Lab 1: Sorting Algorithms

hod12, hortoa5

February 2026

Contents

List of Figures

0.1 Executive Summery

- Learned the relationship between minimum vertex covers and maximum independent sets
- Learned how to design our own experiments
- Gained a further understanding of how DFS and BFS are implemented
- Learned the relationship between cycles and the number of edges/vertices in a graph
- Learned the relationship Connections and number of edges/vertices in a graph

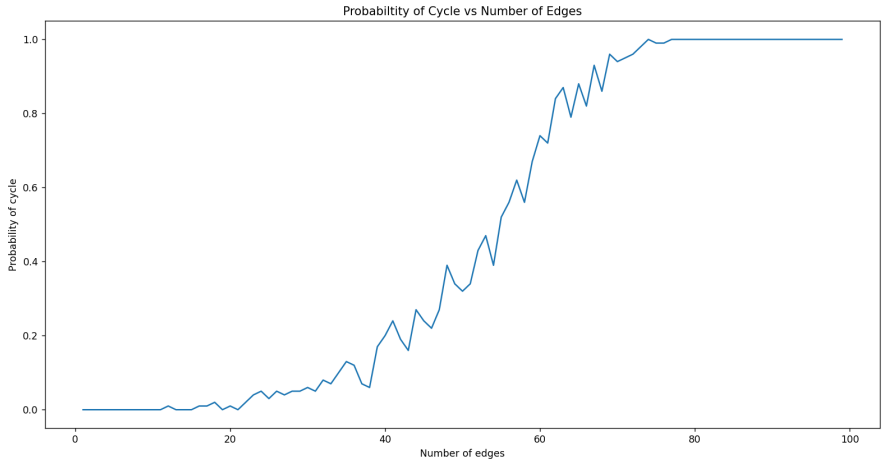
0.2 Part 1

0.2.1 BFS and DFS

0.2.2 Experiment 1

This experiment was run with lists of randomly generated graphs of length 100. Each graph had 100 nodes and the number of edges in each list increases by 1 from 1 to 100. The probability of a cycle was calculated by computing number of graphs with a cycle over total number of graphs in each sub-list of graphs. This experiment was run 5 times before the current data was taken.

Figure 1:

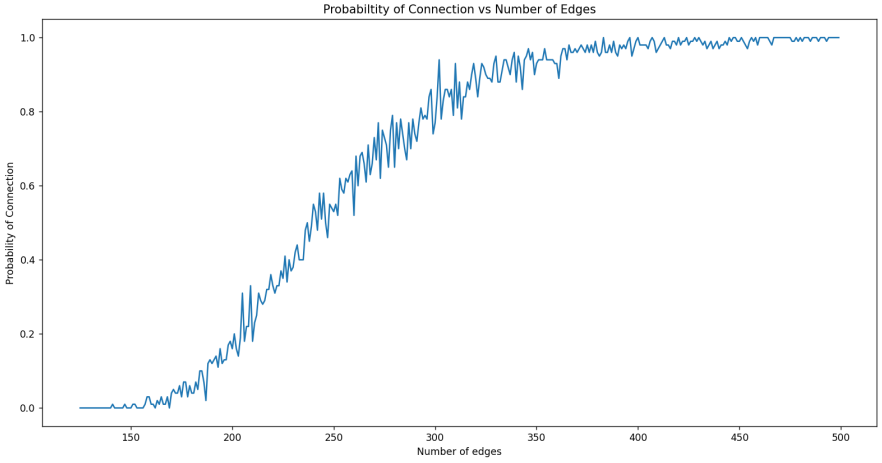


As you can see in Figure 1, the odds of a graph having a cycle with less than 40 edges is quite low. However, once you pass 80 edges it is almost guaranteed. This makes sense as our graphs each have 100 nodes so the closer to 100 edges you get the much more likely you are the have a cycle. In fact, there is likely a point where you mathematically have to have a cycle based on the number of edges for each node. In this scenario it looks to be around 80.

0.2.3 Experiment 2

This experiment was run with lists of randomly generated graphs of length 100. Each graph had 100 nodes and the number of edges in each list increases by 1 from 125 to 500. The probability of connection was calculated by computing number of graphs that are connected over total number of graphs in each sublist of graphs. This experiment was run 5 times before the current data was taken.

Figure 2:



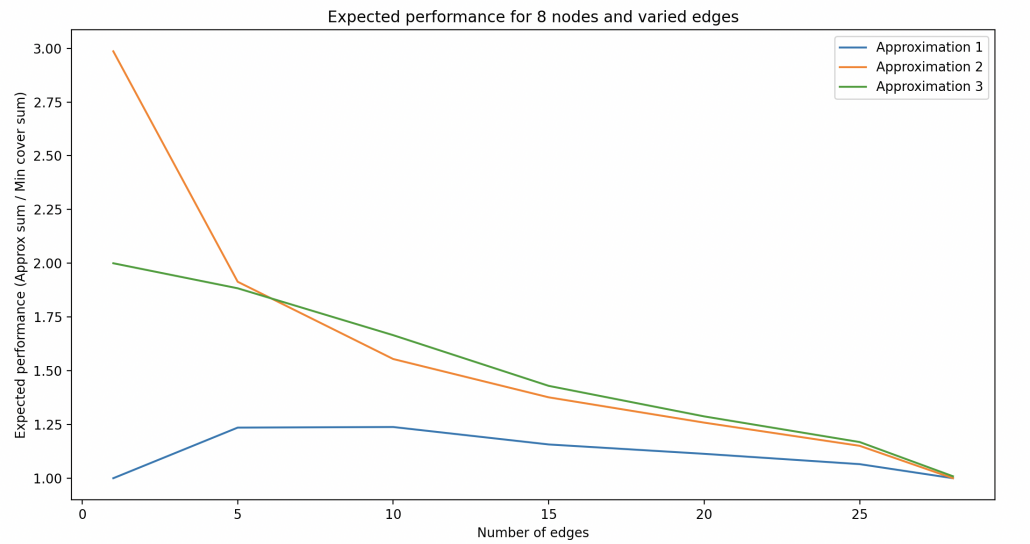
As you can see in Figure 2, the probability of a graph being connected starts to go up from 0 around 150 edges. Slowly approaching 100 around 400 edges. This makes sense because being connected is a much stronger relation than having a cycle. It is easy for a graph to have a cycle but not every component be connected. So it is no surprise that a lot more edges is needed to increase the odds of the graph being connected. It also makes sense that under 100 edges the probability is 0 as there just isn't enough edges to connect all 100 nodes.

0.3 Part 2

0.3.1 Varied edges

Constructed 1000 graphs with 8 nodes for varying edge amounts (1,5,10,15,20,25,28). For each of these graphs I computed the Minimum Vertex Cover and the 3 approximations, then computed the size of each of these covers. Finally, I computed the expected performance (approx size / minimum cover size). The experiment was run 5 times before the following figure was captured.

Figure 3:



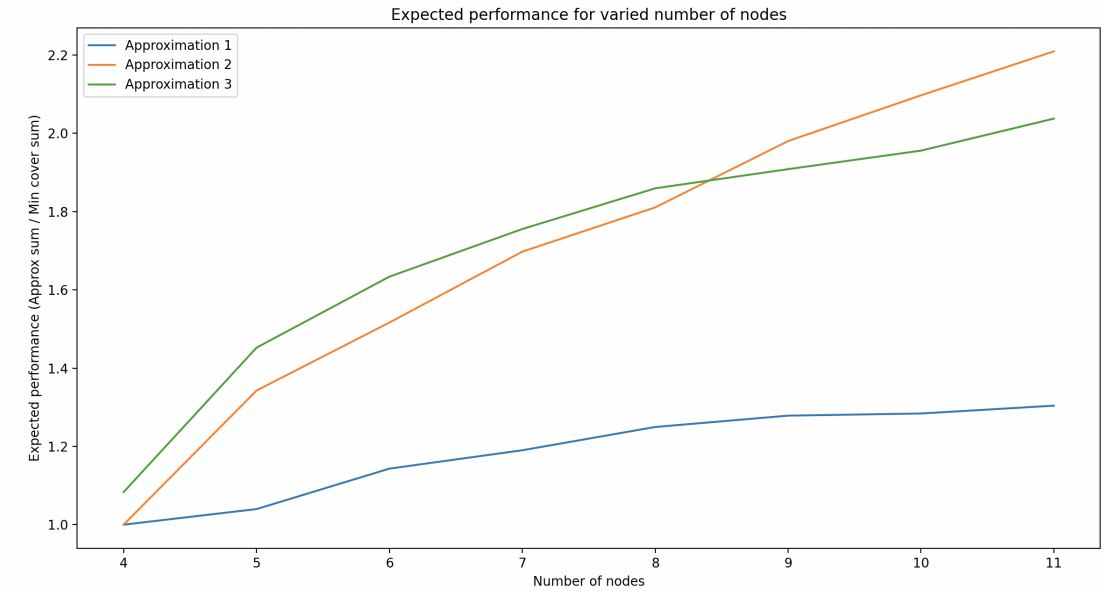
A y-value closer to 1 represents a better approximation, as the approximations size is closer to the minimum cover size, thus the ratio produces a result closer to 1. Meanwhile a larger number means the approximation is larger than the minimum cover. As the number of edges approaches the maximum number of edges, the approximations all converge to the minimum cover size. As the graph becomes less sparse, the approximation becomes closer and closer to the minimum vertex cover. Therefore, based on this experiment, as the number of edges increases, the approximation does as well.
Side note: Approximation 1 appears to perform the best out of all the approximations.

0.3.2 Varied nodes

Constructed 1000 graphs with a fixed amount of edges (6) and varying node amounts (4, 5, 6, 7, 8, 9, 10, 11). Similar to the varied edges experiment, I

computed the minimum vertex cover along with the 3 approximations, then finally computed the expected performance for each of these approximations. The experiment was run 5 times before the following figure was captured

Figure 4:

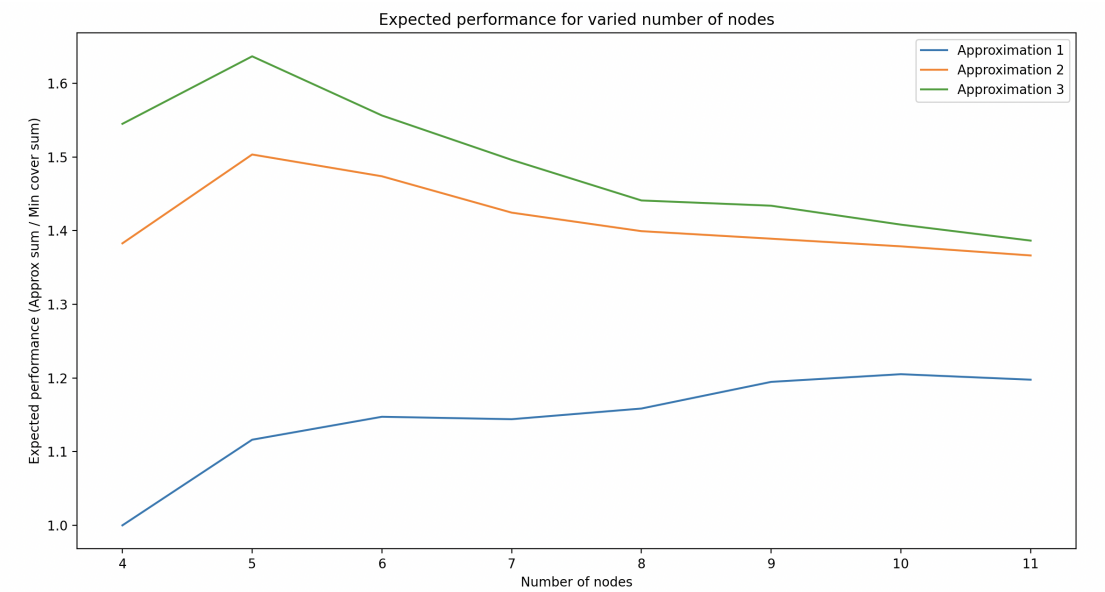


Based on this figure, when you increase the number of nodes for a fixed number of edges across all graphs the approximations become worse. However, I believe this is less to do with how many nodes the graph has and how sparse it is. Since increasing the number of nodes while keeping the number of edges constant makes the graph more sparse.

0.3.3 Proportional experiment

This uses the same code as the varied nodes experiment however, the number of edges is proportional to the number of nodes in the graph. It is kept as half the max number of possible edges. This is so each graph has relatively the same sparsity. The goal is to see if the number of nodes has such an obvious impact on the approximations as previously seen. The experiment was run 5 times before the following figure was captured.

Figure 5:

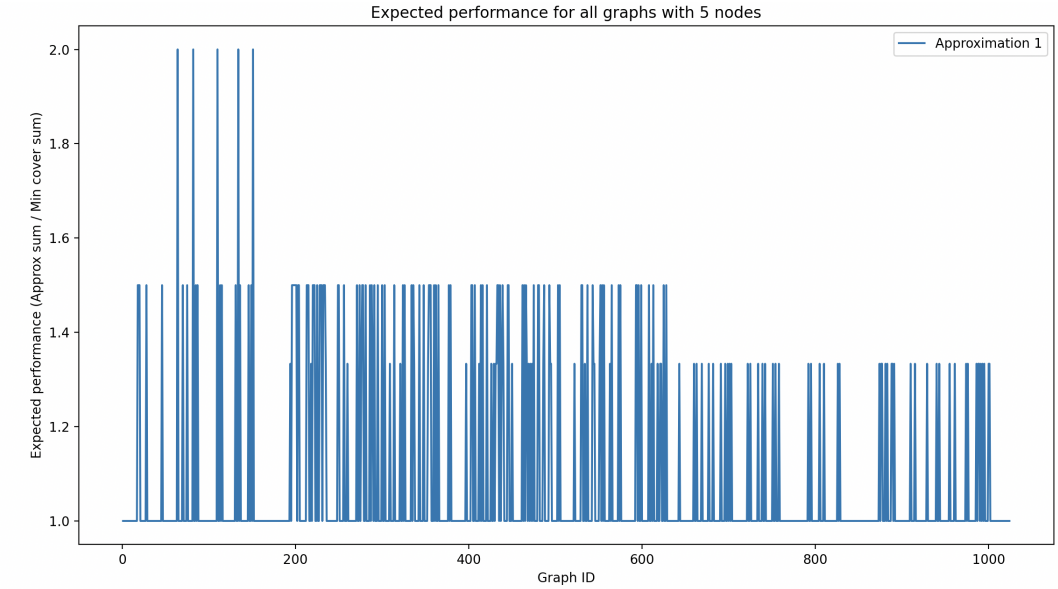


Now that each graph has number of edges proportional to its number of nodes, the decrease in expected performance isn't as obvious. They stay relatively the same. So the number of nodes might not have a direct impact on the expected performance, it is most likely a combination of multiple factors (number of nodes, number of edges, ...)

0.3.4 Worst case

To generate all graphs with 5 nodes, I computed all possible combinations of pairs of nodes. Then using that I computed all possible subsets of those pairs, and then I computed a graph for each one of those subsets. For each graph I computed its expected performance to discover if there were any graphs that performed particularly poorly. The experiment was run 5 times before the following figure was captured.

Figure 6:



I ran this experiment several times, and there were 5 graphs that consistently had an expected performance above 1.8, and almost at 2. They were graphs with edges:
Graph 1: (0, 1), (0, 2), (3, 4)
Graph 2: (0, 1), (1, 2), (3, 4)
Graph 3: (0, 2), (1, 2), (3, 4)
Graph 4: (0, 3), (1, 3), (2, 4)
Graph 5: (0, 4), (1, 4), (2, 3)
They all only have 3 edge, which is not a lot compared to the maximum of 10 possible edges.

0.3.5 The Independent Set Problem

I found that the size of the minimum vertex cover and the maximum independent set always summed to equal the total number of vertices in the graph. When I was running my experiments I noticed that sometimes vertices would be shared between the MVS and the MIS. Which is strange because they always sum to the total number of vertices. What I concluded is that a graph must be able to have multiple MVC or MIS. Every MIS is the complement of a MVS but, empirically, we do not always find the one that is the complement of the other since there can be multiple MVC and MIS for any one graph. This observation makes sense because by definition they are opposites of each other. If b is a minimum vertex cover than v-b is a maximum independent set.

0.4 Appendix

graph.py

Houses experiments 1 and 2 (functions named experiment1(), experiment2()) along with all of the implementations of BFS2, BFS3 and isconnected() under the experiments note. DFS2, DFS3 and hascycle() is up in the code where DFS is defined and is marked by *** that it is our code.