Inference in Bayesian belief networks

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

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
  \[ P(A, B \mid C) = P(A \mid C)P(B \mid C) \]
  \[ P(A \mid C, B) = P(A \mid C) \]
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

$$P(X_i \mid pa(X_i))$$

Where:

$$pa(X_i)$$ - stand for parents of $X_i$

---

<table>
<thead>
<tr>
<th>$P(A \mid B, E)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$B$</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>$T$</td>
</tr>
<tr>
<td>$T$</td>
</tr>
<tr>
<td>$F$</td>
</tr>
<tr>
<td>$F$</td>
</tr>
</tbody>
</table>

---

Bayesian belief network.

<table>
<thead>
<tr>
<th>$P(B)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$B$</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>$0.001$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$P(E)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E$</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>$0.002$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$P(A \mid B, E)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$B$</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>$T$</td>
</tr>
<tr>
<td>$T$</td>
</tr>
<tr>
<td>$F$</td>
</tr>
<tr>
<td>$F$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$P(J \mid A)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>$0.90$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$P(M \mid A)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>$0.7$</td>
</tr>
</tbody>
</table>
**Full joint distribution in BBNs**

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

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

**Example:**

Assume the following assignment of values to random variables

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

Then its probability is:

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

---

**Parameter complexity problem**

- In the BBN the **full joint distribution** is

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

- **What did we save?**

**Parameters:**

- **full joint:** \(2^5 = 32\)
- **BBN:** \(2^3 + 2(2^2) + 2(2) = 20\)

**Parameters to be defined:**

- **full joint:** \(2^5 - 1 = 31\)
- **BBN:** \(2^2 + 2(2) + 2(1) = 10\)
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
Diagnosis of car engine

- Diagnose the engine start problem

Car insurance example

- Predict claim costs (medical, liability) based on application data
(ICU) Alarm network

CPCS

- Computer-based Patient Case Simulation system (CPCS-PM) developed by Parker and Miller (University of Pittsburgh)
- 422 nodes and 867 arcs
QMR-DT

- **Medical diagnosis in internal medicine**

  Bipartite network of disease/findings relations

  ![Diagram of QMR-DT derived from Internist-1/QMR KB]

  534 diseases

  40740 arcs

  4040 findings

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?
BBN for the device example. Inference.

- **Device** (equipment):
  - operating *normally* or *malfunctioning*.
- A sensor indirectly monitors the operation of the device
  - Sensor reading is either *high* or *low*

<table>
<thead>
<tr>
<th>Device status</th>
<th>normal</th>
<th>malfunctioning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.9</td>
<td>0.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Device\Sensor</th>
<th>high</th>
<th>low</th>
</tr>
</thead>
<tbody>
<tr>
<td>normal</td>
<td>0.1</td>
<td>0.9</td>
</tr>
<tr>
<td>malfunctioning</td>
<td>0.6</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Diagnostic inference. Example.

- **Diagnostic inference**: compute the probability of device operating normally given the sensor reading is high

\[
P(D = normal \mid S = high) = \frac{P(D = normal, S = high)}{P(S = high)}
\]

\[
P(D = normal, S = high) = P(S = high \mid D = normal) \cdot P(D = normal)
\]

\[
P(S = high) = \sum_{j \in \{normal, malfunc\}} P(S = high, D = j).
\]

\[
P(S = high) = P(S = high \mid D = normal)P(D = normal) + P(S = high \mid D = malfunc)P(D = malfunc)
\]
Inference in Bayesian network

- **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

\[
\text{Assume we want to compute: } P(J = T)
\]

#### 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: 15
- Number of products: 16*4=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)

\[ P(J = T) = \]

\[ = \sum_{b \in S_1} \sum_{e \in S_2} \sum_{m \in S_3} P(J = T \mid A = a) P(M = m \mid A = a) P(A = a \mid B = b, E = e) P(B = b) P(E = e) \]

\[ = \sum_{b \in S_1} \sum_{e \in S_2} \sum_{m \in S_3} P(J = T \mid A = a) P(M = m \mid A = a) \left( \sum_{e \in S_2} P(A = a \mid B = b, E = e) P(E = e) \right) \]

Computational cost:

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

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

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: \(P(B = T, J = T)\)

\[
P(B = T, J = T) = \sum_{a \in S_1} \left( \sum_{m \in S_2} P(M = m \mid A = a) \right) \left( \sum_{e \in S_3} P(A = a \mid B = b, E = e) P(E = e) \right) \]

• 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: \( P(B = T, J = T) \)

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

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

• 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

\[ P(\text{True}) = 1 = \]

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

\[ f_J(a) \quad f_M(a) \quad f_B(a) \quad f_E(a, b) \]

Variable order:

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:**
    \[
    \tilde{P}(B = T, J = T) = \frac{N_{B=T,J=T}}{N}
    \]
    \(N\) is the total number of samples, \(N_{B=T,J=T}\) is the number of samples with \(B = T, J = T\).

- **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)**
  - Burglary
  - \(P(B) = \begin{array}{c} T \ 0.001 \\ F \ 0.999 \end{array}\)

- **P(E)**
  - Earthquake
  - \(P(E) = \begin{array}{c} T \ 0.002 \\ F \ 0.998 \end{array}\)

- **P(A|B,E)**
  - \(A = \begin{array}{c} T \ 0.90 \\ F \ 0.05 \end{array}\)

- **P(J|A)**
  - JohnCalls
  - \(P(J|A) = \begin{array}{c} T \ 0.90 \\ F \ 0.05 \end{array}\)

- **P(M|A)**
  - MaryCalls
  - \(P(M|A) = \begin{array}{c} T \ 0.7 \\ F \ 0.01 \end{array}\)

CS 1571 Intro to AI
### BBN sampling example

**Burglary**

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

**Earthquake**

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

- **P(A|B,E)**
<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>

**Alarm**

- **P(J|A)**
<table>
<thead>
<tr>
<th>A</th>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.90</td>
<td>0.1</td>
</tr>
<tr>
<td>F</td>
<td>0.05</td>
<td>0.95</td>
</tr>
</tbody>
</table>

**MaryCalls**

- **P(M|A)**
<table>
<thead>
<tr>
<th>A</th>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>0.7</td>
<td>0.3</td>
</tr>
<tr>
<td>F</td>
<td>0.01</td>
<td>0.99</td>
</tr>
</tbody>
</table>

CS 1571 Intro to AI
BBN sampling example

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

|        | P(A|B,E)       |
|--------|---------------|
|        | B  E  T  F    |
| Alarm  | 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)          | P(M|A)          |
|-------|---------------|---------------|
|       | A  T  F       | A  T  F       |
| JohnCalls | T  0.90 0.1  | T  0.7 0.3    |
|         | F  0.05 0.95  | F  0.01 0.99  |

| MaryCalls | T  0.90 0.1  | F  0.05 0.95  |
Monte Carlo approaches

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

- **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