CSCI 432
Project

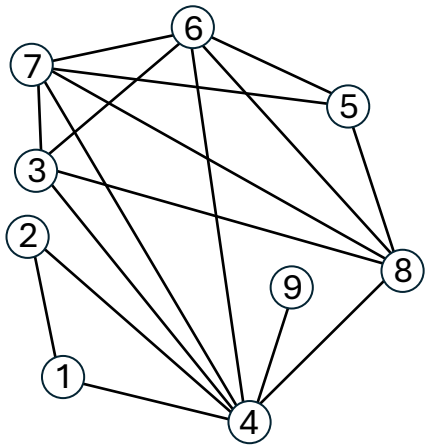
1 Overview

The final project in CSCI 432 is to implement a MAX-CLIQUE solving algorithm and run it on provided test instances. The test instances will be provided to you on 30 November. Your submission will be due on 11 December (at 3:10 PM). Your submission will consist of your code, the maximum clique you found in each of the test instances in the format specified below, and a short (1-2 pages) report explaining how your algorithm works and anything interesting you learned along the way. You will also present to the class a short description of how your algorithm works and any insights you got from testing on 14 December 2026, during our scheduled Final Exam time.

2 Problem

Given an undirected, unweighted, simple graph $G = (V, E)$ (that is connected or unconnected), a clique is is a subset of vertices such that each pair of vertices in the subset forms an edge in the graph. In mathematical terms, a clique is a subset of vertices $V' \subseteq V$ such that for all $u, v \in V'$, with $u \neq v$, $(u, v) \in E$. In the example graph below, the vertices $V' = \{1, 2, 4\}$ form a clique of size three, since there is an edge between each pair of those vertices. However, $V' = \{4, 8, 9\}$ do not for a clique because there is no edge between vertex 8 and vertex 9 in the graph.

The MAX-CLIQUE problem is given an undirected, unweighted, simple graph $G = (V, E)$ (that is connected or unconnected), find the clique of largest size in the graph. In the sample graph below, the maximum clique has size 5.



3 Assignment

Your objective is to write a program that can read in a graph (in the format described below), find the biggest clique you can, and then output the vertices in that clique. MAX-CLIQUE is an NP-Hard problem, so you will very likely be unable to solve it optimally on large graphs. You are free to use any means necessary to complete this task, short of breaking into my house and demanding answers. You can use generative AI as much as you like. You can use techniques we discussed in class. You can use techniques not discussed in class. You can copy stuff from the internet. Whatever. Just find the largest clique you can for each test instance.

4 Input

Your program needs to be able to process graphs that are encoded in .txt files with a header line (first line) consisting of the character 'g ' followed by two positive integers, where the first integer is the number of vertices in the graph (i.e., $|V|$) and the second integer is the number of edges (i.e, $|E|$). Vertex numbers start at 1 and go up to $|V|$. After the header line, every other line in the rest of the file corresponds to a single (undirected) edge. Each of those lines in the text file consists of two positive integers that represent the vertex endpoints of that edge. An example input file for the sample graph above can be found here: `https://www.cs.montana.edu/yaw/teaching/432/project/sample_graph.txt`. A much larger graph input can be found here: `https://www.cs.montana.edu/yaw/teaching/432/project/sample_graph2.txt`. Note that the test instances may differ substantially from these samples.

This is the format that all of the test instances will be provided in. Note: The graph is undirected, so if by chance $(4, 5)$ and $(5, 4)$ are both listed as edges, then that's only one edge. Also, there is no guarantee on the order that edges will be listed in the file, or on the order that the vertices will be listed for a given edge.

5 Output

I am not going to be running your code. You will run your code on each of the (not yet) provided test instances and you will save the largest clique you find to a .txt file. I do not care much what you name the file, though I need to know which test instance that file corresponds to. It needs to be a .txt file. The format of the file is important. There should be two lines. The first line is the size of the maximum clique you could find (i.e., the number of vertices). The second line is the list of vertices that compose the maximum clique you found, separated by spaces. An example output of a clique of size 4 found from the sample graph above can be seen here: `https://www.cs.montana.edu/yaw/teaching/432/project/sample_graph_output.txt`

6 Submission

Your submission will consist of your code, the clique output for each test instance (in separate .txt files) in the format specified above, and a short (1-2 pages) report explaining how your algorithm works and anything interesting you learned along the way. Your report should be concise, but clear. I want to know how your algorithm works, how you came up with that approach, how you did testing, and any changes you made as a result of testing. You can use AI to help you draft it. You will be graded on its content and presentation quality.

7 Presentation

During our regularly scheduled Final Exam time (14 December, 2:00 - 3:50), be prepared to give a brief description of your algorithm and how it functions. Your objective for this presentation is to explain how your algorithm works and to answer questions about it. Even though you are able to use external resources for this project, I expect you to understand what you did. So if you face questions about how/why you did something and can't answer them, your grade may reflect that. We will figure out exactly how much time we'll have per presentation later in the semester, once teams are established.

8 Timeline

Here is a timeline of the project:

- 23 October - Project assigned
- 30 November - Test instances released
- 11 December (3:10 PM) - Code/solutions/report submission due
- 14 December (2:00 - 3:50 PM) - Project presentations

9 Grading

You will not be graded on how effective your algorithm is at finding large cliques. Meaning if someone finds a larger clique than you, that's fine. That is irrelevant to the grading. You will be graded on:

- Your algorithm's validity (are the things you provide actually cliques?)
- Your algorithm's non-triviality (e.g., returning the end points of any random edge and calling it a 2-clique and the biggest thing you found is not impressive)
- The creativity of your approach (kind of vague, but I may allow creative approaches to slack off in other areas)
- Report quality
- Presentation quality and demonstrated understanding of what you did (i.e., answered questions)

10 Rules

There are lots of rules that need to be abided by. Failure to do these could impact your grade:

1. You may work in teams of up to three participants. You will only submit (and present) once for the whole team.
2. Follow the formatting specs detailed in the output section.
3. It is critical that the vertices in your submitted cliques use the original vertex labels from the input file. When you output a clique, I need the vertex labels (numbers) to match what the input was, otherwise I won't know which vertices you are talking about. Basically, if the input says there as an edge between '17' and '34', then that means there needs to be a vertex named 17 and anther named 34 and there needs to be an edge between them. Do not output them renamed to 'f' and 'e' or something else. Retain original labeling.
4. Make sure your report is legible and nicely formatted (use LaTeXor Word if you need to).
5. Don't share actual clique solutions (i.e., list of vertices). Feel free to share the sizes you have found (or add 100 to what you found to keep your friends busy), but not the actual set of vertices.
6. You are expected to understand your solution. If you get help some an external source, you are still responsible for the work. If you submit some massive repository that you did not write, be prepared to answer very specific questions about design choices that you did not make.

11 Glory

I will come up with some poorly devised scheme that ranks each submission based on the sizes of the cliques they found (after I validate that they are actually cliques) and plan on having some sort of prizes for the top performers, presented during the presentations. This does not impact your grade, it is just for fun.