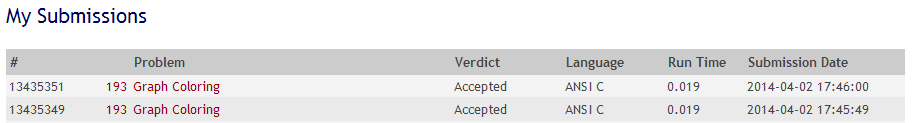
Graph Coloring

UVA Online Judge

DASALGO Problem Code: G3 Status: Accepted

UVA Problem 193 – Graph Coloring Runtime: 0.019 seconds



(Source code with comments at the last page)

Discussion

The problem was to write a program that tries to find an optimal coloring for a given graph. Colors are applied to the nodes of the graph and the only available colors are black and white. The coloring of the graph is called optimal if a maximum of nodes is black. The coloring is restricted by the rule that no two connected nodes may be black.

The specifications imply the use of nodes or linked lists in order to make the graph. For the sake of simplification, I chose to use a two dimensional array instead so that pairing together two nodes would become as simple as setting their coordinates.

For the input, the first line should include the number of graphs *m*, with this we can determine how many times we must repeat the program in order to color (in this case mark) all of the nodes. The second line should include the number of nodes *n,* and the number of edges *k.* The connection of the nodes is given by user input as well. With the parts of the graph(s) complete, all we need to determine is the maximum number of black colored nodes (considering the given rule), as well as their indices.

Here is the step by step algorithm:

1. Determine number of graphs
2. Determine number of nodes and edges
3. Create an array storing the nodes’ indices and a two dimensional array for their pairing/connection
4. Create a function that would set the colors of each node by:

* Setting the first node to color black, while keeping count of the number of black colored nodes.
* Checking if it’s next pair (numerical order) is already ordered

1. if not, set it to white
2. otherwise set it as unmarked

* Check the rest of the unmarked nodes.
* Repeat until all of the nodes have been colored.
* Finally, copy the indices of the black colored nodes to a final array to pass as a result

The confusing part of the code is how to determine the colors of the different branches the current node is connected to. Before proceeding with marking any of them, we have to first determine if they are connected to any black colored nodes. But the answer to the problem was simple, since we already made an array of the nodes, keeping track of their markings; all we have to do is to refer to that list (their indices) when checking if a node is directly connected to any black colored node, thus avoiding the violation of the rule that no two black colored nodes should be connected.

The only thing we did in order to modify the program was to give detailed variables, proper indention, optimization of some useless bits if code, and overall the translation (since the source was in Spanish, which made things a little difficult to understand). As for the compilation, the original code missed a few libraries for some of the functions used in the code, therefore slowing down the compile time.

References

Cespedes, J.(n.d.) *Graph Coloring*. cespedes.org. Retrieved April 02, 2014 from <http://www.cespedes.org/uva-acm/1/193>”

"UVa Online Judge." UVa Online Judge. N.p., n.d. Web. 2 Apr. 2014. <http://uva.onlinejudge.org/index.php?option=com\_onlinejudge&Itemid=8&page= show\_problem&problem=129>