README

Name : Tushar Kumar
NetId : tusharku
Collaborators : None

November 27, 2018

1 Introduction

This project, which was implemented by myself, and **did not have any collaboration from anybody else**, implements four inference algorithms on Bayesian Networks. One is the exact inference method which enumeration method and the other three are approximate inference methods which are Likelihood Weighting, Rejection Sampling and Gibbs Sampling. This project was undertaken as part of the graduate course(CSC 442) at University of Rochester.

2 Technology Used

- Java as the programming language (Java 10.0.2) with no external libraries
- Eclipse used as IDE
- Draw.io for creating design artifacts
- Visual VM for extracting memory usages of implemented algorithms.

3 Building the project

- Download the **UncertainInference-tusharku.zip** file(which you would have if you are reading this file.
- Unzip the file to get the UncertainInference-tusharku folder
- Run the following commands in sequence once you are in the location where you downloaded the zip file: Please mind the line break created because the command being of greater length than width of the page. Actual command would be
  "javac -d executable -sourcepath src
  -cp . src/com/uofr/course/csc442/hw/hw3/bn/inference/BayesianNetworkInference"

```
cd UncertainInference-tusharku

javac -d executable -sourcepath src -cp .
 ↪  src/com/uofr/course/csc442/hw/hw3/bn/inference/BayesianNetworkInference.java
```

In case , you were not able to compile this, not to worry, I have provided the already compiled binaries of classes in the bin folder. So you can use that straight away to run the application

4 Running the application

- Run the following command to run all algorithms for a given network(aima-alarm), a query(B) and some evidence(J true M true).

```
cd UncertainInference-tusharku
java -cp executable
 ↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
 ↪  -samples 1000000 -query B -file aima-alarm -evidence J true M true
 ↪  -algo all -burn-in 10000
```

This, would run all inference algorithms for aima-alarm network with query being B and evidence being J true and M true. It will use a sample size of 1000000 and will use a burn in of 10000 as the burn in sample size for gibbs sampling.

5 Executing Inferences

I would now provide a detailed description of all the command line arguments and how to sue them to evaluate the different inference problems given as part of the project requirement. These are the commandline arguments with their description and possible values that can be provided in order to run the inference.

- **file** : This represents xmlbif file which has details of the network one wants to run inference on.
  Possible Values : aima-alam(for aima-alarm.xml), aima-wet-grass(for aima-wet-grass.xml), dog-problem(for dog-problem.xml)
  Default Value : None(is a required parameter)

- **query** : This represents the variable of the network that you want to run inference on.
  Possible Values : String (must be a variable in network)
  Default Value : None(is a required parameter)

- **samples** : Number of samples which you want to use for approximate inference algorithms
  Possible Values : Integer
  Default Value : 100000

- **evidence** : Evidence to be conditioned on, while running inference
  Possible Values : String key value pairs : -evidence evidence1Name> <evidence1Value> <evidence2Name> <evidence2Value>
  Default Value : None

- **algo** : What algorithm you want to use to run inference on
  Possible Values : enumeration(For Enumeration), likelihood(For Likelihood Weighting), rejection(For Rejection Sampling), gibbs(For Gibbs Sampling), all(For running all algorithms)
  Default Value : enumeration

- **burn-in** : Number of samples ignored as part of burn in samples for gibbs sampling.
  Possible Values : Integer
  Default Value : 0

- **filepath** : Location of the xml file if its not one of aima-alarm/aima-wet-grass/dog-problem. This is provided in case one wants to try on a new xml file with a different network
  Possible Values : String
  Default Value : None

5.1 Running example inference on Aima-Alarm

Here are some sample scenarios with their commands provided for Aima-Alarm:

- **B given J true M true with all algorithms**

```
java -cp executable
 ↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
 ↪  -samples 1000000 -file aima-alarm -query B -evidence J true M true
 ↪  -algo all -burn-in 10000
```

- **B given J true M true with enumeration**

```
java -cp executable
 ↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
 ↪  -file aima-alarm -query B -evidence J true M true -algo
 ↪  enumeration
```

- **B given J true M true with gibbs sampling**

```
java -cp executable
↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
↪  -samples 1000000 -file aima-alarm -query B -evidence J true M true
↪  -algo gibbs -burn-in 10000
```

- **M given A true B true E false with all algorithms**

```
java -cp executable
↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
↪  -samples 1000000 -file aima-alarm -query M -evidence A true B true
↪  E false -algo all -burn-in 10000
```

- **M given A true B true E false with Likelihood Weighting**

```
java -cp executable
↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
↪  -samples 1000000 -file aima-alarm -query M -evidence A true B true
↪  E false -algo likelihood
```

- **M given A true B true E false with Rejection Sampling**

```
java -cp executable
↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
↪  -samples 1000000 -file aima-alarm -query M -evidence A true B true
↪  E false -algo rejection
```

5.1.1 Running example inference on Aima-Wet-Grass

- **R given W is true with all algorithms**

```
java -cp executable
↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
↪  -samples 1000000 -file aima-wet-grass -query R -evidence W true
↪  -algo all -burn-in 10000
```

- **R given W is true with enumeration**

```
java -cp executable
↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
↪  -file aima-wet-grass -query R -evidence W true -algo enumeration
```

- **R given W is true with gibbs**

```
java -cp executable
 ↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
 ↪  -samples 1000000 -file aima-wet-grass -query R  -evidence W true
 ↪  -algo gibbs -burn-in 10000
```

- **S given R true with all algorithms**

```
java -cp executable
 ↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
 ↪  -samples 1000000 -file aima-wet-grass -query S -evidence R true
 ↪  -algo all -burn-in 10000
```

- **S given R true with Likelihood Weighting**

```
java -cp executable
 ↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
 ↪  -samples 1000000 -file aima-wet-grass -query S -evidence R true
 ↪  -algo likelihood
```

- **S given R true with Rejection Sampling**

```
java -cp executable
 ↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
 ↪  -samples 1000000 -file aima-wet-grass -query S -evidence R true
 ↪  -algo rejection
```

5.1.2 Running example inference on Dog-Problem

- **family-out given hear-bark is true with all algorithms**

```
java -cp executable
 ↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
 ↪  -samples 1000000 -file dog-problem -query family-out -evidence
 ↪  hear-bark true -algo all -burn-in 10000
```

- **family-out given hear-bark is true with enumeration**

```
java -cp executable
↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
↪  -file dog-problem -query family-out -evidence hear-bark true -algo
↪  enumeration
```

- **family-out given hear-bark is true with gibbs**

```
java -cp executable
↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
↪  -samples 1000000 -file dog-problem -query family-out -evidence
↪  hear-bark true -algo gibbs -burn-in 10000
```

- **light-on given bowel-problem is true with all algorithms**

```
java -cp executable
↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
↪  -samples 1000000 -file dog-problem -query light-on -evidence
↪  bowel-problem true -algo all -burn-in 10000
```

- **light-on given bowel-problem is true with Likelihood Weighting**

```
java -cp executable
↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
↪  -samples 1000000 -file dog-problem -query light-on -evidence
↪  bowel-problem true -algo likelihood
```

- **light-on given bowel-problem is true with Rejection Sampling**

```
java -cp executable
↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
↪  -samples 1000000 -file dog-problem -query light-on -evidence
↪  bowel-problem true -algo rejection
```

5.1.3 Running example inference on different xml file

This section provides details on how to run inference for xml files other than dog-problem or aima-alarm or aima-wet-grass. In order to run inference on a different file than the three example ones provided, we must make use of the filepath argument and provide it with a value that is the absolute path(with the file name) for the xml file. In short, filename argument is used when inference needs to run on one of the three examples provided whereas filepath

needs to be used when inference needs to be run on an altogether different example file. Please note that if filepath and filename(a valid one) both are provided, program will use filename itself. If filename is not valid(one of expected ones), then it will use the filepath.

```
java -cp executable
 ↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
 ↪  -samples 1000000 -filepath absolute/path/to/file.xml -query query
 ↪  -evidence evidence true -algo all -burn-in 10000
```

- **light-on given bowel-problem is true with all algorithms with filepath provided**

```
java -cp executable
 ↪  com.uofr.course.csc442.hw.hw3.bn.inference.BayesianNetworkInference
 ↪  -samples 1000000 -filepath
 ↪  src/com/uofr/course/csc442/hw/hw3/examples/dog-problem.xml -query
 ↪  light-on -evidence bowel-problem true -algo all
```

5.1.4 Running inference on Auto generated complex graphs

This section provides details on running the inference algorithms on auto generated graphs of N nodes where (N <= 20) for enumeration and N $\approx$ 1000 for approximate sampling. The inference simulator can be used for two types of simulations.

1. Comparison of all algorithms on an auto generated graph. Since we need enumeration to run for a comparison for all algorithms, the simulator forces the nodes count to be less than 20.

2. Execute the inference on a graph as big as 10K nodes with an approximate inference algorithm. The simulation and inference procedure work for node sizes upto 10000 nodes on my machine beyond which the program crosses the memory limitations of my ide. But even at 10K nodes, the time taken is well within scalable limits of 5 seconds for gibbs sampling.

- **Inference on an auto generated graph of 15 nodes with all algorithms**

```
javac -d executable -sourcepath src -cp .
 ↪  src/com/uofr/course/csc442/hw/hw3/simulation/InferenceSimulator.java

java -cp executable
 ↪  com.uofr.course.csc442.hw.hw3.simulation.InferenceSimulator -nodes
 ↪  20 -algo all
```

- **Inference on auto generated graph of 10K nodes with gibbs sampling**

```
javac -d executable -sourcepath src -cp .
↪ src/com/uofr/course/csc442/hw/hw3/simulation/InferenceSimulator.java

java -cp executable
↪ com.uofr.course.csc442.hw.hw3.simulation.InferenceSimulator -nodes
↪ 10000 -algo gibbs
```