CSE331 – Project 4
Unweighted Graphs

Due Date: 11:59 pm Nov.13, 2015

1. Project Description

You will implement an effective algorithm to determine whether a graph is bipartite. A bipartite graph, G= (V,E), is a graph such that a set of vertices, V, can be partitioned into two subsets, V1 and V2, and no edge has both its vertices in the same subset.

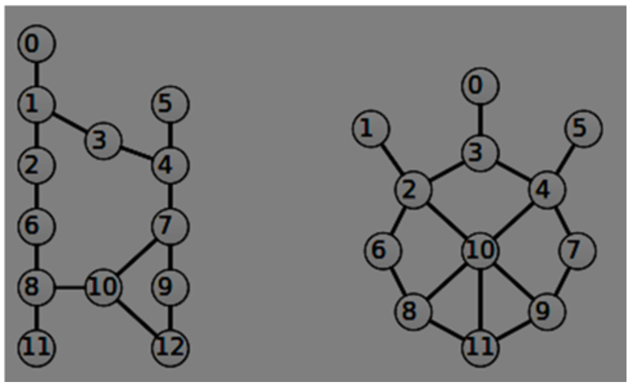
The two sets V1 and V2 may be thought of as a coloring of the graph with two colors: if one colors all nodes in V1 white, and all nodes in V2 black, each edge has endpoints of differing colors, as is required in the graph coloring problem. In contrast, such a coloring is impossible in the case of a non-bipartite graph, such as a triangle (three vertices and three edges): after one node is colored white and another black, the third vertex of the triangle is connected to vertices of both colors, preventing it from being assigned either color.

Your algorithm must run in a linear time. It must accept a single command line argument giving the name of a text file with an adjacency list for a graph. Each line in the text file contains a number indicating which vertex it is, followed by numbers separated by spaces, indicating which vertices this vertex has an edge to.

2. Programming Notes

- For this project you have been given a great deal of latitude in the implementation, however the general restrictions in the project guidelines still apply. You must provide a Makefile (feel free to modify one from a previous project), and it must produce an executable called 'prog'.
- You can assume that a given graph is connected graph
- You can start traversing a graph from any vertex you like. However, you must color the first traversed vertex with WHITE if it is an even number (i.e. 0, 2, 4, etc.). Otherwise, color it with BLACK. After that you color other vertices accordingly while traversing a graph.
- Your executable must output each path it takes, listing the vertices and their colors in the order they are visited. Separate vertex and color with a single space.
- If a graph is a bipartite graph you must output string TRUE at the last output line.
- If a graph is not a bipartite graph you can stop traversing a graph once you found the conflict. You then must output string CONFLICT follow by the conflicted vertices and FALSE. Please see output example below.
- Output examples:

```
>prog g1.txt          >prog g2.txt
0 WHITE               0 WHITE
1 BLACK               3 BLACK
2 WHITE               2 WHITE
3 WHITE               4 WHITE
6 BLACK               1 BLACK
4 BLACK               6 BLACK
8 WHITE               5 BLACK
5 WHITE               7 BLACK
7 WHITE               10 BLACK
10 BLACK              8 WHITE
11 BLACK              9 WHITE
9 BLACK               11 WHITE
12 WHITE              CONFLICT 11 8
TRUE                  FALSE
```



g1.txt g2.txt

3. Project Deliverables

The following files must be submitted via Handin no later than 11:59 pm Friday November 13, 2015:

- main.cpp – the C++ program
- graph.h – your implementation to determine whether a graph is bipartite
- Makefile – produces the executable called 'prog'.