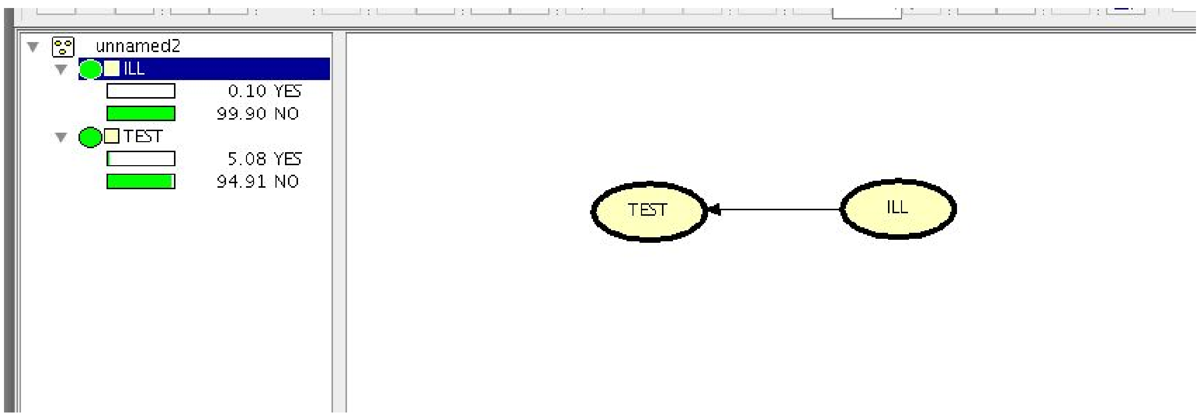
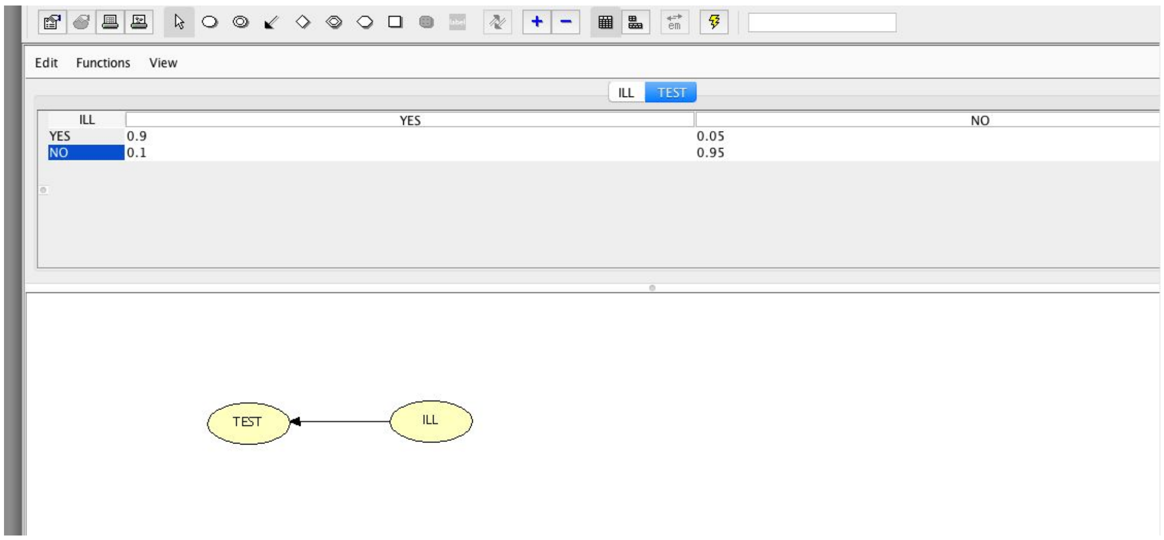
Alberto Lopez Pedroza, Julián Niebieskikwiat Godoy, Marco López

Results table

Query	Ruben's Results	Hugin's result	Our results
+Ill	0.001	0.001	0.001
-Ill	0.999	0.9990	0.999
+Ill | + Test	0.0176991	0.017	0.176691
+Test	0.0585	0.058	0.0585
-Test	0.94915	0.9491	0.94915
+Test | +Ill	0.9	0.9	0.9
+Test | -Ill	0.05	0.05	0.05

ILL -> TEST Diagram


unnamed2
ILL
0.10 YES
99.90 NO
TEST
5.08 YES
94.91 NO


Edit Functions View

ILL TEST

ILL	YES	NO
YES	0.9	0.05
NO	0.1	0.95

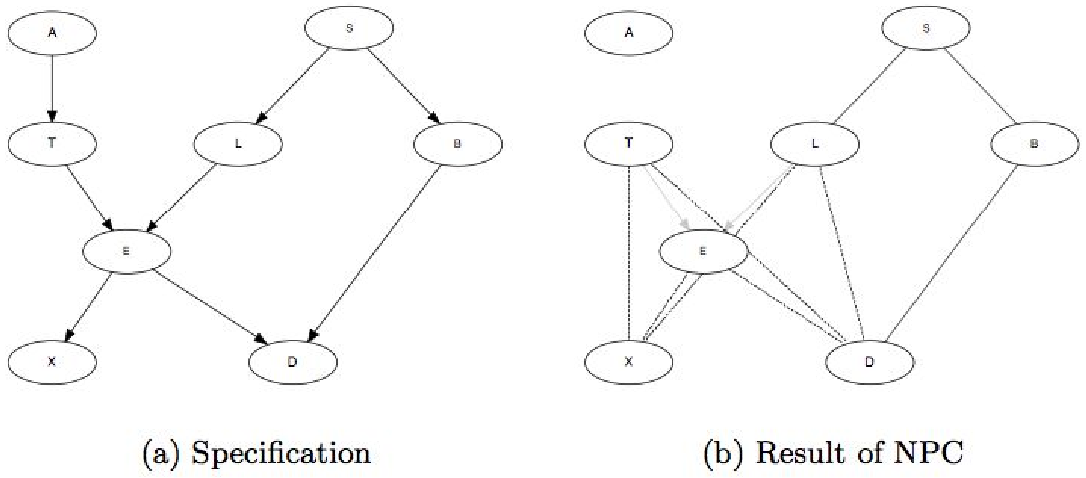
DIFFERENCES

For obvious reasons, Hugin is more complete in its set of features, testing, algorithm selection, optimization, runtime and extra features such as approximations based on Boyen-Koller algorithm and approximations using different probabilistic functions.

The main difference lays on how the algorithm works, which is explained on the next section:

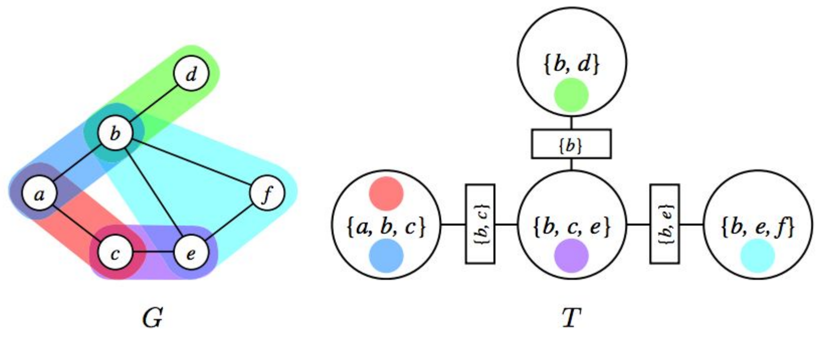
ALGORITHM

When doing constraint-based approach for bayes netowrks,, structured learning is supported, which leaves PC and IC as two valid algorithms. Hugin uses PC algorithm as well as a proposed NPC custom approach which basically analyses a graph network and then identifies DAG cyclic structure to propose a new, more clean, not redundant network for better performance when doing large class queries as well as large data set simulations. [1]



[1]

After cleaning all the unused and irrelevant variables on the previous step, the Hugin Decision Engine makes use of the Junction Tree algorithm to convert the resulting graph into a non-directed one, then treating it again to find a set of of node clusters by the application of the maximal elimination algorithm. [2]



A cluster graph T is a **junction tree** for G if it has these three properties:

1. **singly connected**: there is exactly one path between each pair of clusters.
2. **covering**: for each clique A of G there is some cluster C such that $A \subseteq C$.
3. **running intersection**: for each pair of clusters B and C that contain i, each cluster on the unique path between B and C also contains i.

[2]

Once the clusters have been identifying the message passing protocol is used, which basically states that the clusters can only communicate with other neighbor after all of the other neighbors have finished passing all of their messages to it (this is done when calculating all of the probabilities).[2]

This can be done with two algorithms, the Shafer-Shenoy, which basically multiplies all of the belief/potential of every cluster that's not the receiving one as well as marginilizing its receiving cluster's variables , and then sends the result to the receiving cluster. This means that if I have a graph with N nodes, I clean them and I end up with 5 clusters and I want to know a query that involves a node in a specific cluster A, I'm gonna end up multiplying all of the other cluster's potentials, excluding A's variables, and the Hugin algorithm, which is basically a better Shafer-Shenoy without having to multiply all of the messages every time a cluster belief calculation is done, making it more efficient than Shafer-Shenoy when a set of conditions is met, specifically, when the graph's treewidth is not very large. [2]

The difference with our approach is explained next:

COMMON BASES

While all of them come from the bayes theorem , Hugin does a very well done simplifying and optimizing the node network by cleaning it , grouping and calculating the probabilities based on clusters . Our algorithm visits all of the parent nodes, passing around the probabilities and multiplying on every node.

TOOL FOR REAL LIFE CASES

If I don't have any money, I'd go with libDAI, while it's not the most complete approximate inference software, it's a more than reliable inference engine, which provides support for many techniques like brute force enumeration and junction-trees.. And its free. Bad thing is , it doesn't support more than 2 input files and it has no GUI.

If I had a choice and money I'd go with Bayesserver.com while it's a bit expensive , its universally supported, both for its API's and different languages implementation as well as having one of the best GUI's I've seen (compared with Hugin) but then again, depends on which is the need of the client and if we need structured learning

Sources:

[1] Anders Madsen, Lang, Kjærulff,Jensen, The Hugin Tool for Learning Bayesian Networks. Hugin Expert A/S. Niels Jernes Vej 10, Alborg, Denmark. [Publication] Retrieved via web:

http://people.cs.aau.dk/~uk/papers/madsen-etal-03.pdf

[2] Paskin. The Junction Tree Algorithms. Stanford University. A Short Course on Graphical Models. [Course Presentation] Retrieved via web: http://ai.stanford.edu/~paskin/gm-short-course/lec3.pdf