

# Exact Inference I

Mark Peot

In this lecture we will look at issues associated with exact inference.

## 1.0 Queries

The objective of probabilistic inference is to compute a joint distribution of a set of *query variables* given values for a set of *conditioning* or *evidence variables*. These joint distributions are generally computed by summing out the variables for *nuisance variables*; variables that are neither evidence nor query variables. Say that the set of all variables in a JPD is  $X$ .

$$P\{Q|E\} = \frac{P\{Q, E\}}{P\{E\}} = K_e \sum_N P\{X\} \quad (\text{EQ 1})$$

$K_e$  is a normalization constant that is only a function of  $E$ ,  $K_e = \sum_{X \setminus E} P\{X\}$ .  $N$  is the set of nuisance variables,  $N = X \setminus (Q \cup E)$ .  $\Sigma$  will be understood to be a general summation operator that sums over both discrete and continuous variables.

An important subproblem for inference is to compute the expectation  $E\{g(Q)|E\}$  or  $\langle g(Q)|E \rangle$  of a function (called a *statistic*) on the set of query variables. Some of the more important of these statistics include:

### Probability that $q = k$ :

If  $q$  is discrete: The Kronecker delta<sup>1</sup> or *indicator* function is

$$\delta(q, k) = \begin{cases} 1 & \text{if } (q = k) \\ 0 & \text{otherwise} \end{cases} . P\{q = k\} = \sum_q \delta(q, k) P\{q|E\}$$

If  $q$  is continuous: The Dirac delta function is  $\delta(x) = \lim_{\sigma \rightarrow 0} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$

and has the property that  $\int \delta(q - k) f(q|E) dq = f(q = k)$ .

---

1. Actually, the notation  $\delta_{qk}$  is usually used for a Kronecker delta function.

**Mean of the posterior:**  $g(q) = q \cdot \langle q|E \rangle = \sum_a q P\{q|E\}$ .

**Moments of the posterior:**  $g(q) = q^k \cdot \langle q^k|E \rangle = \sum_a q^k P\{q|E\}$ , the  $k^{\text{th}}$  moment of  $q$ .<sup>1</sup>

**Covariance of the posterior:**  $g(q_1, q_2) = q_1 q_2 - \langle q_1|E \rangle \langle q_2|E \rangle$ .

$$\text{cov}(q_1, q_2|E) = \left( \sum_{q_1, q_2} q_1 q_2 P\{q_1, q_2|E\} \right) - \langle q_1|E \rangle \langle q_2|E \rangle.$$

**Generating Functions:**  $g(q) = e^{qt}$ .  $M(t|E) = \langle e^{qt}|E \rangle$ .

Some algorithms (for example, simulation algorithms) compute conditional expectations of functions without computing  $P\{Q|E\}$ .

We will define a *query* to be a tuple consisting of a set of query variables, a set of evidence variables and, optionally, a function on the query variables.

One key point about this presentation: if there is a choice, I will show the Markov Net version of the algorithm. A Markov Net algorithm can always be used to solve the Markov Net associated with every belief network. Let  $B = (G, P)$  be a belief network.

The Markov Net associated with this belief network is  $M = (G^m, P)$ .

## 2.0 Vertex Elimination

The first algorithm that we will discuss is the *vertex elimination* algorithm [Zhang+Poole, 94]. The vertex elimination algorithm is similar in spirit to the D'Ambrosio's *Symbolic Probabilistic Inference* algorithm and Dechter's *bucket elimination* algorithm.

The input to the algorithm consists of a query  $(Q, E)$  and a Markov net  $B = (G = (X, L), P)$ . The algorithm computes  $P\{Q|E\}$  by incrementally removing (via marginalization) the variables in  $X \setminus (Q \cup E)$ .

### Instantiating Variables

Usually we are given specific values for the evidence variables. If a variable  $e$  is observed, we can simplify the corresponding distribution function or matrix to be a function of all of the variables other than  $e$ . So, we can replace each of the potentials of the

---

1. Alternative notation for expectation

form  $\phi(x_1, \dots, e, \dots, x_n)$  with a corresponding distribution

$\phi_e(x_1, \dots, x_n) = \phi(x_1, \dots, (e = v_e), \dots, x_n)$ . We will call this operation *instantiation with row reduction*.<sup>1</sup> Alternatively, we can add a vector to the Markov net to represent the likelihood of the evidence:  $\lambda(e)$ . If  $\lambda(e) = \delta(e, v_e) = [0, 0, \dots, 1, \dots, 0]$ , that is,  $\lambda$  consists of all zeros except for the  $i^{\text{th}}$  element  $v_e$ , we say that  $e$  has been *instantiated* to its  $i^{\text{th}}$  state.

The algorithm:

Let  $\Phi = \{\phi_1, \phi_2, \dots, \phi_n\}$  be the set of distributions in  $M$ .

Instantiate the distributions in  $\Phi$  that have evidence.

Let  $N = X \setminus (Q \cup E)$  if instantiation with row reduction is used.

Let  $N = X \setminus Q$  if instantiation without row reduction is used.

Let  $\alpha$  be a numbering on  $N$ .

for  $i$  from 1 to  $|N|$ , do

Remove the distributions in  $\Phi$  that contain  $\alpha(i)$ . Call this set  $\Phi'$ .

Let  $\phi_{\text{new}}(X) = \sum_{\alpha(i)} \prod_{\phi_i \in \Phi'} \phi_i(X)$ .

Add  $\phi_{\text{new}}$  to  $\Phi$ .

The answer is derived by multiplying all of the potentials left in  $\Phi$ .

A variant of the algorithm can determine the maximum probability instantiation for all of the variables in  $X \setminus E$ :

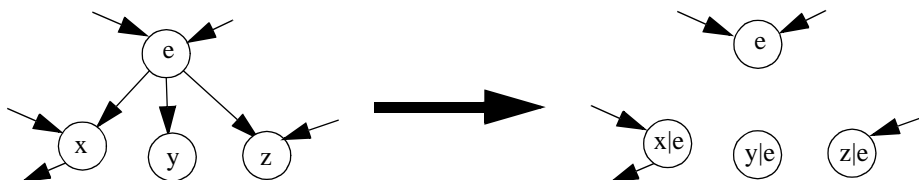
Let  $\Phi = \{\phi_1, \phi_2, \dots, \phi_n\}$  be the set of distributions in  $M$ .

Instantiate the distributions in  $\Phi$  that have evidence.

Let  $N = X \setminus E$  if row reduction  $N = X$  if no row reduction.

Let  $\alpha$  be a numbering on  $N$ .

1. Note that this actually implies a topology change in the belief network: Observing a node decouples that node from its children.



for  $i$  from 1 to  $|N|$ , do

1. Remove the distributions in  $\Phi$  that contain  $\alpha(i)$ . Call this set  $\Phi'$ .
2. Let  $\phi_{\text{new}}(X) = \max_{\alpha(i)} \prod_{\phi_i \in \Phi'} \phi_i(X)$ .
3. Add  $\phi_{\text{new}}$  to  $\Phi$ .

The maximum probability instantiation for the variables in  $X \setminus E$  given  $E$  is given by the

values for  $\alpha(i)$  that maximize the  $\prod_{\phi_i \in \Phi'} \phi_i(X)$  in line (2) above.

### 3.0 The Polytree Algorithm and Cutset Conditioning

The principal disadvantage of the vertex elimination algorithm is that it has to start from scratch to compute each query. Other algorithms, such as the polytree and join tree algorithms attempt to compute marginals for multiple variables simultaneously.

**Read:** Mark Peot and Ross Shachter, Fusion and propagation with multiple observations in belief networks. (Course Reader)

#### 3.1 Input to the Algorithm

The input to the polytree algorithm is a Bayes net  $B = (G = (X, L), P)$  and evidence  $E = e$ . The output is a set of marginals  $\{P\{X_1|e\}, \dots, P\{X_n|e\}\}$ .

#### 3.2 Recursive Conditioning

Note that we do not need to condition each node on every possible instantiation for the cutset variables. Each cutset variable is only relevant to the nodes on the loop that it cuts. This issue is explored in more detail by Javier Diez-Vegas (Recursive Conditioning) and Adnan Darwiche (Dynamic Conditioning).

### 4.0 The Junction Tree Algorithm

The junction tree algorithm (aka *Hugin* aka *Jensen Algorithm* aka *Cluster Tree Algorithm*) computes the joint distribution for each maximal clique in a decomposable graph.<sup>1</sup>

---

1. For those of you that care, the theory for junction trees is developed from a more general concept of cluster trees. I don't think cluster trees are very important so I am moving directly to junction trees.

## 4.1 Junction Graphs and Junction Trees

A *spanning tree* for a connected undirected graph  $G = (X, L)$  is a tree  $G' = (X, L')$  where  $L' \subseteq L$ .

A *junction graph* for an undirected graph  $G$  is an undirected, labelled graph  $J = (C, L)$ . The nodes  $C$  are the cliques in  $G$ . If the intersection of cliques  $C_i$  and  $C_j$  is nonempty then there is an arc between  $C_i$  and  $C_j$  in  $J$ . A *separator* is associated with each link  $C_i - C_j$  that consists of the nodes in  $C_i \cap C_j$ . The *weight* for the link between  $C_i$  and  $C_j$  is the number of nodes its separator. The *weight* of a spanning tree of a junction graph is the sum of the weights of its links.

A *junction tree* is a spanning tree of the junction graph with the property that for each pair of cliques  $C_i$  and  $C_j$ , all nodes on the path from  $C_i$  to  $C_j$  contain  $C_i \cap C_j$ .

**Thm 1:** (Jensen) A graph is triangulated iff its junction graph has a junction tree.

**Thm 2:** (Jensen) A subgraph  $J'$  of junction graph  $J$  of a decomposable graph is a junction tree iff  $J'$  is the spanning tree of maximal weight.

The spanning tree of maximal weight can be found using Kruskal's algorithm. Successively select links of the junction graph of maximal weight unless that link would create a cycle.

The *junction tree corresponding to a Markov Net*<sup>1</sup>  $U = (G = (X, L), \Phi)$  is constructed by:

1. Let  $G_d$  be the decomposable graph derived by triangulating  $G$ .
2. Form a junction tree  $J = (C, L')$  from a junction graph formed from the maximal cliques of  $G_d$ .
3. For each clique  $C_i$  in  $J$ , define a real numbered table  $\phi_C(C_i)$  (called a *potential*) with entries equal to one. Similarly define a table  $\phi_S(S_i)$  for each separator  $S_i$  in  $J$ .
4. For each potential  $\phi_i(X_i) \in \Phi$  select one node  $C_j$  from  $C$ , such that  $X_i \subseteq C_j$ . Multiply  $C_j$ 's table by  $\phi_i(X_i)$ .

---

1.

**Thm 3:** The joint distribution over the variables in  $X$  is equal to the product of the clique tables divided by the product of the separator tables as defined above:

$$P\{X\} = \frac{\prod_{C_i \in \mathcal{C}} \phi_C(C_i)}{\prod_{S_j \in \mathcal{S}} \phi_S(S_j)}. \quad (\text{EQ 2})$$

Conditional probabilities can be computed by multiplying the appropriate clique tables by an indicator function reflecting the observed value of each evidence variable. Let the unobserved variables of  $X$  be  $U = X \setminus E$ .

**Thm 4:**  $P\{U, E = e\} = P\{X\} \prod_{E_i \in E} \delta(E_i, e_i)$ .

## 4.2 Absorption

Absorption is an operation that modifies the clique and separator tables in such a way that the joint probability of the junction tree is preserved. Let  $C_i$  and  $C_j$  be two neighbors in a junction tree with separator  $S$ .

Define  $\text{div}(x, y)$  to be 0 when  $x$  and  $y$  are zero and let  $\text{div}(x, y) = x/y$ , otherwise.

Let

$$\phi_S^*(S) = \sum_{C_i \setminus S} \phi_C(C_i) \quad (\text{EQ 3})$$

$$\phi_C^*(C_j) = \phi_C(C_j) \text{div}(\phi_S^*(S), \phi_S(S)) \quad (\text{EQ 4})$$

Set the tables for  $C_j$  and  $S$  to these new values.

After this operation, we say that  $C_j$  has *absorbed* from  $C_i$ .

Absorption is possible only if  $\phi_S^*(S)$  is zero only where  $\phi_S(S)$  is zero. A link in the junction tree is *supportive* only if it allows absorption in both directions. A junction tree is supportive if all of its links are supportive.

**Thm 5:** (Jensen) Supportiveness is preserved under absorption.

**Thm 6:** (Jensen) If  $J$  is a supportive junction tree, Equation 2 is invariant under absorption.

### 4.3 Global consistency

Two neighboring cliques are *consistent* if their potentials do not change after absorption.

$$\text{That is } \sum_{C_i \setminus S} \phi_C(C_i) = \phi_S(S) = \sum_{C_j \setminus S} \phi_C(C_j).$$

A junction tree is *globally consistent* if  $\sum_{C_i \setminus I} \phi_C(C_i) = \sum_{C_j \setminus I} \phi_C(C_j)$  for any two cliques  $C_i$  and  $C_j$  with intersection  $I$ .

Consider the following message-passing scheme:  $C_i$  can only absorb from  $C_j$  if  $C_j$  has already absorbed from all of its neighbors except for  $C_i$ . We will call this message passing scheme *propagation*.

**Thm 7:** After propagation, the junction tree is globally consistent.

**Thm 8:** After propagation, every clique and separator potential in the junction tree for a Markov Net satisfies  $\sum_{X \setminus C_i} P\{X\} = \phi_C(C_i)$  and  $\sum_{X \setminus S_i} P\{X\} = \phi_S(S_i)$ .

Evidence  $E_i = e_i$  can be added to a junction tree by multiplying the potential for any clique containing  $E_i$  by  $\delta(E_i, e_i)$ , *instantiating*  $E_i = e_i$  in the clique potential.<sup>1</sup>

**Thm 9:** If we instantiate  $E = e$  in the junction tree for a Markov Net, after propagation every clique and separator potential satisfies  $\sum_{X \setminus C_i} P\{X, e\} = \phi_C(C_i, e)$  and

$$\sum_{X \setminus S_i} P\{X, e\} = \phi_S(S_i, e).$$

---

1. Alternatively, we can multiply the clique potentials by evidence likelihoods.

#### **4.4 Implementation issues**

### **5.0 Deriving Good Triangulations**

#### **5.1 Triangulation is necessary: A weak argument.**

Cache argument.

#### **5.2 Cutset Conditioning triangulates the graph.**

Rewrite the sum.

#### **5.3 The min fill heuristic**

#### **5.4 The Geiger/Becker Triangulation Algorithm**

## **6.0 Complexity of Exact Inference**

### **6.1 Introduction to complexity**

### **6.2 Reduction to 3SAT**

### **6.3 3SAT**

### **6.4 What does this mean?**

## **7.0 Gaussian Models**