Design & Analysis of Algorithms - 344: Homework #3

Due on February 3, 2017 at 11:59pm

Professor Dr. Kalantari Section #2

Douglas Rudolph

Question 1: Given an undirected graph G, describe an algorithm that can check if it is bipartite. A graph is bipartite if its vertices are partitioned into two sets A and B where all the edges are of the form (a,b) with a in A, b in B.

Answer 1: When checking to see if a graph is Bipartite, we know that all verteces are required to be split up into 2 sets; we will call these sets A and B. In terms of representing this on a graph, lets assume that for any give vertex V, V is aware of what set it belongs too. We will denote what set a vertex belongs to by writing V_A and or V_B.

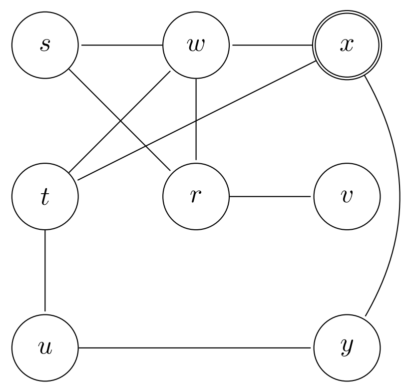
Checking to see if a graph is Bipartite can be accomplished by running BFS on a graph G, but while running BFS, we also have to check and see if that for any given V_i and a neighboring vertex V_j, that $i \neq j$ (i is not from the same set as j.) This conditioned is checked when inserting elements into the FIFO Queue. When all verticies meet this pre-condition, and the algorithm doesn't break prematurely, we know that a graph is considered Bipartite.

Question 2: Do a BFS of the undirected graph whose adjacency list is

s	→	w	→	r			
r	→	v	→	s			
v	→	r					
w	→	s	→	t	→	x
t	→	w	→	x	→	u
u	→	t	→	y			
x	→	t	→	w	→	y
y	→	u	→	x			

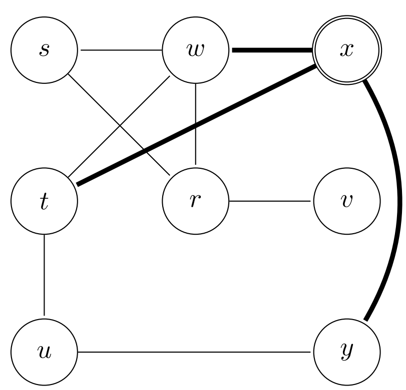
where starting vertex is x and vertices are placed on queue in alphabetic order.

Answer 2: To start, we first create a graph G that corresponds to the adjacency list given in the problem (**NOTE***: Any time we dequeue a vertex from the queue Q, we signify this process by adding a circle around the given vertex V. Also, the vertex x is given a circle in the beginning to signify that x is the starting node.



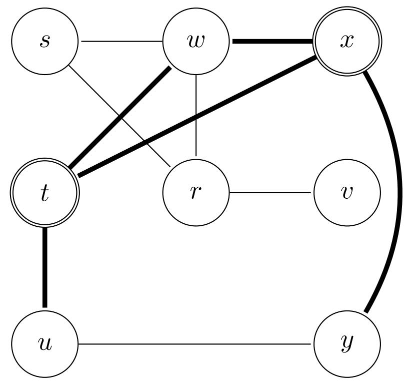
To start running BFS (Breadth First Search), all nodes that are adjacent to x are inserted into Q. After inserting the nodes into Q, G and Q will look as follows:

$$Q = [\text{t,w,y}]$$



Now that all the nodes that x is adjacent too are added into Q, t is dequeued from Q, BFS branches out from the vertex t to all adjacent nodes and adds them to Q.

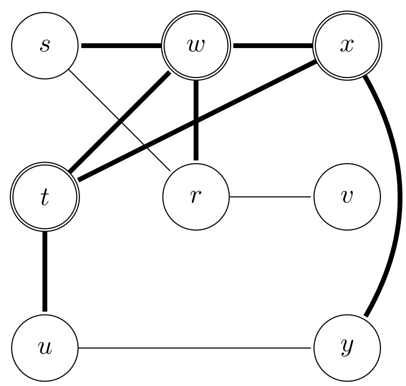
(Q **before** adding adjacent nodes of t) Q = [w,y]



(Q **after** adding adjacent nodes of t) Q = [w,y,u]

We then repeat the same process that was performed with t with the vertex w.

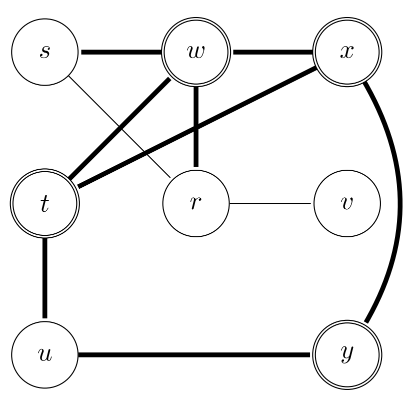
(Q **before** adding adjacent nodes of w) Q = [y,u]



(Q **after** adding adjacent nodes of w) Q = [y,u,r,s]

For the next iteration, we go y, and repeat the previous process.

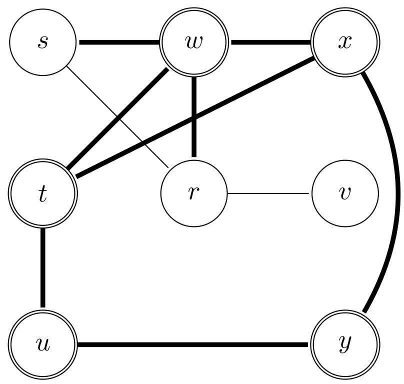
(Q **before** adding adjacent nodes of y) Q = [u,r,s]



(Q **after** adding adjacent nodes of y) Q = [u,r,s]

Again, the process is repeated on the vertex u.

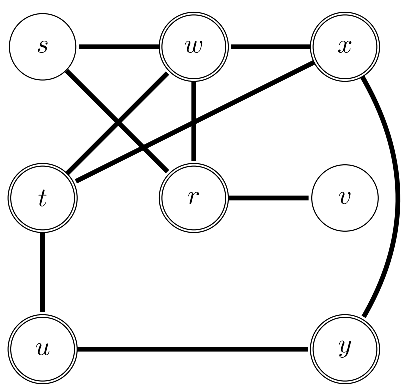
(Q **before** adding adjacent nodes of u) Q = [r,s]



(Q **after** adding adjacent nodes of u) Q = [r,s]. (***NOTE** In this case, all adjacent nodes of u were viewed, therefore no nodes were added to Q.

Moving forward from the previous iteration, the previous process is repeated on the vertex r.

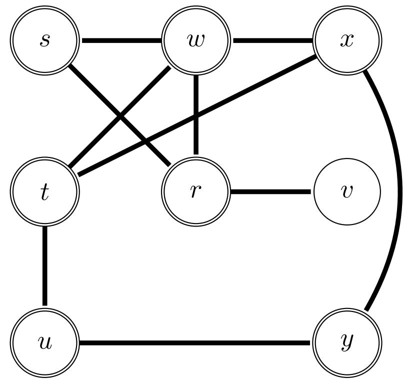
(Q **before** adding adjacent nodes of r) Q = [s]



(Q **after** adding adjacent nodes of r) Q = [s,v].

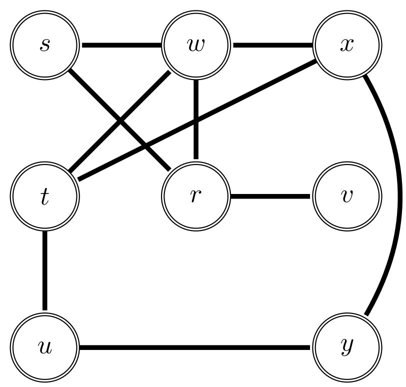
Now, all edges have been seen by BFS, but all nodes have yet been visited, therefore, must continue running BFS until Q is empty.

(Q **before** adding adjacent nodes of s) Q = [v]



(Q **after** adding adjacent nodes of s) $Q =$ [v].

(Q **before** adding adjacent nodes of v) $Q =$ [ ]



(Q **after** adding adjacent nodes of v) $Q =$ [ ].

Q is now empty and all nodes were visited. This concludes the process of BFS.

Question 3: Given a tree T=(V,E) compute its diameter, i.e. the longest path between two vertices. Do BFS at a node s, find the farthest from s to get a vertex, say t. Then do BFS to from t to find the farthest from t. Prove this gives diameter of tree. Diameter of a tree is the longest path in the tree.

Answer 3:

Question 4: Give an efficient algorithm that takes as input a directed graph G =(V,E) and and determines if there is a vertex s in V from which all other vertices can be reached.

Answer 4:

Question 5: Show that in an indirected graph the sum of the degrees of the vertices is twice the number of edges.

Answer 5: