1st Project ASA 2019/2020 Report

Group: tp039

Students: Pavle Arandjelovic (93745) & António Marques Murteira (90706)

__Preface__: In this report “Strongly Connected Component” will be abbreviated to “SCC”

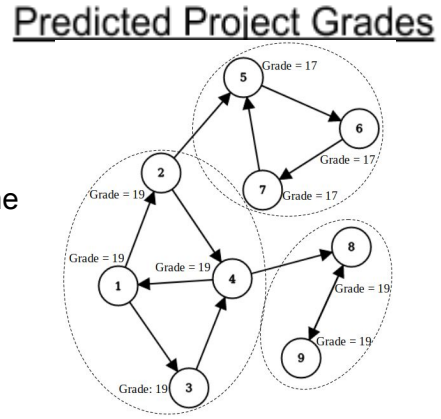
Description of the Problem:

The project in question required us to create a program, where given:

- the number of students N (N > 1)
- the number of relationships between students M (M > 0)
- the grade for the first test of each student
- friendships between students

would predict the project grade that each student would receive based on their first test grade and the relationships that each student has with other students.

Analysis of the Problem at Hand:



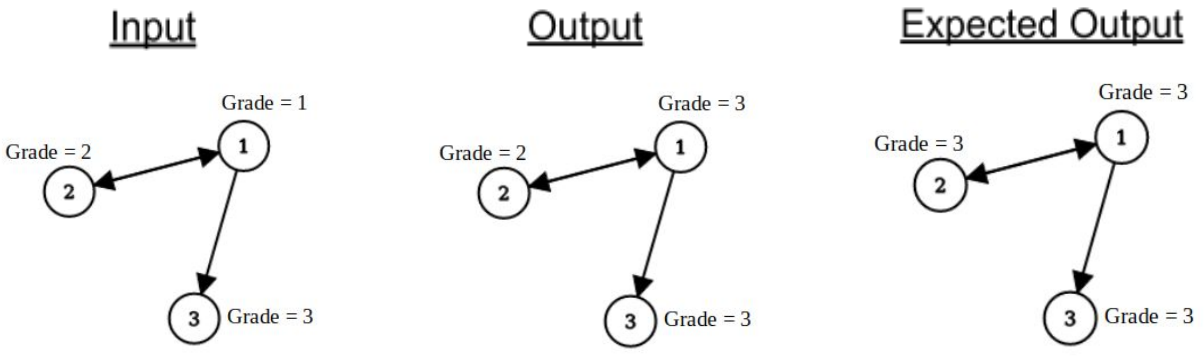
To fully understand the problem at hand, we first constructed a graph with the example given in the project brief. Each student represented one node in our graph and each node contained the students first test grade. We noticed that nodes which were in the same SCC would all end up having the same project grade and if an SCC (SCC1) contains a cross edge which connects it to another SCC (SCC2), then the grades in all the nodes of SCC1 would match that of the max grade in (SCC1, SCC2). See diagram for further clarification (The boundaries drawn with “- - - -” delimit the SCCs in the graph):

Solution of the Problem:

We decided that the best-fit algorithm to help us solve this problem would be the Tarjan search algorithm which would help us identify SCCs in a graph. Our program was coded in C++ and contains a large portion of code from GeeksForGeeks [(1)] .

(1) Tarjan's Algorithm to find Strongly Connected Components

Our version of the algorithm behaves similarly to a normal Tarjan search, with the difference that when we find an SCC in our graph, we register the max grade value within the SCC and set the grade value of each of the student nodes to that value. This however did not cover all possible graphs, we ran into a problem where the following graph would not be solved correctly:



1st Project ASA 2019/2020 Report

Group: tp039

Students: Pavle Arandjelovic (93745) & António Marques Murteira (90706)

We solved this by implementing a second stack (st_SCC). Each time we find the head of an SCC (discovery = low), we pop each member of the main stack, and push them into st_SCC and set the max grade value in the group of popped nodes. We then continue to pop each element from st_SCC and set its grade to the max one found in that group.

Theoretical Analysis of our Solution:

Looking at the flow of our program, we broke it down into steps with time-complexity analyses:

- The process of reading and processing data given to us depended linearly on how many students and relationships we had to process.
  Time Complexity -> Θ(|V| + |E|)
- Create graph object and initializations: Constant time as we are only allocating memory for a fixed amount of data structures.
  Time Complexity -> O(1)
- Altered Tarjan Algorithm: Equal to that of a normal Tarjan search.
  Time Complexity -> O(|V| + |E|)
- Representation of data to Stdout: This is done in linear time as it is merely a loop which iterates through an array and prints out the result.
  Time Complexity -> O(|V|)

TOTAL TIME COMPLEXITY -> O(|V| + |E|)

Rundown of our Experimental Results:

We created a large dataset and timed it; below are specifications used for our testing environment:

- CPU - Intel(R) Core(TM) i7-6500U CPU @ 2.50GHz
- Enabled cores - 2
- Threads - 4
- RAM - 8GiB

We can conclude that we have a linear relationship between the amount of students (V) + relationships (E) and the time it takes to process the predicted project grade for all students which lines up with the global predicted time complexity for our program: **O(|V| + |E|).**