Assignment: TC2011 - Bayes Networks

Team: Ramón Romero, Alejandro López, Eduardo Larios
Professor: Ruben Stranders
25/03/2019 - 11:59:59 PM

Introduction

On this assignment we'll build upon the concept seen in class in order to model a Bayesian Network that is able to predict the probability of events with multiple dependencies through the computing of the joint probability functions of the query and the evidence.

We based our implementation on the enumeration algorithm for bayesian inference. Our implementation is able to distinguish between children and parent nodes, and how they relate to calculate the total joint probability of events.

Comparison to Hugin Lite

Hugin first pre-computes all the total probabilities for each of the nodes in the network, before even attempting to resolve any queries. In comparison, using the enumeration algorithm we try to resolve just the dependencies related to our query, so unneeded values are not computed.

We think that Hugin works via instantiating the nodes and the relationship/dependencies between them first, then working over all of them. Also Hugin seems to add for each node the values of its parents probabilities to simplify their calculation as simple conditional probability, and get the total probability for the child as a basic multiplication of probabilities.

Now having all the total probabilities of each node it's considerably easier to calculate the probability of the occurrence of any node given the simple probabilities, already precalculated, just by following its chain of "dependencies".

In comparison our implementation just computes its probability chain depending on the queries that we receive. Under the hood both implementations probably use the Bayes rule as the basis for their inference, and should only differ on the way they represent their data, and the underlying data structures.

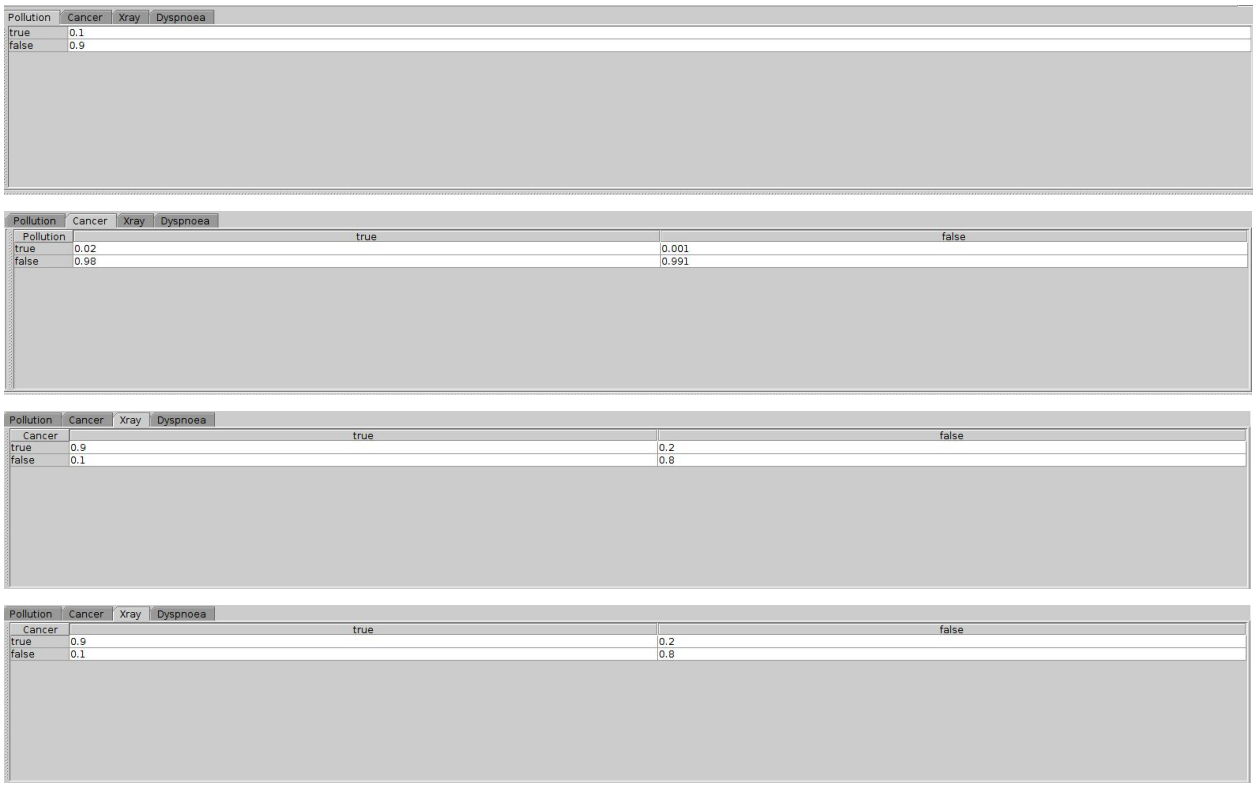
Where we use the chain rule via the enumeration algorithm; Hugin Lite may be using a simpler approach via the total probability equation, as the total joint probabilities are computed on startup, which simplifies the chain rule process.

However one slices it, both should have the same fundamental approach to inference. We think that the approach taken by Hugin Lite simplifies giving a continuous representation of the network that makes it really simply to visualize conditionality and dependence for each node, this is helped even more by showing a graphical user interface to see the state of the network at a glance. On the other hand the pre-calculation of probabilities could be prohibitively costly on a larger network where our algorithm and implementation may better suit its needs, being less

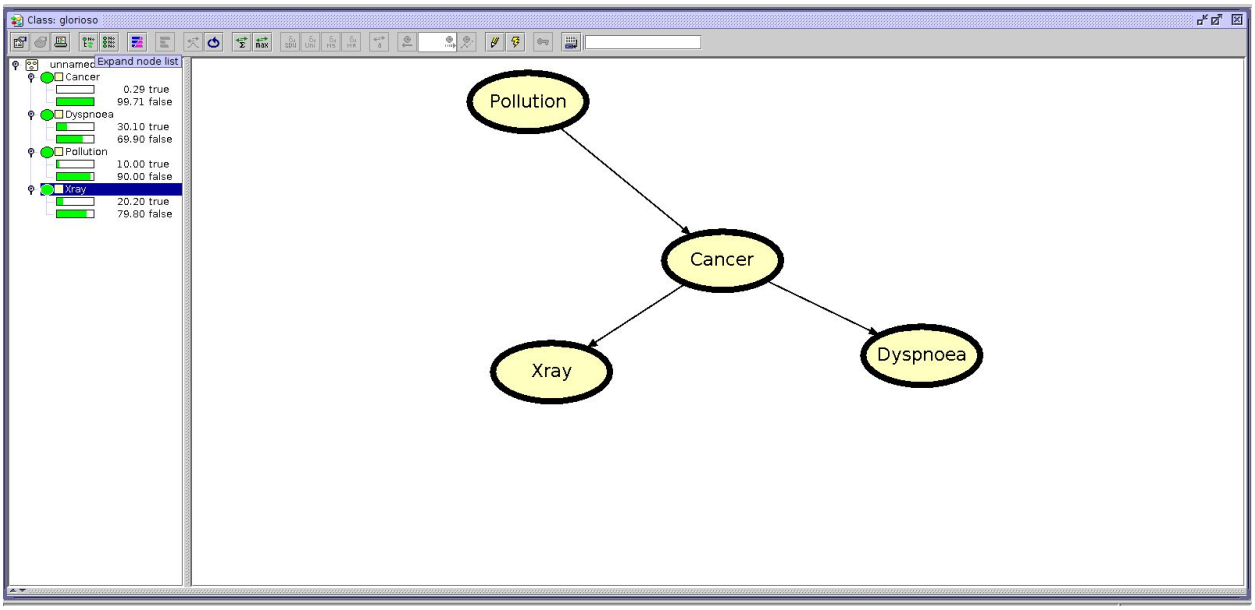
expensive on memory and startup time, however, this is obviously offset by the extra cost taken by each query to complete if many nodes are unknown.

For quick usage; we'll use our implementation since it's easier to just choose our queries and avoid computing unnecessary information. Maybe the redeeming factor of Hugin is just how it's easier to create new nodes and add their probabilities.

Custom Test Model



Nodes



Graphical Representation