Homework 5: Q2

**Name:** Xinkai Lin, xinkaili

*Don’t forget to input your list of collaborators and sources on* ***AutoLab****.*

**Please submit this file as a PDF.**

# **Algorithm Idea**

I will perform this algorithm as BFS algorithm. I will first select a start node and find all the connected components {G1, G2, … Gk} of G with BFS algorithm. The select start node will be giving a label as either moth or butterfly. Then, we can run BFS search in each Gi and get a tree Ti with the root ri. finally, for each Gi, if it has not visited yet, set its value to 0 or 1 as moth or butterfly base on the same or different to its parent. If it has been visited, compare the label with giving definitive judgment, if it is not the same, then it is inconsistent.

1. **Algorithm Details**

**Algorithm:** Algorithm1

Input: A set of n caterpillars and m definitive judgments (same, different, ambiguous)

Construct the graph G = (V, E), s∈V

Consistent = true

LAYER = {s}

Allocate length n array LABEL and set LABEL[i]=-1 for every 0<= I < n

LABEL[s] = 0 // I will assign s as starting node as moth, 0 mean moth, 1 mean butterfly

While LAYER ≠{} do

LAYER = {}

For u ∈ CURRLAYER do

If LABEL[u] == -1 then //if it hasn’t visited

Add u to LAYER

If the judgment is same then

LABEL[u] = 0 //u is moth

Else LABEL[u] = 1 //u is butterfly

End if

Else //it has visited

if the label ≠ definitive judgment

consistent = false // inconsistent

end if

end for

CURRLAYER = LAYER

end while

return consistent

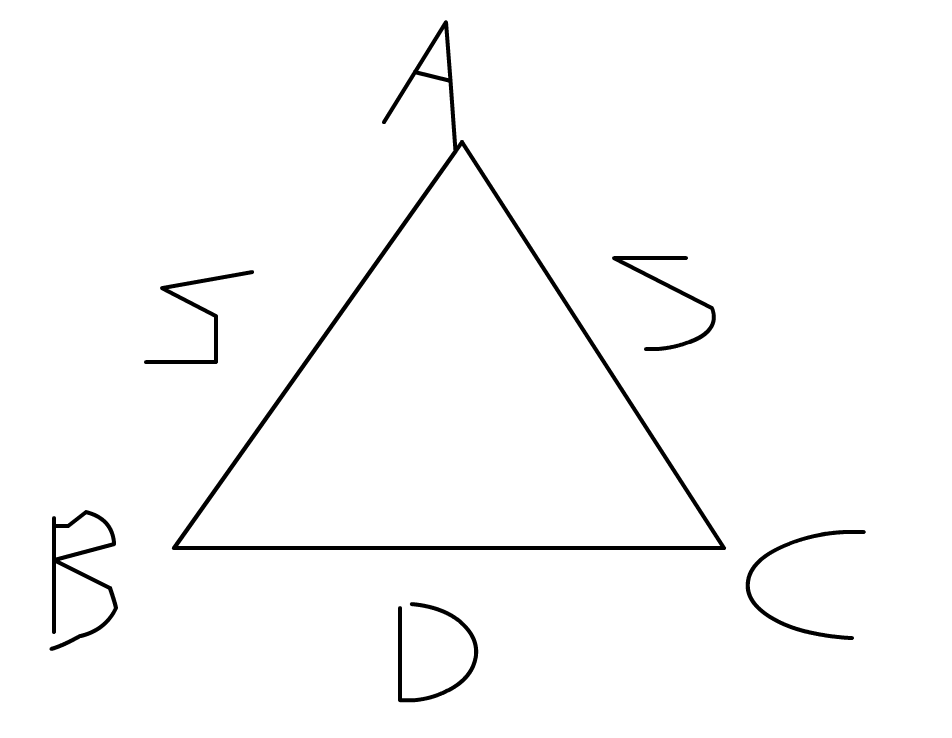
1. **Proof of Correctness Idea**

To prove the algorithm’s correctness, we will show that the judgment is consistent or inconsistent. In this algorithm, it will iterate all the node and mark it as either moth or butterfly depend on their parent node. Once it has its label, we will compare with the definitive judgment, if the label satisfied the definitive judgment, then it is consistent, else it is inconsistent.

1. **Proof Details**

In this proof, we will know that for any giving input, the algorithm will help us to find out whether or not it is consistent or inconsistent. Consider the following example:

Giving input n = 3, 3 caterpillars and m = 3, with 3 pairs of definitive judgment,

<<A, B>, S>, <<A, C>, S>, <<B, C>, D> S is same, and D is different.

let’s set the start node as A. As the algorithm goes, it will first check on all the neighbor node that has not been visited yet, which here are B and C. We found out that B is same A, and C is also same as A. Suppose we know that A is moth. As right now, we can say that B and C is also moth. In other hand, as the algorithm goes, we will find same neighbor node that are already visited. In addition, base on the previous result it will has its label as whether is moth or butterfly. As right now, we know that B and C are same with A which they all moth. Since A have iterated through all its edges. we move to the next layer of the graph. Here we have B and C. Now, we will look at B’s neighbor node. Finally, we found that when we compare B with C. the case fail. From previous we set B and C both as moth, but when we look at the definitive judgment it says different. The label didn’t match the definitive judgment. So, we concluded that this graph is inconsistent. Similarly, for any giving input n and m. By using this BFS algorithm, we will find out if it is consistent or inconsistent.

1. **Runtime Analysis**

In this algorithm, first we will use the for loop which set all the caterpillar as unvisited. There is n caterpillars, each caterpillars will take O(1) time. When the for loop finished, it will run O(n) time. Then, we have a for loop inside the while loop, which the for loop will run O(nu) time, and the while loop will run until the list is empty total run time of ∑u O(nu) = O(m). Finally, we can say that our total run times is O(m+n). Also in class, we have proven that the running time of BFS algorithm is O(m+n) time.