UMEÅ UNIVERSITY **October 15, 2009**
Institution of Computer Science
Lab report

Assignment 2

Artificial Intelligence

Bayesian Networks

Name	Fredrik Söderberg	Eric Jönsson
E-mail	ei98fsg@cs.umu.se	ericj@cs.umu.se
Path	~ei98fsg/edu/ai/lab2	~ericj/edu/ai/lab2

Grader
Johan Granberg

Contents

1 Problem specifikation

1.1 Summary

The purpose of this assignment was to get a hands-on experience of Bayesian Networks, constructs used to evaluate logical statements. Given text files that specify the layout of these networks and a set of logical queries to process, we were to write a program that parsed these text files and answered the queries correctly.

1.2 Original specification

The original specification for this assignment can be downloaded from the following location:

`http://www.cs.umu.se/kurser/5DV019/HT09/bayes.pdf`

2 File access and user guide

2.1 File paths

All files used for this assignment can be downloaded from both of the following paths:

`~ei98fsg/edu/ai/lab2`

`~ericj/edu/ai/lab2`

2.2 Compile and run

The program can be compiled by typing:

```
$ ant
```

This constructs a jar-file named `Bayes.jar`. To run it, type the following:

```
$ java -jar Bayes.jar <network-file.txt>
```

This runs the program using the `network-file.txt` specification file for the Bayesian Network.

3 System description

3.1 Overview

The application uses a slight variant of the naive *enumeration ask*-algorithm specified in the course litterature. During development, it seemed easier to implement our own version rather than the complex and confusing one found in the book.

When designing this algorithm, we paid little regard to the book's version and instead focused on the theory behind it. Going one step at a time, trying to sum up the mathematical formulas worked better for us than trying to implement an algorithm none of us really understood.

In addition, we believe the algorithm as printed in the book works badly for multiple query variables. Our version handles several with ease, i.e `Q(Burglary, Earthquake | Alarm)`.

4 Test runs

4.1 Overview

For the test runs, we were given three test files, `net1.txt`, `net2.txt`, `net3.txt` and `net4.txt`.

The fourth text file described a Bayesian Net with 33 nodes. Using our implementation of the naive *enumeration ask*-algorithm, the running time became ridiculously long. Therefore, we've chosen not to go for the extra credit points and settle with the first three files.

4.2 Results

```
$ java -jar Bayes.jar net1.txt
Q(Disease|!Symptom) = 0.18333333333333332
Q(Risk|!Symptom) = 0.7166666666666667
Q(Disease|Symptom,!Risk) = 0.1

$ java -jar Bayes.jar net2.txt
Q(Disease-1|!Disease-2) = 0.4
Q(Disease-1|Symptom) = 0.8069056108087822
Q(Disease-1|Symptom,!Disease-2) = 0.8421052631578947

$ java -jar Bayes.jar net3.txt
Q(Disease|Symptom-1,!Risk-1) = 0.5836128413991375
Q(Symptom-2|Disease,Risk-1) = 0.92
Q(Symptom-1) = 0.23506699999999997
```

5 Final thoughts

We conclude this report by noting the following:

- The algorithm descriptions given in the course material are, generally speaking, confusing.
- Bayesian Networks are hard. Very hard. Even after completing the assignment, it all seems – in some ways – like black magic.
- Having query results alongside the actual test queries would have been nice during development. As it stands now, we *think* our numbers in section 4 are correct, but we're not 100% sure.

- We've had enough of Bayesian Nets for a long, long time. This assignment consumed way more time than originally expected.

```
Q(SickOfBayesNets | OverDueAssignment, TiredStudents) = 1.0
```