CS/SE 2XC3 Lab 10 Report

Glotov, Oleg
L03, 400174037
glotovo@mcmaster.ca

Willson, Emma
L02, 400309856
willsone@mcmaster.ca

April 4, 2022

This report includes the main observations that we found in this week's lab, along with the analysis of our results.

1 Vertex Cover

In this section, we discuss algorithms for approximating the minimal vertex cover of a graph and their accuracies.

1.1 Approximation Implementation

We created three algorithms to approximate the minimal vertex cover of a given graph, g. Our algorithms all started with an empty list `cover` and added nodes to that list to build a vertex cover. Our algorithms all run in polynomial time.

1. `vc_approx1()` Creates a list of all edges in the graph using the helper function `edgeListForGraph()` and randomly selects (and removes) an edge. The algorithm adds both nodes incident on that edge to the cover, and then removes all edges propagating from those nodes from the list of edges. The algorithm repeats this removal process until the list of edges is exhausted.

2. `vc_approx2()` Creates a list of all nodes and their adjacent nodes using the helper function `nodeListForGraph()` and returns the set of nodes with more than one adjacent node. Removing "leaf" nodes always returns a vertex cover because leaf nodes are only incident on one edge.

3. `vc_approx3()` Creates a list of all nodes and their adjacent nodes using `nodeListForGraph()` and adds nodes to the cover in order of most incident edges. Each time a node is added, the algorithm uses `is_vertex_cover()` to check if the current cover is valid, and returns the first cover it generates.

We verified the validity of our algorithms (with respect to returning vertex covers) using the provided function `is_vertex_cover()` and compared their outputs to `vertex_cover()` for small graphs. However as graph sizes approach 20, testing times for `vertex_cover()` became unreasonably long, so we did not continue verifying our algorithms for large graphs.

1.2 Approximation Testing

*testing method explanation
Since we were unable to use the given implementation to test graphs of size greater than 20, we used data generated by `minimal_cover()` as an optimal solution and as a C^* to compare the outputs of our algorithms. We graphed

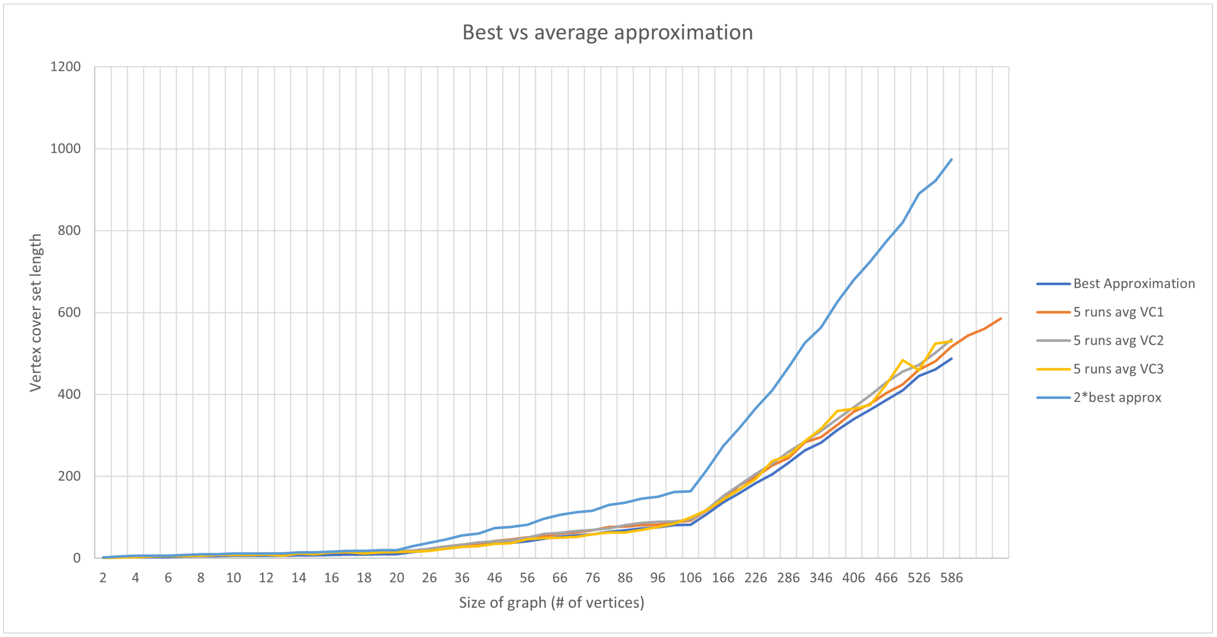

Figure 1: size of approximated minimal vertex covers

As shown in the chart above, all of our approximations had an upper bound of $|C| \leq 2|C^*|$. We do not have a formal proof of this, but we believe these algorithms to be "2-Approximations" of the minimal vertex cover.
We would expect `vc_approx1()` to perform worse on graphs where ...
We would expect `vc_approx2()` to perform worse on graphs without leaf nodes or where each node is incident on at least 2 edges.
We would expect `vc_approx3()` to perform worse on graphs where all of the nodes have an equal number of incident edges because then, the algorithm would simply add each node to the cover, in order, until it was a valid vertex cover. If these nodes are all adjacent, then a needlessly large vertex cover is returned.