Brad Rafferty
brjrafferty@gmail.com

**Bayesian Structure Learning Implementation Summary**

To learn the structure of the Bayesian network, I have implemented the K2 search algorithm in Python. The general process of the entire tool is as follows:

1. Initialize a directed acyclic graph with number of nodes equal to the number of variables in given dataset
2. For each node, consider set of parents and choose the one that most improves the score without adding a cycle into the graph
3. Once a node has reached a prescribed maximum number of parents, continue to the next node
4. Once all nodes are considered, end analysis

To build the directed acyclic graphs as well as check for cycles, the NetworkX package is very useful. In order to reduce runtime due to time constraints, a greedy search that chooses the first parent to add to the current node of interest was implemented. The drawback of this is that you potentially miss local optima at each step; instead, a more optimal strategy could be to consider all parents and choose that which maximized the score at that step. A way to vastly improve the runtime would be to forego the enumeration and indexing of all parent instantiations for each step and instead use a vectorization scheme such as apply_along_axis. Additional ways to improve the resultant structure could include: multiple randomized restarts (of node ordering), and feeding the K2 graph result as a starting graph for local search. The output belief structure for the example.csv is plotted in Figure 1.

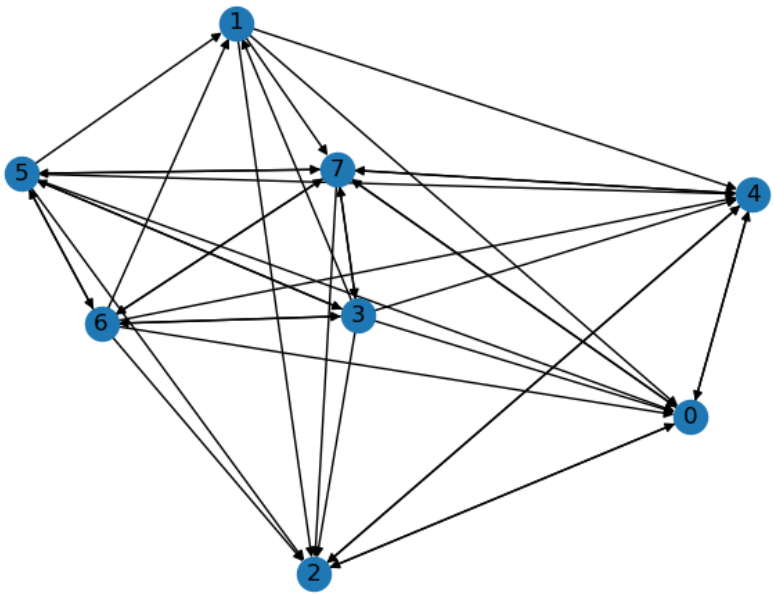


*Figure 1.* Belief structure for a 7-node graph from example.csv