

## Informatics 2D – Reasoning and Agents

Semester 2, 2011-12

Alex Lascarides  
alex@inf.ed.ac.uk

School of  
**informatics**



Lecture 24 – Exact Inference in Bayesian Networks  
16th March 2012  
adapted from slides by Michael Rovatsos

## Where are we?

Last time ...

- ▶ Introduced Bayesian networks
- ▶ Allow for compact representation of JPDs
- ▶ Methods for efficient representations of CPTs
- ▶ But how hard is inference in BNs?

Today ...

- ▶ **Inference in Bayesian networks**

## Inference in BNs

- ▶ Basic task: compute posterior distribution for set of **query variables** given some observed **event** (i.e. assignment of values to **evidence variables**)
- ▶ Formally: determine  $\mathbf{P}(X|\mathbf{e})$  given query variables  $\mathbf{X}$ , evidence variables  $\mathbf{E}$  (and non-evidence or **hidden** variables  $\mathbf{Y}$ )
- ▶ Example:  $\mathbf{P}(\text{Burglary} | \text{JohnCalls} = \text{true}, \text{MaryCalls} = \text{true}) = (0.284, 0.716)$
- ▶ First we will discuss exact algorithms for computing posterior probabilities then approximate methods later

## Inference by enumeration

- ▶ We have seen that any conditional probability can be computed from a full JPD by summing terms
- ▶  $\mathbf{P}(X|\mathbf{e}) = \alpha \mathbf{P}(X, \mathbf{e}) = \alpha \sum_{\mathbf{y}} \mathbf{P}(X, \mathbf{e}, \mathbf{y})$
- ▶ Since BN gives complete representation of full JPD, we must be able to answer a query by computing sums of products of conditional probabilities from the BN
- ▶ Consider query  
 $\mathbf{P}(\text{Burglary} | \text{JohnCalls} = \text{true}, \text{MaryCalls} = \text{true}) = \mathbf{P}(B|j, m)$
- ▶  $\mathbf{P}(B|j, m) = \alpha \mathbf{P}(B, j, m) = \alpha \sum_{\mathbf{e}} \sum_{\mathbf{a}} \mathbf{P}(B, \mathbf{e}, \mathbf{a}, j, m)$

## Inference by enumeration

- Recall  $P(x_1, \dots, x_n) = \prod_{i=1}^n P(x_i | \text{parents}(X_i))$
- We can use CPTs to simplify this exploiting BN structure
- For *Burglary = true*:

$$P(b|j, m) = \alpha \sum_e \sum_a P(b)P(e)P(a|b, e)P(j|a)P(m|a)$$

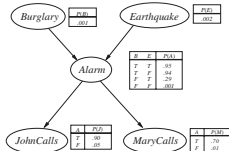
- But we can improve efficiency of this by moving terms outside that don't depend on sums

$$P(b|j, m) = \alpha P(b) \sum_e P(e) \sum_a P(a|b, e)P(j|a)P(m|a)$$

- To compute this, we need to loop through variables in order and multiply CPT entries; for each summation we need to loop over variable's possible values

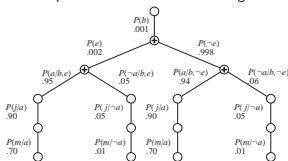
## Example

- New burglar alarm has been fitted, fairly reliable but sometimes reacts to earthquakes
- Neighbours John and Mary promise to call when they hear alarm
- John sometimes mistakes phone for alarm, and Mary listens to loud music and sometimes doesn't hear alarm



## The variable elimination algorithm

- Despite improvements, enumeration method is computationally quite hard ( $O(2^n)$  instead of  $O(n2^n)$ )
- Often computes same things twice, e.g.  $P(j|a)P(m|a)$  and  $P(j|\neg a)P(m|\neg a)$  for each value of  $e$
- Evaluation of expression shown in the following tree:



## The variable elimination algorithm

- Idea of **variable elimination**: avoid repeated calculations
- Basic idea: store results after doing calculation once
- Works bottom-up by evaluating subexpressions
- Assume we want to evaluate

$$P(B|j, m) = \alpha \underbrace{P(B)}_{f_1(B)} \sum_e \underbrace{P(e)}_{f_2(E)} \sum_a \underbrace{P(a|B, e)}_{f_3(A, B, E)} \underbrace{P(j|a)}_{f_4(A)} \underbrace{P(m|a)}_{f_5(A)}$$

- We've annotated each part with a **factor**.
- A factor is a **matrix**, indexed with its argument variables. E.g:

- Factor  $f_5(A)$  corresponds to  $P(m|a)$  and depends just on  $A$  because  $m$  is fixed (it's a  $2 \times 1$  matrix).

$$f_5(A) = \langle P(m|a), P(m|\neg a) \rangle$$

- $f_3(A, B, E)$  is a  $2 \times 2 \times 2$  matrix for  $P(a|B, E)$



## Summary

- ▶ Inference in Bayesian Networks
- ▶ Exact methods: enumeration, variable elimination algorithm
- ▶ Computationally intractable in the worst case
- ▶ Next time: **Approximate inference in Bayesian Networks**