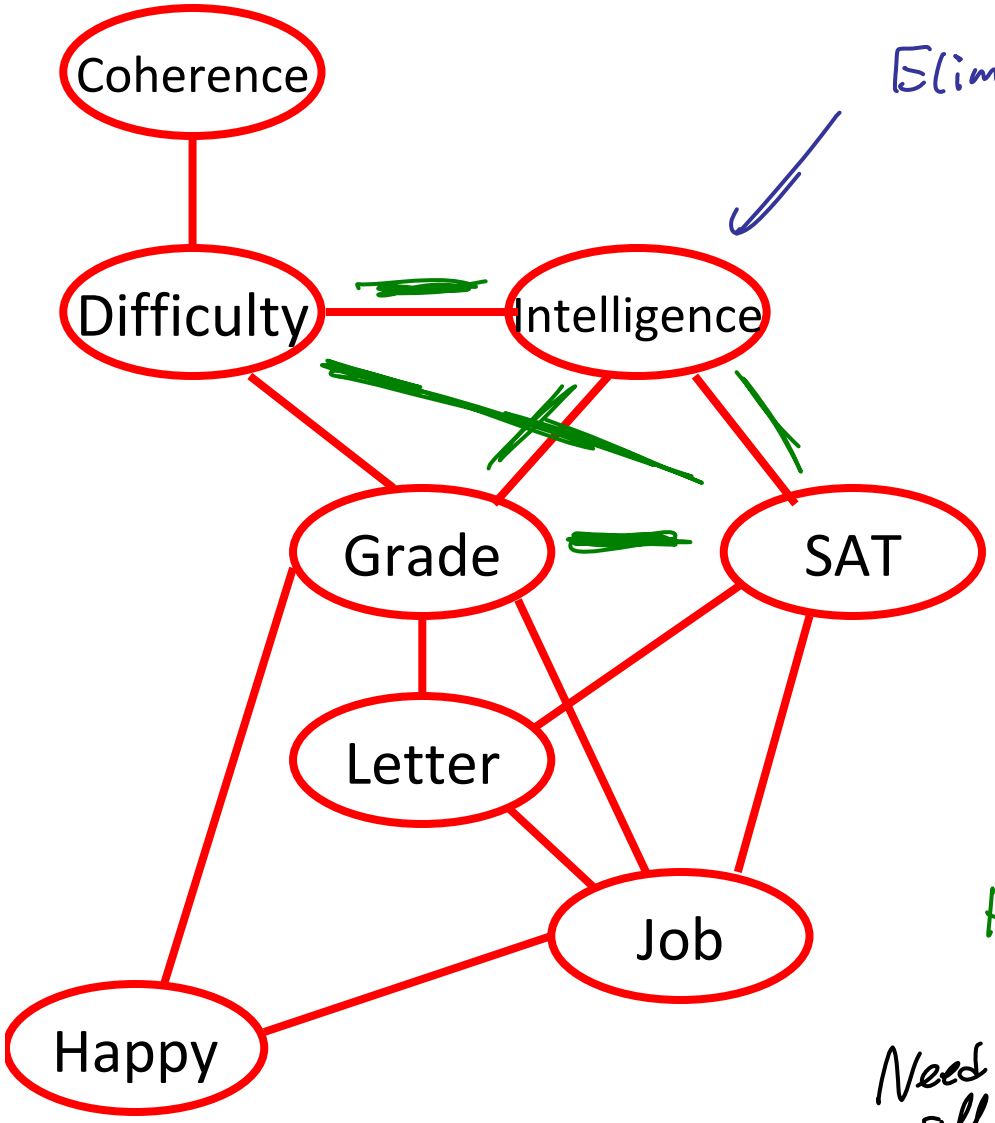
Elimination as graph transformation: Moralization



Want that all factors of G are supported on cliques of G'



Elimination: Filling edges



Eliminate

⇒ Introduce new factor

$$g = \sum_{I} \prod_{\substack{\text{Cliques } C \\ \text{incident} \\ \text{with } I}} f_C$$

$$\text{Scope}(g) = \bigcup_{\substack{\text{Cliques } C \\ \text{inc. w } I}} \text{Scope}(f_C) \setminus \{I\}$$

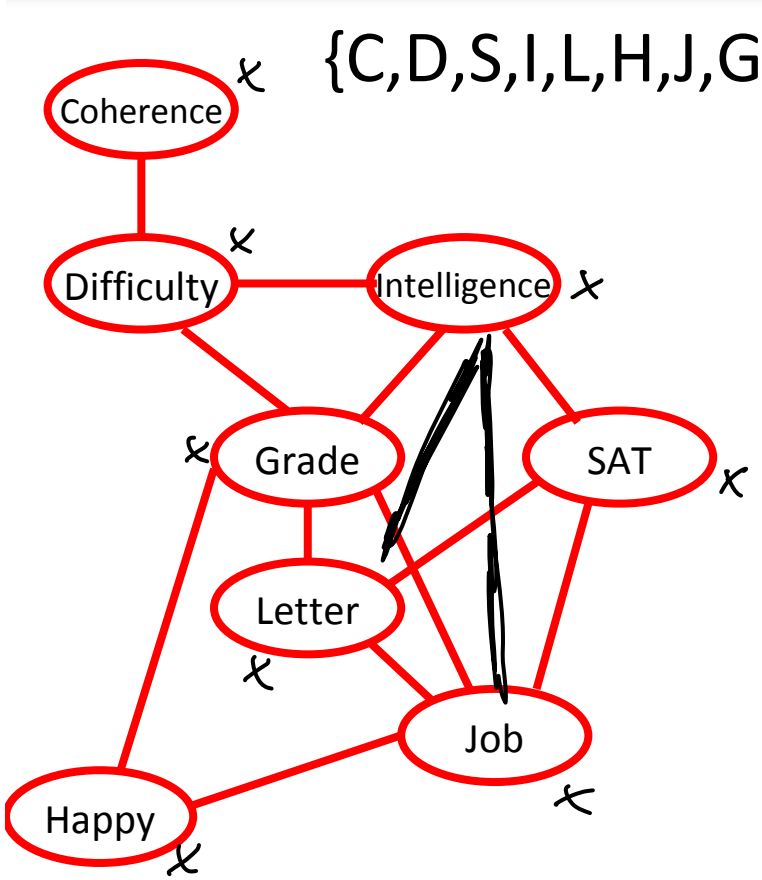
Want to make sure that resulting graph supports g

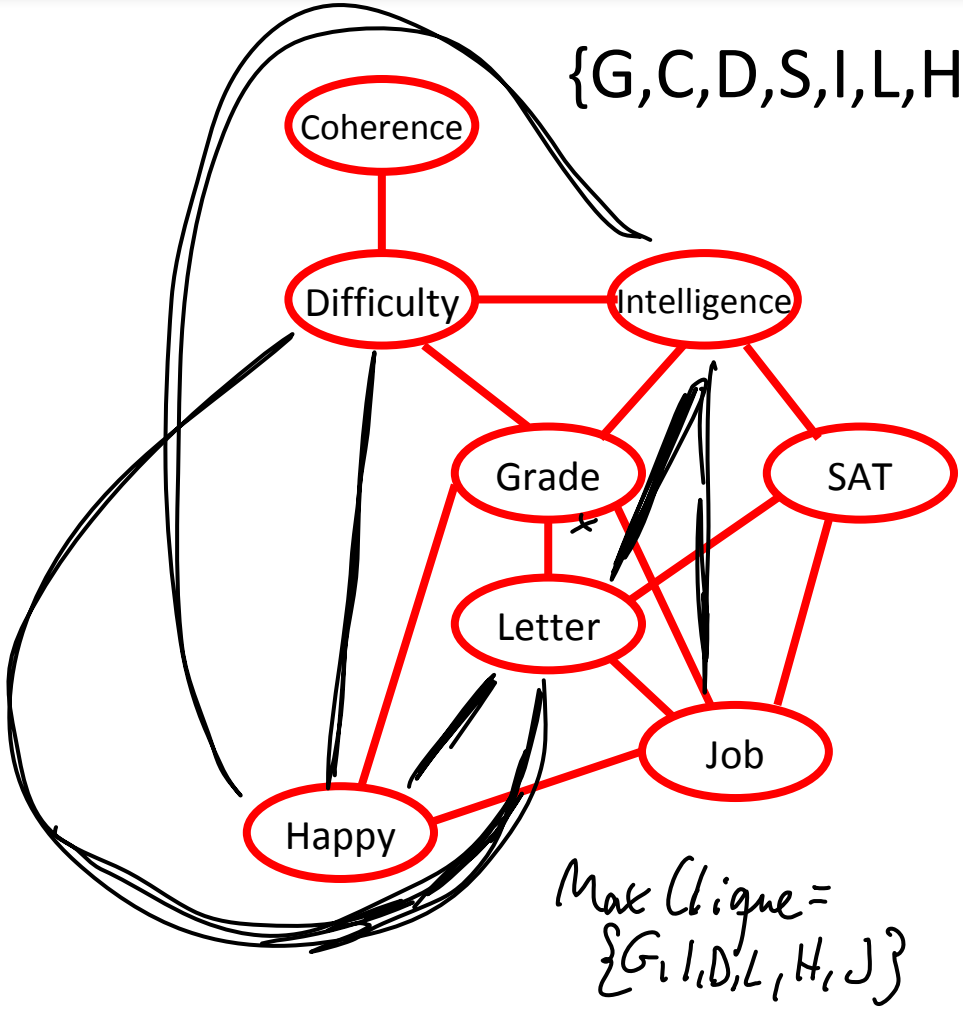
Here: $\text{Scope}(g) = \{D, G, S\}$

Not supported!

Need to fill in edges by connecting all neighbors of I into clique

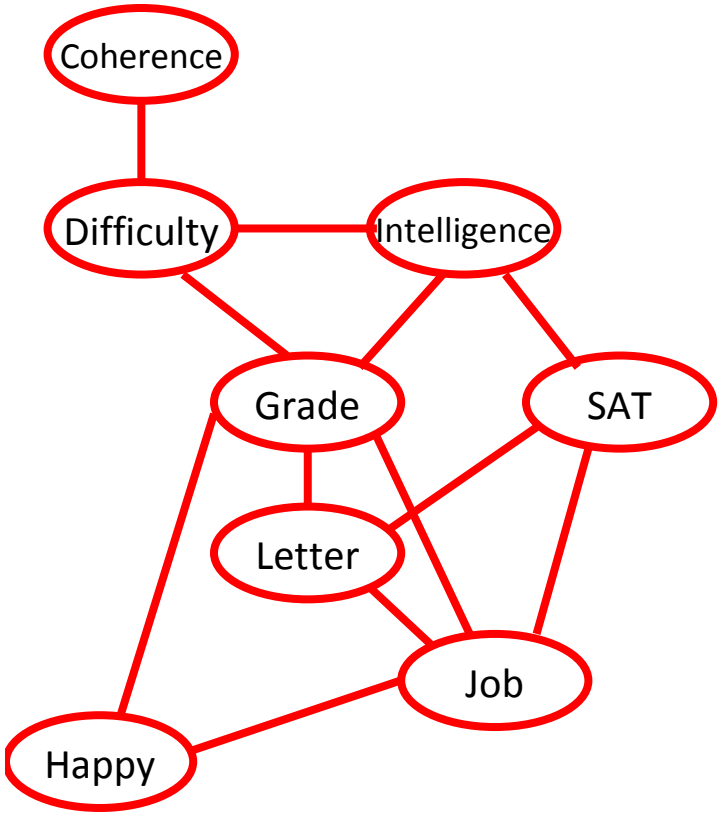
Impact of elimination order





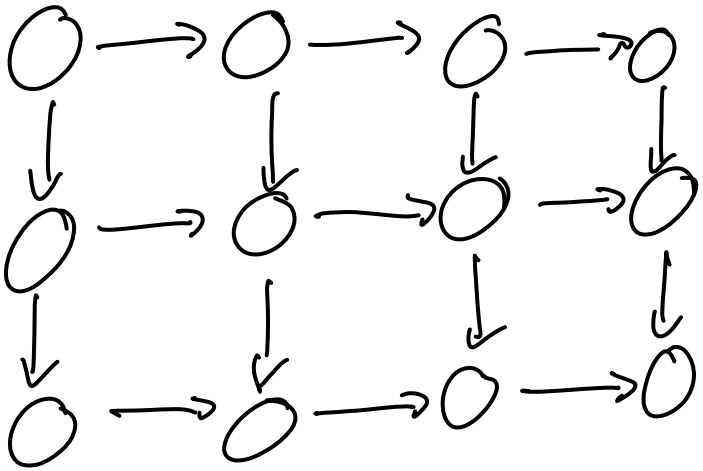
- Different elim. order induce different graphs!

Induced graph and VE complexity

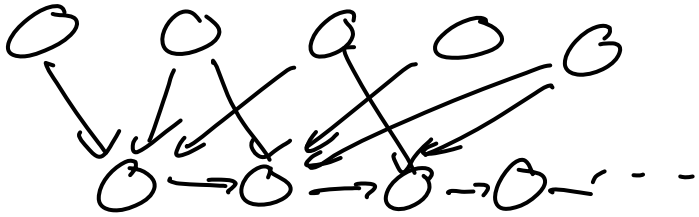


- **Theorem**:
  - All factors arising in VE are defined over cliques (fully connected subgraphs) of the induced graph
  - All maximal cliques of induced graph arise as factors in VE
- Treewidth for ordering = Size of largest clique of induced graph -1
- Treewidth of a graph = minimal treewidth under optimal ordering
- VE exponential in treewidth!

Compact representation → small treewidth?



n x n grid
treewidth n
parents / node = 2



Compact
parents ≤ 4

treewidth ??

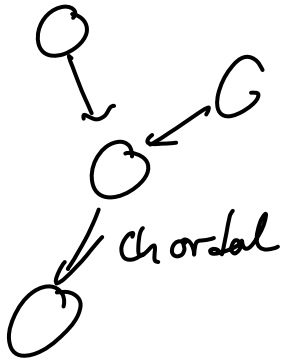
Finding the optimal elimination order

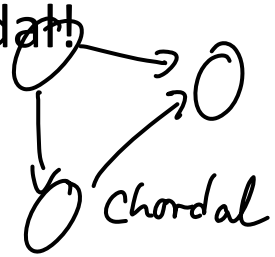
- **Theorem**: Deciding whether there exists an elimination order with induced with at most K is NP-hard
  - Proof by reduction from MAX-CLIQUE
- In fact, can find elimination order in time exponential in treewidth
- Finding optimal ordering as hard as inference…
- For which graphs can we find optimal elimination order?

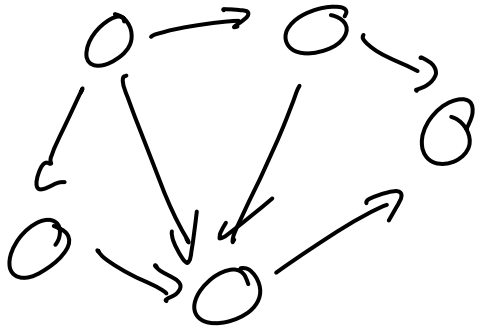
Finding optimal elimination order

- For trees can find optimal ordering (saw before)

- A graph is called chordal if every cycle of length ~~\geq~~ 4 has a chord (an edge between some pair of non-consecutive nodes)
  - Every tree is chordal!







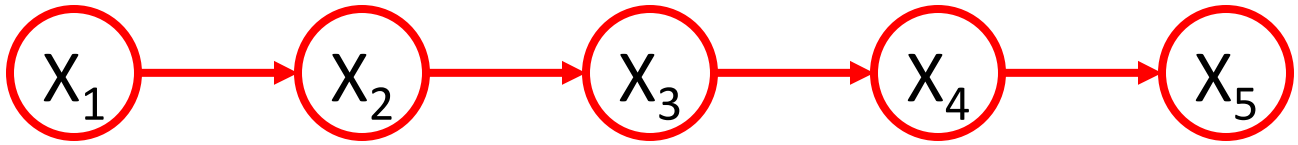
- Can find optimal elimination ordering for chordal graphs

Summary so far

- Variable elimination complexity exponential in induced width for elimination ordering
- Finding optimal ordering is NP-hard
- Many good heuristics
  - Exact for trees, chordal graphs
- Ultimately, inference is NP-hard
- Only difference between cond. prob. queries and MPE is $\sum$ vs. max

- Variable elimination building block for many exact and approximate inference techniques

Answering multiple queries



$$P(x_1, x_5) = P(x_1) \sum_{x_2} P(x_2|x_1) \sum_{x_3} P(x_3|x_2) \sum_{x_4} P(x_4|x_3) P(x_5|x_4)$$

Can compute in $O(n)$ operations (t, k)

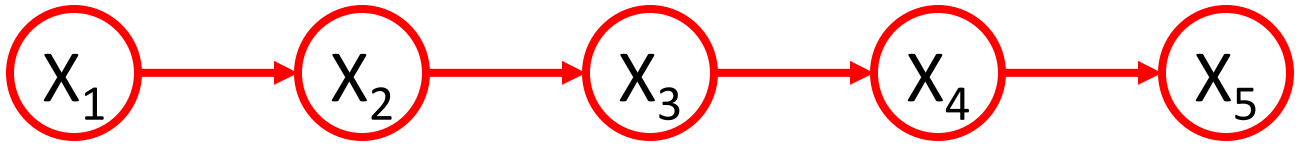
$P(x_2, x_5)$ costs $O(n)$ ops.

$P(x_i, x_5)$ — "$O(n)$ ops

If compute $P(x_i|x_j)$ $\forall i$ $\Rightarrow O(n^2)$

Can we reduce this to $O(n)$??

Reusing computation



$$P(x_1, x_5) = P(x_1) \sum_{x_2} P(x_2 | x_1) \underbrace{\sum_{x_3} P(x_3 | x_2) \underbrace{\sum_{x_4} P(x_4 | x_3) P(x_5 | x_4)}_{g_4(x_3, x_5)}}_{g_3(x_2, x_5)}$$

$$P(x_2, x_5) = \sum_{x_1} P(x_1) P(x_2 | x_1) \underbrace{\sum_{x_3} P(x_3 | x_2) \underbrace{\sum_{x_4} P(x_4 | x_3) P(x_5 | x_4)}_{g_4(x_3, x_5)}}_{\text{already computed!}}$$

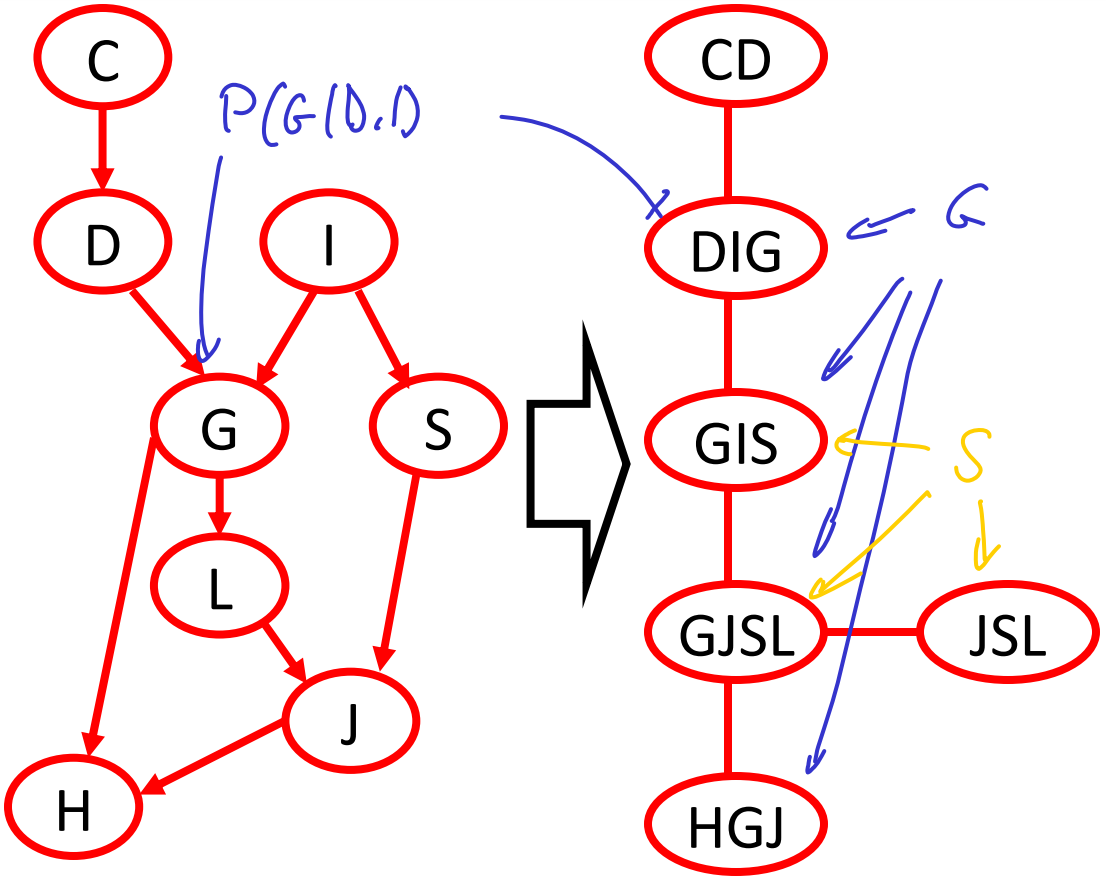
Want to "cache" our computations!

Next

- Will learn about algorithm for efficiently computing all marginals $P(X_i \mid \mathbf{E=e})$ given fixed evidence **E=e**
- Need appropriate data structure for storing the computation

  ➔ Junction trees

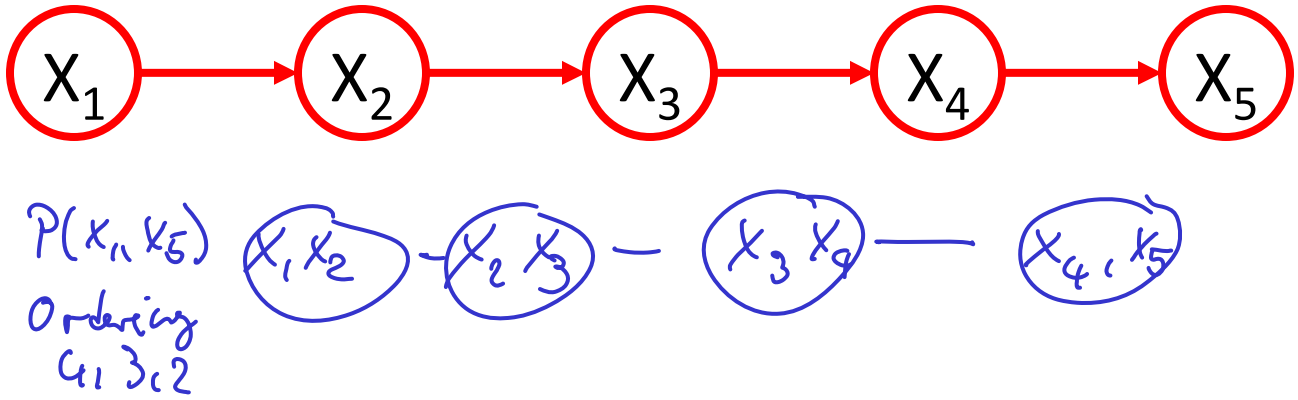
Junction trees



A junction tree for a collection of factors:

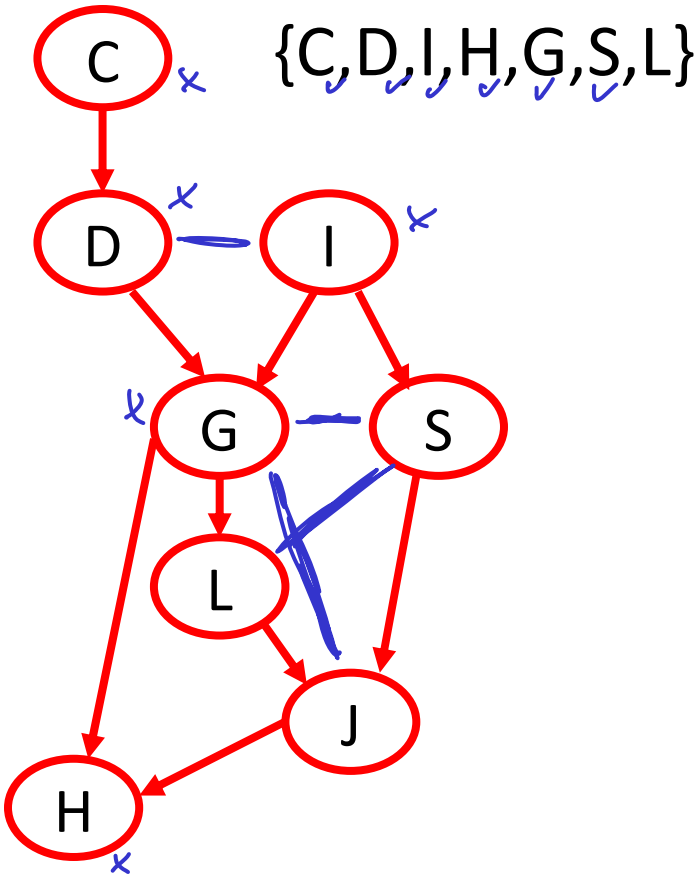
- A tree, where each node is a cluster of variables
- Every factor contained in some cluster C_i
- **Running intersection property**: If $X \in C_i$ and $X \in C_j$, and C_m is on the path between C_i and C_j, then $X \in C_m$

VE constructs a junction tree

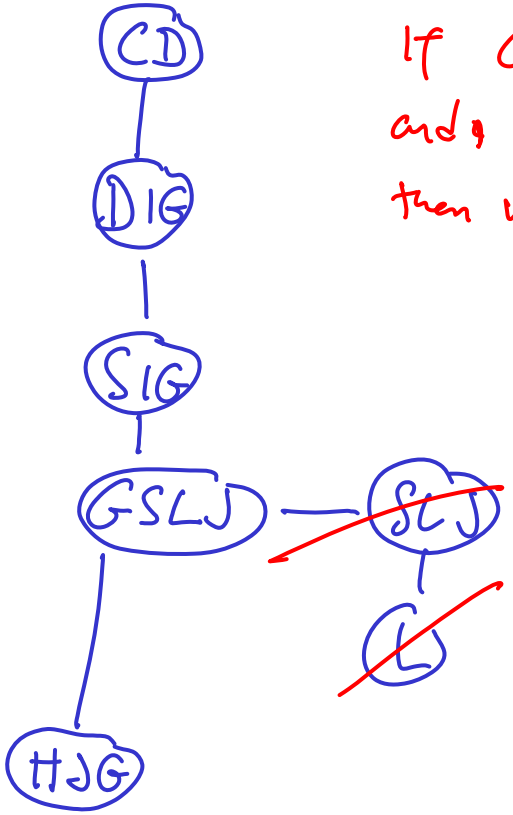


- One clique C_i for each factor f_i created in VE
- C_i connected to C_j if f_i used to generate f_j
- Every factor used only once ➔ Tree
- **Theorem**: resulting tree satisfies RIP

Example: JT from VE

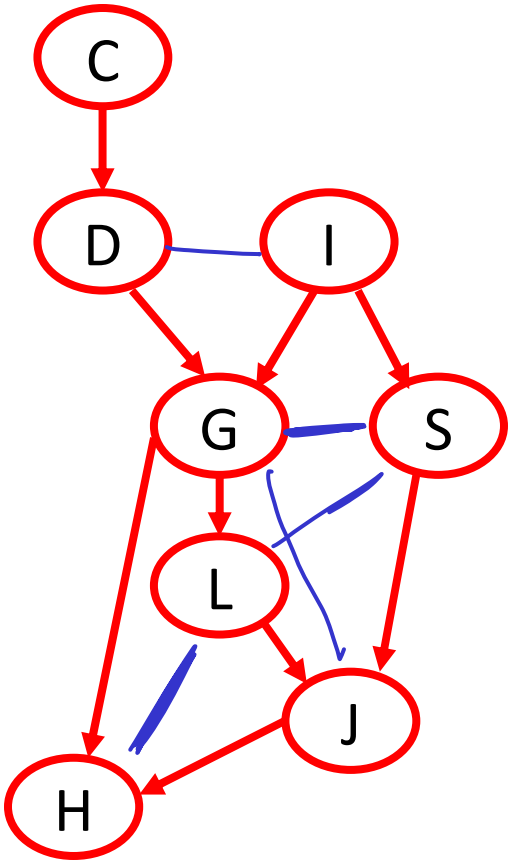


{C,D,I,H,G,S,L}

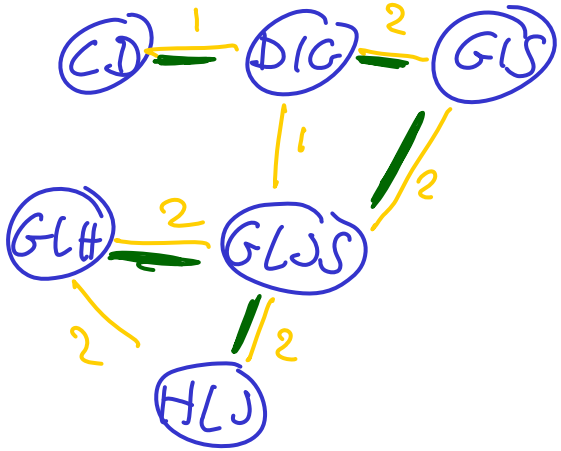


If $C_i - C_j$ and $C_i \subseteq C_j$ then merge C_i, C_j

Constructing JT from chordal graph

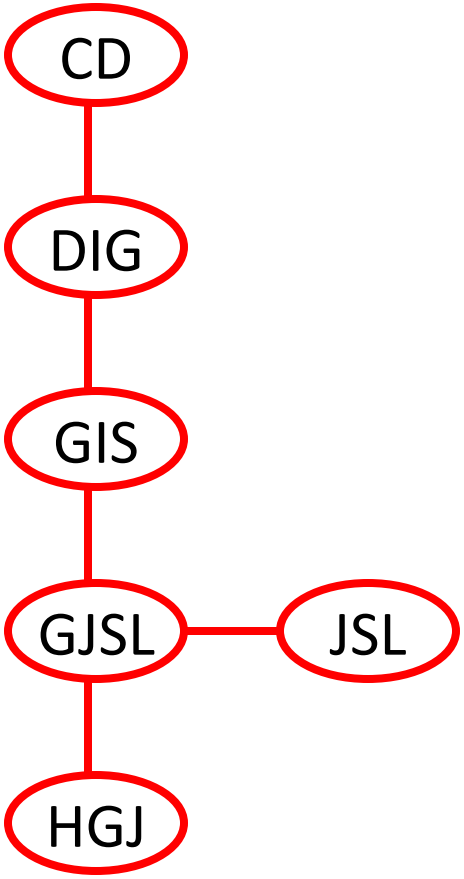


1. Moralize
2. Triangulate (make chordal)
3. Identify max. cliques
4. Connect cliques into undirected graph
   $w(C_i, C_j) = |C_i \cap C_j|$
5. Find Max ST



⇒ Results in valid junction tree

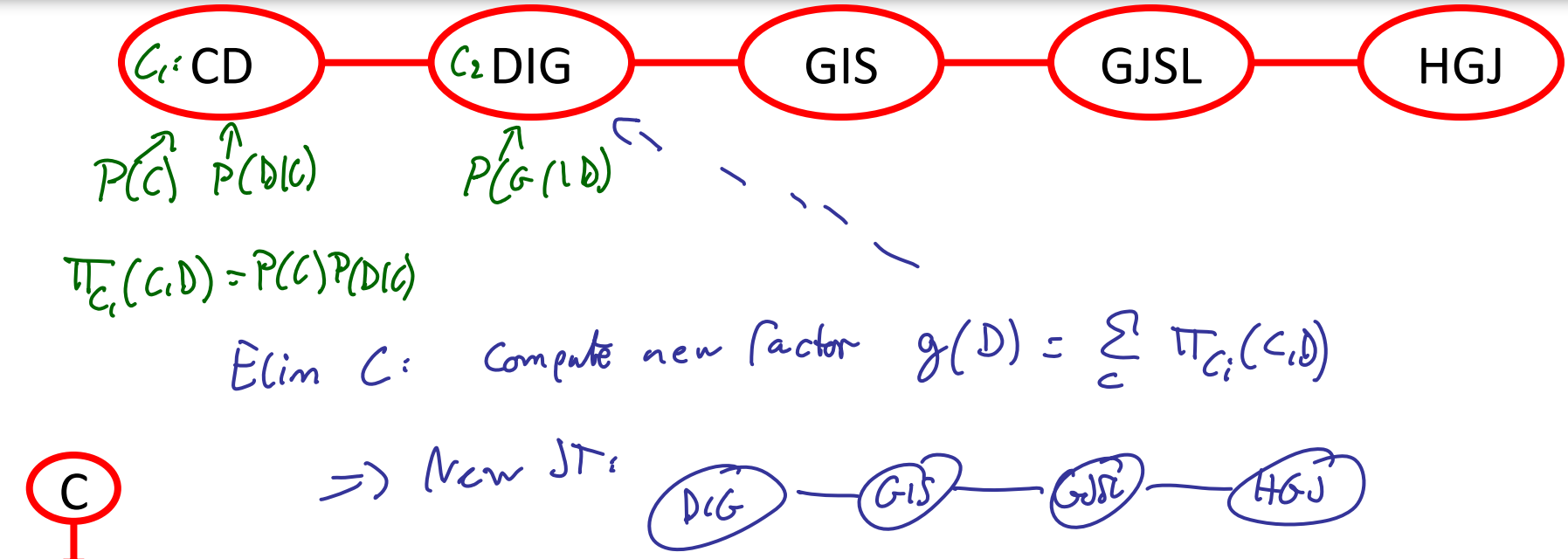
Junction trees and independence

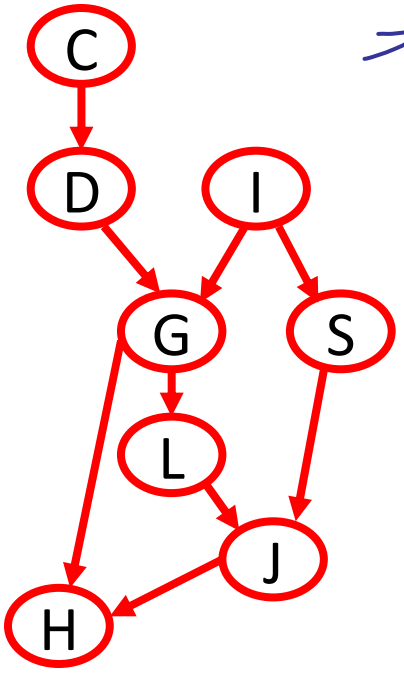


Theorem:

- Suppose
  - T is a junction tree for graph G and factors F
  - Consider edge $\mathbf{C_i} - \mathbf{C_j}$ with separator $\mathbf{S_{i,j}} = C_i \cap C_j$
  - Variables **X** and **Y** on opposite sites of separator
- Then $\mathbf{X} \perp \mathbf{Y} \mid \mathbf{S_{i,j}}$
- Furthermore, $I(T) \subseteq I(G)$

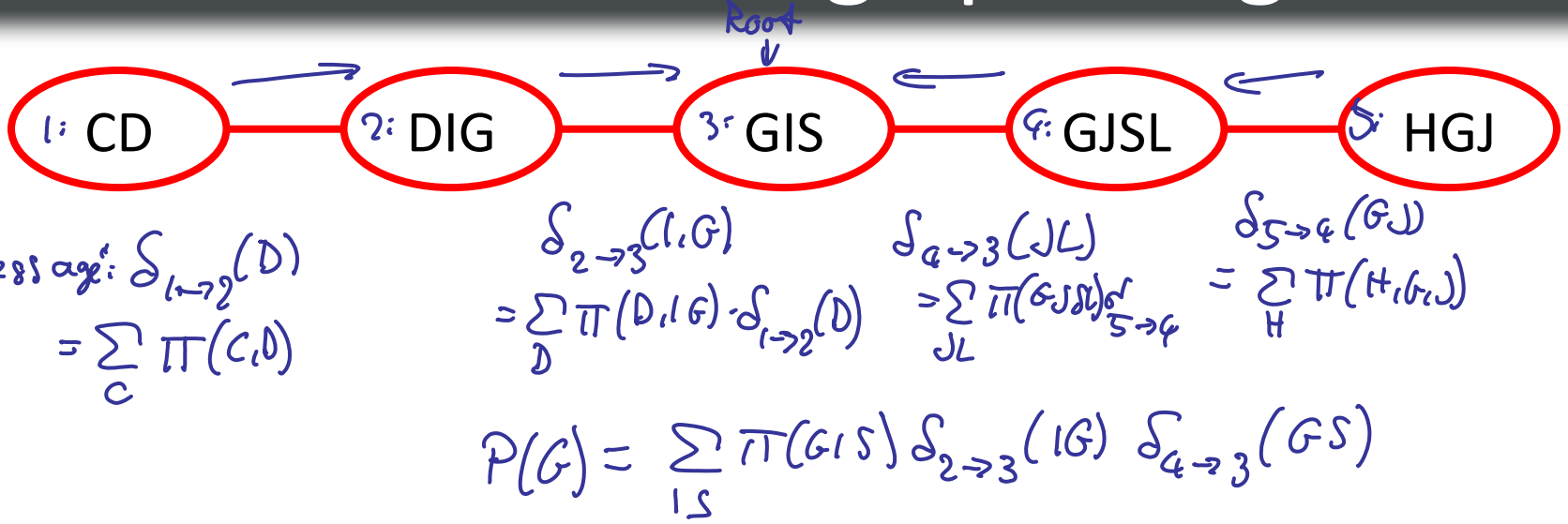
Variable elimination in junction trees

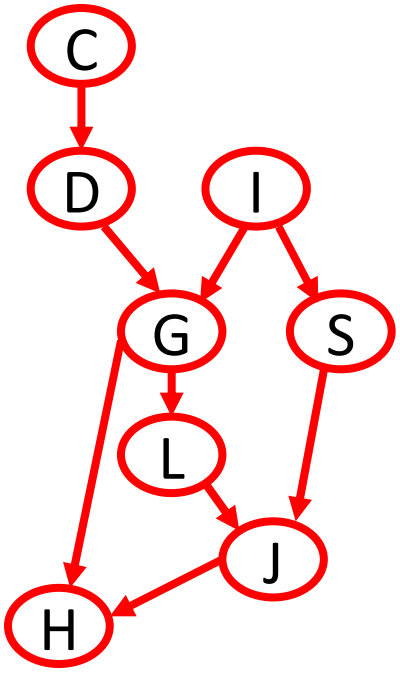




- Associate each CPTs with a clique
- Potential π_C of clique C is product of assigned CPTs

VE as message passing





VE for computing X_i

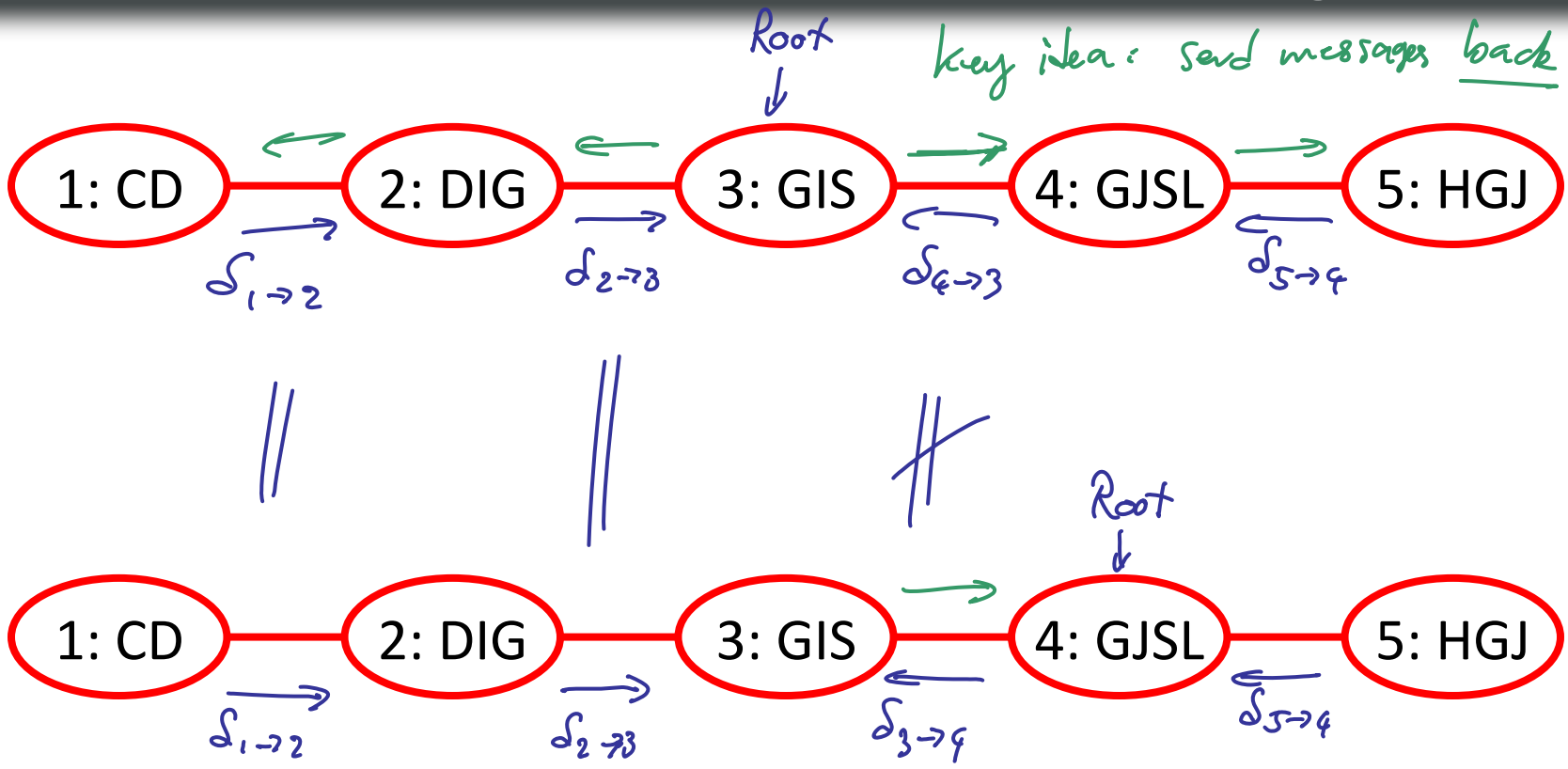
- Pick root (any clique containing X_i)
- Don't eliminate, only send messages recursively from leaves to root
  - Multiply incoming messages with clique potential
  - Marginalize variables not in separator
- Root "ready" when received all messages

Correctness of message passing

- **Theorem**: When root ready (received all messages), all variables in root have correct potentials
  - Follows from correctness of VE

- So far, no gain in efficiency ☹

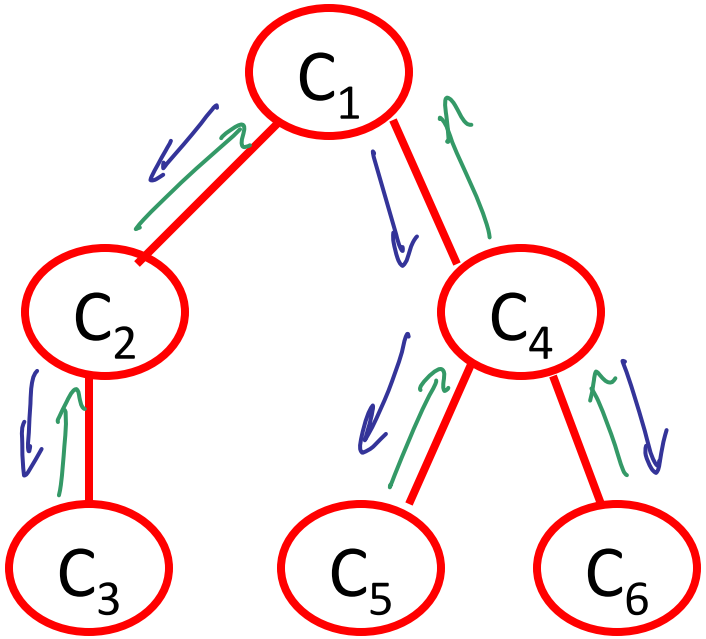
Does the choice of root affect messages?



Instead of $O(n^2) \to O(n)$

1 message per edge per direction

Shenoy-Shafer algorithm



- Clique i ready if received messages from all neighbors but 1
  - Leaves always ready
- While there exists a message $\delta_{i \to j}$ ready to transmit send message

Complexity? $O(n\, 2^{\text{treewidth}})$

1 msg per edge per direction

"Only" exp. in treewidth

Theorem: At convergence, every clique has correct beliefs

Inference using VE

- Want to incorporate evidence E=e
- Multiply all cliques containing evidence variables with indicator potential 1_e

(AB) $1_{A=T}(a,b) = \begin{cases} 1 & \text{if } a = T \\ 0 & \text{if } a = F \end{cases}$

- Perform variable elimination

Summary so far

- Junction trees represent distribution
  - Constructed using elimination order
  - Make complexity of inference explicitly visible
- Can implement variable elimination on junction trees to compute correct beliefs on all nodes

- Now:
  - **Belief propagation** – an important alternative to VE on junction trees.
  - Will later generalize to approximate inference!
  - Key difference: Messages obtained by division rather than multiplication