Exact and Approximate Inference

Inference

Professor Ajoodha

Lecture 5

School of Computer Science and Applied Mathematics
The University of the Witwatersrand, Johannesburg

ExplainableAI Lab
— MODELLING. DECISION MAKING. CAUSALITY —

Problem Statement

We now discuss the problem of inference in graphical models.

We will begin by solving the conditional probability query:

Problem

$$P(\mathbf{Y} \mid \mathbf{E} = \mathbf{e}) = \frac{P(\mathbf{Y}, \mathbf{e})}{P(\mathbf{e})} = \frac{\sum_w P(\mathbf{y}, \mathbf{e}, \mathbf{w})}{\sum_{y,w} P(\mathbf{e})}$$

The effectiveness of inference depends on the network structure (conditional independence and factorisation of joint)

Solving this is easy! Just **summing out** (marginalise) all the nuisance variables and then divide.

Example with tabular CPDs

y^1	e^1	w^1	0.18
y^1	e^1	w^2	0.09
y^1	e^2	w^1	0.01
y^1	e^2	w^2	0.37
y^2	e^1	w^1	0.13
y^2	e^1	w^2	0.16
y^2	e^2	w^1	0.05
y^2	e^2	w^2	0.01

$P(y, e, w)$

$\rightarrow \sum_w P(y, e, w) \rightarrow$

y^1	e^1	0.27
y^1	e^2	0.37
y^2	e^1	0.29
y^2	e^2	0.06

$P(y, e)$

y^1	e^1	0.27
y^1	e^2	0.37
y^2	e^1	0.29
y^2	e^2	0.06

$P(y, e)$

$\rightarrow \sum_y P(y, e) \rightarrow$

e^1	0.57
e^2	0.43

$P(e)$

$$P(y^1 \mid e^1) = \frac{P(y^1, e^1)}{P(e^1)} = \frac{0.27}{0.57} = 0.47$$

Note that $P(y^1 \mid e^1)$ is **not the same thing** as $P(y^1, e^1)$

Problem Statement

This approach to exact inference is not feasible since it leads to the exponential "blow-up" of the joint distribution.

Unfortunately, in the worst case we cannot avoid this outcome

This makes exact inference in graphical models $\mathcal{NP}$-hard.

To make matters worst, approximate inference is also $\mathcal{NP}$-hard!

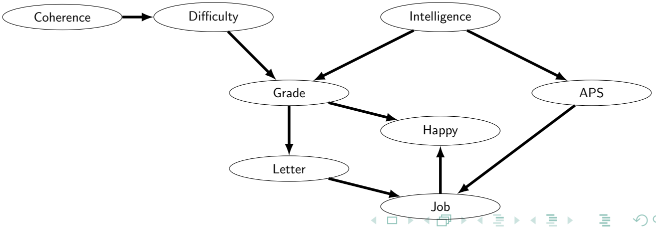
However, these results are for the **worst case** and inference with graphical models is **mostly manageable for practical applications**.

Variable Elimination: Intuition

Variable elimination is at the heart of exact inference.

Exploits two properties of the structure of a Bayesian network:

- Some sub-expressions of the joint only depend on a small number of variables.
- By caching results we do no have to recompute them.
- Some important properties:
  1. $P(A) = \sum_B P(A,B)$
  2. $\sum_C P(A) P(B,C) = P(A) \sum_C P(B,C)$



Variable Elimination Algorithm

Suppose you want to calculate $P(J)$ for the extended Student Network?
Representing the joint distribution using the chain rule for Bayesian networks:

$$\begin{aligned} P(C,D,I,G,A,L,J,H) = & P(C)P(D \mid C)P(I) \\ & P(G \mid I,D)P(A \mid I)P(L \mid G) \\ & P(J \mid L,A)P(H \mid G,J) \end{aligned}$$

As a set of factors each with a scope:

$$\begin{aligned} P(C,D,I,G,A,L,J,H) = & \phi_C(C)\phi_D(C,D)\phi_I(I) \\ & \phi_G(G,I,D)\phi_A(A,I)\phi_L(L,G) \\ & \phi_J(J,L,A)\phi_H(H,G,J) \end{aligned}$$

Variable Elimination Algorithm

Suppose you want to calculate $P(J)$ for the extended Student Network?
Representing the joint distribution using the chain rule for Bayesian networks:

$$\begin{aligned} P(C,D,I,G,A,L,J,H) =& P(C)P(D \mid C)P(I) \\ & P(G \mid I,D)P(A \mid I)P(L \mid G) \\ & P(J \mid L,A)P(H \mid G,J) \end{aligned}$$

As a set of factors each with a scope:

$$\begin{aligned} P(C,D,I,G,A,L,J,H) =& \phi_C(C)\phi_D(C,D)\phi_I(I) \\ & \phi_G(G,I,D)\phi_A(A,I)\phi_L(L,G) \\ & \phi_J(J,L,A)\phi_H(H,G,J) \end{aligned}$$

Exact and Approximate Inference
Professor Ajoodha
Problem Statement
Variable Elimination
Cluster Graphs
Message Passing
Sum-Product
Tree Calibration
Approximate Inference
MAP Inference

Elimination ordering: C D I H G A L

P(J) =

$$\sum_{C,D,I,H,G,A,L} \phi_C(C)\phi_D(C,D)\phi_I(I)\phi_G(G,I,D)\phi_A(A,I)\phi_L(L,G)\phi_J(J,L,A)\phi_H(H,G,J)$$

We can **push in** the summation according to the elimination ordering:

$$= \sum_{L}\sum_{A}\phi_J(J,L,A)\sum_{G}\phi_L(L,G)\sum_{H}\phi_H(H,G,J)\sum_{I}\phi_I(I)\phi_A(A,I)\sum_{D}\phi_G(G,I,D)\sum_{C}\phi_C(C)\phi_D(C,D)$$

Elimination ordering: C D I H G A L

P(J) =

$$\sum_{C,D,I,H,G,A,L} \phi_C(C)\phi_D(C,D)\phi_I(I)\phi_G(G,I,D)\phi_A(A,I)$$
$$\phi_L(L,G)\phi_J(J,L,A)\phi_H(H,G,J)$$

We can **push in** the summation according to the elimination ordering:

$$= \sum_{L}\sum_{A} \phi_J(J,L,A) \sum_{G} \phi_L(L,G) \sum_{H} \phi_H(H,G,J)$$
$$\sum_{I} \phi_I(I)\phi_A(A,I) \sum_{D} \phi_G(G,I,D) \sum_{C} \phi_C(C)\phi_D(C,D)$$

Elimination Ordering: C D I H G A L

Elimination ordering: C D I H G A L

P(J) =

$$\sum_{C,D,I,H,G,A,L} \phi_C(C)\phi_D(C,D)\phi_I(I)\phi_G(G,I,D)\phi_A(A,I)$$
$$\phi_L(L,G)\phi_J(J,L,A)\phi_H(H,G,J)$$

We can **push in** the summation according to the elimination ordering:

$$= \sum_{L} \sum_{A} \phi_J(J,L,A) \sum_{G} \phi_L(L,G) \sum_{H} \phi_H(H,G,J)$$
$$\sum_{I} \phi_I(I)\phi_A(A,I) \sum_{D} \phi_G(G,I,D) \sum_{C} \phi_C(C)\phi_D(C,D)$$

Now we can multiply and marginalise until we reach $P(J)$.

Lets introduce new notation for the marginalised factor (τ) and the product factor (ψ).

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G) \sum_H \phi_H(H, G, J) \sum_I \phi_I(I)\phi_A(A, I) \sum_D \phi_G(G, I, D) \sum_C \phi_C(C)\phi_D(C, D)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G) \sum_H \phi_H(H, G, J) \sum_I \phi_I(I)\phi_A(A, I) \sum_D \phi_G(G, I, D) \sum_C \psi_1(C, D)$$

Variable Elimination Algorithm (Calculation 1/7)

Now we can multiply and marginalise until we reach $P(J)$. Lets introduce new notation for the marginalised factor (τ) and the product factor (ψ).

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \phi_G(G,I,D) \sum_C \phi_C(C)\phi_D(C,D)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \phi_G(G,I,D) \sum_C \psi_1(C,D)$$

Variable Elimination Algorithm (Calculation 1/7)

Now we can multiply and marginalise until we reach $P(J)$. Lets introduce new notation for the marginalised factor (τ) and the product factor (ψ).

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \phi_G(G,I,D) \sum_C \phi_C(C)\phi_D(C,D)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \phi_G(G,I,D) \sum_C \psi_1(C,D)$$

Variable Elimination Algorithm (Calculation 1/7)

Now we can multiply and marginalise until we reach $P(J)$. Lets introduce new notation for the marginalised factor (τ) and the product factor (ψ).

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \phi_G(G,I,D) \sum_C \phi_C(C)\phi_D(C,D)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \phi_G(G,I,D) \sum_C \psi_1(C,D)$$

Variable Elimination Algorithm (Calculation 2/7)

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \phi_G(G,I,D)\,\tau_1(D)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \phi_G(G,I,D)\tau_1(D)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \psi_2(G,I,D)$$

Variable Elimination Algorithm (Calculation 2/7)

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \phi_G(G,I,D)\, \tau_1(D)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \phi_G(G,I,D)\tau_1(D)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \sum_D \psi_2(G,I,D)$$

Variable Elimination Algorithm (Calculation 2/7)

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G) \sum_H \phi_H(H, G, J) \sum_I \phi_I(I)\phi_A(A, I) \sum_D \phi_G(G, I, D)\, \tau_1(D)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G) \sum_H \phi_H(H, G, J) \sum_I \phi_I(I)\phi_A(A, I) \sum_D \phi_G(G, I, D)\tau_1(D)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G) \sum_H \phi_H(H, G, J) \sum_I \phi_I(I)\phi_A(A, I) \sum_D \psi_2(G, I, D)$$

Variable Elimination Algorithm (Calculation 3/7)

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I) \tau_2(G,I)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I)\tau_2(G,I)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \psi_3(A,I,G)$$

Variable Elimination Algorithm (Calculation 3/7)

Exact and Approximate Inference
Professor Ajoodha
Problem Statement
Variable Elimination
Cluster Graphs
Message Passing
Sum-Product
Tree Calibration
Approximate Inference
MAP Inference

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I)\,\tau_2(G,I)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I)\tau_2(G,I)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \psi_3(A,I,G)$$

Variable Elimination Algorithm (Calculation 3/7)

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I)\,\tau_2(G,I)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \phi_I(I)\phi_A(A,I)\tau_2(G,I)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G) \sum_H \phi_H(H,G,J) \sum_I \psi_3(A,I,G)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G) \sum_H \phi_H(H, G, J)$$

$\tau_3(A, G)$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G)\, \tau_3(A, G) \sum_H \phi_H(H, G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G) \tau_3(A, G) \sum_H \psi_4(H, G, J)$$

Variable Elimination Algorithm (Calculation 4/7)

$$= \sum_{L} \sum_{A} \phi_J(J, L, A) \sum_{G} \phi_L(L, G) \sum_{H} \phi_H(H, G, J)$$

$\tau_3(A, G)$

$$= \sum_{L} \sum_{A} \phi_J(J, L, A) \sum_{G} \phi_L(L, G) \tau_3(A, G) \sum_{H} \phi_H(H, G, J)$$

$$= \sum_{L} \sum_{A} \phi_J(J, L, A) \sum_{G} \phi_L(L, G) \tau_3(A, G) \sum_{H} \psi_4(H, G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G) \sum_H \phi_H(H, G, J)$$

$\tau_3(A, G)$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G) \, \tau_3(A, G) \sum_H \phi_H(H, G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G) \tau_3(A, G) \sum_H \psi_4(H, G, J)$$

Variable Elimination Algorithm (Calculation 5/7)

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G)\tau_3(A,G)\, \tau_4(G,J)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G)\tau_3(A,G)\tau_4(G,J)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \psi_5(L,A,G,J)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \psi_5(L,A,G,J)$$

$$= \sum_L \sum_A \phi_J(J,L,A)\, \tau_5(L,A,J)$$

Variable Elimination Algorithm (Calculation 5/7)

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G)\tau_3(A, G)\,\tau_4(G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G)\tau_3(A, G)\tau_4(G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \psi_5(L, A, G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \psi_5(L, A, G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A)\,\tau_5(L, A, J)$$

Variable Elimination Algorithm (Calculation 5/7)

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G)\tau_3(A, G)\, \tau_4(G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G)\tau_3(A, G)\tau_4(G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \psi_5(L, A, G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \psi_5(L, A, G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A)\, \tau_5(L, A, J)$$

Variable Elimination Algorithm (Calculation 5/7)

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G)\tau_3(A,G)\,\tau_4(G,J)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \phi_L(L,G)\tau_3(A,G)\tau_4(G,J)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \psi_5(L,A,G,J)$$

$$= \sum_L \sum_A \phi_J(J,L,A) \sum_G \psi_5(L,A,G,J)$$

$$= \sum_L \sum_A \phi_J(J,L,A)\,\tau_5(L,A,J)$$

Variable Elimination Algorithm (Calculation 5/7)

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G)\tau_3(A, G)\, \tau_4(G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \phi_L(L, G)\tau_3(A, G)\tau_4(G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \psi_5(L, A, G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A) \sum_G \psi_5(L, A, G, J)$$

$$= \sum_L \sum_A \phi_J(J, L, A)\, \tau_5(L, A, J)$$

Variable Elimination Algorithm (Calculation 6/7)

$$= \sum_L \sum_A \phi_J(J, L, A)\tau_5(L, A, J)$$

$$= \sum_L \sum_A \psi_6(J, L, A)$$

$$= \sum_L \sum_A \psi_6(J, L, A)$$

$$= \sum_L \tau_6(J, L)$$

$$= \sum_L \psi_6(J, L)$$

$$= \sum_{L} \sum_{A} \phi_J(J, L, A)\tau_5(L, A, J)$$

$$= \sum_{L} \sum_{A} \psi_6(J, L, A)$$

$$= \sum_{L} \sum_{A} \psi_6(J, L, A)$$

$$= \sum_{L} \tau_6(J, L)$$

$$= \sum_{L} \psi_6(J, L)$$

Variable Elimination Algorithm (Calculation 6/7)

$$= \sum_L \sum_A \phi_J(J, L, A)\tau_5(L, A, J)$$

$$= \sum_L \sum_A \psi_6(J, L, A)$$

$$= \sum_L \sum_A \psi_6(J, L, A)$$

$$= \sum_L \tau_6(J, L)$$

$$= \sum_L \psi_6(J, L)$$

Variable Elimination Algorithm (Calculation 6/7)

$$= \sum_L \sum_A \phi_J(J, L, A)\tau_5(L, A, J)$$

$$= \sum_L \sum_A \psi_6(J, L, A)$$

$$= \sum_L \sum_A \psi_6(J, L, A)$$

$$= \sum_L \tau_6(J, L)$$

$$= \sum_L \psi_6(J, L)$$

Variable Elimination Algorithm (Calculation 6/7)

$$= \sum_L \sum_A \phi_J(J, L, A)\tau_5(L, A, J)$$

$$= \sum_L \sum_A \psi_6(J, L, A)$$

$$= \sum_L \sum_A \psi_6(J, L, A)$$

$$= \sum_L \tau_6(J, L)$$

$$= \sum_L \psi_6(J, L)$$

Variable Elimination Algorithm (Calculation 7/7)

$$= \sum_{L} \tau_7(J, L)$$

$$= \tau_7(J)$$

Variable Elimination Algorithm (Calculation 7/7)

$$= \sum_{L} \tau_7(J, L)$$

$$= \tau_7(J)$$

Summary

We can summarise the run of VE neatly in a table:

Step	Var Eli	Factors used	Variable Involved	New Factor
1	C	$\phi_C(C), \phi_D(D,C)$	C,D	$\tau_1(D)$
2	D	$\phi_G(G,I,D), \tau_1(D)$	G,I,D	$\tau_2(G,I)$
3	I	$\phi_I(I), \phi_A(A,I), \tau_2(G,I)$	G,A,I	$\tau_3(G,A)$
4	H	$\phi_H(H,G,J)$	H,G,J	$\tau_4(G,J)$
5	G	$\tau_4(G,J), \tau_3(G,A), \phi_L(L,G)$	G,J,L,A	$\tau_5(J,L,A)$
6	A	$\tau_5(J,L,A), \phi_J(J,L,A)$	J,L,A	$\tau_6(J,L)$
7	L	$\tau_6(J,L)$	J,L	$\tau_7(J)$

Here is another run of VE with a different elimination ordering:

Step	Var Eli	Factors used	Variable Involved	New Factor
1	G	$\phi_G(G,I,D), \phi_L(L,G), \phi_H(H,G,J)$	G,I,D,L,J,H	$\tau_1(I,D,L,J,H)$
2	I	$\phi_I(I), \phi_A(A,I), \tau_1(I,D,L,A,J,H)$	A,I,D,L,J,H	$\tau_2(D,L,A,J,H)$
3	A	$\phi_J(J,L,A), \tau_2(D,L,A,J,H)$	D,L,A,J,H	$\tau_3(D,L,J,H)$
4	L	$\tau_3(D,L,J,H)$	D,L,J,H	$\tau_4(D,J,H)$
5	H	$\tau_4(D,J,H)$	D,J,H	$\tau_5(D,J)$
6	C	$\phi_C(C), \phi_D(D,C)$	D,J,C	$\tau_6(D)$
7	D	$\tau_5(D,J), \tau_6(D)$	D,J	$\tau_7(J)$

Summary

We can summarise the run of VE neatly in a table:

Step	Var Eli	Factors used	Variable Involved	New Factor
1	C	$\phi_C(C), \phi_D(D,C)$	C,D	$\tau_1(D)$
2	D	$\phi_G(G,I,D), \tau_1(D)$	G,I,D	$\tau_2(G,I)$
3	I	$\phi_I(I), \phi_A(A,I), \tau_2(G,I)$	G,A,I	$\tau_3(G,A)$
4	H	$\phi_H(H,G,J)$	H,G,J	$\tau_4(G,J)$
5	G	$\tau_4(G,J), \tau_3(G,A), \phi_L(L,G)$	G,J,L,A	$\tau_5(J,L,A)$
6	A	$\tau_5(J,L,A), \phi_J(J,L,A)$	J,L,A	$\tau_6(J,L)$
7	L	$\tau_6(J,L)$	J,L	$\tau_7(J)$

Here is another run of VE with a different elimination ordering:

Step	Var Eli	Factors used	Variable Involved	New Factor
1	G	$\phi_G(G,I,D), \phi_L(L,G), \phi_H(H,G,J)$	G,I,D,L,J,H	$\tau_1(I,D,L,J,H)$
2	I	$\phi_I(I), \phi_A(A,I), \tau_1(I,D,L,A,J,H)$	A,I,D,L,J,H	$\tau_2(D,L,A,J,H)$
3	A	$\phi_J(J,L,A), \tau_2(D,L,A,J,H)$	D,L,A,J,H	$\tau_3(D,L,J,H)$
4	L	$\tau_3(D,L,J,H)$	D,L,J,H	$\tau_4(D,J,H)$
5	H	$\tau_4(D,J,H)$	D,J,H	$\tau_5(D,J)$
6	C	$\phi_C(C), \phi_D(D,C)$	D,J,C	$\tau_6(D)$
7	D	$\tau_5(D,J), \tau_6(D)$	D,J	$\tau_7(J)$

Dealing with Evidence

What if we would like to complete a conditional probability query with evidence? i.e. $P(J \mid i^1, h^0)$

1. Simply calculate $P(J, i^1, h^0)$ using Variable Elimination.
2. Use $P(J, i^1, h^0)$ to calculate $P(i^1, h^0)$.
3. Finally compute:

$$P(J \mid i^1, h^0) = \frac{P(J, i^1, h^0)}{P(i^1, h^0)}$$

STOP! We need some definitions before proceeding

Induced Subgraph

Let $\mathcal{K} = (\mathcal{X}, \mathcal{E})$, and $\mathbf{X} \in \mathcal{X}$, then an induced subgraph, denoted $\mathcal{K}[\mathbf{X}]$ is a graph $(\mathbf{X}, \mathcal{E}')$ where $\mathcal{E}'$ are all the edges $X \leftrightharpoons Y \in \mathcal{E}'$ such that $X, Y \in \mathbf{X}$.

A Complete Graph (Clique)

A subgraph over $\mathbf{X}$ is complete if every two nodes in $\mathbf{X}$ are connected by some edge. The set $\mathbf{X}$ is called a clique. A clique $\mathbf{X}$ is maximal if for any superset of nodes $\mathbf{Y} \supset \mathbf{X}$, $\mathbf{Y}$ is not a clique.

Upward Closure

A subset of nodes $\mathbf{X} \in \mathcal{X}$ is upwardly closed in $\mathcal{K}$ if, for any $\mathbf{X} \in \mathcal{X}$, we have that the Boundary$_X \subset \mathbf{X}$. We define upward closure of $\mathbf{X}$ to be the minimally upward closed subset $\mathbf{Y}$ that contains $\mathbf{X}$.

Induced Graph

The elimination ordering is an important choice.

Recall that the following run of VE was achieved with the following ordering: $<= \{C, D, I, H, G, S, L\}$

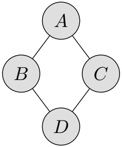
Step	Var Eli	Factors used	Variable Involved	New Factor
1	C	$\phi_C(C), \phi_D(D, C)$	C,D	$\tau_1(D)$
2	D	$\phi_G(G, I, D), \tau_1(D)$	G,I,D	$\tau_2(G, I)$
3	I	$\phi_I(I), \phi_A(A, I), \tau_2(G, I)$	G,A,I	$\tau_3(G, A)$
4	H	$\phi_H(H, G, J)$	H,G,J	$\tau_4(G, J)$
5	G	$\tau_4(G, J), \tau_3(G, A), \phi_L(L, G)$	G,J,L,A	$\tau_5(J, L, A)$
6	A	$\tau_5(J, L, A), \phi_J(J, L, A)$	J,L,A	$\tau_6(J, L)$
7	L	$\tau_6(J, L)$	J,L	$\tau_7(J)$

Induced Graph: We can define an induced graph, denoted $\mathcal{I}_{G,<}$, as an undirected graph over $\mathcal{X}$, where X_i and X_j, are connected by an edge if they both appear in any ψ in the VE algorithm.

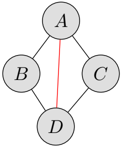
Notice that the induced graph has **moralised edges** that were not in the original.

Chodal Graphs

1. Let $X_1 - X_2 - \ldots - X_k - X_1$ be a loop in a graph. A chord in a loop is an edge connecting X_i and X_j for two nonconsecutive nodes X_i, X_j.
2. An undirected graph $\mathcal{H}$ is said to be chordal if a loop $X_1 - X_2 - \ldots - X_k - X_1$ for $k > 4$ has a chord.
3. Example:



NOT chordal

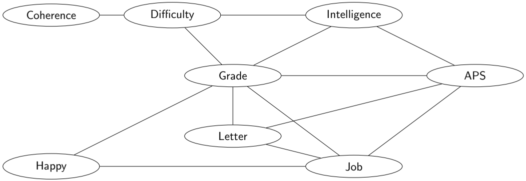


chordal

4. A directed graph $\mathcal{K}$ is said to be chordal if its underlying undirected graph is chordal.

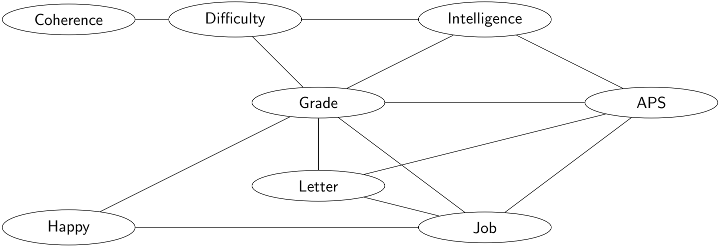
Induced Graph

Step	Var Eli	Factors used	Variable Involved	New Factor
1	C	$\phi_C(C), \phi_D(D,C)$	C,D	$\tau_1(D)$
2	D	$\phi_G(G,I,D), \tau_1(D)$	G,I,D	$\tau_2(G,I)$
3	I	$\phi_I(I), \phi_A(A,I), \tau_2(G,I)$	G,A,I	$\tau_3(G,A)$
4	H	$\phi_H(H,G,J)$	H,G,J	$\tau_4(G,J)$
5	G	$\tau_4(G,J), \tau_3(G,A), \phi_L(L,G)$	G,J,L,A	$\tau_5(J,L,A)$
6	A	$\tau_5(J,L,A), \phi_J(J,L,A)$	J,L,A	$\tau_6(J,L)$
7	L	$\tau_6(J,L)$	J,L	$\tau_7(J)$



The width of an induced graph is the number of nodes in the largest clique minus 1.
Finding an elimination ordering that provides a minimal induced width is $\mathcal{NP}$−hard.

Clique Tree for the Induced Graph



Finally we can construct a clique tree from induced graph by looking at the cliques:

Time and Space complexity

1. **The induced graph and size of the maximal cliques heavily depend on the elimination ordering.**
2. The **time and space complexity** of Variable Elimination $2^{O(K)}$, where K is the width of the induced graph.
3. In other words, good orderings will keep factor sizes small and bad ones will yield them unmanageable for time and computational space.

Variable Elimination using Message Passing

So far we have implemented the variable elimination algorithm as follows:

1. Create a factor ψ_i by multiplying existing factors.
2. A variable is eliminated in ψ_i to generate a new factor τ_i
3. which is then multiplied into another factor ...

This process can also be done in a different way using message passing:

1. We consider the factor ψ_i to be a computational data structure.
2. ψ_i takes `messages` τ_j which is generated by other factors ψ_j and generates a new message τ_i that is used by another function ψ_l.

Cluster Graph

Before we can discuss message passing in more detail, we first need to define **cluster graphs**.

A cluster graph is an indirected graph that we use to visualise message passing.

Each node is a subset of variables (cluster): $\mathbf{C}_i \subseteq \mathcal{X}$.

Each edge subset of variables (cluster): edge between $\mathbf{C}_i$ and $\mathbf{C}_j$ is associated with sepset: $S_{i,j} = \mathbf{C}_i \cap \mathbf{C}_j$.

A sepset is the intersection of the scopes of both nodes.

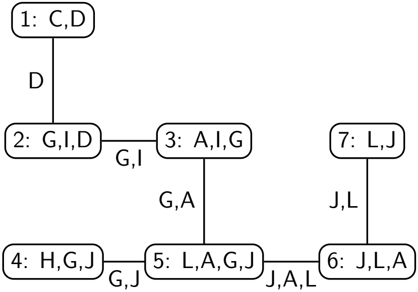
The intuition is that:

1. The nodes "know" about a subset of variables;
2. and can "talk" about the sepset of variables through the edges.

Cluster Graph

We can obtain a cluster graph from each ψ_i from a VE run.

1	$\psi_1(C, D)$
2	$\psi_2(G, I, D)$
3	$\psi_3(A, I, G)$
4	$\psi_4(H, G, J)$
5	$\psi_5(L, A, G, J)$
6	$\psi_6(J, L, A)$
7	$\psi_7(L, J)$



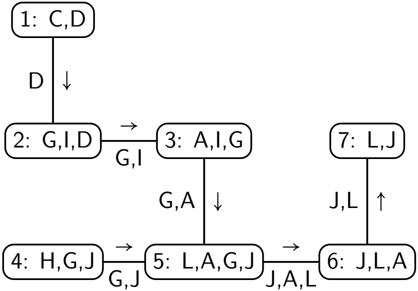
What is the difference between this cluster graph and the clique tree we saw earlier? (non-maximal cliques 6 and 7 are missing)

Message flow from Variable Elimination

We can also demonstrate the flow of message when we executed variable elimination (a few slides ago).

Indicated by arrows:

1	$\psi_1(C, D)$
2	$\psi_2(G, I, D)$
3	$\psi_3(A, I, G)$
4	$\psi_4(H, G, J)$
5	$\psi_5(L, A, G, J)$
6	$\psi_6(J, L, A)$
7	$\psi_7(L, J)$



Cluster 7 is called the root (messages are sent up to the root)

Cluster Graph Properties

There are two properties that a Cluster Graph needs to satisfy:

1. **Family Preservation:**
   - Each cluster needs to be assigned a factor.
   - Therefore, for each factor there should be an appropriate cluster to be assigned to.
2. **Running Intersection Property:**
   - For any factor X, the set of clusters and sepsets containing X form a tree.

Family Preservation

Each factor should have an appropriate cluster to be assigned to.

$$P(C, D, I, G, A, L, J, H) = \phi_C(C)\phi_D(C, D)\phi_I(I)$$
$$\phi_G(G, I, D)\phi_A(A, I)\phi_L(L, G)\ \phi_J(J, L, A)\phi_H(H, G, J)$$

Family Preservation

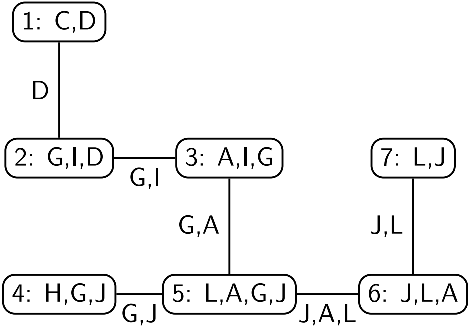
Each factor should have an appropriate cluster to be assigned to.

$$P(C,D,I,G,A,L,J,H) = \phi_C(C)\phi_D(C,D)\phi_I(I)$$
$$\phi_G(G,I,D)\phi_A(A,I)\phi_L(L,G)\ \phi_J(J,L,A)\phi_H(H,G,J)$$

Family Preservation

Each factor should have an appropriate cluster to be assigned to.

$$P(C, D, I, G, A, L, J, H) = \phi_C(C)\phi_D(C, D)\phi_I(I)$$
$$\phi_G(G, I, D)\phi_A(A, I)\phi_L(L, G)\ \phi_J(J, L, A)\phi_H(H, G, J)$$

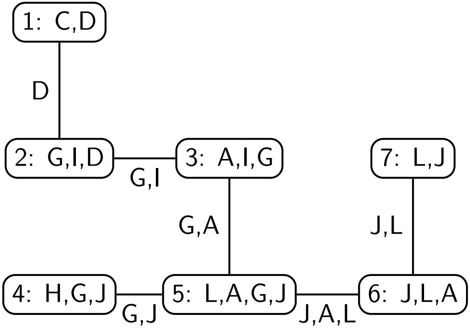


Family Preservation

Each factor should have an appropriate cluster to be assigned to.

$P(C, D, I, G, A, L, J, H) = \phi_C(C)\phi_D(C, D)\phi_I(I)$
$\phi_G(G, I, D)\phi_A(A, I)\phi_L(L, G)\ \phi_J(J, L, A)\phi_H(H, G, J)$

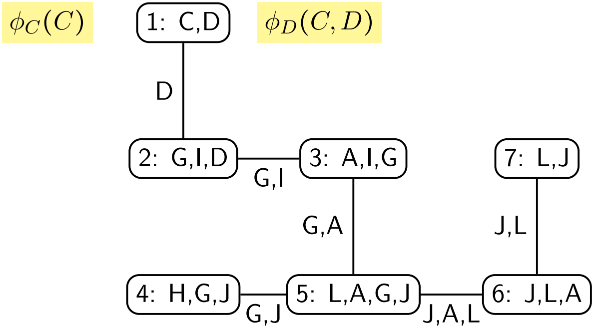
$\phi_C(C)$



Family Preservation

Each factor should have an appropriate cluster to be assigned to.

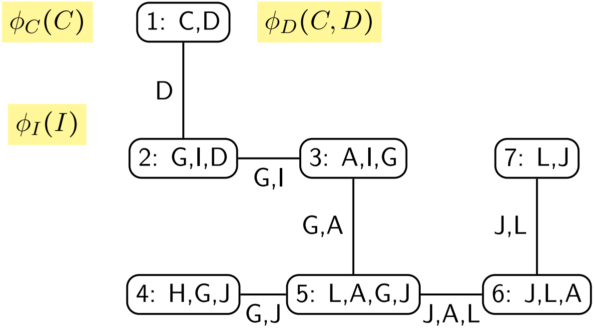
$$P(C,D,I,G,A,L,J,H) = \phi_C(C)\phi_D(C,D)\phi_I(I)$$
$$\phi_G(G,I,D)\phi_A(A,I)\phi_L(L,G)\ \phi_J(J,L,A)\phi_H(H,G,J)$$



Family Preservation

Each factor should have an appropriate cluster to be assigned to.

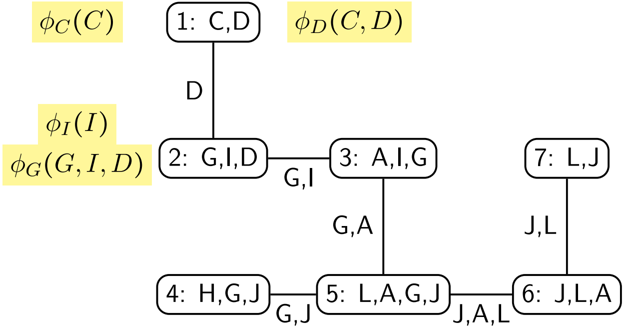
$P(C, D, I, G, A, L, J, H) = \phi_C(C)\phi_D(C, D)\phi_I(I)$
$\phi_G(G, I, D)\phi_A(A, I)\phi_L(L, G)\ \phi_J(J, L, A)\phi_H(H, G, J)$



Family Preservation

Each factor should have an appropriate cluster to be assigned to.

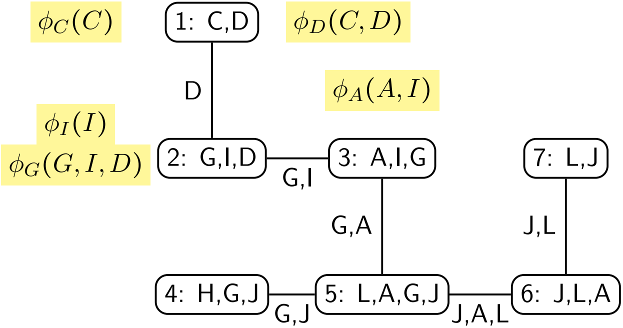
$$P(C,D,I,G,A,L,J,H) = \phi_C(C)\phi_D(C,D)\phi_I(I)$$
$$\phi_G(G,I,D)\phi_A(A,I)\phi_L(L,G)\ \phi_J(J,L,A)\phi_H(H,G,J)$$



Exact and Approximate Inference
Professor Ajoodha
Problem Statement
Variable Elimination
Cluster Graphs
Message Passing
Sum-Product
Tree Calibration
Approximate Inference
MAP Inference

Each factor should have an appropriate cluster to be assigned to.

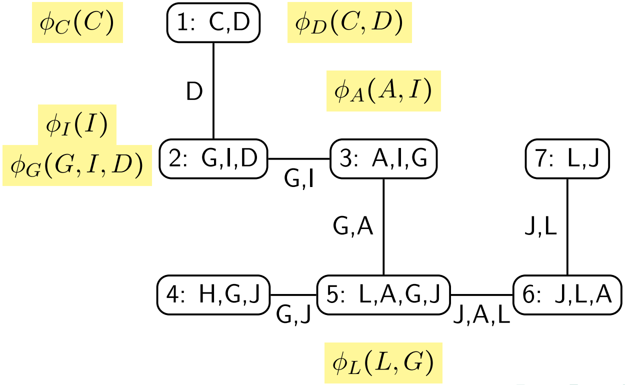
$$P(C,D,I,G,A,L,J,H) = \phi_C(C)\phi_D(C,D)\phi_I(I)$$
$$\phi_G(G,I,D)\phi_A(A,I)\phi_L(L,G)\ \phi_J(J,L,A)\phi_H(H,G,J)$$



Family Preservation

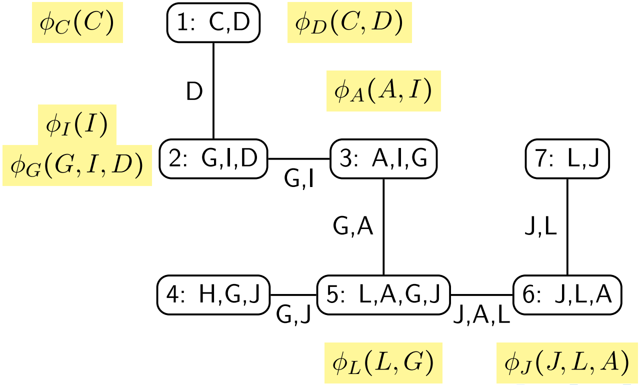
Each factor should have an appropriate cluster to be assigned to.

$$P(C,D,I,G,A,L,J,H) = \phi_C(C)\phi_D(C,D)\phi_I(I)$$
$$\phi_G(G,I,D)\phi_A(A,I)\phi_L(L,G)\ \phi_J(J,L,A)\phi_H(H,G,J)$$



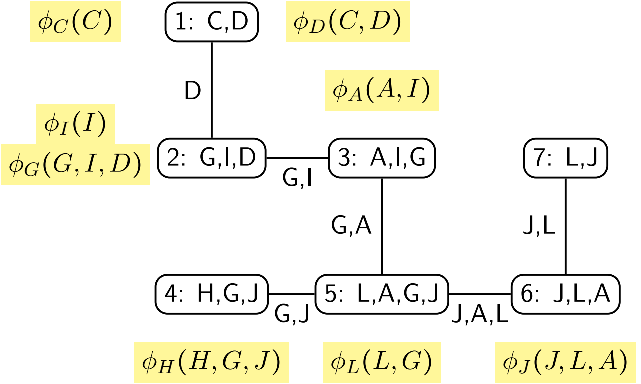
Each factor should have an appropriate cluster to be assigned to.

$$P(C,D,I,G,A,L,J,H) = \phi_C(C)\phi_D(C,D)\phi_I(I)$$
$$\phi_G(G,I,D)\phi_A(A,I)\phi_L(L,G)\ \phi_J(J,L,A)\phi_H(H,G,J)$$



Each factor should have an appropriate cluster to be assigned to.

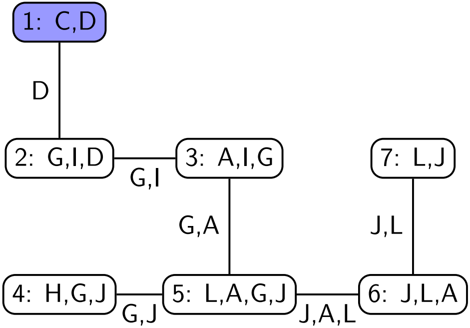
$P(C, D, I, G, A, L, J, H) = \phi_C(C)\phi_D(C, D)\phi_I(I)$
$\phi_G(G, I, D)\phi_A(A, I)\phi_L(L, G)\ \phi_J(J, L, A)\phi_H(H, G, J)$



Running Intersection Property

For any factor X, the set of clusters and sepsets containing X form a tree.

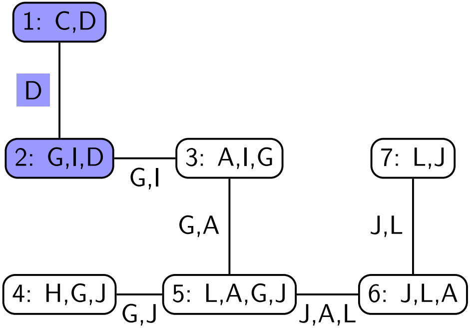
For Factor C,D,I,G,A,L,J,H:



Running Intersection Property

For any factor X, the set of clusters and sepsets containing X form a tree.

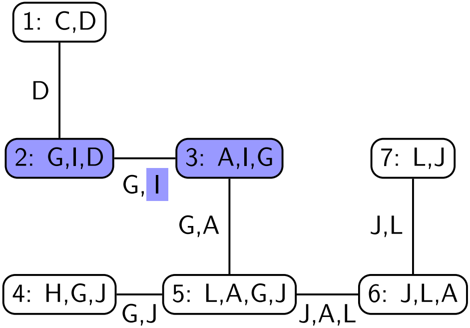
For Factor C, D ,I,G,A,L,J,H:



Running Intersection Property

For any factor X, the set of clusters and sepsets containing X form a tree.

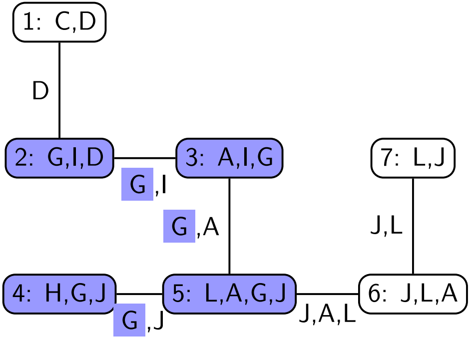
For Factor C,D,I,G,A,L,J,H:



Running Intersection Property

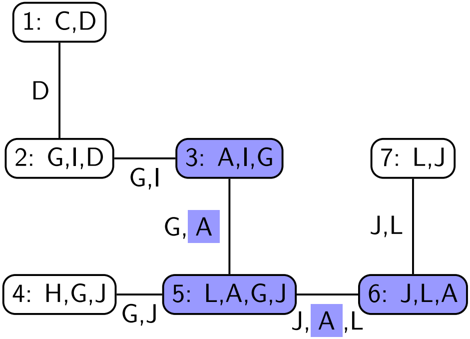
For any factor X, the set of clusters and sepsets containing X form a tree.

For Factor C,D,I,G,A,L,J,H:



For any factor X, the set of clusters and sepsets containing X form a tree.

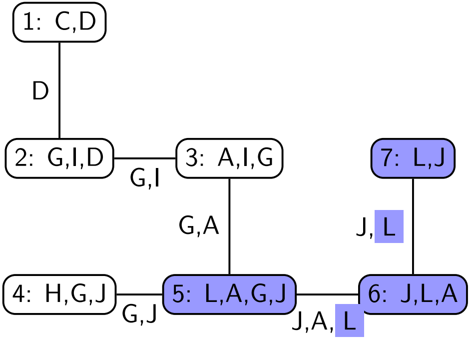
For Factor C,D,I,G, A ,L,J,H:



Running Intersection Property

For any factor X, the set of clusters and sepsets containing X form a tree.

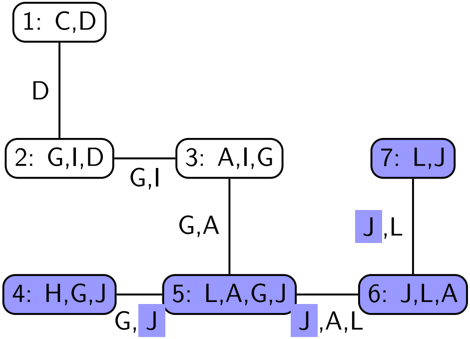
For Factor C,D,I,G,A, L ,J,H:



Running Intersection Property

For any factor X, the set of clusters and sepsets containing X form a tree.

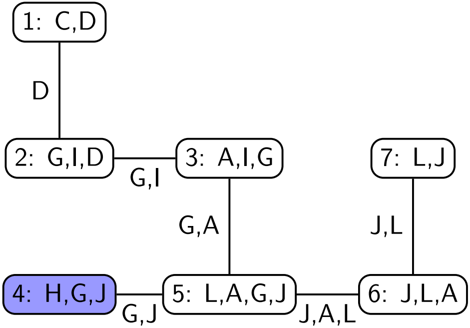
For Factor C,D,I,G,A,L, J ,H:



Running Intersection Property

For any factor X, the set of clusters and sepsets containing X form a tree.

For Factor C,D,I,G,A,L,J, H :



Message Passing: Sum Product

Previously we showed that we can obtain a cluster tree using a run of VE.

Now lets to the reverse, perform VE on a cluster tree.

Quickly convince yourself that this clique tree (that we saw earlier) satisfies the following properties for the Extended Student network:

1. Family preservation
2. Running intersection

Message Passing: Sum Product

Previously we showed that we can obtain a cluster tree using a run of VE.

Now lets to the reverse, perform VE on a cluster tree.

Quickly convince yourself that this clique tree (that we saw earlier) satisfies the following properties for the Extended Student network:

1. Family preservation
2. Running intersection

$\boxed{C,D} \overset{D}{\text{———}} \boxed{G,I,D} \overset{G,I}{\text{———}} \boxed{G,I,A} \overset{G,A}{\text{———}} \boxed{G,J,A,L} \overset{G,J}{\text{———}} \boxed{G,H,J}$

C,D	G,I,D	G,I,A	G,J,A,L	G,H,J
$P(D \mid C)$	$P(G \mid I, D)$	$P(I)$	$P(L \mid G)$	$P(H \mid G, J)$
$P(C)$		$P(A \mid I)$	$P(J \mid L, A)$	

Message Passing: Sum Product

You can use a clique tree to guide your variable elimination process.

The factors ψ_i is computed in the cliques and messages are sent alone the edges.

Each clique:

1. takes the incoming messages (factors)
2. multiplies them
3. sums out one or more variables
4. sends an outgoing message to another clique

If clique **C'** requires a message from **C**, then it must wait for it.

Clique trees can be used for many executions of VE.

It also allows for efficient execution through caching calculations.

Variable Elimination with a Clique Tree

Lets use this clique tree as a data structure for variable elimination to calculate $P(J)$

Step 1: We must first generate **initial potentials**, ψ_i, by multiplying out the factors.

1: C,D —D— 2: G,I,D —G,I— 3: G,I,A —G,A— 5: G,J,A,L —G,J— 4: G,H,J

Variable Elimination with a Clique Tree

Lets use this clique tree as a data structure for variable elimination to calculate $P(J)$

Step 1: We must first generate **initial potentials**, ψ_i, by multiplying out the factors.

1: C,D —D— 2: G,I,D —G,I— 3: G,I,A —G,A— 5: G,J,A,L —G,J— 4: G,H,J

1: C,D	2: G,I,D	3: G,I,A	5: G,J,A,L	4: G,H,J
$\phi_D(D,C)$	$\phi_G(G,I,D)$	$\phi_I(I)$	$\phi_L(L,G)$	$\phi_H(H,G,J)$
$\phi_C(C)$		$\phi_A(A,I)$	$\phi_J(J,L,A)$	

Variable Elimination with a Clique Tree

Lets use this clique tree as a data structure for variable elimination to calculate $P(J)$

Step 1: We must first generate **initial potentials**, ψ_i, by multiplying out the factors.

1: C,D	2: G,I,D	3: G,I,A	5: G,J,A,L	4: G,H,J
$\phi_D(D,C)$	$\phi_G(G,I,D)$	$\phi_I(I)$	$\phi_L(L,G)$	$\phi_H(H,G,J)$
$\phi_C(C)$		$\phi_A(A,I)$	$\phi_J(J,L,A)$	
$\psi_1(D,C)$	$\psi_2(G,I,D)$	$\psi_3(A,I)$	$\psi_5(J,L,G,A)$	$\psi_4(H,G,J)$

Variable Elimination with a Clique Tree

Step 2: Recall that we are trying to calculate $P(J)$.

So pick any cluster that contains J as our root (so J is not eliminated). Lets pick cluster 5 as our root:

Variable Elimination with a Clique Tree

Step 3: We are now ready to propagate through the clique tree.

We start with any leaf and work our way to the root.

Variable Elimination with a Clique Tree

Step 3: We are now ready to propagate through the clique tree.

We start with any leaf and work our way to the root.

$\psi_1(D, C)$ $\psi_2(G, I, D)$ $\psi_3(A, I)$ $\psi_5(J, L, G, A)$ $\psi_4(H, G, J)$

Variable Elimination with a Clique Tree

Step 3: We are now ready to propagate through the clique tree.

We start with any leaf and work our way to the root.

$\psi_1(D, C)$ $\psi_2(G, I, D)$ $\psi_3(A, I)$ $\psi_5(J, L, G, A)$ $\psi_4(H, G, J)$

$$\begin{tikzcd} \boxed{1:\ C,D} \arrow[r, "D"'] & \boxed{2:\ G,I,D} \arrow[r, "G,I"'] & \boxed{3:\ G,I,A} \arrow[r, "G,A"'] & \boxed{5:\ G,J,A,L} & \boxed{4:\ G,H,J} \arrow[l, "G,J"] \end{tikzcd}$$

$$\boxed{\begin{array}{c} \delta_{1\to 2}(D): \\ \sum_C \psi_1(\mathbf{C}_1) \end{array}}$$

Variable Elimination with a Clique Tree

Step 3: We are now ready to propagate through the clique tree.

We start with any leaf and work our way to the root.

$\psi_1(D, C)$ $\psi_2(G, I, D)$ $\psi_3(A, I)$ $\psi_5(J, L, G, A)$ $\psi_4(H, G, J)$

$$\boxed{1:\ C,D} \xrightarrow[D]{} \boxed{2:\ G,I,D} \xrightarrow[G,I]{} \boxed{3:\ G,I,A} \xrightarrow[G,A]{} \boxed{5:\ G,J,A,L} \xleftarrow[G,J]{} \boxed{4:\ G,H,J}$$

$$\boxed{\begin{array}{c}\delta_{1\to2}(D): \\ \sum_C \psi_1(\mathbf{C}_1)\end{array}} \quad \boxed{\begin{array}{c}\delta_{2\to3}(G, I): \\ \sum_D(\psi_2(\mathbf{C}_2) \\ \times \\ \delta_{1\to2})\end{array}}$$

Exact and Approximate Inference
Professor Ajoodha
Problem Statement
Variable Elimination
Cluster Graphs
Message Passing
Sum-Product
Tree Calibration
Approximate Inference
MAP Inference

Variable Elimination with a Clique Tree

Step 3: We are now ready to propagate through the clique tree.

We start with any leaf and work our way to the root.

$\psi_1(D,C)$ $\psi_2(G,I,D)$ $\psi_3(A,I)$ $\psi_5(J,L,G,A)$ $\psi_4(H,G,J)$

$$\text{1: C,D} \xrightarrow{\text{D}} \text{2: G,I,D} \xrightarrow{\text{G,I}} \text{3: G,I,A} \xrightarrow{\text{G,A}} \boxed{\text{5: G,J,A,L}} \xleftarrow{\text{G,J}} \text{4: G,H,J}$$

$\delta_{1\to 2}(D)$: $\sum_C \psi_1(\mathbf{C}_1)$	$\delta_{2\to 3}(G,I)$: $\sum_D(\psi_2(\mathbf{C}_2) \times \delta_{1\to 2})$	$\delta_{3\to 5}(G,A)$: $\sum_I(\psi_3(\mathbf{C}_3) \times \delta_{2\to 3})$

Variable Elimination with a Clique Tree

Step 3: We are now ready to propagate through the clique tree.

We start with any leaf and work our way to the root.

$\psi_1(D,C)$ $\psi_2(G,I,D)$ $\psi_3(A,I)$ $\psi_5(J,L,G,A)$ $\psi_4(H,G,J)$

1: C,D $\xrightarrow{D}$ 2: G,I,D $\xrightarrow{G,I}$ 3: G,I,A $\xrightarrow{G,A}$ **5: G,J,A,L** $\xleftarrow{G,J}$ 4: G,H,J

$\delta_{1\to 2}(D)$: $\sum_C \psi_1(\mathbf{C}_1)$	$\delta_{2\to 3}(G,I)$: $\sum_D(\psi_2(\mathbf{C}_2) \times \delta_{1\to 2})$	$\delta_{3\to 5}(G,A)$: $\sum_I(\psi_3(\mathbf{C}_3) \times \delta_{2\to 3})$	$\delta_{4\to 5}(G,J)$: $\sum_H \psi_4(\mathbf{C}_4)$

Variable Elimination with a Clique Tree

Step 4: We can now compute $P(J)$, by getting the final factor:

$$\beta(G, J, A, L) = \psi_5(\mathbf{C}_5) \times \delta_{4\to 5} \times \delta_{3\to 5}$$

And summing out the nuisance variables:

$$\sum_{G,A,L} \beta(G, J, A, L)$$

$$\boxed{1:\ C,D} \xrightarrow[D]{} \boxed{2:\ G,I,D} \xrightarrow[G,I]{} \boxed{3:\ G,I,A} \xrightarrow[G,A]{} \boxed{5:\ G,J,A,L} \xleftarrow[G,J]{} \boxed{4:\ G,H,J}$$

Variable Elimination with a Clique Tree

Step 4: We can now compute $P(J)$, by getting the final factor:

$$\beta(G, J, A, L) = \psi_5(\mathbf{C}_5) \times \delta_{4\to5} \times \delta_{3\to5}$$

And summing out the nuisance variables:

$$\sum_{G,A,L} \beta(G, J, A, L)$$

Variable Elimination with a Clique Tree

Step 4: We can now compute $P(J)$, by getting the final factor:

$$\beta(G, J, A, L) = \psi_5(\mathbf{C}_5) \times \delta_{4\to5} \times \delta_{3\to5}$$

And summing out the nuisance variables:

$$\sum_{G,A,L} \beta(G, J, A, L)$$

$$\boxed{1:\ C,D} \xrightarrow[D]{} \boxed{2:\ G,I,D} \xrightarrow[G,I]{} \boxed{3:\ G,I,A} \xrightarrow[G,A]{} \boxed{5:\ G,J,A,L} \xleftarrow[G,J]{} \boxed{4:\ G,H,J}$$

Variable Elimination with a Clique Tree

Step 4: We can now compute $P(J)$, by getting the final factor:

$$\beta(G, J, A, L) = \psi_5(\mathbf{C}_5) \times \delta_{4\to5} \times \delta_{3\to5}$$

And summing out the nuisance variables:

$$\sum_{G,A,L} \beta(G, J, A, L)$$

$$\boxed{1:\ \text{C,D}} \xrightarrow[\text{D}]{} \boxed{2:\ \text{G,I,D}} \xrightarrow[\text{G,I}]{} \boxed{3:\ \text{G,I,A}} \xrightarrow[\text{G,A}]{} \boxed{5:\ \text{G,J,A,L}} \xleftarrow[\text{G,J}]{} \boxed{4:\ \text{G,H,J}}$$

$$\boxed{\begin{array}{c}\tilde{P}(J) = \\ \sum_{G,A,L} \psi_5(J, L, G, A) \times \delta_{4\to5} \times \delta_{3\to5}\end{array}}$$

Sum-Product Message Passing Algorithm Summary

This type of message passing is called the **sum-product** since the message from $\mathbf{C}_i$ to $\mathbf{C}_j$ is computed as follows:

$$\delta_{i\to j} = \overbrace{\sum_{\mathbf{C}_i - \mathbf{S}_{i,j}}}^{\texttt{SUM}} \left(\psi_i \times \overbrace{\prod_{k\in(Nb_i - \{j\})}}^{\texttt{PRODUCT}} \delta_{k\to i}\right)$$

1. In otherwords, the clique $\mathbf{C}_i$ multiplies all incoming messages from its neighbours with its initial clique potential ψ_i, creating a new factor.
2. Thereafter, it sums all variables except those in the sepset between $\mathbf{C}_i$ to $\mathbf{C}_j$.
3. This new message $\delta_{i\to j}$ is sent to $\mathbf{C}_j$.
4. Finally, when the root ($\mathbf{C}_r$) receives its messages (at the end) it multiplies them with its own initial potential.

$$\tilde{P}_\Phi(\mathbf{C}_r) = \sum_{\mathcal{X} - \mathbf{C}_r} \left(\psi_r \times \prod_{k\in Nb_i} \delta_{k\to r}\right)$$

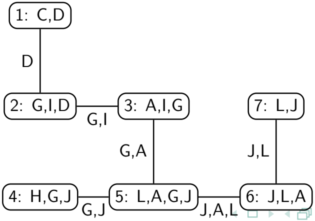
Legal Orderings

It is possible to have many different elimination orderings when using a clique tree.
For examples for the following clique tree, you can have orderings:

1, 2, 3, 7, 6, 4, 5
7, 6, 4, 1, 2, 3, 5

We can also state all possible orderings w.r.t. a root ($\mathbf{C}_5$) and clique tree using an ordering constraint:

$$\{(1 < 2), (2 < 3), (3 < 5), (4 < 5), (7 < 6), (6 < 5)\}$$



Clique Tree Calibration

So far we have learned a lot about exact inference.

But there is a core issue that we have not addressed yet.

We can calculate $P(J)$ or $P(J \mid \mathbf{e})$, but what about the joint $P(C, J)$ or $P(L, J, G, I)$?

To do this we can do the following:

1. Calculate the probability of all variables in the network individually.
2. Calculate the joint probability of each set of variables in each clique.

The second option sounds good since then we can just pick the scopes we want from any clique and eliminate the variables we don't want.

Calibration using Sum-Product

- It turns out that the second idea is a simple extension from what we have done so far.
- This is possible since the cluster graph/clique tree allows us to store messages which we can easily reuse.
- We simply do a upward pass towards any root and then a downward pass to the leaves.
- **This means we can calculate the posterior probability using only twice the computation of the upward pass in the same tree.**
- Remember to use **the initial potentials for all calculations**
- The modified potential contains information from the neighbours which will be double counting evidence.
- The intuition is that the clique will be reaffirming its own "ideas" about the sepset.

Upward pass and downward pass

We need only do a Upward pass and Downward pass.
Continuing from where we left-off:

Upward pass and downward pass

We need only do a Upward pass and Downward pass. Continuing from where we left-off:

$\psi_1(D,C)$ $\psi_2(G,I,D)$ $\psi_3(A,I)$ $\psi_5(J,L,G,A)$ $\psi_4(H,G,J)$

$$\text{1: C,D} \xrightarrow{\text{D}} \text{2: G,I,D} \xrightarrow{\text{G,I}} \text{3: G,I,A} \xrightarrow{\text{G,A}} \text{5: G,J,A,L} \xleftarrow{\text{G,J}} \text{4: G,H,J}$$

$\delta_{1\to 2}(D)$: $\sum_C \psi_1(\mathbf{C}_1)$

$\delta_{2\to 3}(G,I)$: $\sum_D(\psi_2(\mathbf{C}_2) \times \delta_{1\to 2})$

$\delta_{3\to 5}(G,A)$: $\sum_I(\psi_3(\mathbf{C}_3) \times \delta_{2\to 3})$

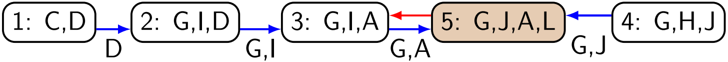
$\delta_{4\to 5}(G,J)$: $\sum_H \psi_4(\mathbf{C}_4)$

Upward pass and downward pass

We need only do a Upward pass and Downward pass. Continuing from where we left-off:

$$\psi_1(D,C) \quad \psi_2(G,I,D) \quad \psi_3(A,I) \quad \psi_5(J,L,G,A) \quad \psi_4(H,G,J)$$

$\delta_{5\to3}(G,A)$: $\sum_{J,L}(\psi_5(\mathbf{C}_5) \times \delta_{4\to5})$



$\delta_{1\to2}(D)$: $\sum_C \psi_1(\mathbf{C}_1)$

$\delta_{2\to3}(G,I)$: $\sum_D(\psi_2(\mathbf{C}_2) \times \delta_{1\to2})$

$\delta_{3\to5}(G,A)$: $\sum_I(\psi_3(\mathbf{C}_3) \times \delta_{2\to3})$

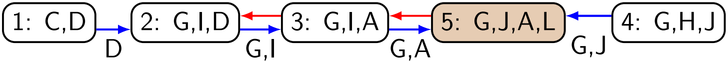
$\delta_{4\to5}(G,J)$: $\sum_H \psi_4(\mathbf{C}_4)$

Upward pass and downward pass

We need only do a Upward pass and Downward pass.
Continuing from where we left-off:

$\psi_1(D,C)$ $\psi_2(G,I,D)$ $\psi_3(A,I)$ $\psi_5(J,L,G,A)$ $\psi_4(H,G,J)$

$$\boxed{\delta_{3\to 2}(G,I): \sum_A(\psi_3(\mathbf{C}_3) \times \delta_{5\to 3})} \quad \boxed{\delta_{5\to 3}(G,A): \sum_{J,L}(\psi_5(\mathbf{C}_5) \times \delta_{4\to 5})}$$



$$\boxed{\delta_{1\to 2}(D): \sum_C \psi_1(\mathbf{C}_1)} \quad \boxed{\delta_{2\to 3}(G,I): \sum_D(\psi_2(\mathbf{C}_2) \times \delta_{1\to 2})} \quad \boxed{\delta_{3\to 5}(G,A): \sum_I(\psi_3(\mathbf{C}_3) \times \delta_{2\to 3})} \quad \boxed{\delta_{4\to 5}(G,J): \sum_H \psi_4(\mathbf{C}_4)}$$

Upward pass and downward pass

We need only do a Upward pass and Downward pass.
Continuing from where we left-off:

$\psi_1(D,C)$ $\psi_2(G,I,D)$ $\psi_3(A,I)$ $\psi_5(J,L,G,A)$ $\psi_4(H,G,J)$

$\delta_{2\to1}(D) : \sum_{G,I}(\psi_2(\mathbf{C}_2) \times \delta_{3\to2})$

$\delta_{3\to2}(G,I) : \sum_{A}(\psi_3(\mathbf{C}_3) \times \delta_{5\to3})$

$\delta_{5\to3}(G,A) : \sum_{J,L}(\psi_5(\mathbf{C}_5) \times \delta_{4\to5})$

$$\text{1: C,D} \underset{D}{\rightleftarrows} \text{2: G,I,D} \underset{G,I}{\rightleftarrows} \text{3: G,I,A} \underset{G,A}{\rightleftarrows} \boxed{\text{5: G,J,A,L}} \xleftarrow[G,J]{} \text{4: G,H,J}$$

$\delta_{1\to2}(D) : \sum_C \psi_1(\mathbf{C}_1)$

$\delta_{2\to3}(G,I) : \sum_{D}(\psi_2(\mathbf{C}_2) \times \delta_{1\to2})$

$\delta_{3\to5}(G,A) : \sum_{I}(\psi_3(\mathbf{C}_3) \times \delta_{2\to3})$

$\delta_{4\to5}(G,J) : \sum_H \psi_4(\mathbf{C}_4)$

Upward pass and downward pass

We need only do a Upward pass and Downward pass.
Continuing from where we left-off:

$\psi_1(D,C)$ $\psi_2(G,I,D)$ $\psi_3(A,I)$ $\psi_5(J,L,G,A)$ $\psi_4(H,G,J)$

$\delta_{2\to1}(D)$: $\sum_{G,I}(\psi_2(\mathbf{C}_2) \times \delta_{3\to2})$	$\delta_{3\to2}(G,I)$: $\sum_{A}(\psi_3(\mathbf{C}_3) \times \delta_{5\to3})$	$\delta_{5\to3}(G,A)$: $\sum_{J,L}(\psi_5(\mathbf{C}_5) \times \delta_{4\to5})$	$\delta_{5\to4}(G,J)$: $\sum_{A,L}\psi_5(\mathbf{C}_5) \times \delta_{3\to5})$

1: C,D —D— 2: G,I,D —G,I— 3: G,I,A —G,A— 5: G,J,A,L —G,J— 4: G,H,J

$\delta_{1\to2}(D)$: $\sum_{C}\psi_1(\mathbf{C}_1)$	$\delta_{2\to3}(G,I)$: $\sum_{D}(\psi_2(\mathbf{C}_2) \times \delta_{1\to2})$	$\delta_{3\to5}(G,A)$: $\sum_{I}(\psi_3(\mathbf{C}_3) \times \delta_{2\to3})$	$\delta_{4\to5}(G,J)$: $\sum_{H}\psi_4(\mathbf{C}_4)$

Calibration using Sum-Product

When the algorithm terminates, then each clique contains a **marginal (unnormalised) probability** over the variables in its scope.

If we are looking for a distribution $P(\mathbf{X})$, then we just pick the most suitable scope that contains $\mathbf{X}$ and **eliminate the redundant variables** in the clique.

It does not matter which clique we select. i.e. If $P(X)$ appears in two cliques, then both cliques must agree on its marginal.

We call this **calibration** since adjacent cliques have the same **modified potential**:

$$\sum_{\mathbf{C}_i - \mathbf{S}_{i,j}} \beta_i(\mathbf{C}_i) = \sum_{\mathbf{C}_j - \mathbf{S}_{i,j}} \beta_j(\mathbf{C}_j)$$

Errors for Approximate Inference

- In some applications we can tolerate some imprecision. That is, is there much difference between the probabilities 0.61 and 0.66?
- An estimate ρ has an **absolute error** ϵ for $P(\mathbf{y} \mid \mathbf{e})$ if:

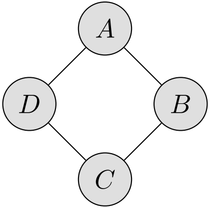
$$|\, P(\mathbf{y} \mid \mathbf{e}) - \rho \,| \leq \epsilon$$

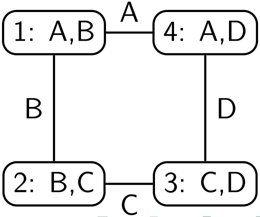
- What if we are trying to compute $P(\text{rare disease}) =$ 0.00001? Then a absolute error of 0.0001 is not acceptable.
- Therefore we need to take into consideration the value of the probability that we are trying to calculate:
- An estimate ρ has **relative error** ϵ for $P(\mathbf{y} \mid \mathbf{e})$ if:

$$\frac{\rho}{1+\epsilon} \leq P(\mathbf{y} \mid \mathbf{e}) \leq \rho(1+\epsilon)$$

Propagation-Based Approximation

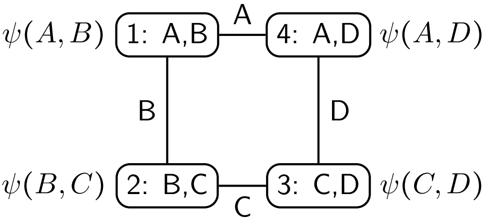
- We now turn our attention to approximate inference.
- Approximate inference is $\mathcal{NP}$-hard.
- This approximate inference strategy uses the same sum-product message passing we saw previously.
- The only difference is that:
  1. Exact inference procedure used clique trees
  2. This approximate inference procedure uses cluster graphs
- Lets consider the misconception network and its cluster graph:





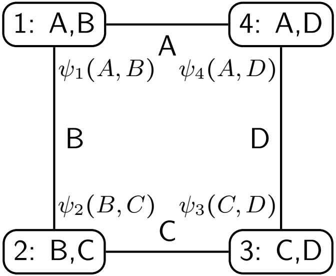
Propagation-Based Approximation

- Notice that this cluster graph is **not a clique tree**

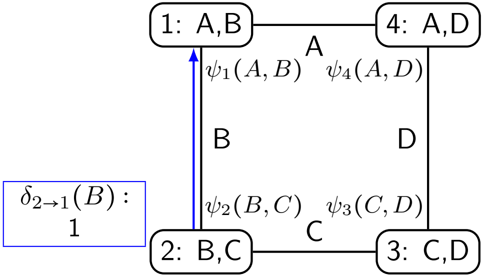


- Since it contains cycles we refer to it as **loopy**.

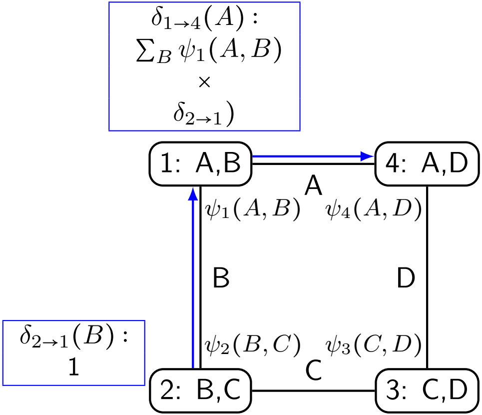
Belief Propagation



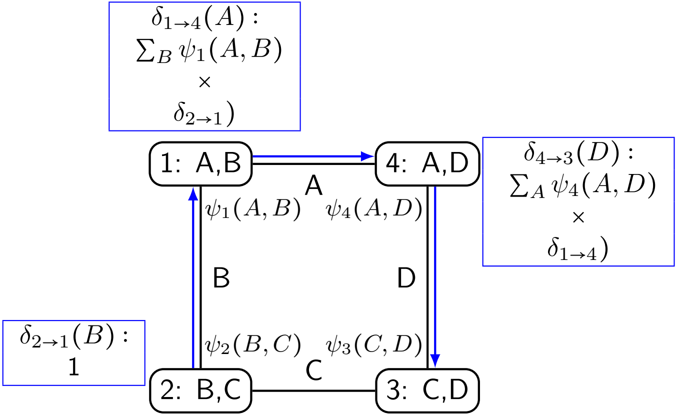
Belief Propagation



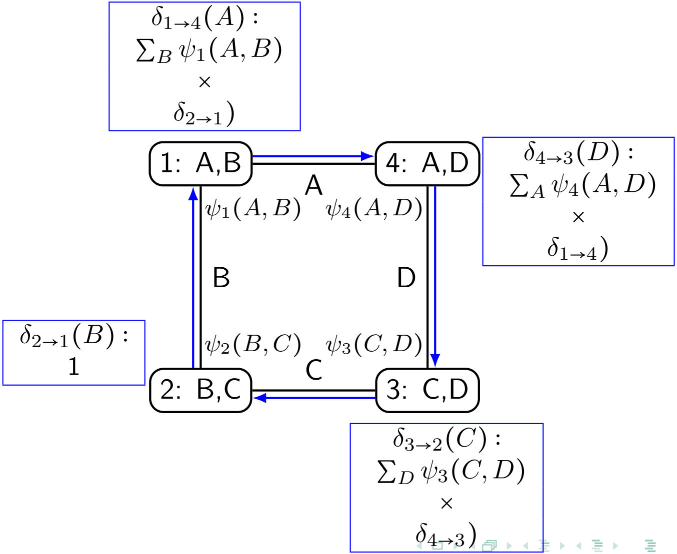
Belief Propagation



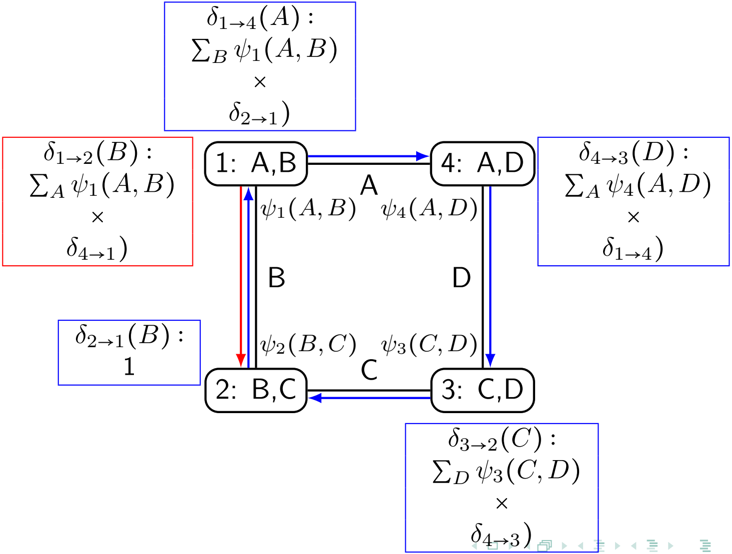
Belief Propagation



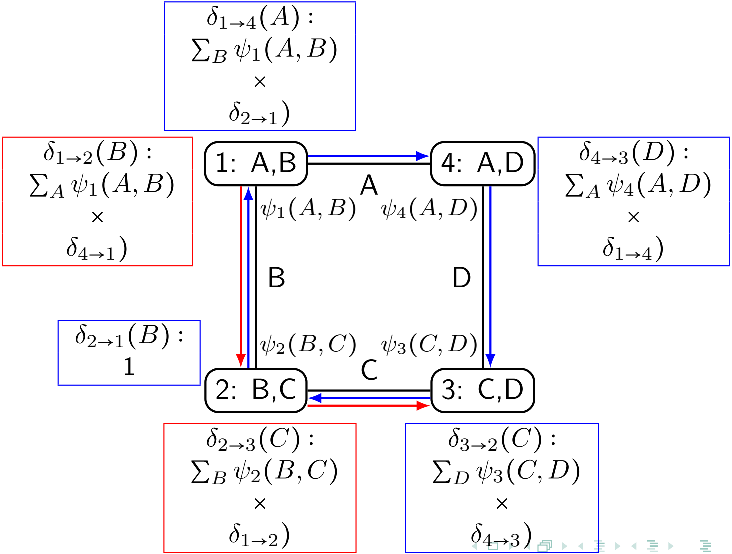
Belief Propagation



Belief Propagation



Belief Propagation

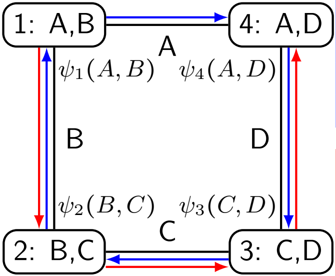


Belief Propagation

$\delta_{1\to4}(A)$: $\sum_B \psi_1(A,B) \times \delta_{2\to1})$

$\delta_{1\to2}(B)$: $\sum_A \psi_1(A,B) \times \delta_{4\to1})$

$\delta_{4\to3}(D)$: $\sum_A \psi_4(A,D) \times \delta_{1\to4})$



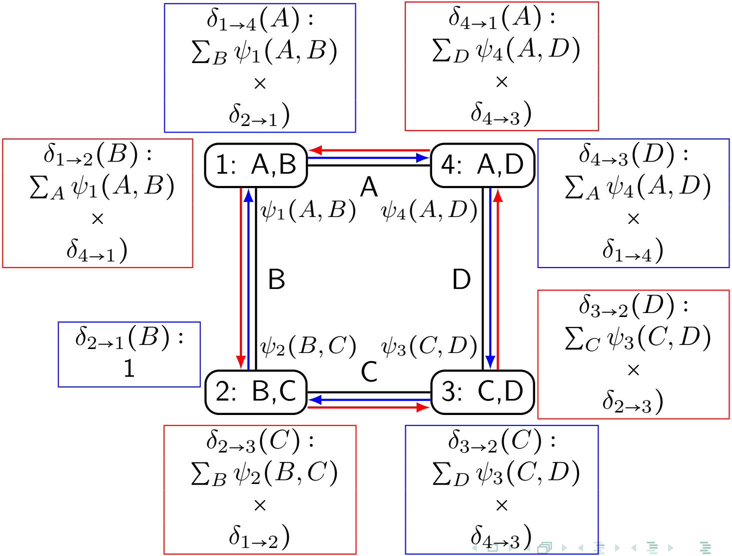
$\delta_{3\to2}(D)$: $\sum_C \psi_3(C,D) \times \delta_{2\to3})$

$\delta_{2\to1}(B)$: 1

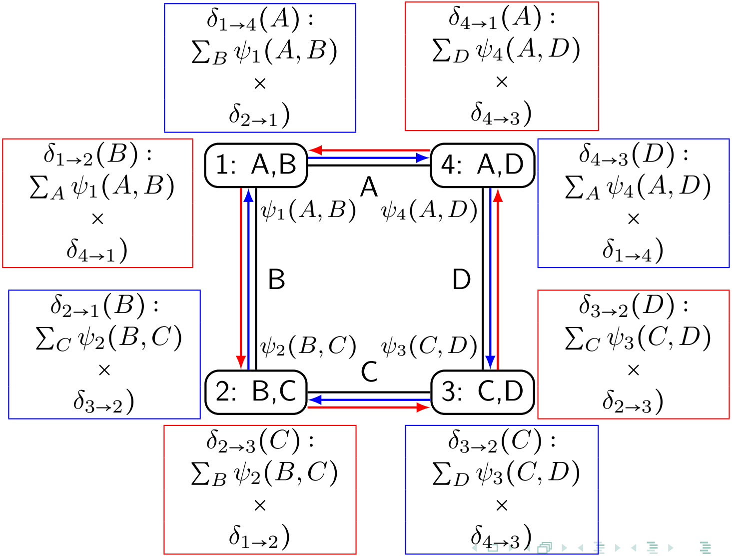
$\delta_{2\to3}(C)$: $\sum_B \psi_2(B,C) \times \delta_{1\to2})$

$\delta_{3\to2}(C)$: $\sum_D \psi_3(C,D) \times \delta_{4\to3})$

Belief Propagation



Belief Propagation



Callibration of Cluster Graphs

- Cluster graphs can also be calibrated if:

$$\sum_{\mathbf{C}_i - \mathbf{S}_{i,j}} \beta_i = \sum_{\mathbf{C}_j - \mathbf{S}_{i,j}} \beta_j$$

- In otherwords, every adjacent set of clusters agree on the marginal of variables in $\mathbf{S}_{i,j}$. (weaker calibration)
- Clique tree calibration the adjacent clusters agreed on all variables they had in common (not only those in the sepset).
- Cluster-graph belief propagation are sometimes preferred:
  - They can be significantly cheaper than exact inference.
  - Clusters are sometimes easier to model for certain models (Markov Net or grids)
- Convergence: Networks with potentials that are closer to deterministic are more likely to have problems with convergence.

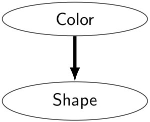
Problem Statement

$$\mathrm{MAP}(\mathbf{Y} = \mathbf{y} \mid \mathbf{E} = \mathbf{e}) = \operatorname*{argmax}_{y} P(\mathbf{Y} = \mathbf{y} \mid \mathbf{E} = \mathbf{e})$$

- Applications:
  - **Signals**: noise reduction, source separation
  - **Images**: Denoising, image segmentation and restoration
  - **Communication**: channel decoding
  - **Finance**: portfolio optimization
  - **Robotics**: localization, mapping (SLAM), planning
  - **Neuroscience**: decode neural activity
  - **Physics**: particle location, learn gravitational wave
  - **Genetics**: linkage analysis, haplotype reconstruction
  - **Epidemiology**: disease transmission models
  - **NLP:** machine translation, speech recognition
  - **Marketing:** customer segmentation and targeting
  - **Social networks:** network analysis and link prediction
  - **Cyber security:** anomaly detection

Problem Statement: Example

Color	
R	B
0.4	0.6

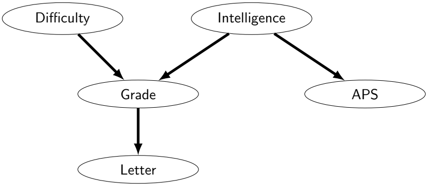


	Shape	
Color	T	S
R	0.1	0.9
B	0.5	0.5

	$P(C,S)$	
Color	Shape	P(C,S)
R	T	0.04
R	S	0.36
B	T	0.3
B	S	0.3

- The following problems are $\mathcal{NP}$-Hard:
  1. Finding a single coherent joint assignment $\mathbf{a}^*$ with highest probability.
  2. Finding an assignment $\mathbf{x}$, such that $P(\mathbf{x}) > p$, where $0 \leq p \leq 1$.

Product to Summation



1. $P(D,I,G,A,L) = P(D)P(I)P(G \mid D,I)P(A \mid I)P(L)$
2. $\operatorname{argmax}\Big(P(D)P(I)P(G \mid D,I)P(A \mid I)P(L)\Big)$
3. $\operatorname{argmax}\Big(\log P(D) + \log P(I) + \log P(G \mid D,I) + \dots\Big)$
4. $\operatorname{argmax}\Big(\theta_D + \theta_I + \theta_{G|D,I} + \theta_{A|I} + \theta_{L|G}\Big)$

Max-Sum Elimination

Find $\text{argmax}_{L,G,I,D} P(A)$?

1. $$\max_L \max_G \max_I \max_D \left(\theta(D)+\theta(I)+\theta(G,D,I)+\theta(A,I)+\theta(L,G)\right)$$
2. $$= \max_L \max_G \max_I \left(\theta(I)+\theta(A,I)+\theta(L,G)+\max_D(\psi_1(G,D,I))\right)$$
3. $$= \max_L \max_G \max_I \left(\theta(I)+\theta(A,I)+\theta(L,G)+\lambda_1(G,I)\right)$$
4. $$= \max_L \max_G \left(\theta(L,G)+\max_I(\lambda_1(G,I)+\theta(I)+\theta(A,I))\right)$$
5. $$= \max_L \max_G \left(\theta_{L,G}+\max_I(\psi_2(G,I,A))\right)$$
6. $$= \max_L \max_G \left(\theta(L,G)+\lambda_2(G,A)\right) = \max_L \left(\lambda_3(A,L)\right) = \lambda_4(A)$$

Factor Summation

A factor summation $(\phi_1 + \phi_2)$ is defined as a factor $\psi : Val(\mathbf{X},\mathbf{Y},\mathbf{Z}) \mapsto \mathbb{R}$ as $\psi(\mathbf{X},\mathbf{Y},\mathbf{Z}) = \phi_1(\mathbf{X},\mathbf{Y}) + \phi_2(\mathbf{Y},\mathbf{Z})$

a^1	b^1	3
a^1	b^2	0
a^2	b^1	-1
a^2	b^2	1

$\phi(A,B)$

+

b^1	c^1	4
b^1	c^2	1.5
b^2	c^1	0.2
b^2	c^2	2

$\phi(B,C)$

=

a^1	b^1	c^1	$3+4=7$
a^1	b^1	c^2	$3+1.5=4.5$
a^1	b^2	c^1	$0+0.2=0.2$
a^1	b^2	c^2	$0+2=2$
a^2	b^1	c^1	$-1+4=3$
a^2	b^1	c^2	$-1+1.5=0.5$
a^2	b^2	c^1	$1+0.2=1.2$
a^2	b^2	c^2	$1+2=3$

$\psi(A,B,C)$

Factor Maximisation

A factor maximisation ($\max_X \phi(\mathbf{X}, \mathbf{Y})$) is defined as a factor $\psi : Val(\mathbf{Y}) \mapsto \mathbb{R}$ as $\max_X \phi_1(\mathbf{X}, \mathbf{Y}) = \psi(\mathbf{Y})$

a^1	b^1	c^1	7
a^1	b^1	c^2	4.5
a^1	b^2	c^1	0.2
a^1	b^2	c^2	2
a^2	b^1	c^1	3
a^2	b^1	c^2	0.5
a^2	b^2	c^1	1.2
a^2	b^2	c^2	3

$\psi(A, B, C)$

$$\rightarrow \max_B \psi(A, B, C) \rightarrow$$

a^1	c^1	7
a^1	c^2	4.5
a^2	c^1	3
a^2	c^2	3

$\phi(A, C)$

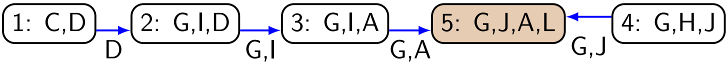
Max-Sum in Clique Trees (MAP Marginal)

Lets now consider the extended student network. Again, we need only do a Upward pass and Downward pass to reach convergence:

Max-Sum in Clique Trees (MAP Marginal)

Lets now consider the extended student network. Again, we need only do a Upward pass and Downward pass to reach convergence:

$\theta_1(D, C) \quad \theta_2(G, I, D) \quad \theta_3(A, I) \quad \theta_5(J, L, G, A) \quad \theta_4(H, G, J)$



Max-Sum in Clique Trees (MAP Marginal)

Lets now consider the extended student network. Again, we need only do a Upward pass and Downward pass to reach convergence:

$\theta_1(D,C)$ $\theta_2(G,I,D)$ $\theta_3(A,I)$ $\theta_5(J,L,G,A)$ $\theta_4(H,G,J)$

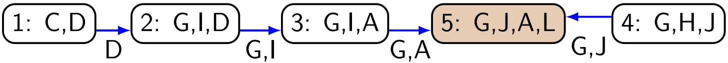
1: C,D $\xrightarrow{D}$ 2: G,I,D $\xrightarrow{G,I}$ 3: G,I,A $\xrightarrow{G,A}$ 5: G,J,A,L $\xleftarrow{G,J}$ 4: G,H,J

$$\lambda_{1\to 2}(D): \max_C \psi_1(\mathbf{C}_1)$$

Max-Sum in Clique Trees (MAP Marginal)

Lets now consider the extended student network. Again, we need only do a Upward pass and Downward pass to reach convergence:

$\theta_1(D,C)$ $\theta_2(G,I,D)$ $\theta_3(A,I)$ $\theta_5(J,L,G,A)$ $\theta_4(H,G,J)$



$$\lambda_{1\to 2}(D): \max_{C} \psi_1(\mathbf{C}_1)$$

$$\lambda_{2\to 3}(G,I): \max_{D}(\psi_2(\mathbf{C}_2) + \lambda_{1\to 2})$$

Max-Sum in Clique Trees (MAP Marginal)

Lets now consider the extended student network. Again, we need only do a Upward pass and Downward pass to reach convergence:

$\theta_1(D,C)$ $\theta_2(G,I,D)$ $\theta_3(A,I)$ $\theta_5(J,L,G,A)$ $\theta_4(H,G,J)$

$$\boxed{1:\ C,D} \xrightarrow[D]{} \boxed{2:\ G,I,D} \xrightarrow[G,I]{} \boxed{3:\ G,I,A} \xrightarrow[G,A]{} \boxed{5:\ G,J,A,L} \xleftarrow[G,J]{} \boxed{4:\ G,H,J}$$

$$\boxed{\lambda_{1\to2}(D): \max_C \psi_1(\mathbf{C}_1)}$$

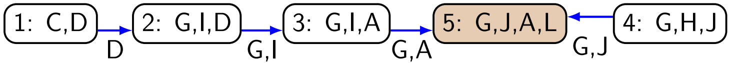
$$\boxed{\lambda_{2\to3}(G,I): \max_D(\psi_2(\mathbf{C}_2) + \lambda_{1\to2})}$$

$$\boxed{\lambda_{3\to5}(G,A): \max_I(\psi_3(\mathbf{C}_3) + \lambda_{2\to3})}$$

Max-Sum in Clique Trees (MAP Marginal)

Lets now consider the extended student network. Again, we need only do a Upward pass and Downward pass to reach convergence:

$\theta_1(D,C)$ $\theta_2(G,I,D)$ $\theta_3(A,I)$ $\theta_5(J,L,G,A)$ $\theta_4(H,G,J)$



$\lambda_{1\to 2}(D): \max_C \psi_1(\mathbf{C}_1)$

$\lambda_{2\to 3}(G,I): \max_D(\psi_2(\mathbf{C}_2) + \lambda_{1\to 2})$

$\lambda_{3\to 5}(G,A): \max_I(\psi_3(\mathbf{C}_3) + \lambda_{2\to 3})$

$\lambda_{4\to 5}(G,J): \max_H \psi_4(\mathbf{C}_4)$

Max-Sum in Clique Trees (MAP Marginal)

Lets now consider the extended student network. Again, we need only do a Upward pass and Downward pass to reach convergence:

$\theta_1(D,C) \quad \theta_2(G,I,D) \quad \theta_3(A,I) \quad \theta_5(J,L,G,A) \quad \theta_4(H,G,J)$

$$\lambda_{5\to3}(G,A) : \max_{J,L}(\psi_5(\mathbf{C}_5) + \lambda_{4\to5})$$

$$\lambda_{1\to2}(D) : \max_{C}\psi_1(\mathbf{C}_1)$$

$$\lambda_{2\to3}(G,I) : \max_{D}(\psi_2(\mathbf{C}_2) + \lambda_{1\to2})$$

$$\lambda_{3\to5}(G,A) : \max_{I}(\psi_3(\mathbf{C}_3) + \lambda_{2\to3})$$

$$\lambda_{4\to5}(G,J) : \max_{H}\psi_4(\mathbf{C}_4)$$

Max-Sum in Clique Trees (MAP Marginal)

Lets now consider the extended student network. Again, we need only do a Upward pass and Downward pass to reach convergence:

$\theta_1(D,C)$ $\theta_2(G,I,D)$ $\theta_3(A,I)$ $\theta_5(J,L,G,A)$ $\theta_4(H,G,J)$

$$\lambda_{3\to2}(G,I) : \max_{A}(\psi_3(\mathbf{C}_3) + \lambda_{5\to3})$$

$$\lambda_{5\to3}(G,A) : \max_{J,L}(\psi_5(\mathbf{C}_5) + \lambda_{4\to5})$$

$$\lambda_{1\to2}(D) : \max_{C}\psi_1(\mathbf{C}_1)$$

$$\lambda_{2\to3}(G,I) : \max_{D}(\psi_2(\mathbf{C}_2) + \lambda_{1\to2})$$

$$\lambda_{3\to5}(G,A) : \max_{I}(\psi_3(\mathbf{C}_3) + \lambda_{2\to3})$$

$$\lambda_{4\to5}(G,J) : \max_{H}\psi_4(\mathbf{C}_4)$$

Max-Sum in Clique Trees (MAP Marginal)

Lets now consider the extended student network. Again, we need only do a Upward pass and Downward pass to reach convergence:

$\theta_1(D,C)$ $\theta_2(G,I,D)$ $\theta_3(A,I)$ $\theta_5(J,L,G,A)$ $\theta_4(H,G,J)$

$$\lambda_{2\to1}(D): \max_{G,I} \psi_2(\mathbf{C}_2) + \lambda_{3\to2})$$

$$\lambda_{3\to2}(G,I): \max_{A}(\psi_3(\mathbf{C}_3) + \lambda_{5\to3})$$

$$\lambda_{5\to3}(G,A): \max_{J,L}(\psi_5(\mathbf{C}_5) + \lambda_{4\to5})$$

1: C,D —D— 2: G,I,D —G,I— 3: G,I,A —G,A— 5: G,J,A,L —G,J— 4: G,H,J

$$\lambda_{1\to2}(D): \max_{C} \psi_1(\mathbf{C}_1)$$

$$\lambda_{2\to3}(G,I): \max_{D}(\psi_2(\mathbf{C}_2) + \lambda_{1\to2})$$

$$\lambda_{3\to5}(G,A): \max_{I}(\psi_3(\mathbf{C}_3) + \lambda_{2\to3})$$

$$\lambda_{4\to5}(G,J): \max_{H} \psi_4(\mathbf{C}_4)$$

Max-Sum in Clique Trees (MAP Marginal)

Lets now consider the extended student network. Again, we need only do a Upward pass and Downward pass to reach convergence:

$\theta_1(D,C)$ $\theta_2(G,I,D)$ $\theta_3(A,I)$ $\theta_5(J,L,G,A)$ $\theta_4(H,G,J)$

$\lambda_{2\to1}(D): \max_{G,I} \psi_2(\mathbf{C}_2) + \lambda_{3\to2})$

$\lambda_{3\to2}(G,I): \max_{A}(\psi_3(\mathbf{C}_3) + \lambda_{5\to3})$

$\lambda_{5\to3}(G,A): \max_{J,L}(\psi_5(\mathbf{C}_5) + \lambda_{4\to5})$

$\lambda_{5\to4}(G,J): \max_{A,L} \psi_5(\mathbf{C}_5) + \lambda_{3\to5})$

1: C,D —D— 2: G,I,D —G,I— 3: G,I,A —G,A— 5: G,J,A,L —G,J— 4: G,H,J

$\lambda_{1\to2}(D): \max_{C} \psi_1(\mathbf{C}_1)$

$\lambda_{2\to3}(G,I): \max_{D}(\psi_2(\mathbf{C}_2) + \lambda_{1\to2})$

$\lambda_{3\to5}(G,A): \max_{I}(\psi_3(\mathbf{C}_3) + \lambda_{2\to3})$

$\lambda_{4\to5}(G,J): \max_{H} \psi_4(\mathbf{C}_4)$

Decoding a MAP Assignment

- If the MAP assignment is unique:
  1. There is a single maximising assignment at each clique.
  2. Whose value is the θ value of the MAP Assignment.
  3. Due to calibration, all choices must agree.
- If the MAP assignment is NOT unique:
  1. Add a tiny random noise to each factor to make them unique.

Summary

- Inference is probabilistic queries or MAP assignments
- These tasks are done exact or approximate inference
- Both are $\mathcal{NP}$-hard, however it is used practical applications
- Exact inference which is deeply rooted in variable elimination
- Variable elimination can be sum-product or max-sum
- Elimination orderings affect the computational time
- Approximate inference: cluster-based belief propagation, calibration, and convergence