Modeling uncertainty with probabilities

• **Knowledge based system era (70s – early 80’s)**
  – **Extensional non-probabilistic models**
    – Solve the space, time and acquisition bottlenecks in probability-based models
    – froze the development and advancement of KB systems and contributed to the slow-down of AI in 80s in general

• Breakthrough (late 80s, beginning of 90s)
  – **Bayesian belief networks**
    • Give solutions to the space, acquisition bottlenecks
    • Partial solutions for time complexities
  • Bayesian belief network
Bayesian belief networks (BBNs)

**Bayesian belief networks.**
- Represent the full joint distribution over the variables more compactly with a **smaller number of parameters**.
- Take advantage of **conditional and marginal independences** among random variables

- **A and B are independent**
  \[ P(A, B) = P(A)P(B) \]
- **A and B are conditionally independent given C**
  \[
  P(A, B | C) = P(A | C)P(B | C) \\
  P(A | C, B) = P(A | C)
  \]

Bayesian belief networks (general)

Two components: \( B = (S, \Theta_S) \)

- **Directed acyclic graph**
  - Nodes correspond to random variables
  - (Missing) links encode independences

- **Parameters**
  - Local conditional probability distributions for every variable-parent configuration

Where:
- \( pa(X_i) \) - stand for parents of \( X_i \)

<table>
<thead>
<tr>
<th>B</th>
<th>E</th>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>0.95</td>
<td>0.05</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>0.94</td>
<td>0.06</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>0.29</td>
<td>0.71</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>0.001</td>
<td>0.999</td>
</tr>
</tbody>
</table>
Bayesian belief network.

\[
\begin{array}{c|c|c|c}
\text{Burglary} & T & F \\
0.001 & 0.999 \\
\end{array}
\quad
\begin{array}{c|c|c|c}
\text{Earthquake} & T & F \\
0.002 & 0.998 \\
\end{array}
\quad
\begin{array}{c|c|c|c|c|c}
\text{Alarm} & \text{B} & \text{E} & T & F \\
& & & 0.95 & 0.05 \\
& & & 0.94 & 0.06 \\
& & & 0.29 & 0.71 \\
& & & 0.001 & 0.999 \\
\end{array}
\quad
\begin{array}{c|c|c|c|c|c}
\text{JohnCalls} & \text{A} & \text{T} & \text{F} \\
& 0.90 & 0.1 \\
& 0.05 & 0.95 \\
\end{array}
\quad
\begin{array}{c|c|c|c|c|c}
\text{MaryCalls} & \text{M} & \text{A} & \text{T} & \text{F} \\
& 0.7 & 0.3 \\
& 0.01 & 0.99 \\
\end{array}
\]

Full joint distribution in BBNs

**Full joint distribution** is defined in terms of local conditional distributions (obtained via the chain rule):

\[
P(X_1, X_2, \ldots, X_n) = \prod_{i=1, \ldots, n} P(X_i \mid \text{pa}(X_i))
\]

**Example:**
Assume the following assignment of values to random variables:
\[B=T, E=T, A=T, J=T, M=F\]

Then its probability is:
\[
\]
Bayesian belief networks (BBNs)

Bayesian belief networks
• Represent the full joint distribution over the variables more compactly using the product of local conditionals.
• But how did we get to local parameterizations?

Answer:
• **Graphical structure** encodes **conditional and marginal independences** among random variables

  • **A and B are independent** \[ P(A, B) = P(A)P(B) \]
  • **A and B are conditionally independent given C**
    \[ P(A | C, B) = P(A | C) \]
    \[ P(A, B | C) = P(A | C)P(B | C) \]
  • **The graph structure implies the decomposition !!!**

Independences in BBNs

3 basic independence structures:

1. [Diagram of independence structure 1]
2. [Diagram of independence structure 2]
3. [Diagram of independence structure 3]
Independences in BBN

- BBN distribution models many conditional independence relations among distant variables and sets of variables
- These are defined in terms of the graphical criterion called d-separation
- **D-separation and independence**
  - Let \( X, Y \) and \( Z \) be three sets of nodes
  - If \( X \) and \( Y \) are d-separated by \( Z \), then \( X \) and \( Y \) are conditionally independent given \( Z \)
- **D-separation**
  - \( A \) is d-separated from \( B \) given \( C \) if every undirected path between them is blocked with \( C \)
- **Path blocking**
  - 3 cases that expand on three basic independence structures

Independences in BBNs

- Earthquake and Burglary are independent given MaryCalls \( F \)
- Burglary and MaryCalls are independent (not knowing Alarm) \( F \)
- Burglary and RadioReport are independent given Earthquake \( T \)
- Burglary and RadioReport are independent given MaryCalls \( F \)
Bayesian belief networks (BBNs)

**Bayesian belief networks**
- Represents the full joint distribution over the variables more compactly using the product of local conditionals.
- So how did we get to local parameterizations?

\[ P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i \mid pa(X_i)) \]

- The decomposition is implied by the set of independences encoded in the belief network.

---

**Full joint distribution in BBNs**

Rewrite the full joint probability using the product rule:

\[ P(B=T, E=T, A=T, J=T, M=F) = \]

\[ = P(J=T \mid B=T, E=T, A=T, M=F)P(B=T, E=T, A=T, M=F) \]

\[ = \underbrace{P(J=T \mid A=T)} P(B=T, E=T, A=T, M=F) \]

\[ P(M=F \mid B=T, E=T, A=T)P(B=T, E=T, A=T) \]

\[ P(M=F \mid A=T)P(B=T, E=T, A=T) \]

\[ P(A=T \mid B=T, E=T)P(B=T, E=T) \]

\[ = P(J=T \mid A=T)P(M=F \mid A=T)P(A=T \mid B=T, E=T)P(B=T)P(E=T) \]
Bayesian belief network.

- In the BBN the **full joint distribution** is expressed using a set of local conditional distributions.

\[ P(B) \]

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.001</td>
<td>0.999</td>
</tr>
</tbody>
</table>

\[ P(E) \]

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.002</td>
<td>0.998</td>
</tr>
</tbody>
</table>

\[ P(A|B,E) \]

<table>
<thead>
<tr>
<th></th>
<th>B</th>
<th>E</th>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>0.95</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>0.94</td>
<td>0.06</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>0.29</td>
<td>0.71</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>0.001</td>
<td>0.999</td>
<td></td>
</tr>
</tbody>
</table>

\[ P(J|A) \]

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.90</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>0.05</td>
<td>0.95</td>
<td></td>
</tr>
</tbody>
</table>

\[ P(M|A) \]

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.7</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>0.01</td>
<td>0.99</td>
<td></td>
</tr>
</tbody>
</table>

Parameter complexity problem

- In the BBN the **full joint distribution** is defined as:
  \[ P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | pa(X_i)) \]

- **What did we save?**
  
  Alarm example: 5 binary (True, False) variables

  \# of parameters of the full joint:
  \[ 2^5 = 32 \]
  
  One parameter is for free:
  \[ 2^5 - 1 = 31 \]

  \# of parameters of the BBN:
  \[ 2^3 + 2(2^2) + 2(2) = 20 \]

  One parameter in every conditional is for free:
  \[ 2^2 + 2(2) + 2(1) = 10 \]
Model acquisition problem

The structure of the BBN
• typically reflects causal relations
  (BBNs are also sometime referred to as causal networks)
• Causal structure is intuitive in many applications domain and it is relatively easy to define to the domain expert

Probability parameters of BBN
• are conditional distributions relating random variables and their parents
• Complexity is much smaller than the full joint
• It is much easier to obtain such probabilities from the expert or learn them automatically from data

BBNs built in practice
• In various areas:
  – Intelligent user interfaces (Microsoft)
  – Troubleshooting, diagnosis of a technical device
  – Medical diagnosis:
    • Pathfinder (Intellipath)
    • CPSC
    • Munin
    • QMR-DT
  – Collaborative filtering
  – Military applications
  – Business and finance
    • Insurance, credit applications
Inference

•

Inference in Bayesian networks

• BBN models compactly the full joint distribution by taking advantage of existing independences between variables
• Simplifies the acquisition of a probabilistic model
• But we are interested in solving various inference tasks:
  – Diagnostic task. (from effect to cause)
    \[ P(\text{Burglary} \mid \text{JohnCalls} = T) \]
  – Prediction task. (from cause to effect)
    \[ P(\text{JohnCalls} \mid \text{Burglary} = T) \]
  – Other probabilistic queries (queries on joint distributions).
    \[ P(\text{Alarm}) \]
• Main issue: Can we take advantage of independences to construct special algorithms and speeding up the inference?
Inference in Bayesian networks

• **Bad news:**
  – Exact inference problem in BBNs is NP-hard (Cooper)
  – Approximate inference is NP-hard (Dagum, Luby)
• **But** very often we can achieve significant improvements
• Assume our Alarm network

• Assume we want to compute: \( P(J = T) \)

---

Inference in Bayesian networks

**Computing:** \( P(J = T) \)

**Approach 1. Blind approach.**

- Sum out all un-instantiated variables from the full joint,
- express the joint distribution as a product of conditionals

\[
P(J = T) = \\
= \sum_{b \in T, F} \sum_{e \in T, F} \sum_{a \in T, F} \sum_{m \in T, F} P(B = b, E = e, A = a, J = T, M = m) \\
= \sum_{b \in T, F} \sum_{e \in T, F} \sum_{a \in T, F} \sum_{m \in T, F} P(J = T | A = a) P(M = m | A = a) P(A = a | B = b, E = e) P(B = b) P(E = e)
\]

**Computational cost:**
- Number of additions: ?
- Number of products: ?
Inference in Bayesian networks

Computing: \( P(J = T) \)


- Sum out all un-instantiated variables from the full joint,
- express the joint distribution as a product of conditionals

\[
P(J = T) = \\
= \sum_{b \in T, F} \sum_{e \in T, F} \sum_{a \in T, F} \sum_{m \in T, F} P(B = b, E = e, A = a, J = T, M = m) \\
= \sum_{b \in T, F} \sum_{e \in T, F} \sum_{a \in T, F} \sum_{m \in T, F} P(J = T | A = a) P(M = m | A = a) P(A = a | B = b, E = e) P(B = b) P(E = e)
\]

Computational cost:

Number of additions: 15
Number of products: 64
Inference in Bayesian networks

Approach 2. Interleave sums and products

- Combines sums and product in a smart way (multiplications by constants can be taken out of the sum)

\[ P(J = T) = \]

\[ = \sum_{b \in T,F} \sum_{o \in T,F} \sum_{m \in T,F} P(J = T | A = a) P(M = m | A = a) P(A = a | B = b, E = e) P(B = b) P(E = e) \]

\[ = \sum_{b \in T,F} \sum_{o \in T,F} \sum_{m \in T,F} P(J = T | A = a) P(M = m | A = a) P(B = b) \left[ \sum_{o \in T,F} P(A = a | B = b, E = e) P(E = e) \right] \]

\[ = \sum_{m \in T,F} P(M = m | A = a) \left[ \sum_{b \in T,F} P(B = b) \left[ \sum_{o \in T,F} P(A = a | B = b, E = e) P(E = e) \right] \right] \]

**Computational cost:**

Number of additions: \(1 + 2 \times [1 + 1 + 2 \times 1] = 9\)

Number of products: \(2 \times [2 + 2 \times (1 + 2 \times 1)]\)
Inference in Bayesian networks

Approach 2. Interleave sums and products

- Combines sums and product in a smart way (multiplications by constants can be taken out of the sum)

\[ P(J = T) = \]
\[ = \sum_{\text{bc}, \text{tc}, \text{sc}} \sum_{\text{bc}, \text{tc}, \text{sc}} \sum_{\text{mc}, \text{fc}} P(J = T | A = a)P(M = m | A = a)P(A = a | B = b, E = e)P(B = b)P(E = e) \]
\[ = \sum_{\text{bc}, \text{tc}, \text{sc}} \sum_{\text{bc}, \text{tc}, \text{sc}} \sum_{\text{mc}, \text{fc}} P(J = T | A = a)P(M = m | A = a)P(B = b)\left( \sum_{\text{bc}, \text{tc}, \text{sc}} P(A = a | B = b, E = e)P(E = e) \right) \]
\[ = \sum_{\text{bc}, \text{tc}, \text{sc}} P(J = T | A = a)\left( \sum_{\text{mc}, \text{fc}} P(M = m | A = a) \right)\left( \sum_{\text{bc}, \text{tc}, \text{sc}} P(B = b) \right)\left( \sum_{\text{bc}, \text{tc}, \text{sc}} P(A = a | B = b, E = e)P(E = e) \right) \]

**Computational cost:**

Number of additions: 1+2*\(1+1+2*1\)=9

Number of products: 2*[2+2*(1+2*1)]=16

---

Inference in Bayesian networks

- The smart interleaving of sums and products can help us to speed up the computation of joint probability queries

- What if we want to compute: \( P(B = T, J = T) \)

\[ P(B = T, J = T) = \]
\[ = \sum_{\text{mc}, \text{fc}} P(J = T | A = a)\left( \sum_{\text{mc}, \text{fc}} P(M = m | A = a) \right)\left( \sum_{\text{bc}, \text{tc}, \text{sc}} P(B = T) \right)\left( \sum_{\text{bc}, \text{tc}, \text{sc}} P(A = a | B = T, E = e)P(E = e) \right) \]

\[ P(J = T) = \]
\[ = \sum_{\text{mc}, \text{fc}} P(J = T | A = a)\left( \sum_{\text{mc}, \text{fc}} P(M = m | A = a) \right)\left( \sum_{\text{bc}, \text{tc}, \text{sc}} P(B = b) \right)\left( \sum_{\text{bc}, \text{tc}, \text{sc}} P(A = a | B = b, E = e)P(E = e) \right) \]

- A lot of shared computation
  - Smart cashing of results can save the time for more queries
Inference in Bayesian networks

- The smart interleaving of sums and products can help us to speed up the computation of joint probability queries.
- What if we want to compute: \( P(B = T, J = T) \)

\[
P(B = T, J = T) = 
\sum_{a \in F} P(J = T | A = a) \left[ \sum_{m \in F} P(M = m | A = a) \right] P(B = T) \left[ \sum_{k \in F} P(A = k | B = T, E = e) P(E = e) \right]
\]

\[
P(J = T) = 
\sum_{a \in F} P(J = T | A = a) \left[ \sum_{m \in F} P(M = m | A = a) \right] \sum_{k \in F} P(B = k) \left[ \sum_{e \in F} P(A = e | B = b, E = e) P(E = e) \right]
\]

- A lot of shared computation
  - Smart cashing of results can save the time if more queries

Inference in Bayesian networks

- When cashing of results becomes handy?
- What if we want to compute a diagnostic query:

\[
P(B = T | J = T) = \frac{P(B = T, J = T)}{P(J = T)}
\]

- Exactly probabilities we have just compared!!
- There are other queries when cashing and ordering of sums and products can be shared and saves computation

\[
P(B | J = T) = \frac{P(B, J = T)}{P(J = T)} = \alpha P(B, J = T)
\]

- General technique: Variable elimination
Inference in Bayesian networks

- General idea of variable elimination

\[
P(\text{True}) = 1 = \\
\sum_{a \in F} \left[ \sum_{j \in J} P(J = j | A = a) \right] \left[ \sum_{m \in M} P(M = m | A = a) \right] \left[ \sum_{b \in B} P(B = b) \right] \left[ \sum_{e \in E} P(E = e) \right] \\
\]

\[
\begin{align*}
& f_J(a) \\
& f_M(a) \\
& f_E(a,b) \\
& f_B(a) \\
& 
\end{align*}
\]

Variable order:

![Diagram showing variable order with nodes J, M, B, E and edges connecting them.]

Results cashed in the tree structure

Inference in Bayesian network

- **Exact inference algorithms:**
  - Variable elimination
  - Symbolic inference (D’Ambrosio)
  - Recursive decomposition (Cooper)
  - Message passing algorithm (Pearl)
  - Clustering and joint tree approach (Lauritzen, Spiegelhalter)
    - Arc reversal (Olmsted, Schachter)

- **Approximate inference algorithms:**
  - Monte Carlo methods:
    - Forward sampling, Likelihood sampling
    - Variational methods
Monte Carlo approaches

- **MC approximation:**
  - The probability is approximated using sample frequencies
  - Example:
    \[ \tilde{P}(B = T, J = T) = \frac{N_{B=T,J=T}}{N} \]
    
    \# samples with \( B = T, J = T \)
    total \# samples

- **BBN sampling:**
  - Generate sample in a top down manner, following the links
  - One sample gives one assignment of values to all variables

---

BBN sampling example

- **P(B):**
  - T: 0.001, F: 0.999

- **P(E):**
  - T: 0.002, F: 0.998

- **P(A|B,E):**
  - T T: 0.95, T F: 0.94, F T: 0.29, F F: 0.001

- **P(J|A):**
  - A T: 0.90, A F: 0.05

- **P(M|A):**
  - A T: 0.7, A F: 0.01
BBN sampling example

<table>
<thead>
<tr>
<th>Event</th>
<th>P(B)</th>
<th>P(E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.001</td>
<td>0.002</td>
</tr>
<tr>
<td>F</td>
<td>0.999</td>
<td>0.998</td>
</tr>
</tbody>
</table>

| Condition | P(A|B,E) | P(J|A) | P(M|A) |
|-----------|---------|-------|--------|
| B: T E: T | 0.95   | 0.90  | 0.7    |
| B: T E: F | 0.05   | 0.1   | 0.3    |
| B: F E: T | 0.94   | 0.05  | 0.01   |
| B: F E: F | 0.06   | 0.95  | 0.99   |

| Condition | P(A|B,E) | P(J|A) | P(M|A) |
|-----------|---------|-------|--------|
| B: T E: T | 0.95   | 0.90  | 0.7    |
| B: T E: F | 0.05   | 0.1   | 0.3    |
| B: F E: T | 0.94   | 0.05  | 0.01   |
| B: F E: F | 0.06   | 0.95  | 0.99   |

CS 2710 Foundations of AI
### BBN sampling example

**Table 1:** Probabilities of B, E, A, J, and M conditions.

<table>
<thead>
<tr>
<th></th>
<th>B (T)</th>
<th>B (F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E (T)</td>
<td>0.95</td>
<td>0.05</td>
</tr>
<tr>
<td>E (F)</td>
<td>0.94</td>
<td>0.06</td>
</tr>
<tr>
<td>J (T)</td>
<td>0.90</td>
<td>0.10</td>
</tr>
<tr>
<td>J (F)</td>
<td>0.05</td>
<td>0.95</td>
</tr>
<tr>
<td>M (T)</td>
<td>0.70</td>
<td>0.30</td>
</tr>
<tr>
<td>M (F)</td>
<td>0.01</td>
<td>0.99</td>
</tr>
</tbody>
</table>

**Figure:** A Bayesian network diagram showing the relationships between Burglary, Earthquake, Alarm, JohnCalls, and MaryCalls.
BBN sampling example

Monte Carlo approaches

- **MC approximation of conditional probabilities:**
  - The probability is approximated using sample frequencies
  - **Example:**
    \[
    \tilde{P}(B = T \mid J = T) = \frac{N_{B=T,J=T}}{N_{J=T}}
    \]

- **Rejection sampling:**
  - Generate sample for the full joint by sampling BBN
  - Use only samples that agree with the condition, the remaining samples are rejected

- **Problem:** many samples can be rejected
Likelihood weighting

- **Avoids inefficiencies of rejection sampling**
  - Idea: generate only samples consistent with an evidence (or conditioning event)
  - If the value is set no sampling (random choice occurs)
- **Problem:** using simple counts is not enough since these may occur with different probabilities
- Likelihood weighting:
  - With every sample keep a weight with which it should count towards the estimate

\[
\tilde{P}(B = T \mid J = T) = \frac{\sum_{\text{samples with } B=\text{T and } J=T} W_{B=\text{T}}}{\sum_{\text{samples with any value of } B \text{ and } J=T} W_{B=x}}
\]

### BBN likelihood weighting example

<table>
<thead>
<tr>
<th>Burglary</th>
<th>P(B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.001</td>
</tr>
<tr>
<td>F</td>
<td>0.999</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Earthquake</th>
<th>P(E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.002</td>
</tr>
<tr>
<td>F</td>
<td>0.998</td>
</tr>
</tbody>
</table>

| P(A|B,E) |
|--------|
| B   | E | T | F |
| T   | T | 0.95 | 0.05 |
| T   | F | 0.94 | 0.06 |
| F   | T | 0.29 | 0.71 |
| F   | F | 0.001 | 0.999 |

| P(J|A) |
|------|
| A   | T | F |
| T   | 0.90 | 0.1 |
| F   | 0.05 | 0.95 |

| P(M|A) |
|------|
| A   | T | F |
| T   | 0.7 | 0.3 |
| F   | 0.01 | 0.99 |

J = T (set !!!)
BBN likelihood weighting example

\[ P(B) \]

\[ \begin{array}{cc} T & F \\ 0.001 & 0.999 \end{array} \]

JohnCalls

J = T (set !!!)

Earthquake

\[ P(E) \]

\[ \begin{array}{cc} T & F \\ 0.002 & 0.998 \end{array} \]

MaryCalls

\[ P(J|A) \]

\[ \begin{array}{cc} T & F \\ 0.90 & 0.05 \end{array} \]

\[ P(M|A) \]

\[ \begin{array}{cc} T & F \\ 0.7 & 0.3 \end{array} \]

CS 2710 Foundations of AI
BBN likelihood weighting example

<table>
<thead>
<tr>
<th></th>
<th>P(B)</th>
<th>P(E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.001</td>
<td>0.002</td>
</tr>
<tr>
<td>F</td>
<td>0.999</td>
<td>0.998</td>
</tr>
</tbody>
</table>

|   | P(A|B,E)   | P(M|A)    |
|---|-----------|----------|
| T | 0.90      | 0.7      |
| T | 0.1       | 0.3      |
| F | 0.05      | 0.01     |

J = T (set !!!)

The sample weight: w = 0.05