2XC3 Lab 4 + 5

Investigating Properties of Implemented Graphs and Graph Algorithms
February 26, 2023

Andy Luo
luoa22@mcmaster.ca

Kieran Henderson
hendek12@mcmaster.ca

Corinna Lin
linc94@mcmaster.ca

Table of Contents

Table of Figures

Summary

- The number of nodes in a graph do not have a significant effect on cycle probability, however, the edge to node proportion does
    - A ratio of 2:1 nodes and edges will have a low chance of cycles
    - A ratio of 1:2 nodes to edges will almost guarantee a cycle
- Edge to node proportions are not the only variable that affects the connected probability
    - For a graph with high chances of cycles, a ratio of 1:4 nodes to edges
    - A ratio of less than 1:2 nodes to edges almost guarantees that your graph will not be completely connected.
- Approximation 1 is considerably better, however, is likely due to a small sample size, as we know that as the graph gets larger approximation 2 and 3 will outperform approximation 1
- For a given graph, though there may be more than one MVC and MIS, the MIS will always be the complement of the MVC
    - i.e. if you remove the nodes in the MVC from the graph, you will get an MIS, and vice versa

Part 1

Experiment 1

From my experiment, I have determined that the cycle probability for a graph of a given number of nodes has a sigmoid curve like growth (exponential growth to logarithmic flattening). From further testing, I have also determined that the size of the graph has little to no effect on cycle probability and that only the edge to node proportion affects the cycle probability.

For the first experiment I ran, I had a fixed size of 100 nodes, generated 1000 graphs to calculate cycle probability for a given number of edges, and had edges numbering from 0-300 with steps of 1 (*Figure 1*). What I conclude from this experiment is that before 50 edges, we had an extremely low likelihood of having cycles. However, after 50, there is a huge growth of the probability of having a cycle with it eventually plateauing at around 200-250 after reaching a 100% chance. This indicates that if you want to have a graph that has a low chance of cycles, you should have a ratio of 2:1 nodes to edges and that a ratio of greater than 1:2 nodes to edges essentially guarantees that your graph will have a cycle.

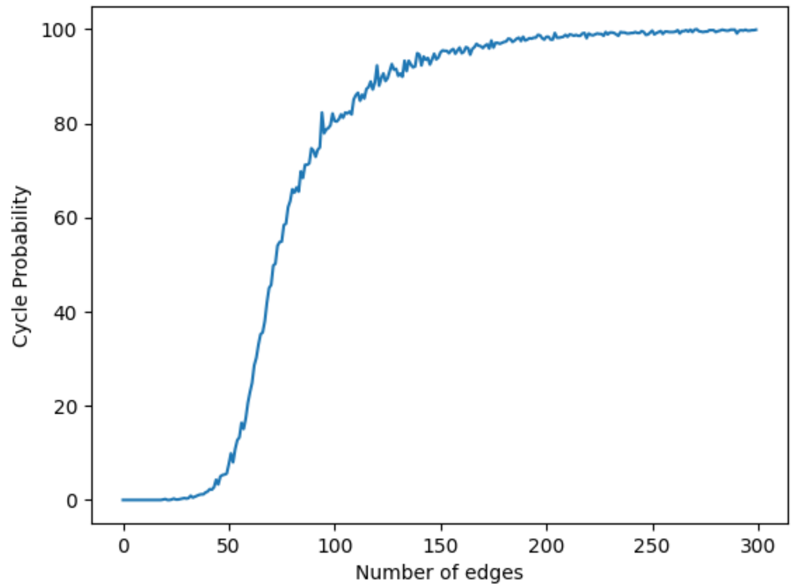

Figure 1. Cycle Probability vs Number of Edges

For the second experiment I ran, I had fixed sizes of 10, 100, and 1000 nodes, generated 100 graphs to calculate cycle probability for a given proportion of edges, and had edge proportions numbering from 0-2 with steps of 0.01 (*Figure 2).* The purpose of this experiment was to verify that node size had no influence over cycle probability and that cycle probability was purely dependent on the proportion of nodes to edges. Based on (*Figure 2),* the fact that all the graphs basically perfectly line up lets me conclude that node size indeed had no influence over cycle probability.

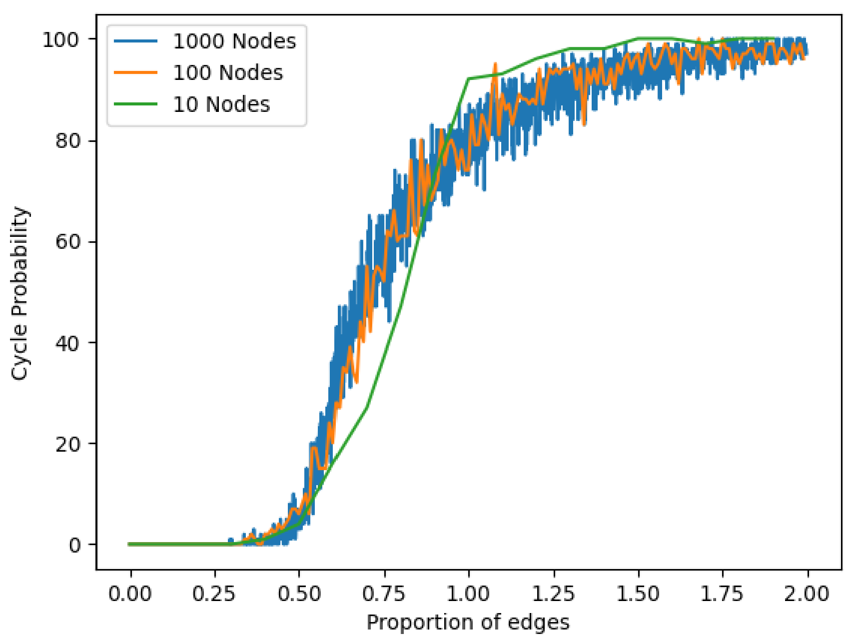

Figure 2. Cycle Probability vs Proportion of Edges

Experiment 2

From my experiment, I have determined that the connected probability for a graph of a given number of nodes has a sigmoid curve like growth (exponential growth to logarithmic flattening). From further testing, I have also determined that the size of the graph has a notable effect on connected probability and that the edge to node proportion is not the only variable that affects the connected probability.

For the first experiment I ran, I had a fixed size of 100 nodes, generated 100 graphs to calculate cycle probability for a given number of edges, and had edges numbering from 0-600 with steps of 1 (*Figure 3*). What I conclude from this experiment is that before 200 edges, we had an extremely low likelihood of having complete connected graphs. However, after 200, there is a huge growth of the likelihood of it with it eventually plateauing at around 400-450 after reaching a 100% chance. This indicates that if you want to have a graph that has a high chance of cycles, you should have a ratio of 1:4 nodes to edges and that a ratio of less than 1:2 nodes to edges essentially guarantees that your graph will not be completely connected.

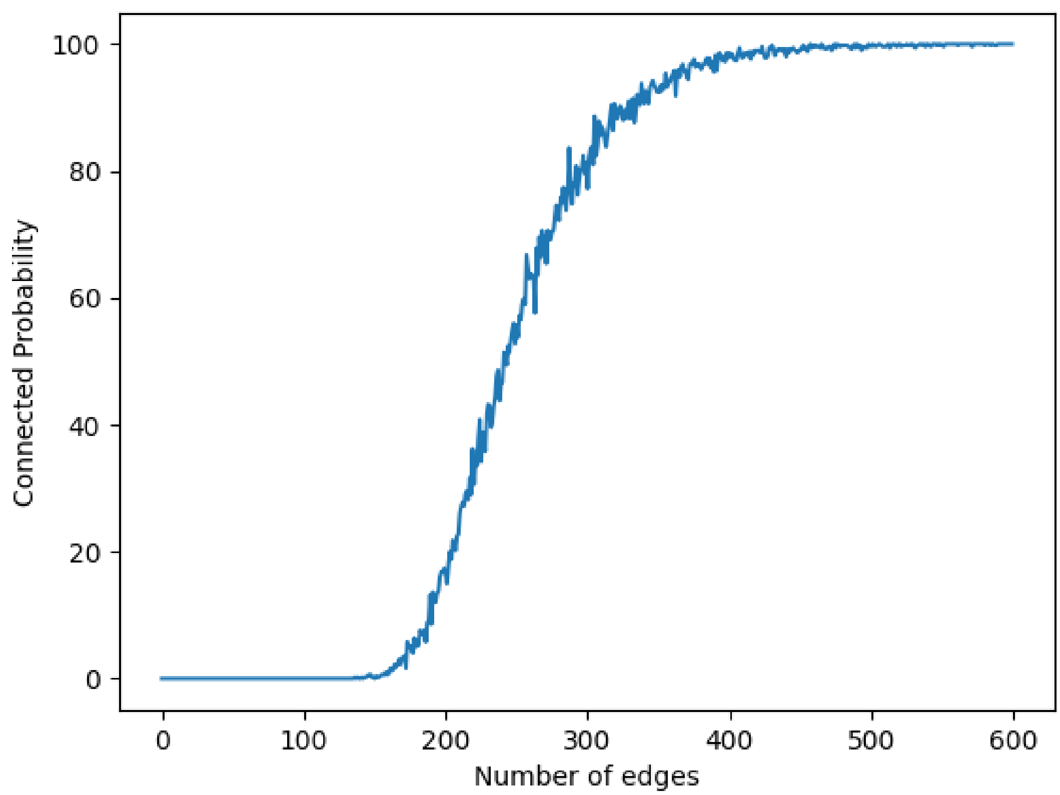

Figure 3. Connected Probability vs Number of Edges

For the second experiment I ran, I had fixed sizes of 100, 200, and 300 nodes, generated 100 graphs to calculate cycle probability for a given proportion of edges, and had edge proportions numbering from 0-6 with steps of 0.01 (*Figure 4).* The purpose of this experiment was to identify the influence node size had over connected probability. Based on (*Figure 4),* the fact that all the graphs are not perfectly aligned lets me conclude that node size has an influence over connected probability and that as you increase node size, it takes a higher proportion of nodes to edges to guarantee connectedness. Based on the fact that the graphs look identical, I can conclude that node size does not influence the growth of the increase in connected probability.

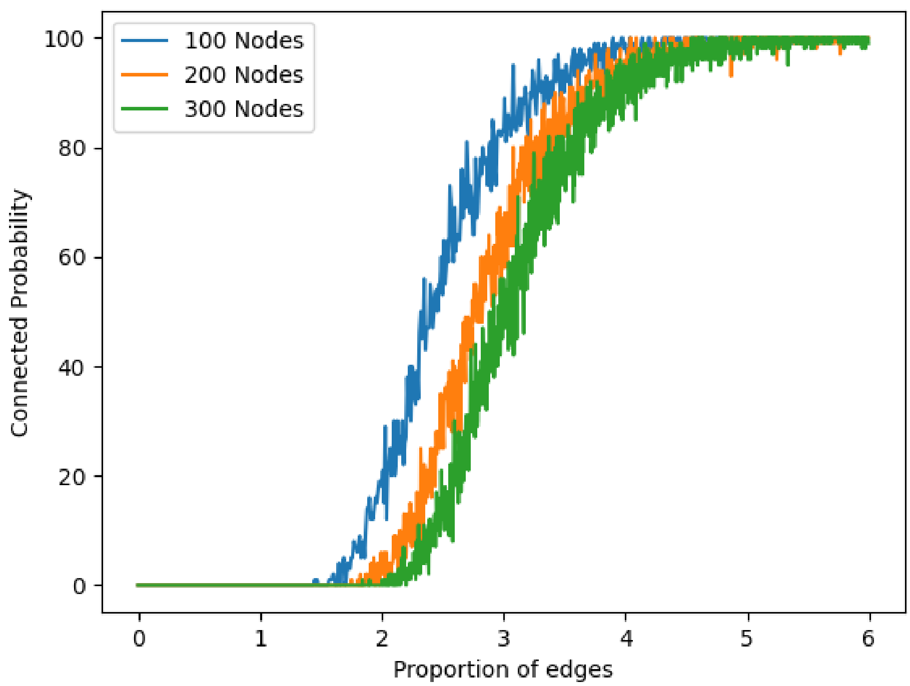

Figure 4. Connected Probability vs Proportion of Edges

Part 2

Approximations and their Experiments

To test the different approximation approaches, the first test we ran had a fixed node count of 8 and a variable edge count from 1 to 30 on 100 different graphs, the second test we ran had a fixed edge count of 10 and a variable amount of nodes from 5 to 15 on 100 different graphs, the final test we ran was to determine the worst case scenarios for each approximation by generating all possible graphs of size 2, 3, 4, and 5 then taking the average of 1000 runs of the process to help minimize the random effects of approximation 2 and 3 to get the worst case scenarios.

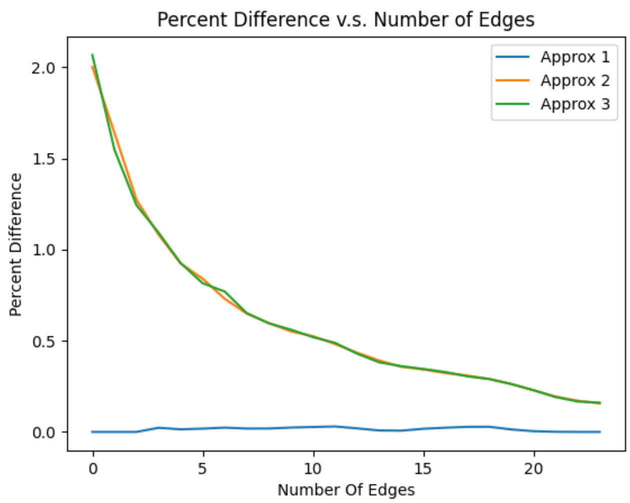

Figure 5. Percent Difference vs Number of Edges

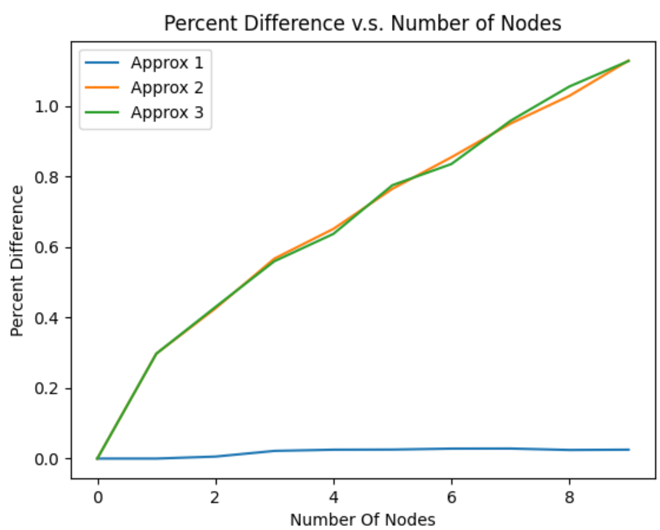

Figure 6. Percent Difference vs Number of Nodes

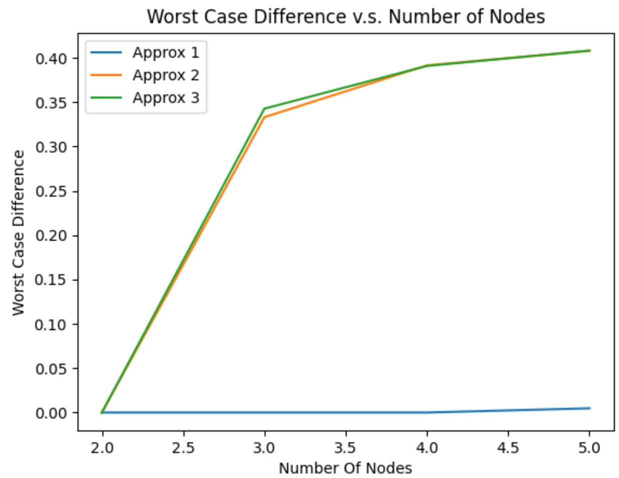

Figure 7. Worst Case Difference vs Number of Nodes

Based on the experiments it appears that approximation 1 is considerably better, however, as we discussed in class this is likely due to a small sample size as we know that as the graph gets larger approximation 2 and 3 will outperform approximation 1. This is because the difference in approximation 1 will become slightly worse for every extra node and edge a graph has meaning that it will eventually get too 300% or more error even though the error is so small with our sample size. On the other hand, Approximations 2 and 3 will continue towards an asymptote the more nodes and edges there are, meaning that they are capped and will not exponentially grow in percent difference. Additionally, approximation 2 and 3 are very similar, we believe this is due to them being fundamentally the same approximation method with the random node choices.

Independent Set Problem

For this experiment, I implemented a function that will return the maximum independent set of a graph. It functions very similarly to the implementation of the minimum vertex cover (quite literally, it's the same function with the 'not' removed from the conditions). The experiment had the following parameters:

- 20 randomly generated graphs
- Each graph with an increasing number of nodes from 0-19
- Both MIS's and MVC's were given on the same generated graph

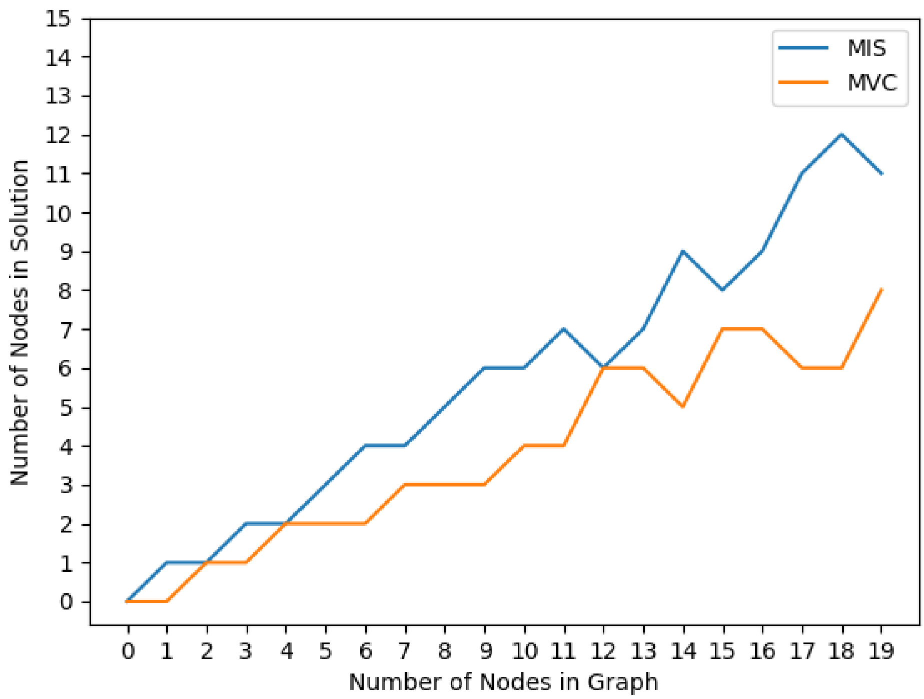

Figure 8. Number of Nodes vs Size of MVC and MIS

When looking at the size of each MIS and MVC in Figure 8, we can see that when you sum the size of the MIS and MVC, we get the size of the graph itself. I tested this for graphs up to size 40 (which took a while) and this property still held. For example, when you have a graph with 19 nodes, the MVC had a size of 8 and the MIS had a size of 11, and 11+8=19. When trying to identify why this was the case I experimented on graphs of random sizes and found that MIS's and MVC's are actually complements of each other. When you take the nodes that are not in the MIS, you will get a MVC, and vice versa. For example, when you have a graph with 6 nodes and 8 edges, say:

```
>>> g = createRandomGraph(6,8)
>>> g.adj
{0: [2, 1, 3], 1: [4, 0, 2, 5], 2: [0, 1, 4], 3: [5, 0], 4: [1, 2], 5: [3, 1]}
>>> MIS(g)
[5, 4, 0]
>>> MVC(g)
[3, 2, 1]
```

Figure 9. MVC and MIS Generated For a Given Graph

As you can see in Figure 9, the MIS and MVC found are complements of each other in respect to the nodes in the graph. Though this is the case, there can be multiple MVC's and MIS's for a given graph, so the complement may not always be shown through the program's results. If we were to return all the MIS's and MVC's for a graph, we would be able to pair each MIS and MVC, because they are complements of each other.

Appendix