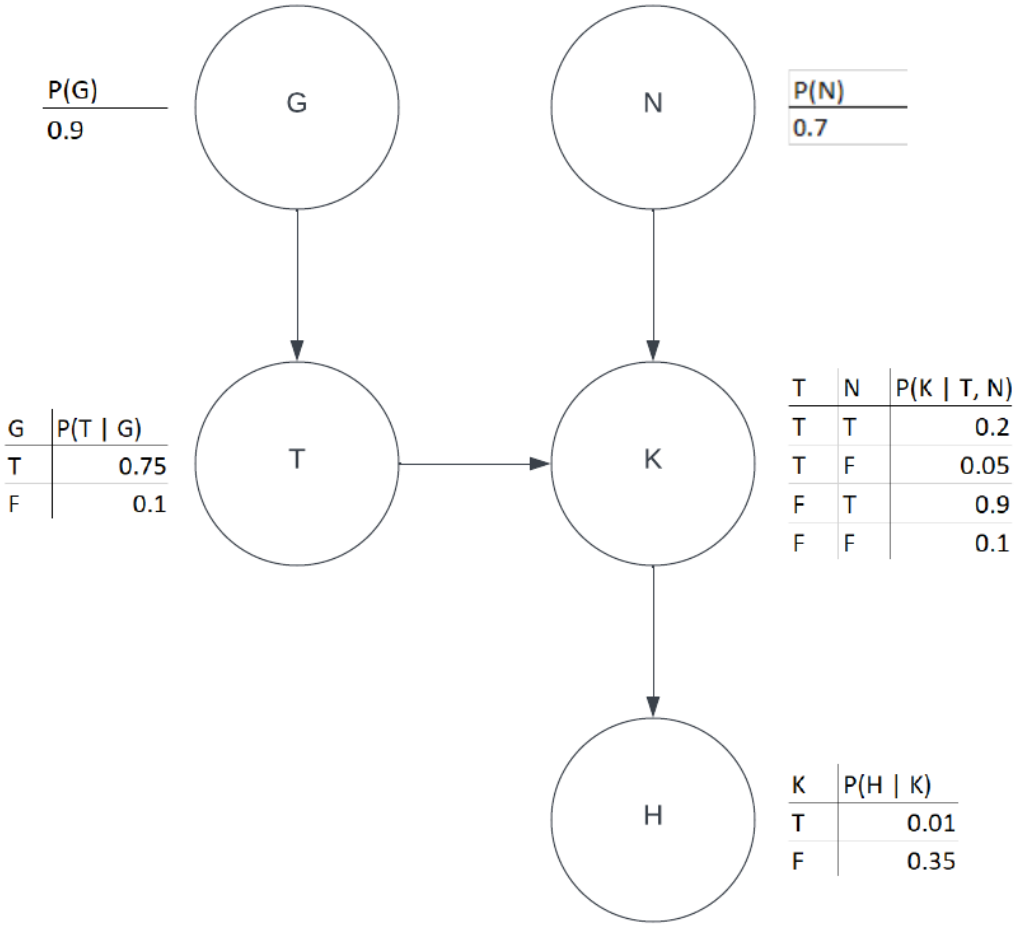
Andrew Davison
CSC 372: Survey of Artificial Intelligence
Dr. Thomas Allen
4/10/2023

A5: Bayesian Networks

Part 1: Designing a Bayesian Network Graph



P(G)
0.9

P(N)
0.7

G	P(T | G)
T	0.75
F	0.1

T	N	P(K | T, N)
T	T	0.2
T	F	0.05
F	T	0.9
F	F	0.1

K	P(H | K)
T	0.01
F	0.35

Part 2: Implementing a Bayesian Network

Code Design and Methodology

I wrote my code in C++ using the exact full joint distribution inference method and the approximate rejection sampling method. My code works by taking input from the command line and storing information in a C++ vector; this is the “assignment” data type. The information is held in a strict topological order, such that the first entry corresponds to the Burglary node, then

Earthquake, then Alarm, then JohnCalls, then MaryCalls. Each slot in the array is given a number 0-4 inclusive. 0 indicates that the value is false and 1 indicates that the value is true according to the query. 2 and 3 indicate false and true, respectively, but also that this node is in the "given clause" of the query. Finally, a value of 4 indicates that this node did not have a specified truth value in the query.

The CPTs of each node were stored in functions titled after the name of the node, taking the truth value of the node and its parents (ex. Alarm(a, b, e)). This function then returns the corresponding probability in the CPT.

The program constructs a full joint distribution through 5 nested loops -- each executes twice – that calculates the corresponding probability of each possible event. This construction happens once every time the program runs. Inference is done by examining the table to find each event that maintains the truth of the input and summing them all together, including keeping a second accumulator for the "given clause" if applicable, dividing at the end according to the definition of conditional probability.

Similarly, the program executes approximate rejection sampling by randomly generating events and then "counting" or "not counting" the event depending on whether it is consistent with the input. There is an accumulator in the rejection sampling program for both the number of accepted trials and the number of accepted trials according to the "given clause." Currently, rejection sampling runs through 10,000,000 iterations of sample generation.

Output

$$P(B, \neg A \mid \neg M)$$

```
PS C:\Users\Andrew Davison\Documents\davison_cs372_AI\BayesianNetworks> ./bayes.exe Bt Af given Mf
Exact inference by enumeration found probability 0.00163428
Approximate inference by rejection sampling found probability 0.00162862
```

$$P(\neg A, E)$$

```
PS C:\Users\Andrew Davison\Documents\davison_cs372_AI\BayesianNetworks> ./bayes.exe Af Et
Exact inference by enumeration found probability 0.018834
Approximate inference by rejection sampling found probability 0.0187757
```

$$P(J, \neg A \mid B, \neg E)$$

```
PS C:\Users\Andrew Davison\Documents\davison_cs372_AI\BayesianNetworks> ./bayes.exe Jt Af given Bt Ef
Exact inference by enumeration found probability 0.0056
Approximate inference by rejection sampling found probability 0.00548905
```

P(B, ¬A, ¬M, J, E)

```
PS C:\Users\Andrew Davison\Documents\davison_cs372_AI\BayesianNetworks> ./bayes.exe Bt Af Mf Jt Et
Exact inference by enumeration found probability 1.2348e-06
Approximate inference by rejection sampling found probability 1.1e-06
```

What I Learned

Through this assignment, I have a much better understanding of how Bayesian Networks operate and, in general, how Bayes' law can be used to simplify computation and probability models. Additionally, I've become a better C++ programmer as I become more comfortable with the capabilities of the language.

Acknowledgements and References

Russell, Stuart, Stuart Jonathan Russell, Peter Norvig, and Ernest Davis. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2010.

Algorithms derived from this book.