Bayesian Network Semantics

- Bayesian networks are representing independence between variables.
  - Two variables are **dependent** if knowing the value of one influences your belief of the value of another.
  - Two variables are **independent** if knowing the value of one does not change your belief in the other.

### Example Bayesian Network

**Nodes:**
- Weather
- Cavity
- Toothache
- Catch
- Alarm
- Earthquake
- MaryCalls
- JohnCalls
- Burglary
- MaryCalls
- JohnCalls

**Arrows:**
- Weather → Cavity
- Toothache → Weather
- Toothache → Catch
- Cavity → Toothache
- Cavity → Alarm
- Weather → Alarm
- Burglary → Alarm
- Earthquake → Alarm
- MaryCalls → Alarm
- JohnCalls → Alarm

**Probabilities:**

<table>
<thead>
<tr>
<th>Event</th>
<th>P(B)</th>
<th>P(E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Burglary</td>
<td>.001</td>
<td>.002</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Burglary</th>
<th>Earthquake</th>
<th>Alarm</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>E</td>
<td>P(A)</td>
</tr>
<tr>
<td>t</td>
<td>t</td>
<td>.95</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>.94</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>.29</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>.001</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>JohnCalls</th>
<th>P(J)</th>
<th>P(A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>.90</td>
<td></td>
</tr>
<tr>
<td>f</td>
<td>.05</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MaryCalls</th>
<th>P(M)</th>
<th>P(A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>.70</td>
<td></td>
</tr>
<tr>
<td>f</td>
<td>.01</td>
<td></td>
</tr>
</tbody>
</table>
Conditional independence

- Previous versions of the book gave better details on dependence/independence
- A node is conditionally independent of non-descendants given its parents
- A node is conditionally independent of all other nodes given its neighbors

D-separation

- Given a network, how can we easily know if two variables are independent or dependent?
- direction-dependent separation (d-separation)
- To tell if X and Y are independent:
  - Look at every undirected path between X and Y
  - Check to see if every path is blocked (given E)
- Use the same information to determine whether links need to be added to a network when adding nodes

D-separation cases (independence)

Example 1

- Burglary
- Earthquake
- Alarm
- JohnCalls
- MaryCalls
Example 2

Example 3

Exact Inference in Bayesian Networks

- Inferences allows us to ask any question about the variables in the network
- Example: What is $P(B=\text{true} \mid j=\text{true} \land m=\text{true})$
  - Hidden variables are alarm & earthquake
  - Enumerate all values for alarm, earthquake, burglary
  - Use ratios to find probabilities
Approximate Inference in Bayesian Networks

- Monte-Carlo approach
  - Starting from the top of the tree:
    - Randomly sample unknown variables according to provided distributions
    - Measure if query is true/false
  - The ratio of the query being true/false will approach the actual ratio

Example

- $P(B \mid j=true \land m=true)$

<table>
<thead>
<tr>
<th>b</th>
<th>e</th>
<th>a</th>
<th>j</th>
<th>m</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>1.197E-07</td>
</tr>
<tr>
<td>t</td>
<td>t</td>
<td>f</td>
<td>t</td>
<td>t</td>
<td>5.000E-11</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>5.910E-04</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
<td>t</td>
<td>t</td>
<td>2.994E-08</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>3.650E-04</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>f</td>
<td>t</td>
<td>t</td>
<td>6.094E-07</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>6.281E-04</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
<td>t</td>
<td>t</td>
<td>4.980E-04</td>
</tr>
</tbody>
</table>

Direct Sampling

- Consider the following (somewhat silly) example
  - Estimate the distribution of a coin coming up heads
    - Given that it is a fair coin (0.5 heads; 0.5 tails)
  - Simulate coin and measure the result
Direct sampling, version 2

- Now consider a simple network
- Simple to estimate probability of bad traffic through sampling & directly
  - $P(S) \cdot P(T | S) + P(\neg S) \cdot P(T | \neg S)$
  - $0.2 \cdot 0.9 + 0.8 \cdot 0.5 = 0.58$

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(int argc, char **argv)
{
    srand(time(0));
    int cnt = 10;
    if (argc > 1)
        cnt = atoi(argv[1]);
    int traffic = 0;
    for (int x = 0; x < cnt; x++)
    {
        bool snow = false;
        if (((random()%10) <= 1)
            snow = true;
        if (((snow) & ((random()%10) != 3))
            traffic++;
        if (((!snow) & ((random()%2) != 0))
            traffic++;
    }
    printf("%d of %d trials have bad traffic (%1.3f)\n", traffic, cnt, (float)traffic/cnt);
    return 0;
}
```
Introducing evidence

• Suppose I want the probability of bad traffic given snow?
  • Like our coin flip, we can compute or sample this directly
    • $P(T \mid S) = 0.9$
  • In general, this only works if our evidence has no parents

Rejection Sampling

• To sample a network with evidence:
  • Sample the network as before
  • Throw out cases where evidence isn’t true
  • Measure probabilities in resulting network

Limitations

• This works so far because we are just sampling probabilities in the network
• What happens if we want to introduce evidence?
  • Depends on where the evidence is introduced
What is $P(Rain|Sprinkler = true)$?

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(int argc, char **argv)
{
    srand(time(0));
    int cnt = 10;
    if (argc > 1)
        cnt = atoi(argv[1]);
    int valid = 0;
    int rainCnt = 0;
    for (int x = 0; x < cnt; x++)
    {
        bool cloudy = false;
        if ((random() % 2) == 1)
            cloudy = true;
        bool sprinkler = false;
        if ((cloudy) && ((random() % 10) == 3))
            sprinkler = true;
        bool rain = false;
        if ((cloudy) && ((random() % 10) == 2))
            rain = true;
        bool wet = false;
        if (cloudy && rain && (random() % 100) < 99)
            wet = true;
        if (cloudy && !rain && (random() % 100) < 90)
            wet = true;
        if (!cloudy && rain && (random() % 100) < 90)
            wet = true;
        if (sprinkler)
        {
            valid++;
            if (rain)
                rainCnt++;
        }
    }
    printf("%d of %d trials are valid. %d of %d with sprinkler have rain (%1.3f)\n", valid, cnt, rainCnt, valid, (float)rainCnt/valid);
    return 0;
}
```

Drawbacks of this approach?

- What if the number of variables we need to sample grows?
- What if the likelihood of the data shrinks?
Likelihood weighting

• How can we measure uncommon events?
  • Fix evidence in bayesian network
  • Sample all other variables
  • Weight sample according to likelihood of sample given the evidence

What is \( P(\text{Rain}|\text{Sprinkler} = \text{true}) \)?

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(int argc, char **argv)
{
    srand(time(0));
    int cnt = 10;
    if (argc > 1)
        cnt = atoi(argv[1]);
    double rainCnt = 0;
    double rainNotCnt = 0;
    double weight = 1;
    for (int x = 0; x < cnt; x++)
    {
        bool cloudy = false;
        if ((random()) % 2 == 1)
            cloudy = true;
        bool sprinkler = true;
        if (cloudy) weight = 0.1;
        if (!cloudy) weight = 0.5;
        bool rain = false;
        if ((cloudy) && ((random()) % 100 < 99))
            rain = true;
        if ((cloudy) && ((random()) % 100 < 90))
            rain = true;
        if (rain)
            rainCnt += weight;
        if (!rain)
            rainNotCnt += weight;
        if ((climate) && (!cloudy) && (random()) % 100 < 90)
            rain = true;
        if (!rain)
            rainNotCnt += weight;
        printf("%.3f weight on rain; %.3f on no rain. (%1.3f normalized chance of rain | sprinkler = true)\n", rainCnt, rainNotCnt, rainCnt/(rainCnt + rainNotCnt));
    }
    return 0;
}
```
Likelihood weighting

- Likelihood weighting uses information from every sample generated
- But:
  - If there is a lot of evidence, the weights will be small (and thus less accurate)
  - If the evidence occurs late in the inference, it may be inconsistent with the rest of the network
    - eg allow Sprinkler & Rain to be false with evidence that the grass is wet [resulting in 0 weight]

Local Search for inference

- Markov-Chain Monte-Carlo sampling
  - Build a complete state of the world
  - Re-sample each non-evidence variable according to probability distribution
  - Count how often query is true

Probability distributions

- How do we compute the chance of a given variable given everything else in the network?
- Markov Blanket: A variable is independent of the rest of the networking given its parents, children, and children’s parents
Markov Blanket

Sampling a variable

- Sample each variable from:
  \[ P(x'_i|\text{mb}(X_i)) = \alpha P(x'_i|\text{parents}(X_i)) \prod_{Y_j \in \text{Children}(X_i)} P(y_j|\text{parents}(Y_j)) \]

- Why is this correct?
- How would this work for our network?