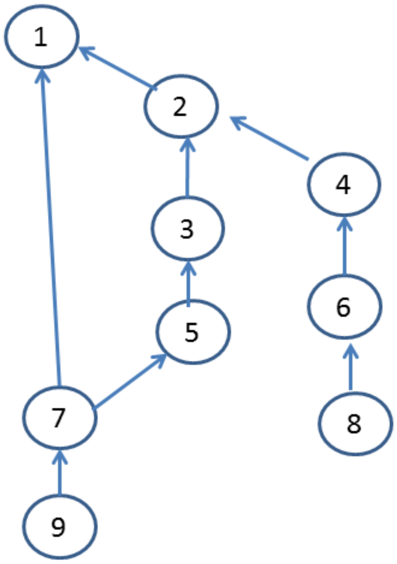
COS 226 Data Structures and Algorithms
Computer Science Department
Princeton University
Fall 2015

Week 8 Activity

1. Shortest Common Ancestor (SCA).

   (a) A common ancestral path between two vertices v and w in a directed graph is a directed path from v to a common ancestor x, together with a directed path from w to the same ancestor x. A shortest common ancestral path is a common ancestral path of minimum total length. We refer to the common ancestor in a shortest ancestral path as a shortest common ancestor (SCA). Given the digraph below, find the Shortest Common Ancestor of vertices 4 and 9 and find the length of a path of minimum length.



   (b) Assume that SCA is an instance of the ShotestCommonAncestor class (as defined in the wordnet assignment) Describe an algorithm for calculating SCA.ancestor(int v, int w). Your algorithm should run in linear time (proportional to V + E) and should work even if the digraph contains cycles.

   (c) (Homework)How would your algorithm differ if V and W are two sets of vertices (instead of a single vertex each) ?

2. Wordnet Assignment.

   (a) What is a synset?

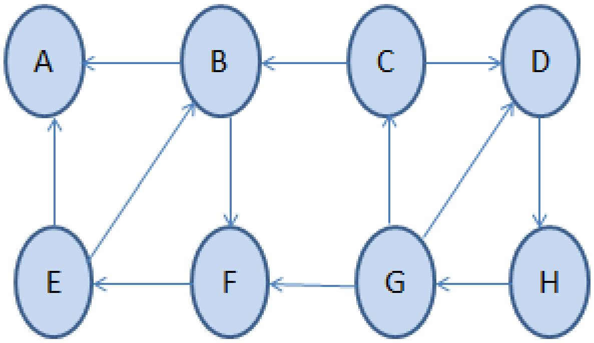
   (b) What is the relation between hyponym and hypernym? Provide an example.

   (c) State the definition of a rooted DAG and devise a simple algorithm to identify a rooted DAG. What is the order of growth of your algorithm?

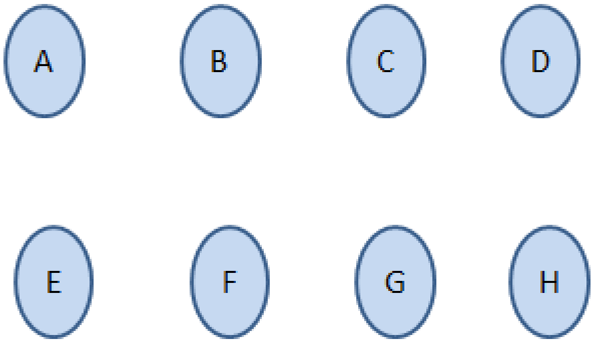
   (d) Discuss the classes and methods in the wordnet assignment, WordNet and ShortestCommonAncestor.

3. Extra Problem. Strong Components (SC)

(a) For this problem, use the digraph below. Assume that adjacency lists appear in alphabetical order. Assume that vertices are processed in alphabetical order. The digraph given below is G.



Draw reverse digraph G^R of G.



(b) What is the order of growth of finding reverse Digraph, given the adjacency-list representation of the graph?

(c) Find a topological sort of G^R. Start with A.

(d) Using DFS, original graph G, and your answer to (c), find the strong components (SC) of G. Do this by writing the id[] of each vertex next to the vertex. Assume the first SC is given id 0, the second SC is given id 1, etc.

(e) For (d), if we used the postorder of G instead of the reverse postorder of G^R, would we get the correct SCs? Hint: The postorder of G is AEFBGHDC.