CSC263: Problem Set 8

December 3, 2019

1 Problem 1

(a) We first prove this in the forward direction, that is, if a graph is bipartite then it contains no odd cycle. So assume that a graph G is bipartite. Then by definition the vertices of graph G can be partitioned into two sets V_1 and V_2 such that the all the edges of G have one endpoint in V_1 and the other endpoint in V_2. Suppose you start at a vertex in V_1 (without loss of generality), then following the edge from the vertex to the next takes you to V_2. In this way, each step either takes you from V_1 to V_2 or V_2 to V_1. Thus, to end up at the original starting point, an even number of steps are needed. So G cannot contain an odd cycle since that would mean there would be two vertices in V_1 (or V_2) who share an edge between them, contradicting the definition of a bipartite graph.

Now we prove this in the reverse direction, that is, if a graph contains no odd cycle then it is bipartite. Suppose a graph G contains only even cycles. Let v_0 be a vertex in the graph G. Now we have to define sets V_1 and V_2. Before that, let $d(m, n)$ be a function that defines the shortest distance between vertices m and n by the minimum number of edges in any path from vertex m to vertex n (i.e. immediate neighbours would have a distance of 1). Let $V_1 = \{x : d(x, v_0) \textit{ is odd}\}$ and $V_2 = \{y : d(y, v_0) \textit{ is even}\}$. Since G is connected, $V_1 \bigcup V_2 = V$, where V is the set of all vertices of G. Additionally, V_1 and V_2 do not share any vertices, that is, their intersection is empty. Now, we show that for every edge, e, in the graph G, $e = xy$, where $x \in V_1$ and $y \in V_2$.

Suppose we have an edge $e \in E$ such that $e = pq$, where $p, q \in V_1$. Then $d(v_0, p)$ and $d(v_0, q)$ will be odd since both p and q belong to V_1. Consider the shortest paths from v_0 to p and from v_0 to q, and let there be a vertex s such that s is the last common vertex on the path from v_0 to p and v_0 to q. Note that s might be v_0 if the paths from v_0 to p and v_0 to q have no vertices in common aside from v_0. Since the paths from v_0 to p and v_0 to q are the shortest paths possible (by assumption), we have that the path from v_0 to s of the path from v_0 to p and the path of v_0 to s of the path from v_0 to q are the same, since if they were not then we would have that one of the paths from v_0 to p or q was not the shortest possible path.

Then it follows that $d(s,p)$ and $d(s,q)$ will both be of the same parity, either even or odd. From this it follows that the path from p to s to q will be of even length since odd plus odd is even and so is even plus even. From this it follows if we create the edge pq then we create an odd length cycle with the following vertices p, q, s and the following paths from p to s, s to q, and q to p. But this contradicts our original assumption of G having only even cycles. Similar logic is applied to vertices from V_2.

Thus, if a graph contains no odd cycles, then it is bipartite.

(b) The correctness condition holds for a graph that is not connected. If a graph G is disconnected, then any cycle in G is contained in one of its connected components. So the condition from part a) can be applied to each connected component. So with each connected component being bipartite, each component will have two sets of vertices, we denotes this with $V_{(1,i)}$ and $V_{(2,i)}$ where i is an element of the naturals and refers to the corresponding connected component. Then for the entire graph, G, we have $V_1 = \bigcup_{i=1}^{n} V_{1,i}$ and $V_2 = \bigcup_{i=1}^{n} V_{2,i}$, where n is the number of connected components in G. Then it follows that G too will be bipartite if all its connected components are bipartite. Thus the correctness condition holds for a graph that is not connected.

(c) We define a function $BIPARTITE(G)$ that returns true if the graph is bipartite or false otherwise based on a modified version of depth first search. Here we use the correctness condition outlined above and we rely on the fact that odd-cycles are not 2-colourable to check to see if there are any odd-cycles in the graph. If there are, then the graph is not bipartite. The proof for this is similar to the proof from part a. Namely suppose we have a graph with an odd cycle, G = (V,E). And we define a function $f : V \to \{0,1\}$. Suppose $C = \{v_0, v_1, ..., v_n\}$ is a cycle in G. So $f(v_0) = 0$, $f(v_1) = 1$, and so on where

$$f(x) = \begin{cases} 0 & k \ mod \ 2 = 0 \\ 1 & k \ mod \ 2 = 1 \end{cases}$$

for k = 1,...,n. v_n will be adjacent to v_0 and we have $f(v_n) = 1$, which implies n is odd. C has $n+1$ vertices, C must be an even cycle.

So if G is 2-colourable, then every cycle in G is even.

Note: for the sake of simplicity, we have included an additional attribute to each vertex, called visited. That is, if the vertex has been visited then it is True and if not, it is False.

```
BIPARTITE(G):
    for every v in V:
        v.visited = False
        v.colour = WHITE
```

```
    while there is p in V such that visited(p) is False:
        p.visited = True
        p.colour = BLACK
        Stack = [p] #stack containing v
        while Stack is not empty:
            u = pop(S)
            for all edges (u,v) in E:
                if v.visited is False:
                    v.visited = True
                    push(Stack,v)
                if v.colour = WHITE:
                    if u.colour == GRAY:
                        v.colour = BLACK
                    if u.colour == BLACK:
                        v.colour = GRAY
                else if v.colour == BLACK:
                    if u.colour != GRAY:
                        return False
                else if v.colour == GRAY:
                    if u.colour != BLACK:
                        return False
    return True
```