# CptS 223 Homework #4 - Graphs

Please complete the homework problems on the following page using a separate piece of paper. Note that this is an individual assignment and all work must be your own. Be sure to show your work when appropriate.

**1. [13]** Define these terms as they relate to graph and graph algorithms:  
 Use mathematical terms where appropriate.

**GRAPH -** A graph is a structure amounting to a set of objects in which some pairs of the objects are in some sense "related". The objects correspond to mathematical abstractions called vertices (also called nodes or points) and each of the related pairs of vertices is called an edge (also called an arc or line Typically, a graph is depicted in diagrammatic form as a set of dots for the vertices, joined by lines or curves for the edges.   
**Vertice** - is a synonym for a node of a graph, i.e., one of the points on which the graph is defined and which may be connected by graph edges.

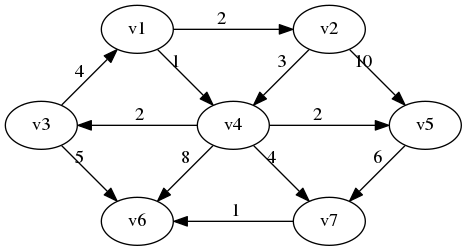
**Edge-** An unordered pair of nodes that specify a line joining these two nodes are said to form an edge.  
**UndirectedGraph** An undirected graph is graph, i.e., a set of objects (called vertices or nodes) that are connected together, where all the edges are bidirectional. An undirected graph is sometimes called an undirected network.   
**Directed Graph** a graph where the edges point in a direction is called a directed,,,graph.  
**Path** In graph theory, a path in a graph is a finite or infinite sequence of edges which connect a sequence of vertices which, by most definitions, are all distinct from one another.   
**Loop-** A loop is commonly defined as an edge (or directed edge in the case of a digraph) with both ends as the same vertex.  
 **Cycle** - are usually defined as closed walks which do not repeat edges or vertices except for the starting and ending vertex. This definition usually allows for cycles of length one (loops) and cycles of length two (parallel edges).  
**Acyclic** An acyclic graph is a graph having no graph cycles.Acyclic graphs are bipartite. A connected acyclic graph is known as a tree, and a possibly disconnected acyclic graphis known as a forest (i.e., a collection of trees).  
**Connected** - A graph is connected when there is a path between every pair of vertices. In a connected graph, there are no unreachable vertices. A graph that is notconnected is disconnected.  
**Sparse**- A graph in which the number of edges is much less than the possible number of edges.

**Weight**- A weighted graph is a graph in which each branch is given a numerical weight. A weighted graph is therefore a special type of labeled graph in which the labels are numbers.

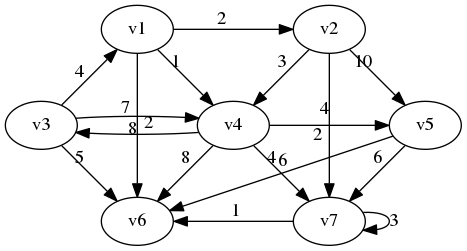
**2. [4]** Under what circumstances would we want to use an adjacency matrix instead of an adjacency list to store our graph?

With adjacency matrices we can answer fast to questions regarding if a specific edge between two vertices belongs to the graph a[i][j] as an O(1) lookup often., and we can also have quick insertions and deletions of edges. The disadvantage is that we have to use excessive space space used is O(n2), especially for graphs with many vertices, which is very inefficient especially if our graph is sparse.  
  
  
  
  
  
  
  
**3. [6]** Name three problems or situations where a graph would be a good data structure to use:

|  |  |
| --- | --- |
|  | **Computer Networks:** Graphs model intuitively model computer networks and the Internet. Often nodes will represent end-systems or routers, while edges represent connections between these systems.  **Data Structures:** Any data structure that makes use of pointers to link data together is making use of a graph of some kind. This includes tree structures and linked lists which are used all the time.  **Pathing and Maps:** Trying to find shortest or longest paths from some location to a destination makes use of graphs. This can include pathing like you see in an application like Google maps, or calculating paths for AI characters to take in a video game, and many other similar problems. |

**4. [4]** What kind of graph is this?  


**5. [4]** Identify the loop in this graph:



1) v3->v4->v3 (or) v4->v3->v4

2) v7->v7

3) v3->v1->v4->v3 (or) v1->v4->v3->v1 (or) v4->v3->v1->v4

4) v1->v2->v4->v3->v1 (or) v2->v4->v3->v1->v2 (or) v4->v3->v1->v2->v4 (or) v3->v1->v2->v4->v3

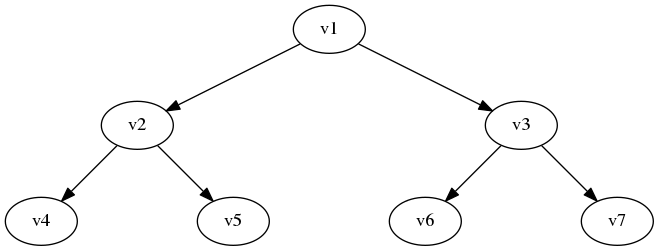
**6. [4]** How many vertices and edges are in this graph:

|  |  |
| --- | --- |
|  | Vertices \_\_\_\_\_7\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Edges \_\_\_\_\_\_\_\_\_\_\_\_17\_\_\_\_\_\_\_\_\_\_\_\_\_ |

**7. [6]** Are these cyclic or acyclic graphs?

|  |  |
| --- | --- |
|  | Cyclic?  No |
| Image result for cyclic graph | Cyclic?  Yes |
| Image result for acyclic graph | Cyclic?  No |

**8. [5]** A tree is a particular kind of graph. What kind of graph is that?



As the given graph has the direction in beteen the vertices if we see as tree then from parent to child, therefore this graph is **directed, acyclic connected graph**.

**9. [4]** What is the difference between a breadth-first search and a depth first search?

1.Depth first utilizes the stack and where as bredth first search utilizes the queue.

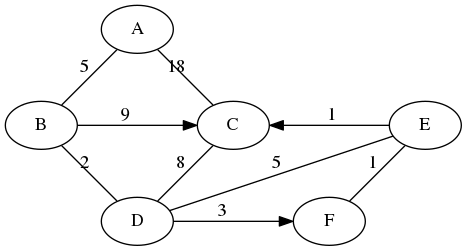
2.A DFS burrows deep inside the child node upto a target is attained.A BFS searches each one soluton in a graph to enlarge its node.

3. breadth first search is initiated at node ,known as root ,seeing its every immediate one close to each other in front of going to other node .depth first search initiate at node and trnsverse to the extent that is viable along with the branch in front of recursing back to the initial node.

4.in BFS the space complexity is more complicated when contrasted with time complexity.but in DFS minor space complexity since simultaneously it requires to preserve single path i.e root to leaf node.

5.DFS Is quicker when compared to BFS.

**10. [10] Dijkstra's Algorithm.** Use Dijkstra's Algorithm to determine the shortest path starting at **A**. Note that edges without heads are bi-directional. To save time, you do not have to add items to the "priority queue" column after it has been discovered (listed in the "distance" column). Use the table below to show your work.

What’s the shortest route (by weight) from A to C? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  


|  |  |
| --- | --- |
| **Node: Distance** | **Priority Queue** |
| **0** |  |
| **5** |  |
| **2** |  |
| **3** |  |
| **1** |  |
| **1** |  |
| **Total: 12** |  |
|  |  |
|  |  |

**11. [10] Topo sort.** Show the final output of running Topo Sort on this graph:

|  |  |
| --- | --- |
|  | What’s the vertice with the largest degree and its value?  8  What’s the vertice with the highest indegree and its value?  3  What’s the vertice with the highest outdegree and its value?  6 |

Topo sort output:

Mac3311, COP3210, MAD 2104, COP3400,CAP3700, MAD3512,MAD3305,CDA4101,COP4555,COP 3530, COP9540, CIS4610 COP5621,CPA4400, COP4610, COP4225