Math 182 Homework #1

Nathan Solomon

January 24, 2025

Problem 0.1.

We showed in math 180 that a tree has Euler characteristic $\chi = 1$, and that the Euler characteristic for any graph is $\chi = |V| - |E|$, where $|V|$ is the number of vertices and $|E|$ is the number of edges. If we let N_i denote the number of vertices with i children, then for a binary tree,

$$\begin{aligned}
|V| &= \sum_i N_i = N_0 + N_1 + N_2 \\
|E| &= \sum_i (iN_i) = N_1 + 2N_2 \\
\Rightarrow \chi = 1 = |V| - |E| &= N_0 - N_2 \\
N_2 &= N_0 - 1.
\end{aligned}$$

Problem 0.2.

Every connected graph G contains a spanning tree (a tree which connects all its vertices) T as a subgraph. Let v be any leaf of that tree, and let $G - v$ denote the graph G with v removed, and with all edges adjacent to v removed. Then $T - v$ is a tree which connects all vertices of G except v, and $T - v$ is a subgraph of $G - v$, so $G - v$ is connected.

Problem 0.3.

Let G' be a copy of G but with the edge (x, y) removed. Then use DFS to see if a path exists between x and y. We already showed in class that DFS on G runs in $O(|V| + |E|)$ time. G' has the same number of vertices and one less edge, so DFS on G' will also run in $O(|V| + |E|)$ time.

The pseudocode for DFS is goven in the class note:

```
DFS(u):
    Mark u as "explored" and add u to R
    For each edge (u, v) incident to u:
        If v is not marked "explored":
            DFS(v)
```

The first and second lines of that are ran once per vertex. The third and fourth lines are ran once per edge. Since each time DFS is called, one more node is marked as “explored” (and nodes are never changed back to “unexplored”), the fifth line is also ran only once per vertex. Therefore, the entire program runs in $O(|V| + |E|)$ time.

G' contains a path from x to y iff G contains multiple paths from x to y, which occurs iff (x, y) is contained in a cycle in G.

Problem 0.4.

Call each judgement an "edge", and call two specimens "adjacent" if they share an edge.

```
CheckConsistency(n specimens, m judgements):
    Label all specimens C
    While there is at least one specimen labeled C:
        Let L[0] be the singleton set containing one such specimen
        Label the specimen in L[0] A
        Let N = 0
        While there is at least one specimen labeled C which is adjacent to an element of L[N]:
            For each specimen X adjacent to an element Y of L[N]:
                If that judgement is "same", add X to L[N]
                If that judgement is "different", add X to L[N+1]
                If X was just added to an even-numbered layer (L[2Z] for some integer Z), label X A
                If X was just added to an odd-numbered layer, label X B
            Increment N
    For each judgement:
        Return False if the labels do not satisfy the judgement
    Return True
```

Let G be the graph where each vertex is a specimen and each edge is a judgement between them. Two vertices are connected iff knowing which group one specimen is in tells you which group the other is in (although it may tell you two different answers).

The inner while loop identifies all vertices connected to the one element of $L[0]$ ("layer 0"). This is exactly the same as using BFS to find a 2-coloring (which we already showed is $O(m+n)$), except that if an edge (between $L[N]$ and an unexplored node) is labeled "same", we add the unexplored node to the previous layer. That tweak will not affect the run time. If the judgements for this connected component are logically consistent, then the labels created in the inner while loop will not contradict each other, because they can all be deduced from the one element of $L[0]$ being labeled A. Of course, if that specimen is actually in group B, then we could swap all the labels for specimens in that connected component from A to B and vice versa, but that would not change whether the judgements are logically consistent, and it would not affect other connected components.

The outer while loop repeats that process for each connected component of G. This whole thing runs in $O(m+n)$ time, for the same reason that the DFS in problem 3 runs in $O(|V|+|E|)$ time. Since this whole process only guesses the label for one element of each connected component, if there is any logically consistent way to label the specimens, this will have found one such labeling (because the inner while loop did that same thing for each connected component).

Lastly, the for loop checks in $O(m)$ times whether the labels are consistent. That means the entire algorithm takes $O(m+n)$ time.

Homework 2

Math 182, Winter 2025

For some questions, you may wish to use the algorithms discussed in class (eg BFS or DFS). If you use them directly, you can simply call them in your algorithm. If you modify them, you must provide details of the modified version.

When designing an algorithm, you must include the following:

1. A short (1-2 sentence) sketch of the main idea of the algorithm.
2. The algorithm itself, written in pseudo-code.
3. The runtime of the algorithm.
4. A proof that the algorithm is correct.
5. A proof that the runtime is correct.

Question 1:

A binary tree is a tree in which every node has at most two children. Prove that the number of nodes with exactly two children is one less than the number of leaves (nodes with no children).

Question 2:

Prove that if G is a connected undirected graph, there is a vertex v such that G remains connected when v, and all edges containing v, are removed.

Hint: Consider the DFS search tree.

Question 3:

Given a connected undirected graph $G = (V, E)$ and an edge $(x, y) \in E$, design an efficient algorithm to detect whether there is a cycle containing the edge (x, y). For full credit, this algorithm should run in $O(|V| + |E|)$ time.

Question 4:

Scientists have collected n specimens and are attempting to sort them into two groups. They do this by examining pairs of specimens, and determining if they belong in the same groups or different groups. If it is unclear, they will simply not make a decision. They worry that they may have made inconsistent judgments: maybe it is impossible to sort the specimens into two groups.

Given n specimens and m judgments of “same” or “different” for pairs of specimens, design an efficient algorithm to determine whether the judgments are consistent. That is, determine whether or not the specimens can be labeled A or B, such that for each pair (i, j) labeled ”same” i and j will be labeled the same, and for each pair (i, j) labeled “different”, i and j will be labeled differently. For full points, the algorithm should run in $O(n + m)$ time.

Question 5:

Extra Practice: This problem will not be graded. Suppose the breadth-first tree and depth-first tree produced from a graph G are the same, and include all vertices of G. Prove that $G = T$. (That is, prove that G cannot contain any edges that do not belong to T.)

Question 6:

Extra Practice: This problem will not be graded. Prove that if G has n vertices, all of which have degree at least $n/2$, then G is connected.