Clique Trees

Amr Ahmed

October 23, 2008

Outline

- Clique Trees
- Representation
- Factorization
- Inference
- Relation with VE

Representation

- Given a Probability distribution, *P*
  - How to represent it?
  - What does this representation tell us about *P*?
    - Cost of inference
    - Independence relationships
  - What are the options?
    - Full CPT
    - Bayes Network (list of factors)
    - Clique Tree

Representation

- FULL CPT
  - Space is exponential
  - Inference is exponential
  - Can read nothing about independence in *P*
  - Just bad

Representation: the past

- Bayes Network
  - List of Factors: $P(X_i|Pa(X_i))$
    - Space efficient
  - Independence
    - Read local Markov Ind.
    - Compute global independence via d-seperation
  - Inference
    - Can use dynamic programming by leveraging factors
    - Tell us little immediately about cost of inference
      - Fix an elimination order
      - Compute the induced graph
      - Find the largest clique size
        - » Inference is exponential in this largest clique size

Representation: Today

- Clique Trees (CT)
  - Tree of cliques
  - Can be constructed from Bayes network
    - Bayes Network + Elimination order → CT
  - What independence can read from CT about *P?*
  - How P factorizes over CT?
  - How to do inference using CT?
  - When should you use CT?

The Big Picture

Full CPT

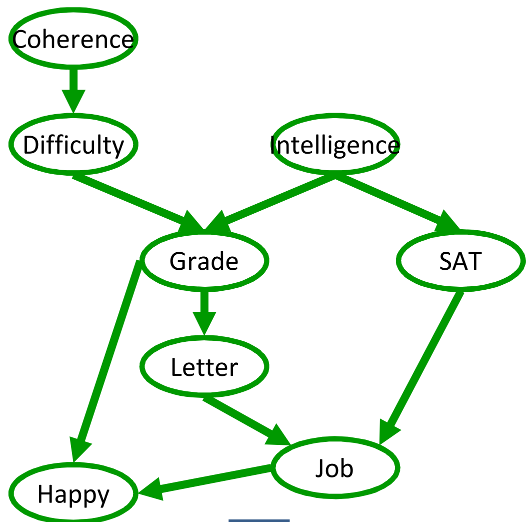
a^1	b^1	c^1	0.5·0.5
a^1	b^1	c^2	0.5·0.7
a^1	b^2	c^1	0.8·0.1
a^1	b^2	c^2	0.8·0.2
a^2	b^1	c^1	0.1·0.5
a^2	b^1	c^2	0.1·0.7
a^2	b^2	c^1	0·0.1
a^2	b^2	c^2	0·0.2
a^3	b^1	c^1	0.3·0.5
a^3	b^1	c^2	0.3·0.7
a^3	b^2	c^1	0.9·0.1
a^3	b^2	c^2	0.9·0.2

many

PDAG,
Minimal
I-MAP,etc

Inference is
Just summation

List of local factors

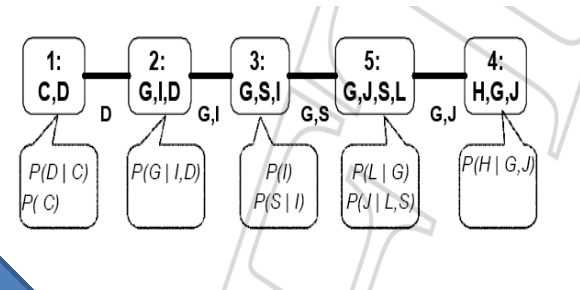


VE operates in factors by caching computations (intermediate factors) within a single inference operation P(X|E)

many

Choose an
Elimination order,
Max-spaning tree

Tree of Cliques



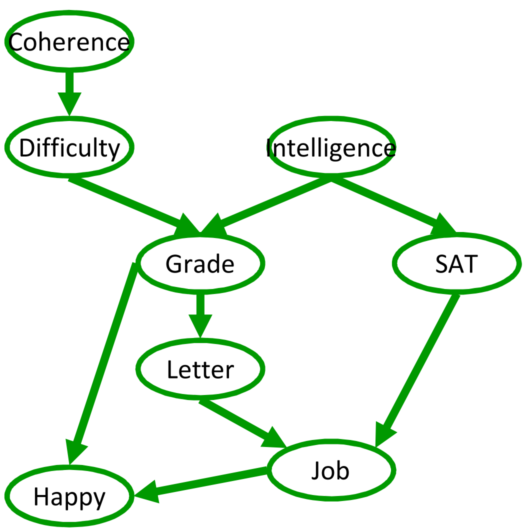
CT enables caching computations (intermediate factors) across multiple inference operations P(X_i,X_j) for all I,j

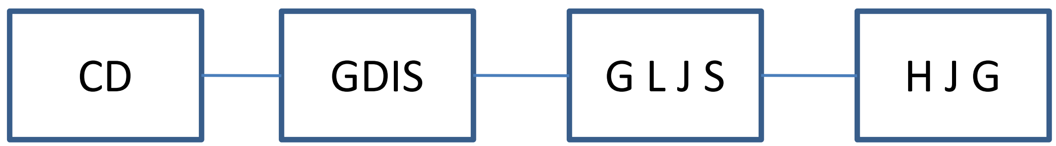
Clique Trees: Representation

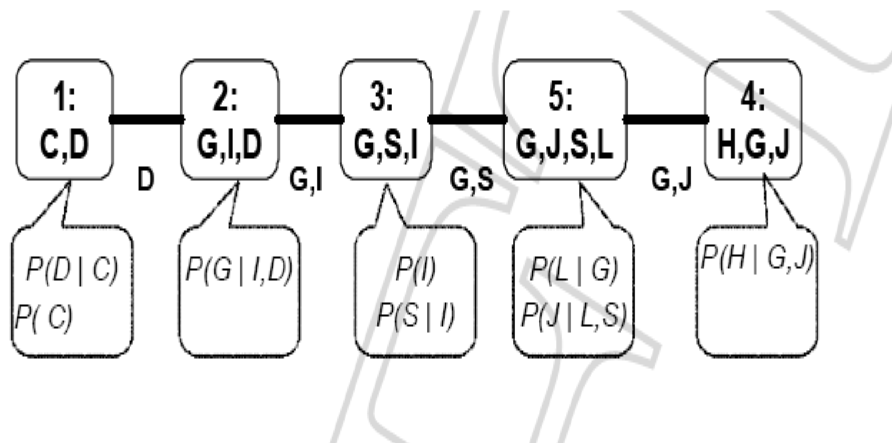
- For set of factors *F* (i.e. Bayes Net)
  - Undirected graph
  - Each node i associated with a cluster $\mathbf{C}_i$
  - *Family preserving*: for each factor $f_j \in F$, $\exists$ node i such that scope[f_i] $\subseteq \mathbf{C}_i$
  - Each edge i – j is associated with a separator $\mathbf{S}_{ij}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

Clique Trees: Representation

- *Family preserving over factors*
- Running Intersection Property
- Both are correct Clique trees

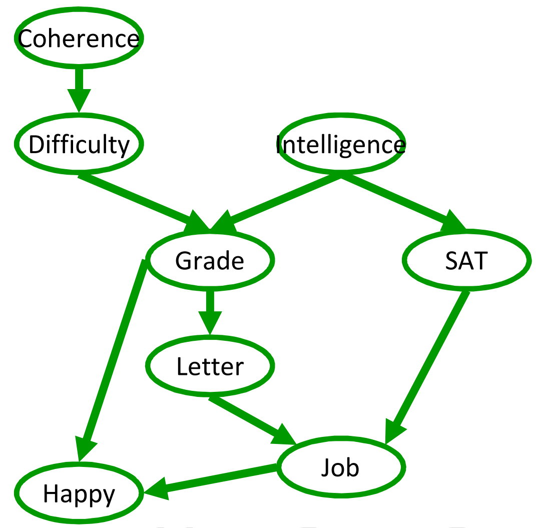






Clique Trees: Representation

- What independence can be read from CT
  - I(CT) subset I(G) subset I(P)
- Use your intuition
  - How to block a path?
    - Observe a separator. Q4

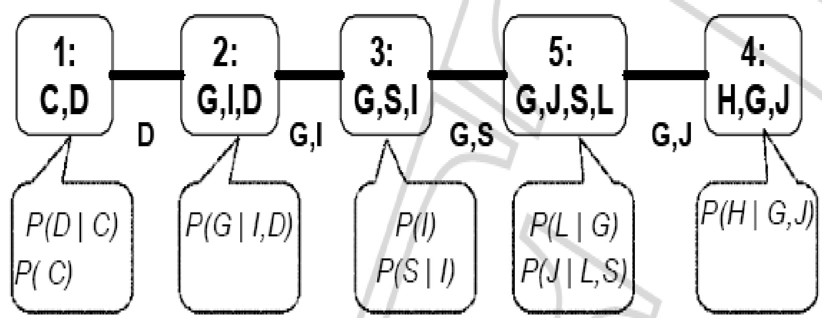


$$C \perp G \mid D$$

$$H \perp I \mid G, J$$

$$H \perp I \mid G, S$$

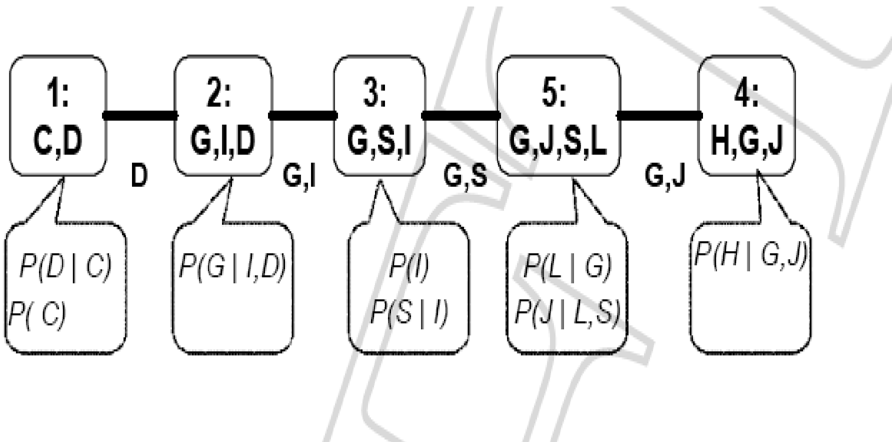
$$CD \perp HJ \mid GI$$

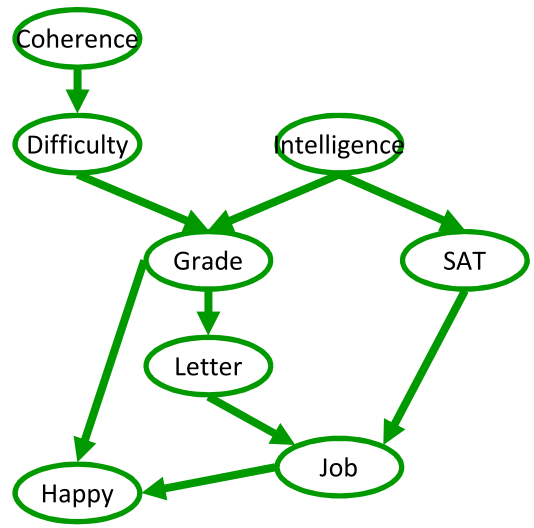


Clique Trees: Representation

- How P factorizes over CT (when CT is calibrated) Q4 (See 9.2.11)

$$P(\mathbf{X}) = \frac{\prod_i P(\mathbf{C}_i)}{\prod_{ij} P(\mathbf{S}_{ij})}$$



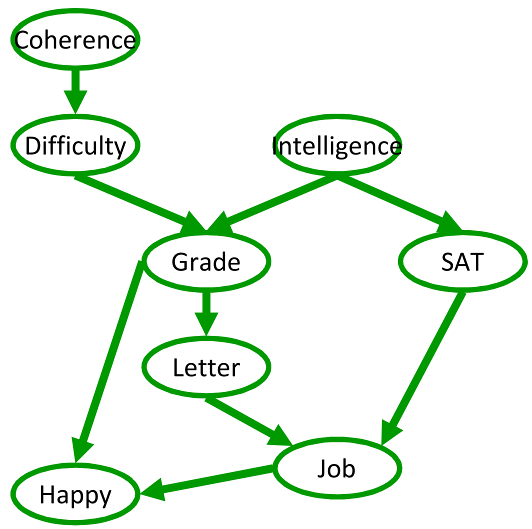


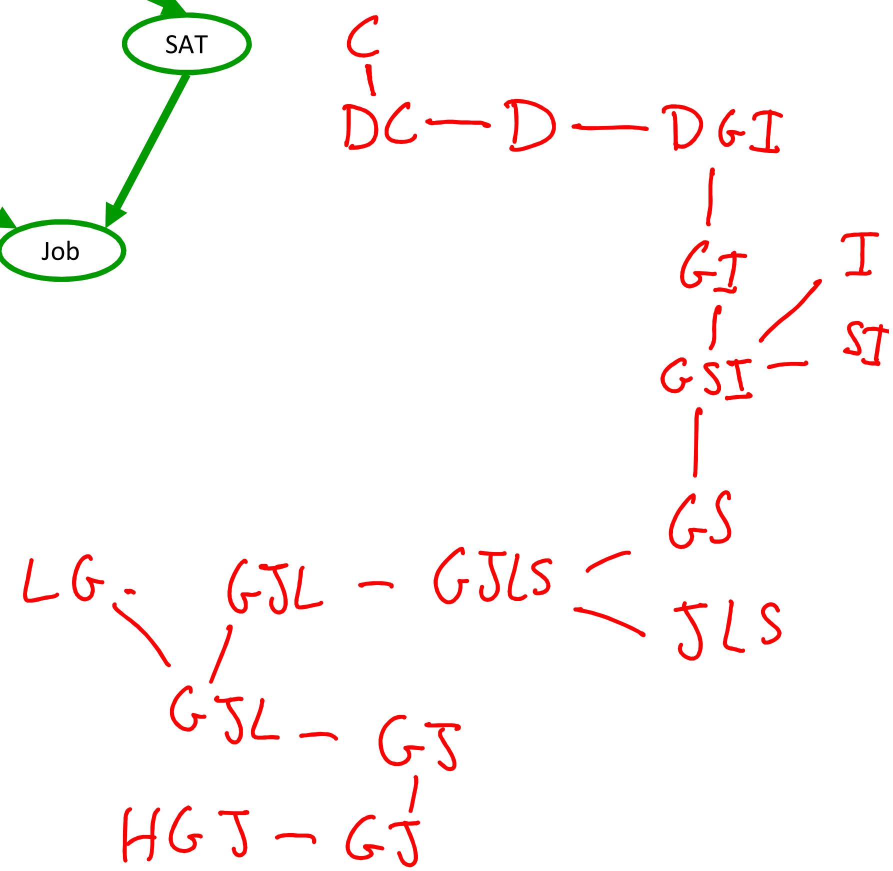
Representation Summary

- Clique trees (like Bayes Net) has two parts
  - Structure
  - Potentials (the parallel to CPTs in BN)
    - Clique potentials
    - Separator Potential
    - Upon calibration, you can read marginals from the cliques and separator potentials
- Initialize clique potentials with factors from BN
  - Distribute factors over cliques (family preserving)
  - Cliques must satisfy RIP
- But wee need calibration to reach a fixed point of these potentials (see later today)
- Compare to BN
  - You can only read local conditionals $P(x_i|pa(x_i))$ in BN
    - You need VE to answer other queries
  - In CT, upon calibration, you can read marginals over cliques
    - You need VE over calibrated CT to answer queries whose scope can not be confined to a single clique

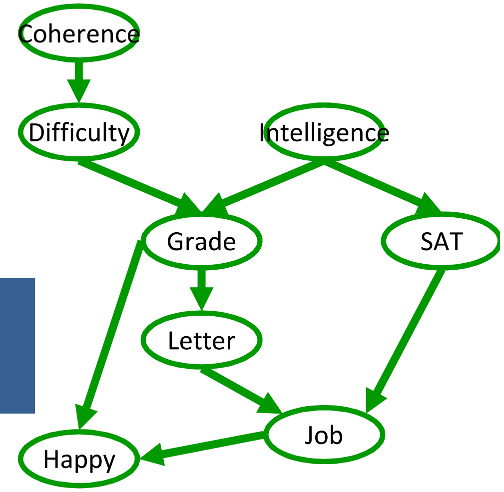
Clique tree Construction

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





Clique tree Construction (details)



- Replay VE with order: C,D,I,H,S, L,J,G

Initial factors: C, DC, GDI, SI, I, LG, JLS, HJG

Eliminate C: multiply CD, C to get factor with CD, then marginalize C To get a factor with D.

C — CD — D

Eliminate D: multiply D, GDI to get factor with GDI, then marginalize D to get a factor with GI

C — CD — D — DGI — GI

Eliminate I: multiply GI, SI, I to get factor with GSI, then marginalize I to get a factor with GS

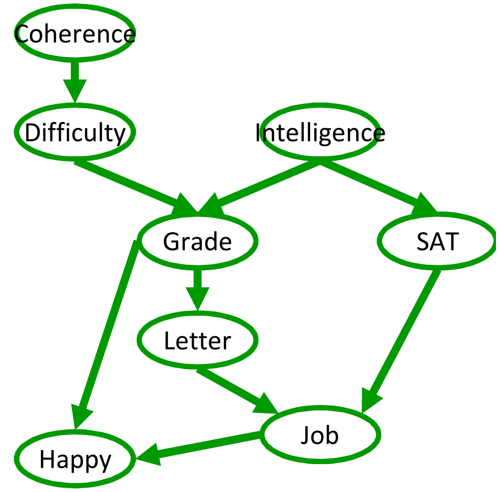
C — CD — D — DGI — GI — GSI — GS

GSI — I

GSI — SI

Clique tree Construction (details)

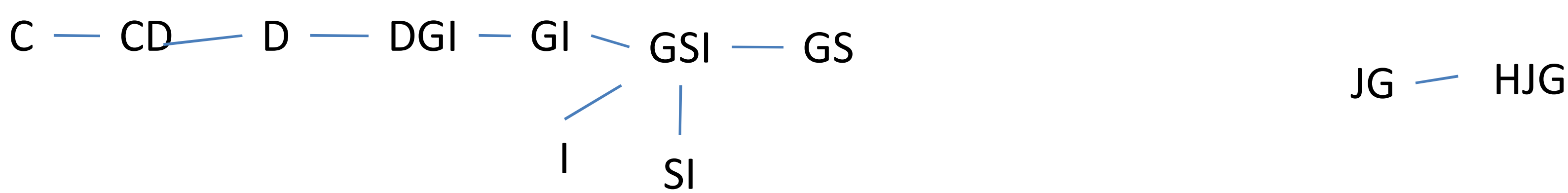
- Replay VE with order: C,D,I,H,S, L,J,G



Initial factors: C, DC, GDI, SI, I, LG, JLS, HJG

Eliminate I: multiply GI, SI, I to get factor with GSI, then marginalize I to get a factor with GS

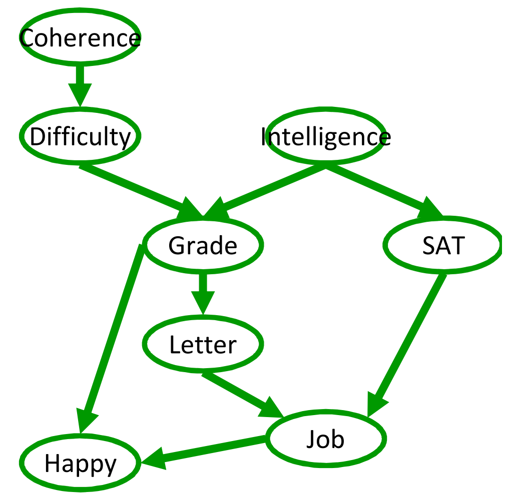
Eliminate H: just marginalize HJG to get a factor with JG



Clique tree Construction (details)

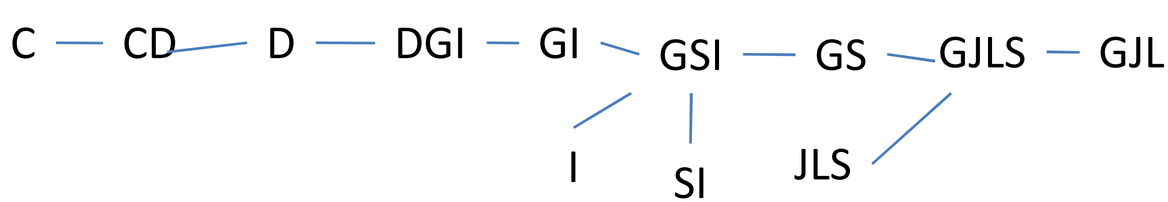
- Replay VE with order: C,D,I,H,S, L,J,G

Initial factors: C, DC, GDI, SI, I, LG, JLS, HJG



Eliminate H: just marginalize HJG to get a factor with JG

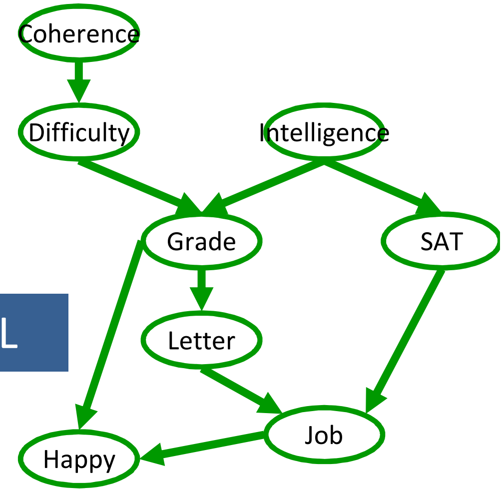
Eliminate S: multiply GS, JLS to get GJLS, then marginalize S to get GJL



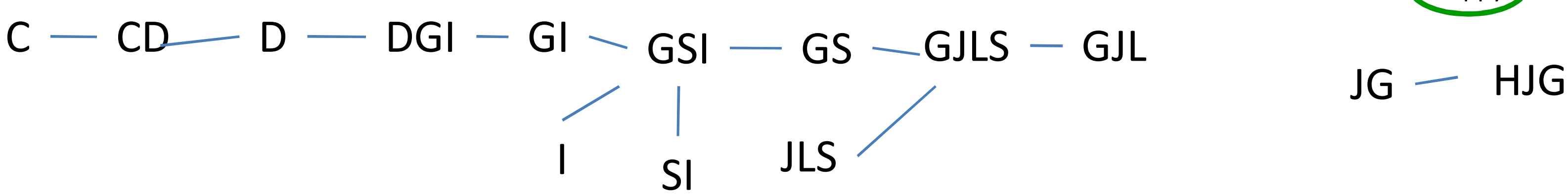
Clique tree Construction (details)

- Replay VE with order: C,D,I,H,S, L,J,G

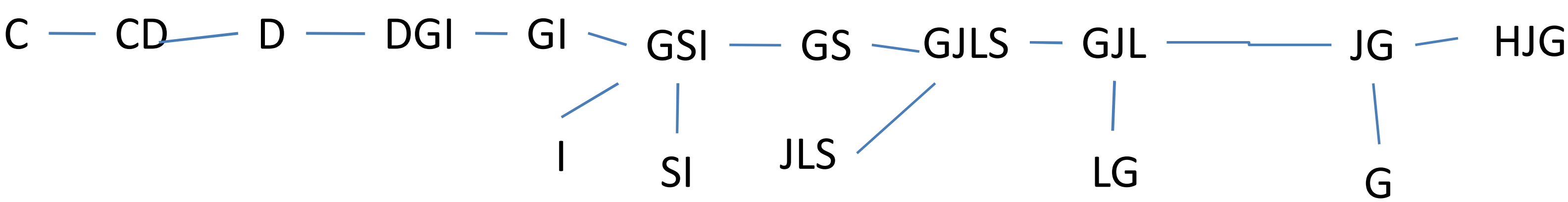
Initial factors: C, DC, GDI, SI, I, LG, JLS, HJG



Eliminate S: multiply GS, JLS to get GJLS, then marginalize S to get GJL



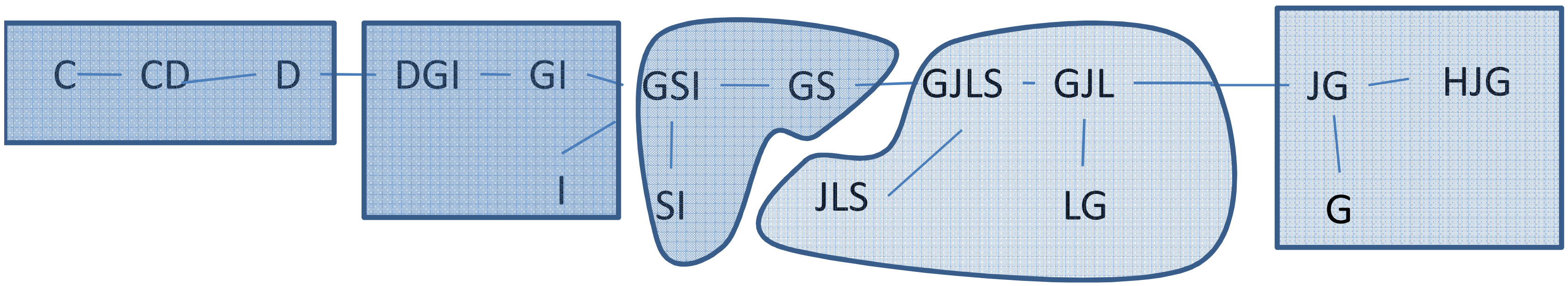
Eliminate L: multiply GJL, LG to get JLG, then marginalize L to get GJ



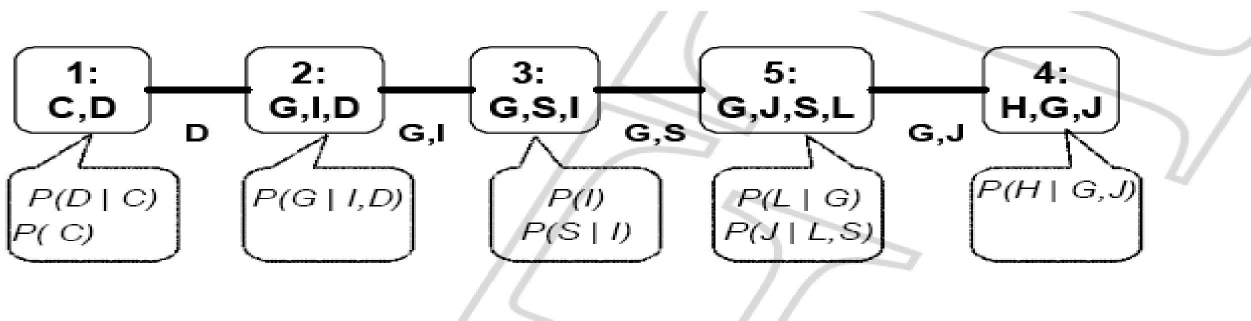
Eliminate L, G:
JG→ G

Clique tree Construction (details)

- Summarize CT by removing subsumed nodes



- Satisfy RIP and Family preserving (always true for any Elimination order)
- Finally distribute initial factor into the cliques, to get initial beliefs (which is the parallel of CPTs in BN) , to be used for inference

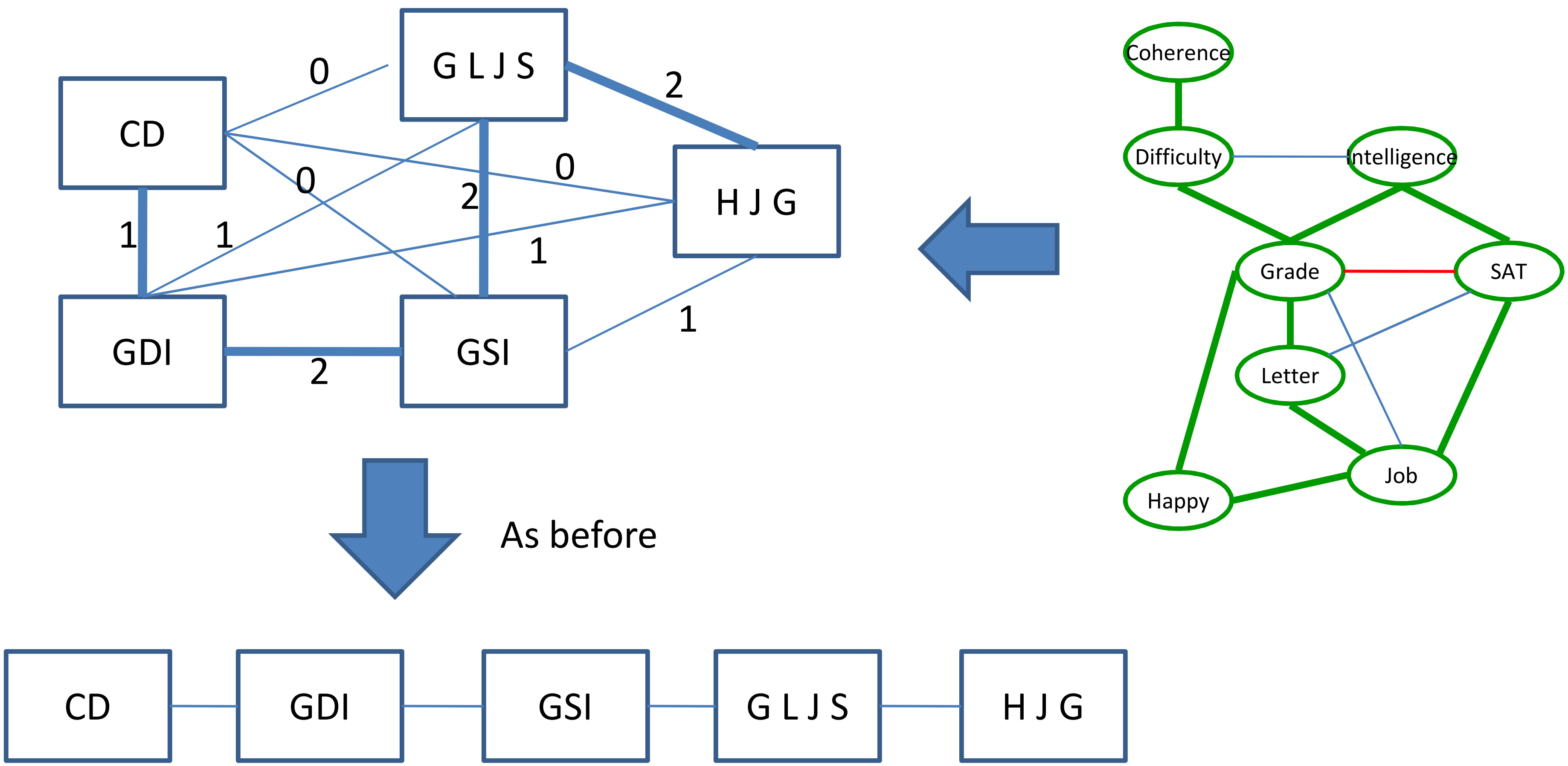


Clique tree Construction: Another method

- From a triangulated graph
  - Still from VE, why?
  - Elimination order → triangulation
  - Triangulation → Max cliques
  - Connect cliques, find max-spanning tree

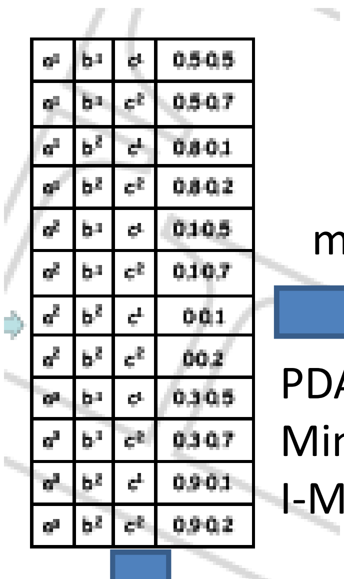
Clique tree Construction: Another method (details)

- Get choral graph (add fill edges) for the same order as before C,D,I,H,S, L,J,G.
- Extract Max cliques from this graph and get maximum-spanning clique tree



The Big Picture

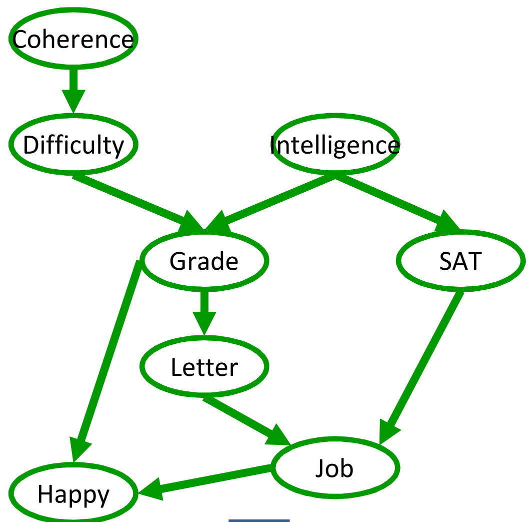
Full CPT



many

PDAG,
Minimal
I-MAP,etc

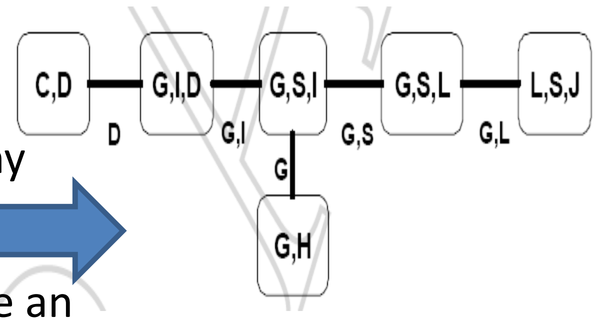
List of local factors



many

Choose an
Elimination order,
Max-spaning tree

Tree of Cliques



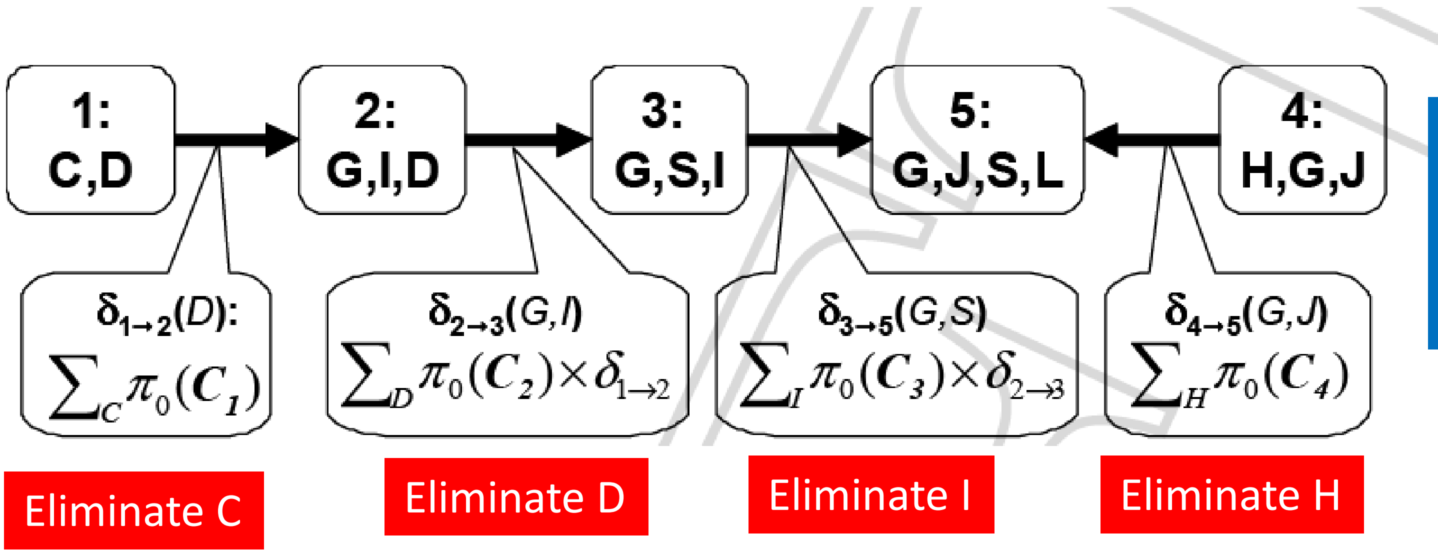
Inference is
Just summation

VE operates in factors by caching computations (intermediate factors) within a single inference operation P(X|E)

CT enables caching computations (intermediate factors) across multiple inference operations $P(X_i,X_j)$ for all I,j

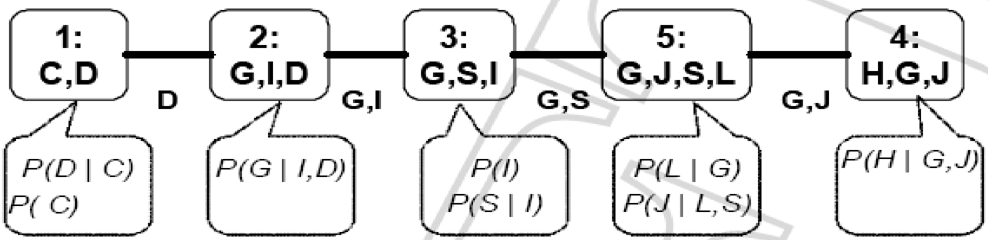
Clique Tree: Inference

- P(X): assume X is in a node (root)
- Just run VE! Using elimination order dictated by the tree and initial factors put into each clique to define \Pi$_0$(C$_i$)
- When done we have P(G,J,S,L)



In VE jargon, we assign these messages names like g1, g2, etc.

Clique Tree: Inference (2)



Initial Local belief

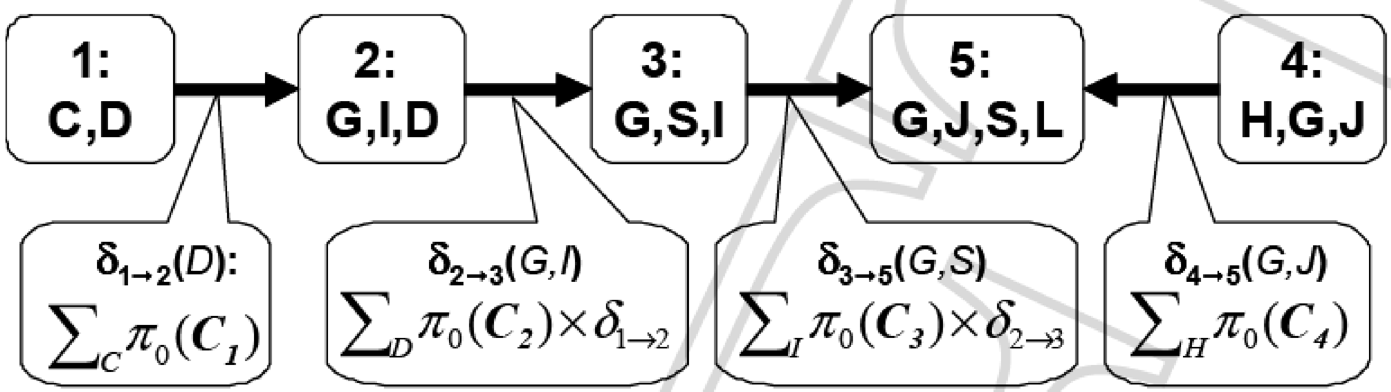
What is: $\underline{\delta_{1->2}(D)} = \sum_C \pi_0(C1) = \sum_C P(C)P(D|C)$

Just a factor over D

What is: $\delta_{2->3}(G,I) = \sum_D \pi_0(C2) \times \underline{\delta_{1->2}(D)} = \sum_C \delta_{1->2}(D) P(G|I,D)$

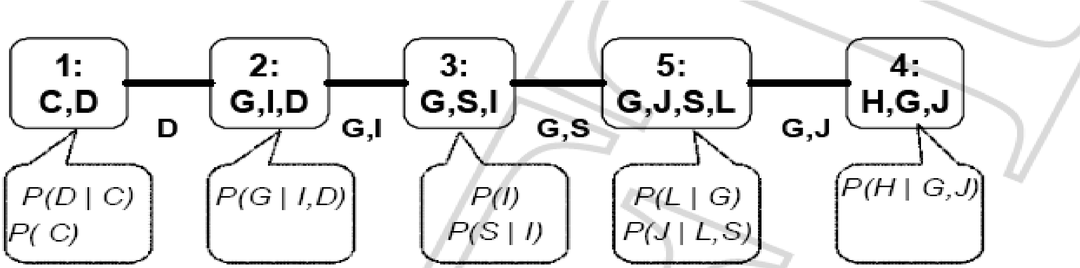
Just a factor over GI

We are simply doing VE along "partial" order determined by the tree: (C,D,I) and H (i.e. H can Be anywhere in the order)



In VE jargon, we call these messages with names like g1, g2, etc.

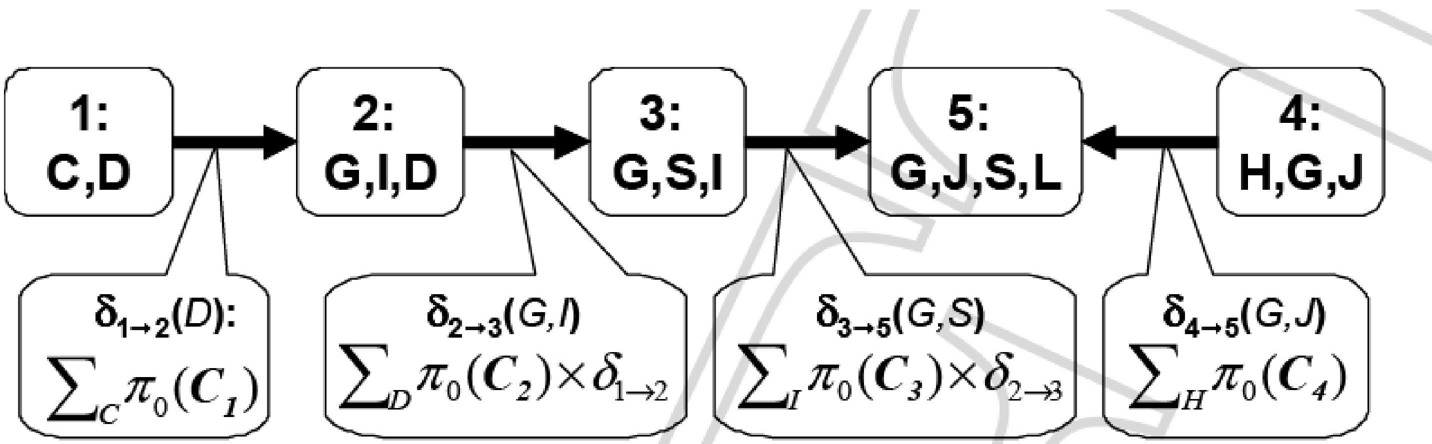
Clique Tree: Inference (3)



Initial Local belief

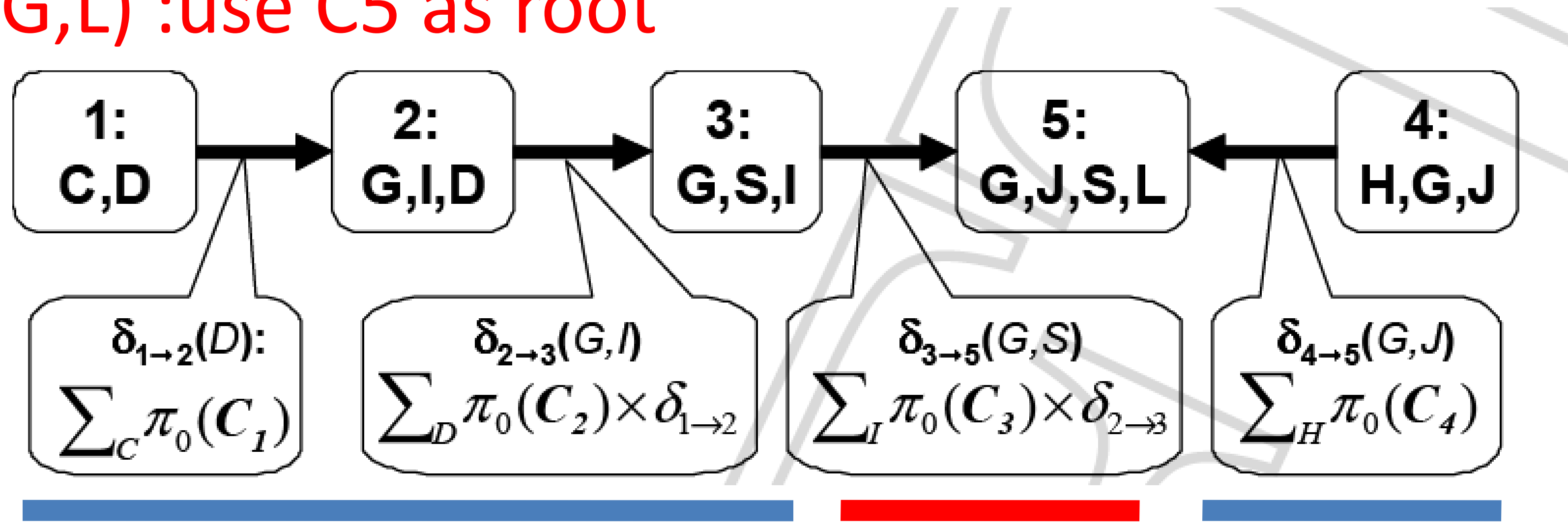
-When we are done, C5 would have received two messages from left and right
- In VE, we will end up with factors corresponding to these messages in addition to all factors that were distributed into C5: P(L|G), P(J|L,G)

-In VE, we multiply all these factors to get the marginals
- In CT, we multiply all factors in C5 : \Pi_0(C_5) with these two messages to get C_5 calibrated potential (which is also the marginal), so what is the deal? Why this is useful?



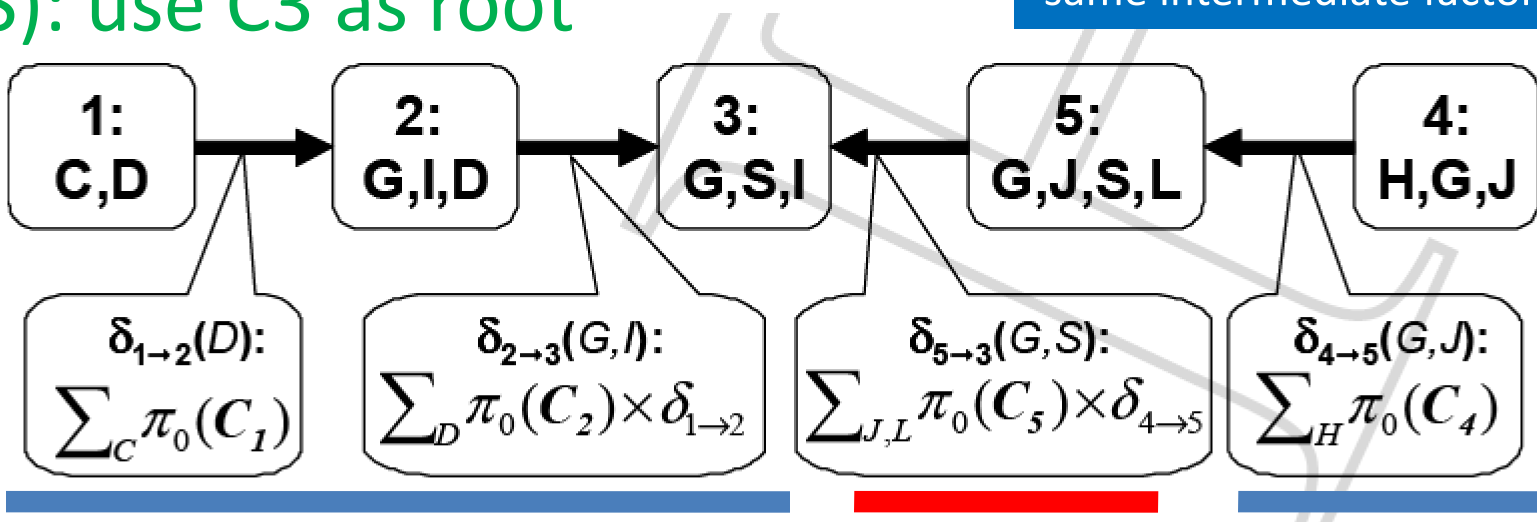
Clique Tree: Inter-Inference Caching

P(G,L) :use C5 as root

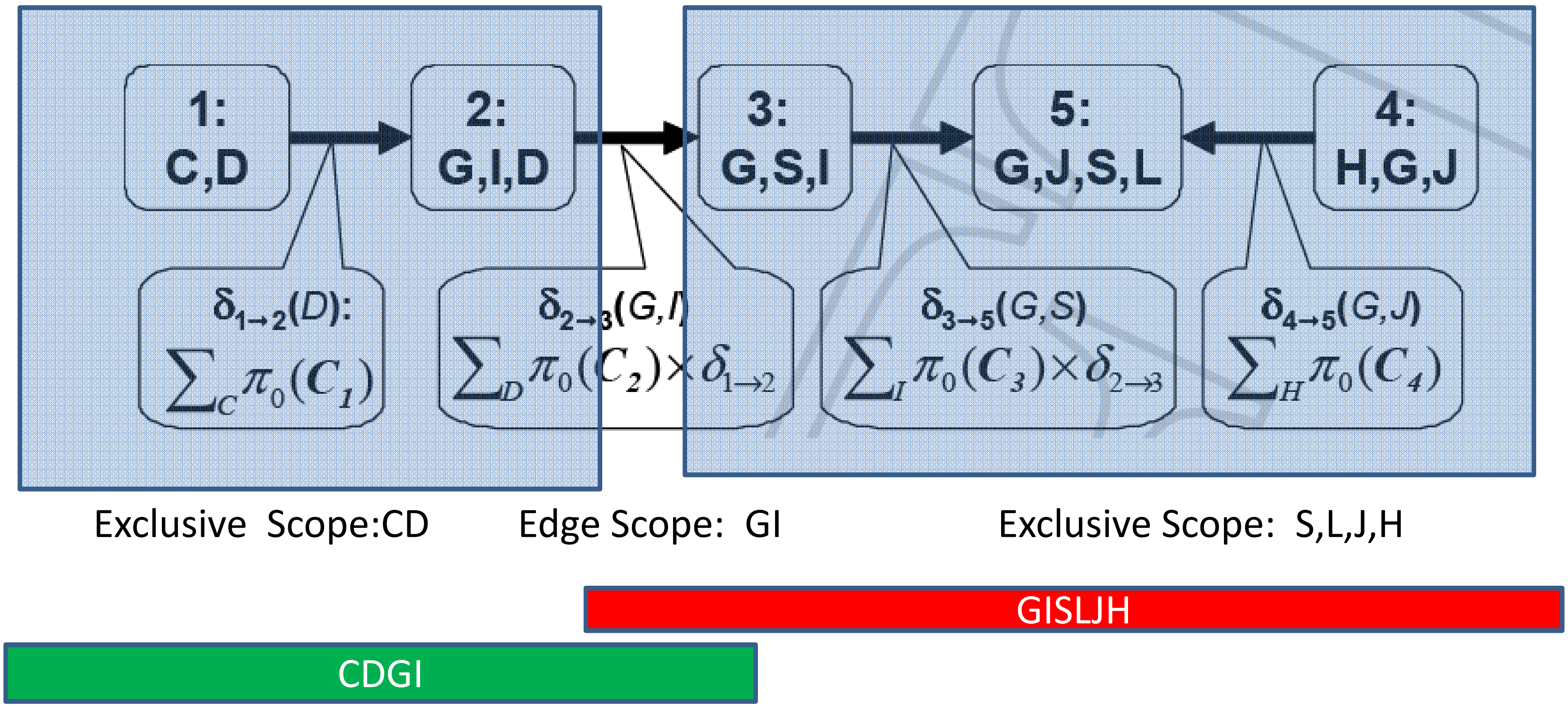


Notice the same 3 messages: i.e. same intermediate factors in VE

P(I,S): use C3 as root



What is passed across the edge?



-The message summarizes what the right side of the tree cares about in the left side **(GI)**

- See Theorem 9.2.3

- Completely determined by the root

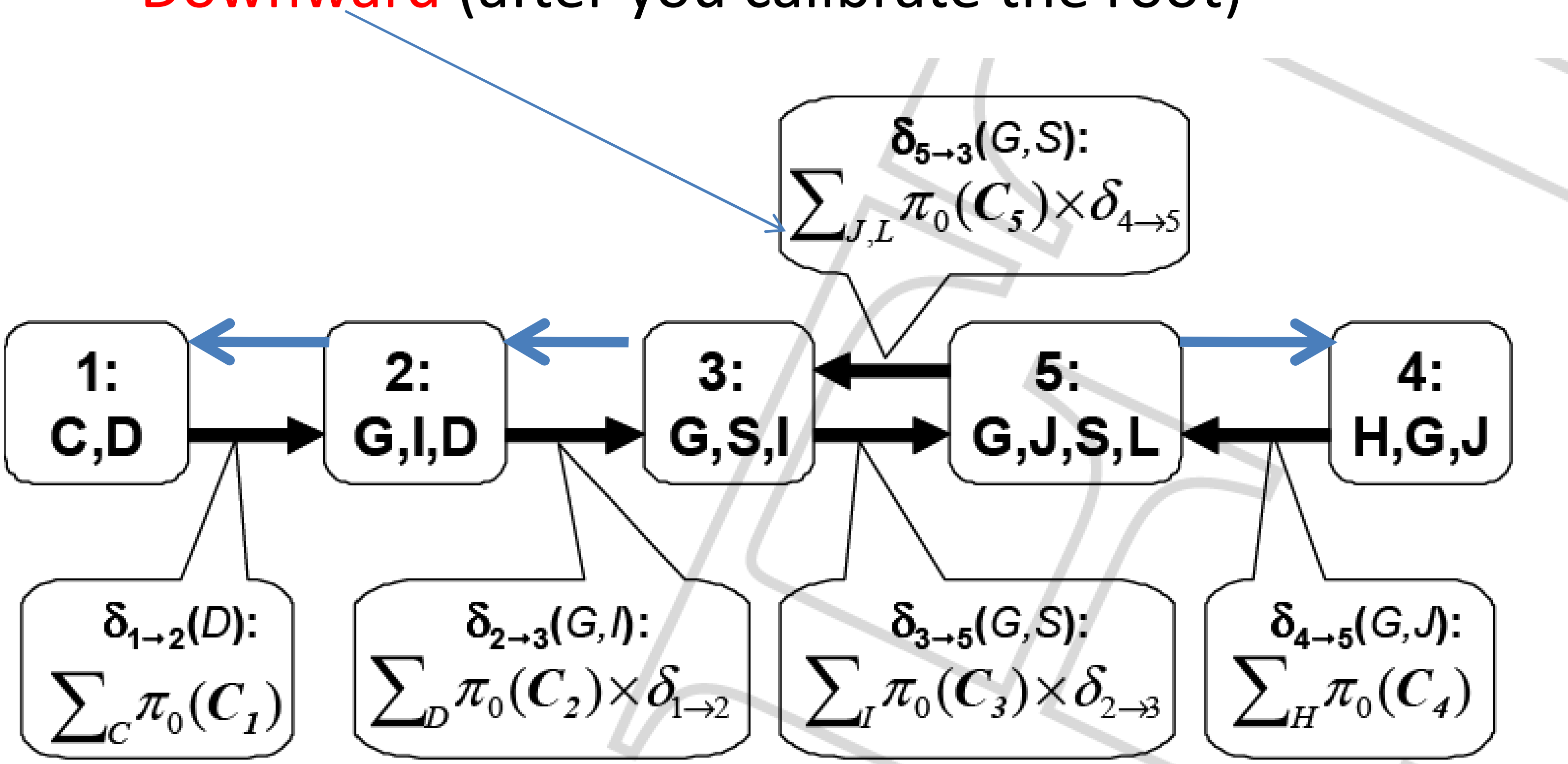
-Multiply all factors in left side

-Eliminate out exclusive variables (but do it in steps along the tree: C then D)

-The message depends ONLY on the direction of the edge!!!

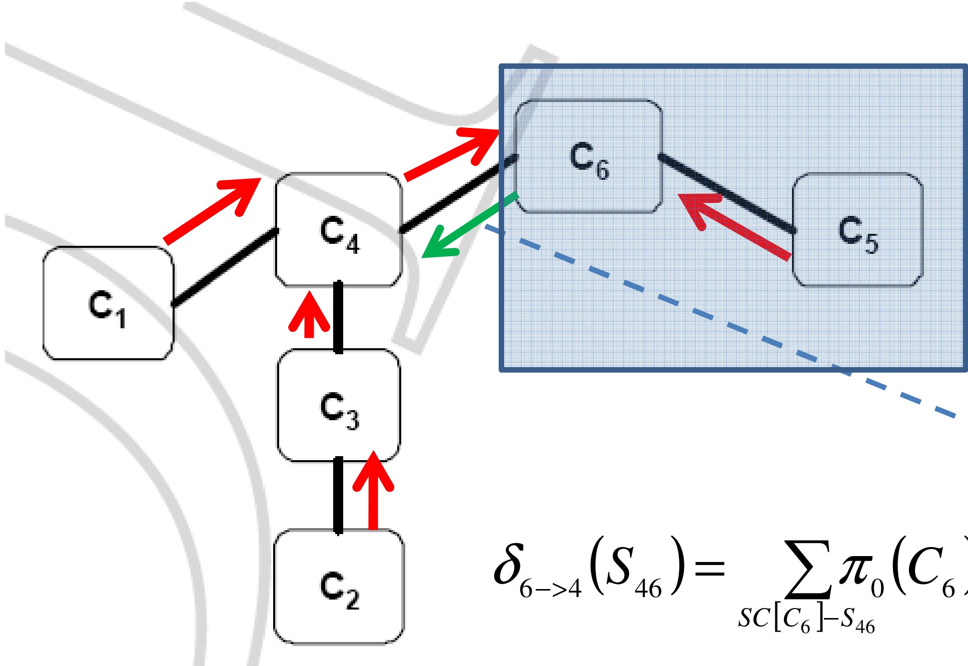
Clique Tree Calibration

- Two Step process:
  – Upward: as before
  – Downward (after you calibrate the root)

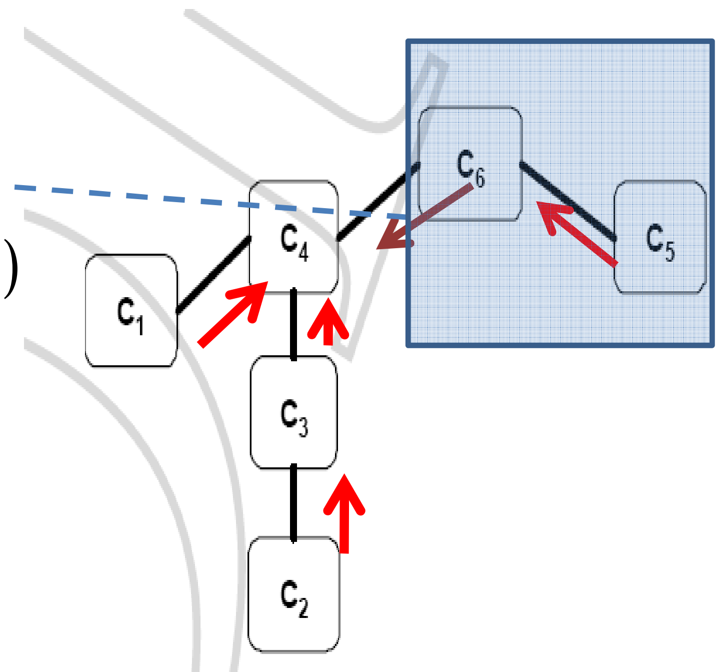


Intuitively Why it works?

Upward Phase: Root is calibrated
Downward : Lets take C4, what if it was a root.



=



$$\delta_{6->4}(S_{46}) = \sum_{SC[C_6]-S_{46}} \pi_0(C_6) \times \delta_{5->6}(S_{56})$$

Now C4 is calibrated and can
Act recursively as a new root!!!

The two tree only differ on
the edge from C4-C6, but else
The same

C4: just needs message from C6
That summarizes the status of the
Separator from the other side of
the tree

Clique Trees

- Can compute all clique marginals with double the cost of a single VE
- Need to store all intermediate messages
  - It is not magic
    - If you store intermediate factors from VE you get the same effect!!
- You lose internal structure and some independency
  - Do you care?
    - Time: no!
    - Space: YES
- You can still run VE to get marginal with variables not in the same clique and even all pair-wise marginals (Q5).
- Good for continuous inference
- Can not be tailored to evidences: only one elimination order

Queries Outside Clique: Q5

- T is assumed calibrated
  - Cliques agree on separators
  - **See section 9.3.4.2, Section 9.3.4.3**