Uncertainty

Inference

Introduction to Artificial Intelligence

Chandra Gummaluru

University of Toronto

Version W22.1

- The following is based on material developed by many individuals, including (but not limited to):
  - Sheila McIlraith
  - Bahar Aameri
  - Fahiem Bacchus
  - Sonya Allin

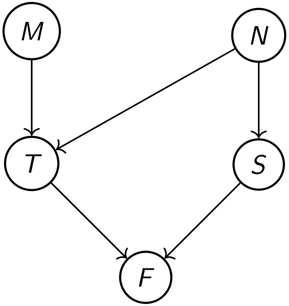
- Given a set of related random variables, $\{X_1, \dots, X_n\}$, we often wish to compute

$$P(Q|E) := P(Q_1, \dots, Q_v|E_1, \dots, E_w),$$

where $Q = \{Q_1, \dots, Q_v\}$, $E = \{E_1, \dots, E_w\} \subseteq \{X_1, \dots, X_n\}$, and $Q \cap E = \emptyset$.

Example: Catching a Flight

- We defined a Bayesian network over $\{F, T, S, M, N\}$, where:
  - F is whether we catch the flight or not
  - T is when we get to the airport
  - S is how long it takes to get through security
  - M is the method of transport we choose
  - N is how many bags we have
- We seek the probability of catching the flight given the method of transport, i.e., $P(F|M)$.



- We can express $P(Q|E)$ in terms of the joint distribution of $X_1, \ldots, X_n$. We have

$$P(Q|E) = \frac{P(Q_1, \ldots, Q_v, E_1, \ldots, E_w)}{P(E_1, \ldots, E_w)}$$

$$= \frac{P(Q_1, \ldots, Q_v, E_1, \ldots, E_w)}{\sum_{Q_i} \sum_{\mathrm{dom}(Q_i)} P(Q_1, \ldots, Q_v, E_1, \ldots, E_w)}$$
$$= \frac{\sum_{X_i \notin Q \cap E} \sum_{\mathrm{dom}(X_i)} P(X_1, \ldots, X_n)}{\sum_{X_i \notin E} \sum_{\mathrm{dom}(X_i)} P(X_1, \ldots, X_n)}$$

Simplifying the Joint Distribution with Conditional Independence

- In general, the joint distribution can be written as

$$P(X_1, \ldots, X_n) = P(X_1) \prod_{i \neq 1} P(X_i | X_1, \ldots, X_{i-1}).$$

- We saw that if $X_1, \ldots, X_n$ are expressed as a Bayesian network, then X_i is independent of its non-descendants given its parents, i.e.,

$$P(X_i | S \cup \text{pts}(X_i)) = P(X_i | \text{pts}(X_i)),$$

where $\text{pts}(X_i)$ are the parents of X and S is a subset of X_i's non-descendants.

Simplifying the Joint Distribution with Conditional Independence

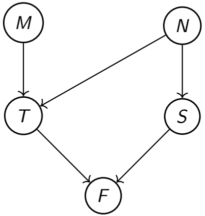
- Since a Bayesian network must be acyclic, if we also assume it to be finite:
  - it must contain at least one node without any parents,
  - no node can be an ancestor and descendant of another node
- Thus, we can assume without loss of generality that $X_1, \ldots, X_n$ ordered such that if X_j is a descendant of X_i, then $j > i$.
- In other words, we can assume that $X_1, \ldots, X_{i-1}$ are not descendants of X_i.
- Therefore, we can write

$$P(X_1, \ldots, X_n) = \prod_i P(X_i | \text{pts} X_i).$$

- **Example:** Catching a Flight
  - Suppose $\text{dom}(F) = \{\text{yes}, \text{no}\}$, $\text{dom}(T) = \{\text{early}, \text{late}\}$, $\text{dom}(S) = \{\text{fast}, \text{slow}\}$, $\text{dom}(M) = \{\text{train}, \text{car}\}$, and $\text{dom}(N) = \{0, 1, 2\}$.
  - The probability of catching the flight if we take the train is $P(F = \text{yes} | M = \text{train})$.

$P(M = \text{train})$
0.6

M	N	$P(T = \text{early} \mid M, N)$
train	0	0.95
train	1	0.8
train	2	0.65
car	0	0.7
car	1	0.7
car	2	0.7



$P(N = 0)$	$P(N = 1)$
0.4	0.5

N	$P(S = \text{fast} \mid N)$
0	0.9
1	0.8
2	0.7

T	S	$P(F = \text{yes} \mid T, S)$
early	fast	0.9
early	slow	0.7
late	fast	0.7
late	slow	0.3

- We first express the probability in terms of the joint distribution, i.e.,

$$P(F = \text{yes} | M = \text{train}) = \frac{P(F = \text{yes}, M = \text{train})}{\sum_{\forall F} P(F, M = \text{train})}$$

where

$$P(F, M = \text{train}) = \sum_{\forall N, T, S} P(M = \text{train}, N, T, S, F)$$

$$= \sum_{\forall N} \sum_{\forall T} \sum_{\forall S} P(M = \text{train}) P(N) P(T | M = \text{train}, N) P(S|N) P(F|T, S)$$

$$= P(M = \text{train}) \underbrace{\sum_{\forall N} P(N) \underbrace{\sum_{\forall T} P(T | M = \text{train}, N) \underbrace{\sum_{\forall S} P(F|T,S) P(S|N)}_{g_1(N,F,T)}}_{g_2(N,F)}}_{g_3(F)}$$

- First, we compute

$$g_1(N, F, T) = P(F|T, S = \text{fast})P(S = \text{fast}|N) + P(F|T, S = \text{slow})P(S = \text{slow}|N)$$

N	F	T	$g_1(N, F, T)$
0	yes	early	$0.9 \times 0.9 + 0.7 \times 0.1 = 0.88$
0	yes	late	$0.7 \times 0.9 + 0.3 \times 0.1 = 0.66$
0	no	early	$0.1 \times 0.9 + 0.3 \times 0.1 = 0.12$
0	no	late	$0.3 \times 0.9 + 0.7 \times 0.1 = 0.34$
1	yes	early	$0.9 \times 0.8 + 0.7 \times 0.2 = 0.86$
1	yes	late	$0.7 \times 0.8 + 0.3 \times 0.2 = 0.62$
1	no	early	$0.1 \times 0.8 + 0.3 \times 0.2 = 0.14$
1	no	late	$0.3 \times 0.8 + 0.7 \times 0.2 = 0.38$
2	yes	early	$0.9 \times 0.7 + 0.7 \times 0.3 = 0.84$
2	yes	late	$0.7 \times 0.7 + 0.3 \times 0.3 = 0.58$
2	no	early	$0.1 \times 0.7 + 0.3 \times 0.3 = 0.16$
2	no	late	$0.3 \times 0.7 + 0.7 \times 0.3 = 0.42$

- Next, we compute

$$g_2(N, F) = P(T = \text{early}|M = \text{train}, N)g_1(N, F, T = \text{early}) \\ + P(T = \text{late}|M = \text{train}, N)g_1(N, F, T = \text{late})$$

N	F	$g_2(N, F)$
0	yes	$0.95 \times 0.88 + 0.05 \times 0.66 = 0.869$
0	no	$0.95 \times 0.12 + 0.05 \times 0.34 = 0.131$
1	yes	$0.80 \times 0.86 + 0.20 \times 0.62 = 0.821$
1	no	$0.80 \times 0.14 + 0.20 \times 0.38 = 0.188$
2	yes	$0.65 \times 0.84 + 0.35 \times 0.58 = 0.749$
2	no	$0.65 \times 0.16 + 0.35 \times 0.42 = 0.251$

- We compute

$$g_3(F) = P(N=0)g_2(N=0,F) + P(N=1)f^{(2)}(N=1,F) + P(N=2)g_2(N=2,F)$$

F	$g_3(F)$
yes	$0.400 \times 0.869 + 0.500 \times 0.821 + 0.100 \times 0.749 = 0.8258$
no	$0.400 \times 0.131 + 0.500 \times 0.188 + 0.100 \times 0.251 = 0.1715$

- Finally, we compute

$$P(F, M = \text{train}) = P(M = \text{train}) f_3(F)$$

$$= \begin{cases} 0.6 \times 0.8258, F = \text{yes} \\ 0.6 \times 0.1715, F = \text{no} \end{cases}$$

$$= \begin{cases} 0.4971, F = \text{yes} \\ 0.1029, F = \text{no} \end{cases}$$

- Marginalizing the joint distribution, we have

$$P(F = \text{yes} | M = \text{train}) = \frac{0.4971}{0.4971 + 0.1029} = \frac{0.4971}{0.6} = 0.8285.$$

- We can similarly compute $P(F = \text{yes} | M = \text{car}) = 0.7958$.

- Let us now formalize the previous procedure into an algorithm.

- We begin by formalizing the concept of factors from earlier.

- A **factor**, f, is a function such that:
  - the scope of the factor, denoted $\text{scp}(f)$ is the set of variables it involves.
  - the input any assignment of f's scope, i.e., $\{X_j = x_j, \forall X_j \in \text{scp}(f)\}$.
  - The output is a real number.
- There are three operations on factors of particular interest:
  1. restrictions
  2. marginalizations
  3. products

- For any factor, f, any $K \subseteq \mathrm{scp}(f)$ and $k \in \prod_{K_i \in K} \mathrm{dom}(K_i)$ let $f_{K=k}$ denote the **restriction** of f under $K = k$, defined so that $\mathrm{scp}(f_{K=k}) = \mathrm{scp}(f) \setminus K$ and

$$f_{K=k}\left(\{X_i = x_i, \forall X_i \in \mathrm{scp}(f_{K=k})\}\right) = f\left(\{X_i = x_i, \forall X_i \in \mathrm{scp}(f_{K=k})\} \cup \{K_i = k_i, \forall K_i \in K\}\right)$$

X	Y	Z	$f(X, Y, Z)$
T	T	T	0.10
T	T	F	0.08
T	F	T	0.35
T	F	F	0.14
F	T	T	0.15
F	T	F	0.12
F	F	T	0.05
F	F	F	0.02

X	Y	$f_{Z=\mathsf{T}}(X, Y)$
T	T	0.10
T	F	0.35
F	T	0.15
F	F	0.05

- For any factor, f and $Z \in \text{scp}(f)$, let $f_{\sum Z}$ denote the **marginalization** of f under Z, defined so that $\text{scp}(f_{\sum Z}) = \text{scp}(f) \setminus Z$, and

$$f_{\sum Z}\left(\{X_i = x_i, \forall X_i \in \text{scp}(f_{\sum Z})\}\right) = \sum_{\forall z \in \text{dom}(Z)} f\left(\{X_i = x_i, \forall X_i \in \text{scp}(f_{\sum Z})\} \cup \{Z = z\}\right)$$

X	Y	Z	$f(X, Y, Z)$
T	T	T	0.10
T	T	F	0.08
T	F	T	0.35
T	F	F	0.14
F	T	T	0.15
F	T	F	0.12
F	F	T	0.05
F	F	F	0.02

X	Y	$f_{\sum Z}(X, Y)$
T	T	0.18
T	F	0.49
F	T	0.27
F	F	0.07

- For any pair of factor, $f^{(1)}$ and $f^{(2)}$, let $f^{(1)}f^{(2)}$ denote their **product** of f under $K = k$, defined so that $\mathrm{scp}(f^{(1)}f^{(2)}) = \mathrm{scp}(f^{(1)}) \cup \mathrm{scp}(f^{(2)})$ and

$$\begin{aligned} f^{(1)}f^{(2)} &\left(\left\{X_i = x_i, \forall X_i \in \mathrm{scp}(f^{(1)}f^{(2)})\right\}\right) \\ &= f^{(1)}\left(\left\{X_i = x_i, \forall X_i \in \mathrm{scp}(f^{(1)})\right\}\right) f^{(2)}\left(\left\{X_i = x_i, \forall X_i \in \mathrm{scp}(f^{(2)})\right\}\right) \end{aligned}$$

X	Y	$f^{(1)}(X, Y)$
T	T	0.2
T	F	0.7
F	T	0.3
F	F	0.1

Y	Z	$f^{(2)}(Y, Z)$
T	T	0.5
T	F	0.4
F	T	0.5
F	F	0.2

X	Y	Z	$f^{(1)}f^{(2)}(X, Y, Z)$
T	T	T	0.10
T	T	F	0.08
T	F	T	0.35
T	F	F	0.14
F	T	T	0.15
F	T	F	0.12
F	F	T	0.05
F	F	F	0.02

- We seek $P(Q|E)$ for some $Q, E \subseteq \{X_1, \dots, X_n\}$, where $Q \cap E = \emptyset$.

- It is sufficient to find $P(Q, E)$. We proceed as follows:

  1. Define a set of factors, $F = \left\{f^{(1)}, \dots, f^{(n)}\right\}$, where $f^{(i)} = P(X_i|\text{pts}(X_i))$.
  2. For each $f \in F_0$ such that $\text{scp}(f) \cap E \neq \emptyset$, replace f with a restriction of $\text{scp}(f) \cap E$.
  3. Eliminate Z_i, for each $Z_i \notin Q \cup E$ as follows:
     a. Define $F(Z_i) := \{f : Z_i \in \text{scp}(f)\}$
     b. Replace F with $F \setminus F(Z_i)$.
     c. Compute and add the following factor to F:

$$\sum_{z_i \in \text{dom}(Z_i)} \prod_{f \in F(Z_i)} f$$

  4. We can now compute $P(Q, E) = \prod_{f \in F} f$.

- The order in which we eliminate the variables can have a significant impact on the size of the resulting factors.

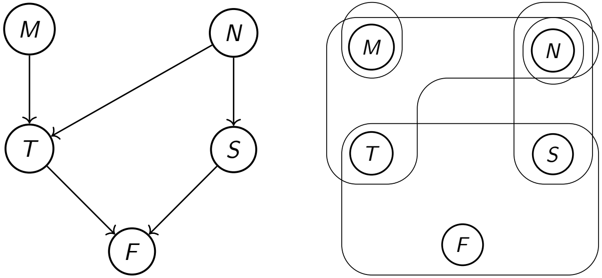
- **Example**: Catching a Flight
  - Using the elimination ordering, S, T, N, we needed to compute the factors:
    - $g_1(N, F, T)$, whose size is $|\text{dom}(N)| \times |\text{dom}(F)| \times |\text{dom}(T)| = 3 \times 2 \times 2 = 12$.
    - $g_2(N, F)$, whose size is $|\text{dom}(N)| \times |\text{dom}(F)| = 3 \times 2 = 6$
    - $g_3(F)$, whose size is $|\text{dom}(F)| = 2$.
  - If we had instead used the elimination ordering, N, T, S, we would need to compute the factors:
    - $g_1(S, T)$, whose size is $|\text{dom}(S)| \times |\text{dom}(T)| = 2 \times 2 = 4$.
    - $g_2(S, F)$, whose size is $|\text{dom}(S)| \times |\text{dom}(F)| = 2 \times 2 = 4$.
    - $g_3(F)$, whose size is $|\text{dom}(F)| = 2$.
  - We would have halved the total entries.

- Let us formally analyze the complexity of the variable elimination algorithm.
- To do this, we will need to introduce the concept of a hyper-graph.
- A **hyper-graph** is a generalization of a graph in which the edges, called hyper-edges, can contain more than two vertices.

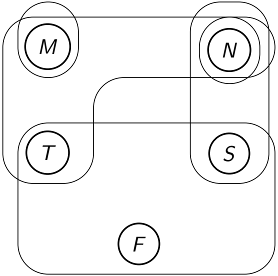
- Whenever a variable, X, is eliminated, the resulting factor's scope consists of:
    - X's children
    - X's parents
    - the other parents of X's children, not including X itself
- We refer to these variables as X's **Markov blanket**, denoted $\text{mbk}(X)$.
- On a hyper-graph, this is equivalent to taking all hyper-edges that include X, replacing them with their union minus X, and then removing X.

Example: Catching a Flight

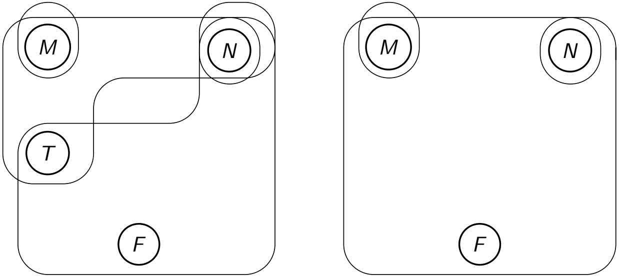
- The Bayesian network on the left can be represented by the hyper-graph on the right.



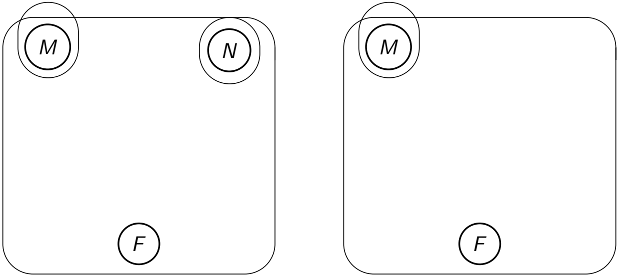
- To eliminate S, we first replace the hyper-edges $\{S, N\}, \{F, T, S\}$ with $\{F, T, N\}$ and then remove S from the hyper-graph.



- To eliminate T, we first replace the hyper-edges $\{F, T, N\}, \{T, M, N\}$ with $\{F, N, M\}$ and then remove T from the hyper-graph.



- To eliminate N, we first replace the hyper-edges $\{N\}, \{F, M, N\}$ with $\{F, M\}$ and then remove N from the hyper-graph.



- The resulting hyper-edges will only include M and F.

- We see that variable elimination creates a sequence of hyper-edges that depends on the elimination ordering.
- The elimination width, k of the sequence is the maximum number of variables over all hyper-edges.
- Thus, variable elimination must create and store a factor whose number of entries is exponential in k, i.e., the time/space complexity is $O(D^k)$, where D is the maximum cardinality over the variables' domains.
- Finding the optimal ordering, i.e., the one that minimizes the elimination width is an NP-Hard problem. Thus, we often use heuristics instead.

- We present some commonly used heuristics:
  1. Eliminate the variable with the fewest parents.
  2. Eliminate the variable whose parent set has the smallest domain, where

$$|\mathrm{dom}(\mathrm{pnt}(X))| = \prod_{Z \in \mathrm{pnt}(X)} |\mathrm{dom}(Z)|.$$

  3. Eliminate the variable with the smallest Markov blanket
  4. Eliminate the variable whose Markov blanket has the smallest domain, where

$$|\mathrm{dom}(\mathrm{mbk}(X))| = \prod_{Z \in \mathrm{mbk}(X)} |\mathrm{dom}(Z)|.$$