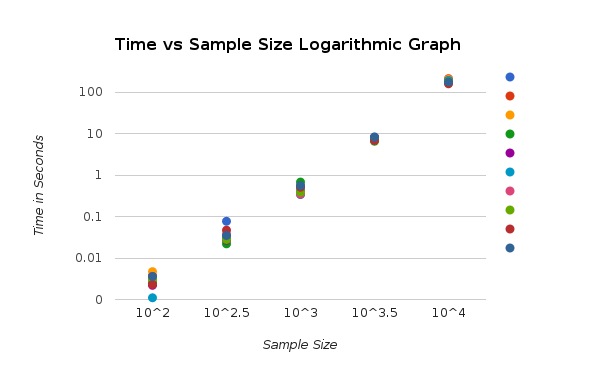
Empirical Results

We wrote the algorithm and problem set generation code in python. It was run on a standard GCC laptop running Arch Linux. The problem sets were generated in the manner described in the spec, choosing a random starting node and depth-first expansion from there. The problem sets were then run using the same starting node as was used for generation.



Discussion

One interesting thing was how the spread of the running times converged as the problem sets got larger. This actually supports our given average running time of O(E) for this algorithm, as the spread of edges for the number of vertices taken is vastly different with a small number of vertices, but gets closer and closer the more vertices we used. The data we used was relatively sparse, so the algorithm ran quite fast, and thus was not close to the theoretical worst case of O(VE).