COURSE HANDOUT

Course Code	ACSC13
Course Name	Design and Analysis of Algorithms
Class / Semester	IV SEM
Section	A-SECTION
Name of the Department	CSE-CYBER SECURITY
Employee ID	IARE11023
Employee Name	Dr K RAJENDRA PRASAD
Topic Covered	Graph traversals: Breadth first search
Course Outcome/s	Use BFS for generation of optimal tree from the graphs
Handout Number	23
Date	

Content about topic covered: Graph traversals: Breadth first search

- In a breadth-first search, we start by going to the start vertex, which is labelled v. Next, all of the unvisited vertices adjacent to v are visited.
- Then, the unvisited vertices adjacent to the vertices that have just been visited are visited, and so on.
- The end result of a BFS travel through a graph is a spanning tree. A graph without any loops is called a "spanning tree." For BFS traversal of a graph, we use a data structure called a "Queue" that has a maximum size equal to the number of nodes in the graph.

Step 1: Define a Queue of size total number of vertices in the graph.

Step 2: Select any vertex as starting point for traversal. Visit that vertex and insert it into the Queue.

Step 3: Visit all the adjacent vertices of the vertex which is at front of the Queue which is not visited and insert them into the Queue.

Step 4: When there is no new vertex to be visit from the vertex at front of the Queue then delete that vertex from the Queue.

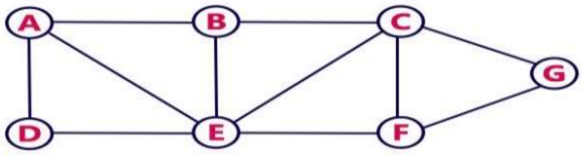
Step 5: Repeat step 3 and 4 until queue becomes empty.

Step 6: When queue becomes Empty, then produce final spanning tree by removing unused edges from the graph

Analysis Of BFS:

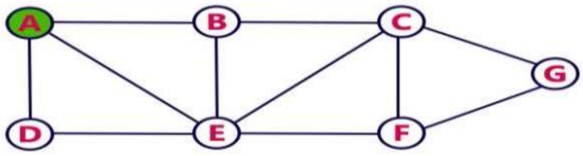
Each visited vertex enters the queue exactly once. So the while loop is iterated at most n times If an adjacency matrix is used the loop takes O(n) time for each vertex visited. The total time is therefore, $O(n^2)$. If adjacency lists are used the loop has a total cost of $d_0 + \ldots + d_{n-1} = O(e)$, where d is the degree of vertex i. As in the case of DFS all visited vertices together with all edges incident to them, form a connected component of G.

Consider the following example graph to perform BFS traversal



Step 1:

- Select the vertex **A** as starting point (visit **A**).
- Insert **A** into the Queue.

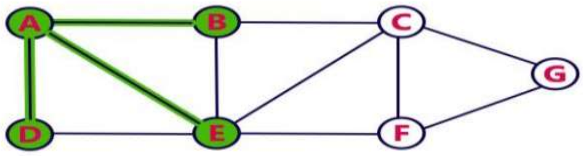


Queue

A						

Step 2:

- Visit all adjacent vertices of **A** which are not visited (**D**, **E**, **B**).
- Insert newly visited vertices into the Queue and delete A from the Queue..

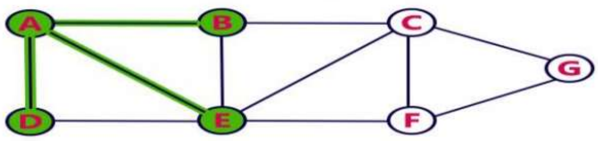


Queue

	D	E	B			

Step 3:

- Visit all adjacent vertices of **D** which are not visited (there is no vertex).
- Delete D from the Queue.

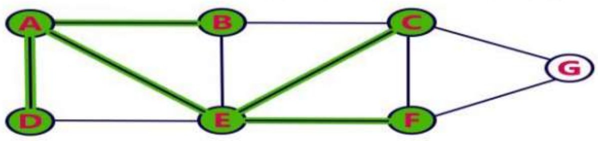


Queue

		E	B			

Step 4:

- Visit all adjacent vertices of **E** which are not visited (**C**, **F**).
- Insert newly visited vertices into the Queue and delete E from the Queue.

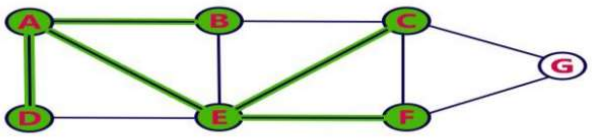


Queue

			B	C	F	

Step 5:

- Visit all adjacent vertices of **B** which are not visited (**there is no vertex**).
- Delete **B** from the Queue.

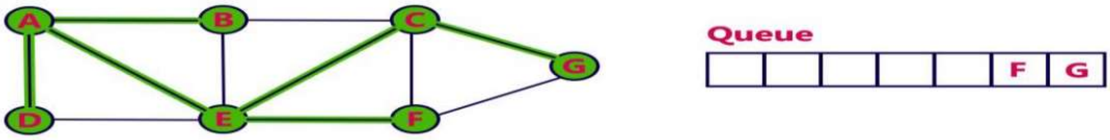


Queue

				C	F	

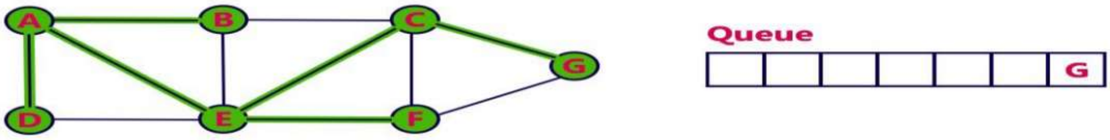
Step 6:

- Visit all adjacent vertices of **C** which are not visited (**G**).
- Insert newly visited vertex into the Queue and delete **C** from the Queue.



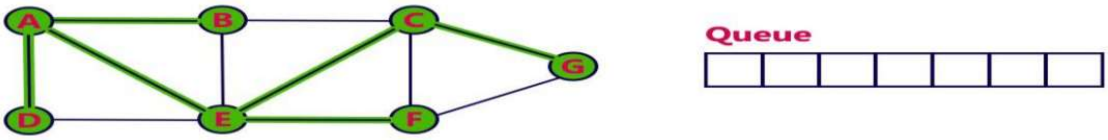
Step 7:

- Visit all adjacent vertices of **F** which are not visited (**there is no vertex**).
- Delete **F** from the Queue.



Step 8:

- Visit all adjacent vertices of **G** which are not visited (**there is no vertex**).
- Delete **G** from the Queue.



- Queue became Empty. So, stop the BFS process.
- Final result of BFS is a Spanning Tree as shown below...