COMP3308 Assignment 2
Bayesian Networks

Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865

1 Aim

blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah

2 Methods

2.1 Bayesian Networks

Bayesian networks are simple, probabilistic graphical models that represent sets of random variables and their conditional dependencies.

2.2 Variable Elimination

blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah

2.3 Likelihood Weighting

blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah

3 Results and Discussion

3.1 Question 1

Metastatic cancer is a possible cause of a brain tumour and is also an explanation for increased total serum calcium. In turn, either of these could explain a patient falling into a coma. Severe headache is also possibly associated with a brain tumour.

- The prior probability of metastatic cancer $P(m)$ is 0.2.
- The conditional probability of increased total serum calcium $P(I|M)$ is: $P(i|m) = 0.8$ and $P(i|\neg m) = 0.2$
- The conditional probability of brain tumor $P(B|M)$ is: $P(b|m) = 0.2$ and $P(b|\neg m) = 0.05$
- The conditional probability of coma $P(C|I, B)$ is: $P(c|i, b) = 0.8$, $P(c|\neg i, b) = 0.8$, $P(c|i, \neg b) = 0.8$ and $P(c|\neg i, \neg b) = 0.05$.
- The conditional probability of severe headache $P(S|B)$ is $P(s|b) = 0.8$ and $P(s|b) = 0.6$.

a) Construct and show the equivalent graphical model.

Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865

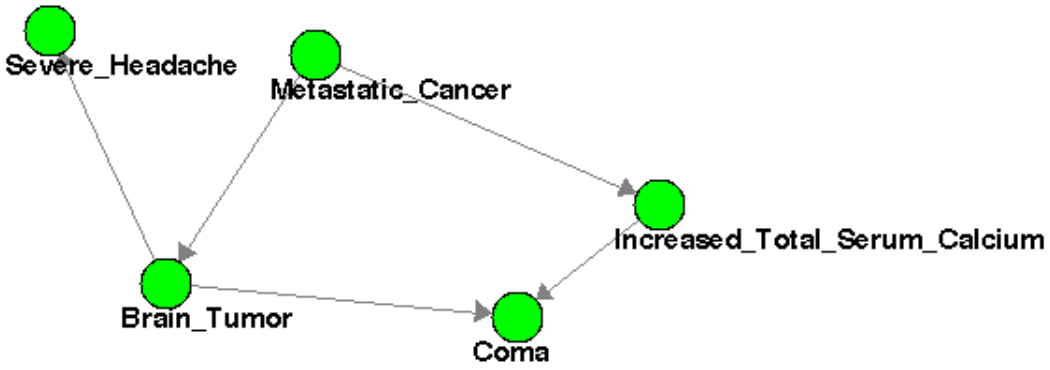

Figure 1: Equivalent graphical model created using JavaBayes

b) What is the prior probability of coma $P(C)$?

```
Posterior distribution:
        probability (  "Coma" ) { //1 variable(s) and 2 values
                table
                        0.32    // p(true | evidence )
                        0.68;   // p(false | evidence );
        }
```

Figure 2: Probability of coma query output using JavaBayes

Using the `Query` function in JavaBayes, $P(C) = 0.32$.

c) What is the probability of metastatic cancer given the patient has severe headaches and has not fallen into coma?

```
Posterior distribution:
        probability (  "Metastatic_Cancer" ) { //1 variable(s) and 2 values
                table
                        0.12087912087912088     // p(true | evidence )
                        0.8791208791208791;     // p(false | evidence );
        }
```

Figure 3: $P(M|S, \neg C)$ query output using JavaBayes

Using the `Observe` and `Query` functions in JavaBayes, $P(M|S, \neg C) = 0.12087912087912088$.

d) What is the Markov blanket of coma?

In a Bayesian network, the Markov blanket of node A includes its parents, children and the other parents of all of its children.

Therefore the Markov blanket of coma are brain tumor and increased total serum calcium.

e) Are increased total serum calcium and brain tumor independent given coma? Explain.

Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865

No, because of explaining away otherwise known as Berkson's Paradox.

Normally, total serum calcium and brain tumor are independent, but if we are given coma they become dependent since they share the same child.

f) What is the probability of fallen into coma given the patient has metastatic cancer?

```
Posterior distribution:
          probability ( "Coma" ) { //1 variable(s) and 2 values
                    table
                              0.68     // p(true | evidence )
                              0.32;    // p(false | evidence );
          }
```

Figure 4: $P(C|M)$ query output using JavaBayes

Using the `Observe` and `Query` functions in JavaBayes, $P(C|M) = 0.68$.

3.2 Question 2

WUT I DUN EVEN

3.3 Question 3

something something diagnosis

3.4 Question 4

3.4.1 Likelihood Weighting

We implemented the likelihood weighting in Python 2.7.5 to calculate the probability of $P(cloudy|sprinkler, wetgrass)$.

Our program takes in the number of samples used to construct the estimate (N) as input and then runs M times which is set to 1000 by default. It then returns the mean, variance and standard deviation of the probability calculated using the likelihood weighting algorithm.

Below in Table 1, are the results for N=10, 100, 1000, 5000.

N	Mean	Variance	Standard Deviation
10	0.5069	0.02524239	0.158878538513
100	0.50004	0.0025593984	0.0505904971314
1000	0.499573	0.000222700671	0.0149231588814
5000	0.5001358	4.979223836e-05	0.00705636155253

Table 1: Accuracy results for likelihood weighting

blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah discuss table?

Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865

3.4.2 Exact Inference using JavaBayes

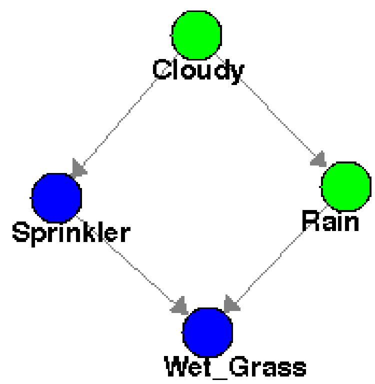

Figure 5: Equivalent graphical model created using JavaBayes. Sprinkler and Wet Grass set to true.

```
Posterior distribution:
        probability ( "Cloudy" ) { //1 variable(s) and 2 values
                table
                        0.17475728155339806     // p(true | evidence )
                        0.825242718446602;      // p(false | evidence );
        }
```

Figure 6: $P(Cloudy|Sprinkler, WetGrass)$ query output using JavaBayes

blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah discuss exact inference and compare to other thing

4 Conclusions

blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah

5 Reflection

blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah

Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865

6 Code

Our implementation of Likelihood Weighting was written in Python 2.7.5.

This implementation was designed to work with any Bayesian network and not just the Cloudy-Rain-Sprinkler-WetGrass network.

6.1 XML Format

The program reads an XML file which has the structure of the network and probability values and makes the appropriate data structures.

The XML of the Cloudy-Rain-Sprinkler-WetGrass network can seen in the figure below.

```
<network>
        <node>
                <id>C</id>
                <name>Cloudy</name>
                <probability>0.5</probability>
        </node>

        <node>
                <id>S</id>
                <name>Sprinkler</name>
                <parent>C</parent>
                <probability given="C">0.1</probability>
                <probability given="-C">0.5</probability>
        </node>

        <node>
                <id>R</id>
                <name>Rain</name>
                <parent>C</parent>
                <probability given="C">0.8</probability>
                <probability given="-C">0.2</probability>
        </node>

        <node>
                <id>W</id>
                <name>Wet Grass</name>
                <parent>S</parent>
                <parent>R</parent>
                <probability given="S, R">0.99</probability>
                <probability given="S, -R">0.9</probability>
                <probability given="-S, R">0.9</probability>
                <probability given="-S, -R">0.00</probability>
        </node>
</network>
```

Figure 7: Cloudy-Rain-Sprinkler-WetGrass-Network.xml

Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865

6.2 Instructions

To run our implementation, change directory to the where the code is. Make sure that Cloudy-Rain-Sprinkler-WetGrass-Network.xml file is present.

Then run:

```
python likelihood.py <N>
```

where N is the number of samples used.

6.2.1 Example

```
$ python likelihood.py 1000
---------- Likelihood Weighting Sampling ----------
Estimating the probability of P(Cloudy | Sprinkler, Wetgrass)
---- Summary ----
N: 1000
M: 1000
Mean: 0.499573
Variance: 0.000222700671
Standard Deviation: 0.0149231588814
```


Woo Hyun Jung 310250811
Khanh Cao Quoc Nguyen 311253865