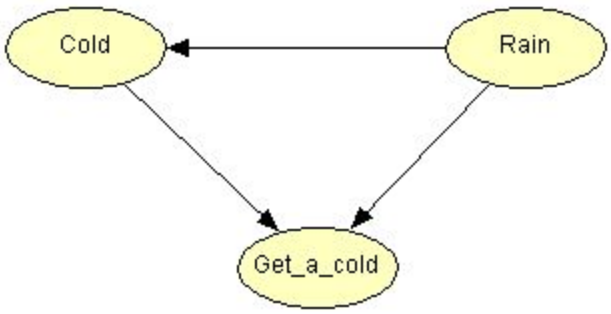
Andrea Hernandez Monterrubio - A01205154
Marc Abonce Seguin - A01200356

Bayes Network Lab

Our example problem is based on if a person could **Get a cold** given that it is Raining or it is Cold.




```
[Nodes]
Rain, Cold, Get_a_cold

[Probabilities]
+Rain = 0.3
+Cold|+Rain=0.8
+Cold|-Rain=0.4
+Get_a_cold|+Cold, +Rain = 0.8
+Get_a_cold|-Cold, +Rain = 0.3
+Get_a_cold|+Cold, -Rain = 0.6
+Get_a_cold|-Cold, -Rain = 0.1
```

Using Hugin Lite we defined the values for each node as following:

Rain | Cold | Get_a_cold

True	0.3
False	0.7

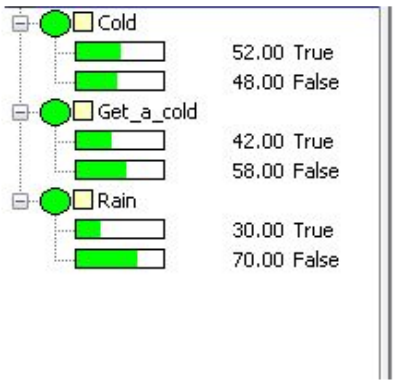
Rain | Cold | Get_a_cold

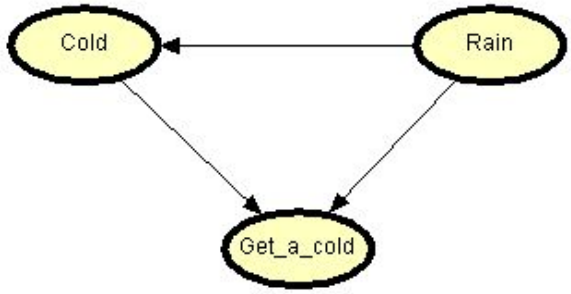
Rain	True	False
True	0.8	0.4
False	0.2	0.6

Rain | Cold | Get_a_cold

Cold	True		False	
Rain	True	False	True	False
True	0.8	0.6	0.3	0.1
False	0.2	0.4	0.7	0.9

And the final result is the network below:



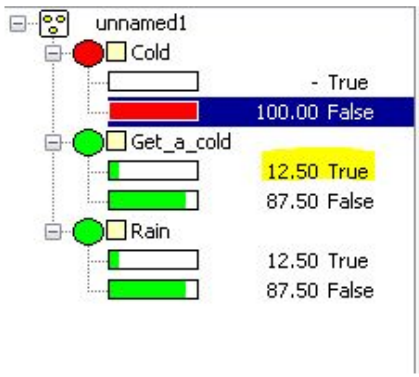


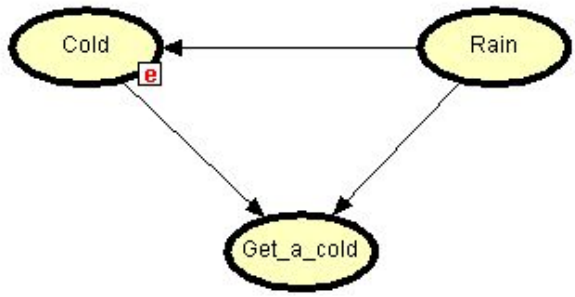
Now that we defined the network and the probabilities, it's time to test it! Clicking the node that will be our evidence, the other nodes will update the probability.

The queries used for this example are the following:

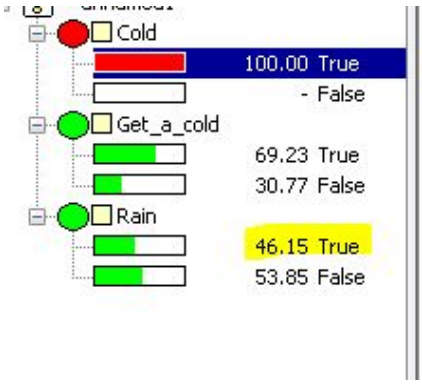
```
[Queries]
+Get_a_cold|-Cold
+Rain|+Cold
+Cold|-Rain,+Get_a_cold
+Rain|-Get_a_cold
```

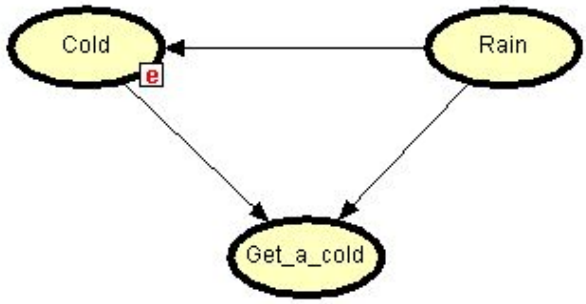
Query 1



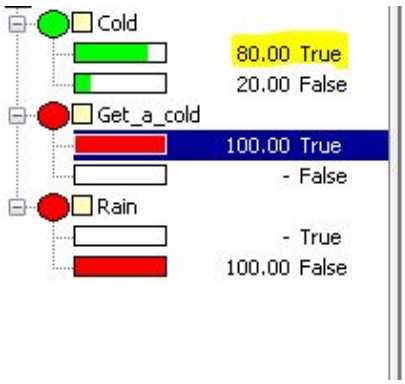


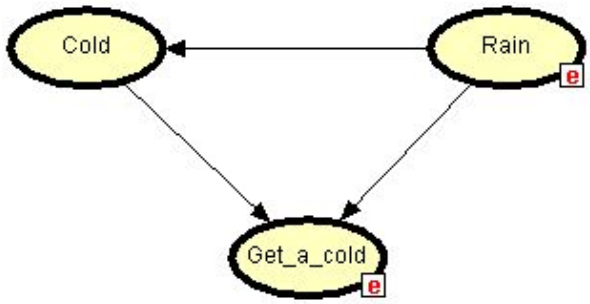
Query 2



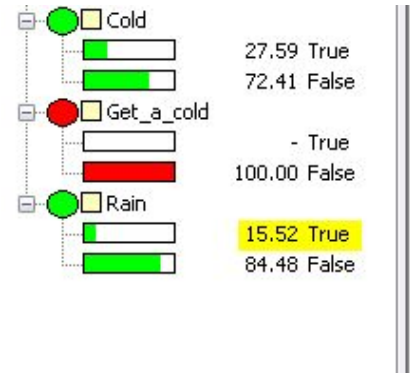


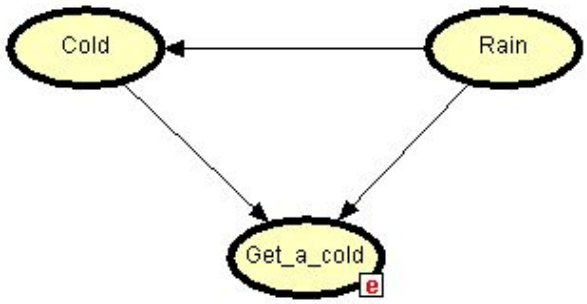
Query 3





Query 4





Values given by our program:

```
0.125
0.4615385
0.8
0.1551724
```

In both Hugin Lite and our lab we get:

1. 0.125
2. 0.4615
3. 0.8
4. 0.1552

Comparison between our implementation and Hugin Lite

Both Hugin Lite and our implementation create a user provided Bayes network and can calculate new probabilities based on the evidence they get. One difference is that Hugin Lite propagates all the probabilities for the network given any evidence, while our lab program will only answer a single specific query at a time. Also, Hugin Lite is graphical, unlike our text based implementation. This means that once you have built your network, Hugin Lite is faster to use than our lab and the answers

make more sense because you can see the graphical representation of the entire network being updated.

The algorithms used are also different between our lab and Hugin Lite. Our implementation uses enumeration to get each probability, which is the simplest algorithm, but not the most efficient one. On the other hand, Hugin Lite seems to use the junction tree algorithm to propagate the probabilities throughout the network. Both algorithms provide exact inferences though, so in theory they should always return the same results for all queries. However, Hugin's algorithm is more efficient and therefore allows bigger and more complex networks to be run, unlike our lab which would most likely run very slowly given a big Bayes network and a complex query.

For a real life application, we would most certainly not use our own implementation because of the limitations it has given the inefficiency of the enumeration algorithm which would likely make the tool unusable for practical problems. However, depending on the magnitude of the problem, Hugin Lite may not be enough for all problems either, because it is limited to 50 states before upgrading to the Developer/Research versions. There are many open source libraries that already implement Bayes networks though, so we would most likely use any of those libraries for a real life application.