

Properties of a DAG:

- All edges are directed (if A is connected to B then B is not necessarily connected to A)
- The graph must be acyclic

Properties That Allow Us to Use Dijkstra's

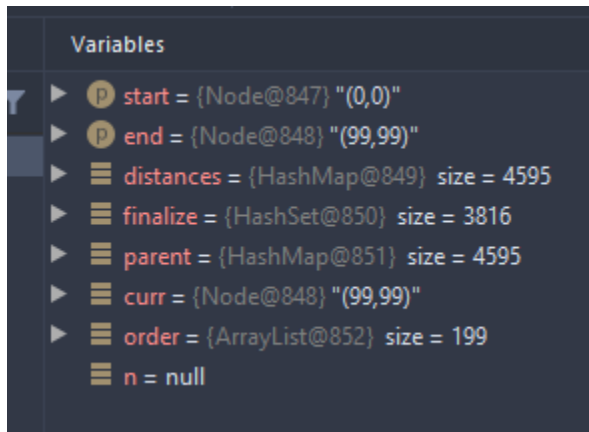
- The edges have weights
- The weights are non-negative
- The graph is connected to the extent that there exists a path such that the starting node and ending node are connected by that path.

Admissible heuristic for maze:

- An admissible heuristic for the maze is the Manhattan distance between the current node and the destination node.
- This is the theoretical shortest distance a node can have to a destination so it will therefore always underestimate the distance to destination node. The function will also always return the same value for a particular input value.

Extra Credit:

- A* : finalized had 3816 elements



- Dijkstra's

```
george@betty:~/435$ java Main
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
    at java.util.HashMap.resize(HashMap.java:704)
    at java.util.HashMap.putVal(HashMap.java:663)
    at java.util.HashMap.put(HashMap.java:612)
    at WeightedGraph.addWeightedEdge(WeightedGraph.java:20)
    at Main.createRandomCompleteWeightedGraph(Main.java:19)
    at Main.main(Main.java:5)
```

The nature of the two graphs were extremely different however the space complexity is similar for my two algs and the code was ran on the same # of nodes. Considering that Dijkstra's ran out of memory – we can reasonably assume that it required a lot more space to operate and therefore also had a large finalized array. This a huge difference and can illustrate how A* improves on Dijkstra's alg