

# Exact Inference on Bayesian Networks

## PAI. Tutorial 4

Gideon Dresdner

ETH Zürich

October 18, 2019

# The Plan

## Homework 2

- ▶  $d$ -separation problem (Q1)
- ▶ Outline of  $d$ -separation coding (Q3)
- ▶ Variable elimination problem (Q2)
- ▶ Selections of variable elimination problem (Q4)

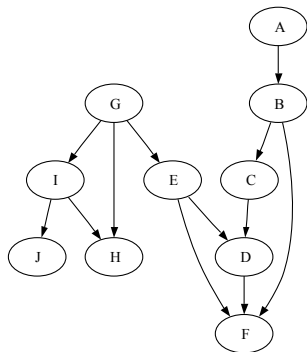
## Recap on Inference

1. The problem of inference
2. The basic approaches
3. Review of sum-product algorithm

## Question 1: $d$ -separation

- ▶  $d$ -sep  $\implies$  independence
  - ▶ Why not the other direction?
  - ▶ Bayesian networks don't need to be “minimal”
- ▶ the opposite of  $d$ -separation is active trail — “information flow” — good for drawing
- ▶ What happens when you condition-on / observe  $B$ :
  - ▶  $A \rightarrow B \leftarrow C$
  - ▶ Now do the same but connect  $A \rightarrow C$ .

## Question 1: $d$ -separation



- ▶ Let's query. . . any suggestions?
- ▶ Of the form:

$$X \text{ } d\text{-sep} Y \mid (Z_1, Z_2, \dots)$$

- ▶ Consider  $I \text{ } d\text{-sep} C \mid H, F$ .  
Conditioning on  $H$  leaves the path  $I - G - E$  open.  
Conditioning on  $F$  activates the path between  $E - D - C$ . Thus there is an active trail, thus not  $d$ -sep.

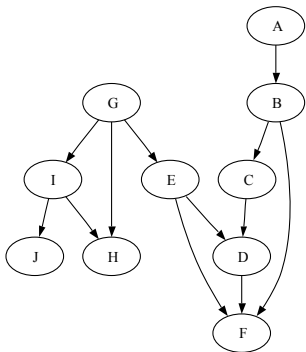
### Question 3: $d$ -sep coding exercise

Remember, we solve this by *finding all the active paths*. Go to vim to see code.

## Question 2: variable elimination

1. Setup initial factors:

$$\begin{aligned} &P(A)P(B | A)P(C | B) \\ &\cdot P(G)P(E | G)P(D | C, E)P(F | B, D, E) \\ &\cdot P(I | G)P(J | I)P(H | G, I) \end{aligned}$$



2. Marginalize in some order  
(given in problem as  
 $A, B, C, D, E, F, G, H, I, J$ ):

$$g_1(B) = \sum_a P(A)P(B | A)$$

$$g_2(C, F, D, E) = \sum_b g_1(B)P(C | B)P(F | B, D, E)$$

3. etc. (blackboard)

## Reminder: variable elimination context

- ▶ What are we missing here?
- ▶ queries and observations (e.g. data)
- ▶ For observations we “clamp,” e.g. multiply joint distribution by  $[A = 0]$ .<sup>1</sup>
- ▶ We remove queried variables from the list of variables to be eliminated.

---

<sup>1</sup>This notation is called the “Iverson Bracket.”

## Question 4: Querying using variable elimination

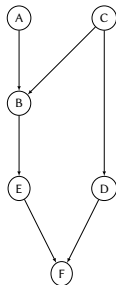


Figure 3: Bayesian network for problem 4.

- Queries of the form  
 $P(A = a \mid B = b, D = d)$

$$P(A = t) = 0.3 \quad (1)$$

$$P(C = t) = 0.6 \quad (2)$$

Table 1: CPTs for problem 1.

$A$	$C$	$P(B = t)$	$C$	$P(D = t)$	$B$	$P(E = t)$	$D$	$E$	$P(F = t)$
$f$	$f$	0.2	$f$	0.9	$f$	0.2	$f$	$f$	0.95
$f$	$t$	0.8	$t$	0.75	$t$	0.4	$f$	$t$	1
$t$	$f$	0.3					$t$	$f$	0
$t$	$t$	0.5					$t$	$t$	0.25

## Question 4: Querying using variable elimination

- ▶ What are the variables we need to eliminate?
- ▶  $C, E, F$
- ▶ Write down the initial factors  
... actually this is already done for us in the conditional probability tables.
- ▶ Eliminate  $C, E, F$ . What order?  
... let's try  $F, E, C$   
... Why this one?  
... Leaves to root (also known as topological order) is best.

## Question 4: Querying using variable elimination

### Initial Factors

$g_1(A)$		$g_2(C)$		$g_3(C, D)$		$g_4(B, E)$	
$A$		$C$		$C$	$D$	$B$	$E$
$f$	0.7	$f$	0.4	$f$	$f$	$f$	$f$
$t$	0.3	$t$	0.6	$f$	$t$	$f$	$t$
				$t$	$f$	$t$	$f$
				$t$	$t$	$t$	$t$

$g_5(A, B, C)$			$g_6(D, E, F)$		
$A$	$B$	$C$	$D$	$E$	$F$
$f$	$f$	$f$	$f$	$f$	$f$
$f$	$f$	$t$	$f$	$f$	$t$
$f$	$t$	$f$	$f$	$t$	$f$
$f$	$t$	$t$	$f$	$t$	$t$
$t$	$f$	$f$	$t$	$f$	$f$
$t$	$f$	$t$	$t$	$f$	$t$
$t$	$t$	$f$	$t$	$t$	$f$
$t$	$t$	$t$	$t$	$t$	$t$

## Question 4: Querying using variable elimination

### Initial Factors

### Eliminate $F$

- ▶ Find all the factors with  $F$ :  $g_6$
- ▶ Marginalize

$$\sum_f g_6(D, E, f) := g_7(D, E) = 1$$

- ▶ Factors:  $g_1, g_2, g_3, g_4, g_5, \cancel{g_6}, g_7$

## Question 4: Querying using variable elimination

### Eliminate $E$

- ▶ Find all the factors with  $E$ :  $g_4, g_7$
- ▶ Form product  $f_1(B, D, E) := g_4(B, E)g_7(D, E)$
- ▶ Eliminate  $E$

$$\sum_e f_1(B, D, e) = \sum_e \underbrace{g_4(B, e)}_{\rightarrow P(e|B)} \underbrace{g_7(D, e)}_1 := g_8(B, D) = 1.$$

- ▶ Factors:  $g_1, g_2, g_3, \cancel{g_4}, g_5, \cancel{g_6}, \cancel{g_7}, g_8$

## Question 4: Querying using variable elimination

### Eliminate $C$

- Factors:

$$g_1(A), g_2(C), g_3(C, D), \cancel{g_4}, g_5(A, B, C), \cancel{g_6}, \cancel{g_7}, g_8(B, D)$$

- Form the product

$$f_3(A, B, C, D) := g_2(C)g_3(C, D)g_5(A, B, C)$$

$A$	$B$	$C$	$D$	$f_3(A, B, C, D)$
$f$	$f$	$f$	$f$	$0.04 \times 0.8 = 0.032$
$f$	$f$	$f$	$t$	$0.36 \times 0.8 = 0.288$
$f$	$f$	$t$	$f$	$0.15 \times 0.2 = 0.030$
...				...

- Marginalize...  $g_9(A, B, D)$

## Question 4: Querying using variable elimination

### Answering queries

- ▶ All we have is an unnormalized density

$$P(a \mid b, d) \propto g_1(a)g_8(b, d)g_9(a, b, c)$$

- ▶ We need to normalize:

$$P(A = t \mid b, d) = \frac{g_1(A = t)g_8(b, d)g_9(A = t, b, c)}{g_1(A = t)g_8(b, d)g_9(A = t, b, c) + g_1(A = f)g_8(b, d)g_9(A = t, b, c)}$$

# Bayesian inference in a nutshell

## The Problem(s)

1. Want to evaluate queries.
2. Queries are essentially summation problems.
3. Broadly speaking, there are a few basic approaches.

## Approaches

1. **Exact inference and inspired methods (loopy belief propagation)**
2. Deterministic approximation (e.g. variational inference, end of previous lecture)
3. Stochastic approximation (e.g. sampling, future lectures)

## Sum-product algorithm (aka Belief Propagation)

- ▶ Interesting part of the algorithm: recurrence relation
- ▶ The full algorithm

# Sum-product algorithm (aka Belief Propagation)

## Recurrence Relation

- ▶ Notation —  $v$ : nodes,  $u$ : factors,  $N(a)$ : neighbors in the factor graph,  $\mathbf{x}_u \sim x_v$ : table of neighbors of factor  $u$  with node  $v$  node clamped to value  $x_v$ .
- ▶ Messages from  $u \rightarrow v$  are given in terms of messages from  $v \rightarrow u$  and also the other way around.
- ▶ A node,  $v$ , of the Bayesian Network only keeps track of it's own marginal.

$$\mu_{v \rightarrow u}(x_v) = \prod_{u' \in N(v) \setminus \{u\}} \mu_{u' \rightarrow v}(x_v)$$

- ▶ Basic idea: marginalize to make the dimension work out.

$$\mu_{u \rightarrow v}(x_v) = \sum_{\mathbf{x}_u \sim x_v} f(\mathbf{x}_u) \prod_{v' \in N(u) \setminus \{v\}} \mu_{v' \rightarrow u}(x_{v'})$$

## Sum-product algorithm (aka Belief Propagation)

- ▶ Take the polytree, select a root node.
- ▶ Initialize the leaf messages to 1, i.e.  $\mu_{v \rightarrow u} = 1$ .
- ▶ Pass all messages
- ▶ Use messages to compute what you want:

$$P(X_v = x_v) \propto \prod_{u' \in N(v)} \mu_{u' \rightarrow v}(x_v)$$

- ▶ Note, you need to renormalize. Ok when discrete and reasonable number of possible values.