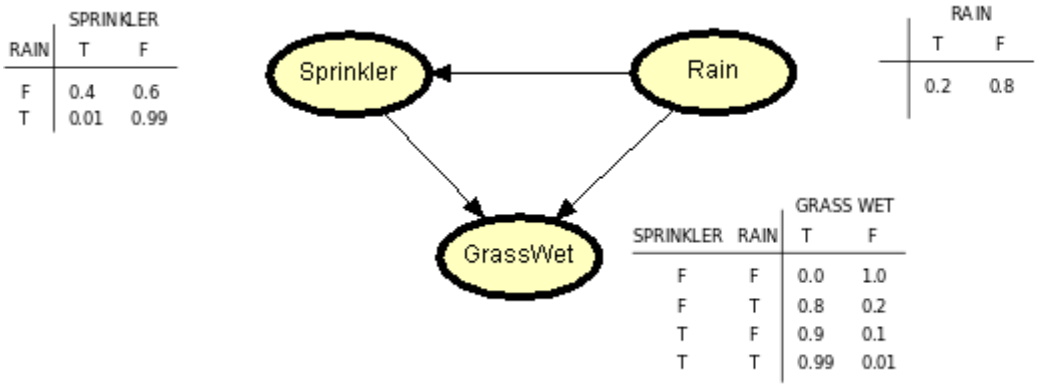
Ángel Alberto Flores Rodríguez A01201645
Antonio Esper Cook A01202743
Francisco Nuñez Gómez A01202727

Bayes Network diagram



Comparison between Hugin Lite and our code

Query	Hugin Lite	Our Code
+Rain	0.2	0.2
-Rain	0.8	0.8
+Sprinkler	0.322	0.322
-Sprinkler	0.678	0.678
+GrassWet	0.4484	0.44838
-GrassWet	0.5516	0.55162
+GrassWet|+Sprinkler	0.9006	0.900559
+GrassWet|-Sprinkler	0.2336	0.233628
+GrassWet|+Sprinkler, +Rain	0.99	0.99
-GrassWet|+Sprinkler, -Rain	0.1	0.1

As seen in the table above, the results are nearly the same. Some results vary due to the calculations and type of variables. Our code handles doubles, so it will express the result more accurate, while with the Hugin Lite tool it expresses the result in percentage or -e notation, giving two decimals or 4 with the exponential notation. While Hugin uses the PC algorithm to remove independence from the queries, we implemented the enumeration algorithm. Though different algorithms are used, they both run the queries and extract the valuable information. For the PC algorithm, they remove independence between nodes if any. This is done with every node causing independence between variables, so its computationally inefficient, because the number of tests it does grow exponentially in the number of variables or nodes in the network. The enumeration algorithm takes the original network and “rewrites” it into a general probability function, given by the sum of all known and unknown nodes. For the unknown variables, it collects all possible outcomes for each nodes and evaluates them with the known nodes. This can be very inefficient if the network is composed of many nodes. In fact, for each query it takes, the number of functions it must run will grow 2^n where n is the number of unknown (hidden) nodes. While the emuneration algorithm can perform well in small networks, it will require more time to find the answer to some queries with big networks as explained above. The PC algorithm can be useful

in this situations when getting the answer due to its tendency to remove independent nodes, but causing finding independence and removing it to be very inefficient