Readings

- [Lecture Notes Chapter 16: Graph Traversals: BFS & DFS]

Problems

Problem 1

1. Which type of edges are found in a DFS traversal of an undirected graph?

2. The presence of which type of edge indicates a cycle in a DFS traversal?

3. Consider an edge \((u, v)\) when the edge is first explored. Determine the classification of the edge in the following scenarios:

   - If \(v\) is white, \((u, v)\) is a ______
   - If \(v\) is grey, \((u, v)\) is a ______
   - If \(v\) is black, \((u, v)\) is a ______

4. (True / False) If there is a path from \(u\) to \(v\) in a directed graph, \(G\), and \(d[u] < d[v]\) in a DFS traversal, then \(v\) is descendant of \(u\) in the DFS forest.

Solution

1. Which type of edges are found in a DFS traversal of an undirected graph? **Tree Edge and Back Edge**

2. The presence of which type of edge indicates a cycle in a DFS traversal? **Back Edge**

3. Consider an edge \((u, v)\) when the edge is first explored. Determine the classification of the edge in the following scenarios:

   - If \(v\) is white, \((u, v)\) is a **tree edge**
   - If \(v\) is grey, \((u, v)\) is a **back edge**
   - If \(v\) is black, \((u, v)\) is a **forward edge or cross edge**

4. (True / False) If there is a path from \(u\) to \(v\) in a directed graph, \(G\), and \(d[u] < d[v]\) in a DFS traversal, then \(v\) is descendant of \(u\) in the DFS forest.

   **False.** Consider the vertices \(y\) and \(v\) on the following graph. If the DFS traversal starting at \(q\) explored the path \(q \rightarrow t \rightarrow y\) first, then \(d[y] < d[v]\). There is also clearly a path from \(y\) to \(v\). However, \(v\) is not a descendant of \(y\) in the DFS tree rooted at \(q\).
Problem 2

The CIS Department wants to assign prerequisites to their $n$ classes. They have a list of $m$ prerequisite pairings, such as CIS 110 is a prerequisite for CIS 120. Given a list of prerequisite pairings, design an algorithm that determines if this list of pairings is a valid list of pairings. For example, (CIS 110, CIS 120), (CIS 120, CIS 121), (CIS 121, CIS 110) is not a valid list of prerequisites. Analyze the runtime of this algorithm.

Solution

Algorithm:
1. Construct a graph $G$, where each of the vertices in $G$ are the $n$ classes in the CIS department and there is an edge $(u, v)$ if $u$ is a prerequisite for $v$.
2. Run DFS on $G$.
3. If the DFS traversal finds a back edge during its execution, return that this list is invalid. Otherwise, return that this is list is valid.

Runtime:

It takes $O(n + m)$ time to construct $G$ if we represent $G$ as an adjacency list. It takes $O(n + m)$ to run DFS on $G$ and determine if there were any back edges during the traversal. Thus, the overall runtime of this algorithm is simply $O(n + m) + O(n + m) = \boxed{O(n + m)}$. 

[Diagram of a graph with nodes q, n, s, t, y, r, v, w, x, z and edges connecting them]
Additional Problems

Problem 3

Recursively generate all the permutations of the character sequence 'ABCD'.

Solution

The key to understanding how we can generate all permutations of a given string is to imagine the string (which is essentially a set of characters) as a complete graph where the nodes are the characters of the string. This basically reduces the permutations generating problem into a graph traversal problem: given a complete graph, visit all nodes of the graph without visiting any node twice. How many different ways are there to traverse such a graph?

It turns out, each different way of traversing this graph is one permutation of the characters in the given string!

We can use DFS to traverse this graph of characters. The important thing to keep in mind is that we must not visit a node twice in any "branch" of the depth-first tree that runs down from a node at the top of the tree to the leaf which denotes the last node in the current "branch".

The code solution is on this website: [http://exceptional-code.blogspot.com/2012/09/generating-all-permutations.html](http://exceptional-code.blogspot.com/2012/09/generating-all-permutations.html)