Computer Architecture
Assignment 5 - Optimization

As always it is important you adhere to the following guidelines:

(1) Work individually;

(2) The submission date is January 20, 23:59;

(3) Submission is via the "submit" system;

(4) Ensure that your submission compiles and runs without errors or warnings on BIU's servers before submitting. Failure to do so will result in docked points;

(5) At the beginning of every file you submit, add your Teudat Zehut and name in a comment. For example: /* 123456789 Israel Israeli */;

(6) It is forbidden to use AI tools like ChatGPT when writing your homework. Doing so is tantamount to cheating, and will be treated as such;

A popular problem in computer science is the problem of finding a Maximum Clique. This problem is known to be NP-Hard, meaning we don't know how to solve this problem in polynomial time (and we don't know if it is possible at all). In this problem, we are given a graph $G = (V, E)$, presented as an adjacency matrix $A \in \{0,1\}^{|V|\times|V|}$, which satisfies $A_{ij} = 1 \iff (i,j) \in E$. Your goal is to find a subset $C \subseteq V$ such that:

$$\max_{C\subseteq V} |C| \text{ subject to } \forall u \neq v \in C : (u,v) \in E$$

In other words, find the largest subset of vertices where every vertex is connected to every other vertex in the subset. If the graph is directed, either direction of an edge is considered a connection.

You are given an unoptimized implementation of this code in a file named "max-clique.c" which implements 4 functions:

(1) `isClique` check if some subset of vertices forms a clique.

(2) `generateCombinations` recursive code that generates possible combinations of vertices.

(3) `findMaxClique` the main called function, needs to find the maximum clique.

(4) `printSolution` utility function to print the vertices in the clique.

You are also given a “main.c” that asks for an input and runs your code. Your assignment is to rewrite the code in order to optimize it. **Note:** You are not restricted to the above functions, you may rewrite the whole file as long as the output is the same. **You cannot change the main file**, only the “max-clique.c” file.

You can assume that the input for the problem (which is an adjacency matrix of size n) is valid.

In order to help you test your code, you have been given a Python script `create-matrix.py`. Use it to create randomly generated adjacency matrices of varying sizes, which you can check on. **The Python script is optional and provided for convenience only. You may adapt it as you wish or completely ignore it and use the `main.c` file directly.**

Hint: The `>` and `<` commands redirect output and input, respectively. For example:

```
python create-matrix.py n > graph.txt
```

creates a file `graph.txt` with a generated adjacency matrix. You can then run:

```
./clique < graph.txt > output.txt
```

This executes your program with the generated matrix as input and writes the output to `output.txt`.

It is important that your code will return **the same** result as the original code, and that you compile with the “**-O0**” flag, which means that the compiler cannot add optimizations!

For consistent timing results, we recommend running your code on the BIU servers and using the same input for time testing. For example, the default code’s runtime for a random graph with 30 nodes is approximately 110 seconds. Use this as a reference to evaluate the performance of your optimized code.

Then after you optimize the code, rerun it with the generated matrices and compare your new results to your old results using bash commands like `diff` (as always `man diff` for more information).

Your grade will be determined based on how quickly your code runs (on BIU servers) relative to your peers. Do not unionize and all submit bad code: code which runs at subpar speed will receive a low grade even if it is the quickest code submitted.

The best of luck and may the best person win (get 100)!

What to Submit

You must submit the following files using the submit system:

(1) The original `main.c` file.

(2) The modified `max-clique.c` file.