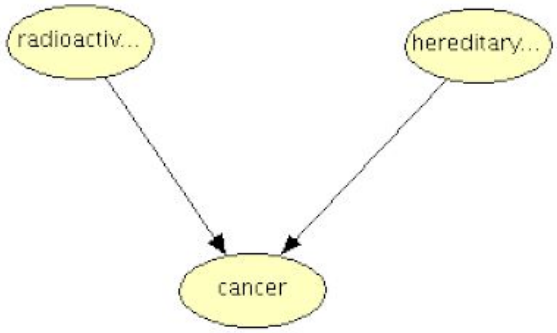
Martín Alejandro Molinero Morales A01204818
Diego Alfredo Ballesteros Bautista A01271588

Bayesian Network report, compare with Hugin Lite

The example we created is a 3 node network that represents two possible causes of getting cancer. The first one, is being on a radioactive place, and the second one is by an hereditary disease.
The created network has the following representation:



With the following probability tables for each node:

- Hereditary disease:

<heredita...	0.65
<heredita...	0.35

- Radioactive_place:

<radioactive_place>true	0.3
<radioactive_place>false	0.7

- Cancer:

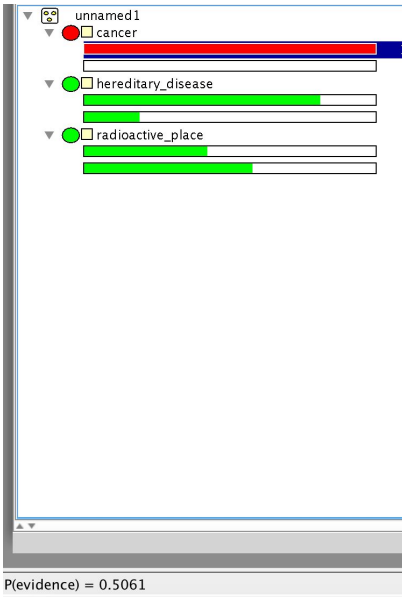
hereditary_dis...	<hereditary_disease>true		<hereditary_disease>false	
radioactive_pl...	<radioactive_place>true	<radioactive_place>false	<radioactive_place>true	<radioactive_place>false
<cancer>true	0.98	0.48	0.22	0.3
<cancer>false	0.02	0.52	0.78	0.7

We were looking for three main queries, which have the following results:

- In Hugin lite software, we only needed to provide the node probability values that we had as an evidence.
- In our program, we needed to write the full query in order that it can produce the result.

The following queries were executed on both tools and we reached the same results.

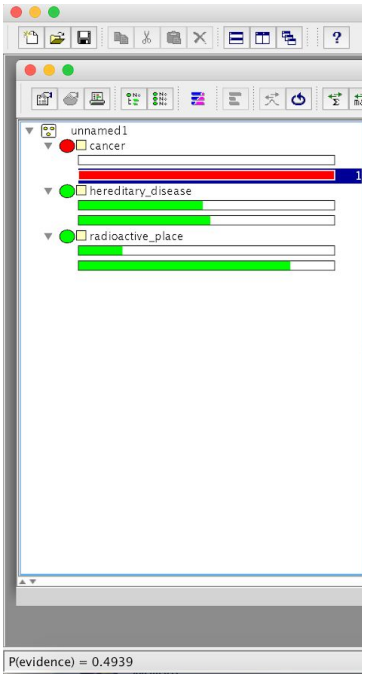
- +Cancer
    - Hugin Lite: It was needed to select, from all the nodes value, only the evidence we had in our formula, that was +Cancer, in this case, we take the value of the bottom left part of the program, showing p(evidence) = P(+cancer) = 0.5061



- Bayes program

```
diegoballest@ubuntu ...ayes/Bayes-network-lab master python3 bayes.py <
cancer.in
0.5061
```

- -Cancer
  - Hugin Lite: Same case as the previous one, just changing the evidence to -Cancer.

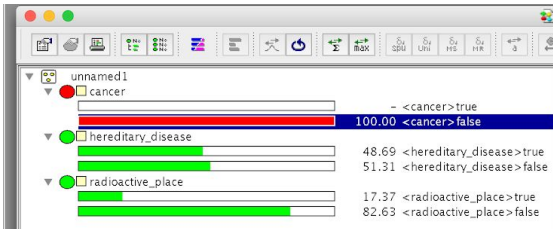


  - Bayes program

```
0.4939
```

- +Hereditary_disease|-Cancer
  - Hugin Lite: In this case, to obtain the probability is different from the previous cases. We need to select the probability of the evidence that we have (-Cancer) and look for the value in the probability table and node of the

hypothesis, in this case(+Hereditary_disease). The value in that node key, is the probability that we are looking for. p(+Hereditary_disease|-Cancer) = 0.4869



- Bayes Program

The algorithm in our implementation is the following. First, as we read the comma separated node names from the input, we create an object Node for each of them. Then, we read how much probability values of the network will be given and proceed to compute them. For every given probability, if it can be split, we associate to the first place of that split (the original node) its parents from the second place of the split and then, we assign a new key from a hash, containing the concatenation of both node name and parents with the numerical probability value. As we enter this value to the original probability table, the complement of that key (1-P(key)) is also calculated and inserted into the probability table.

Then, we process the queries received from the input and decide if they have to be computed with a conditional rule or by our recursive function, which receives just a comma separated nodes query. The recursive function then decides, if the query is composed just by a single node and it has no parents, look for the query in the node's probability table, otherwise, calculate total probability of the node. If the query is composed of multiple nodes and some of them are hidden (unsigned), generate the new queries making the product of each hidden node with the signs (+,-) and making a concatenation with the original query, then add every result of the chain rule of the new queries to obtain the total.

What our Bayes implementation does, is obtaining the probability of an event or series of events to happen by calculating the join probability of each related node in the problem. We achieve this by applying the enumeration algorithm and obtaining the values of the generated and original queries given from the probability table of the related nodes.

The Hugin tool allows you to build a probabilistic graphical model such as Bayesian networks, building it with graphical Influence diagrams and writing the knowledge base information to each node. Some of the main differences between the Hugin tool and our program is that Hugin has a breakthrough algorithm, making this tool one of the most efficient and robust inference engines, also having an object-oriented modeling, allowing it to learn from continuous and discrete variables, value of information and sensitivity analysis, and our implementation uses just the Enumeration algorithm, creating new inferences with each query that our program receives. But one common base between our application and the Hugin tool is that in both, they solve a specific query of a Bayesian network given all the different probabilities and their dependencies, each one obtaining the answer of the query with help of the total probability theorem and chain rule theorem, but with a different implementation in Hugin.

Martín Alejandro Molinero Morales A01204818
Diego Alfredo Ballesteros Bautista A01271588

To decide which tool to use depends on the scenario, if you want it for educational purposes of how a Bayesian network works, then this is a great tool to teach or understand more about this topic, this tool shows you in real time and with a good GUI all the information that you want. But in some cases, using the GUI might be slow (giving the inputs and manage it), and considering all the time it might take when you have a big network of nodes, on the other hand, if you just want to get a specific query fast, you can use the application developed and if you want to get information, you can modify the code to see each operation of the Enumeration algorithm.