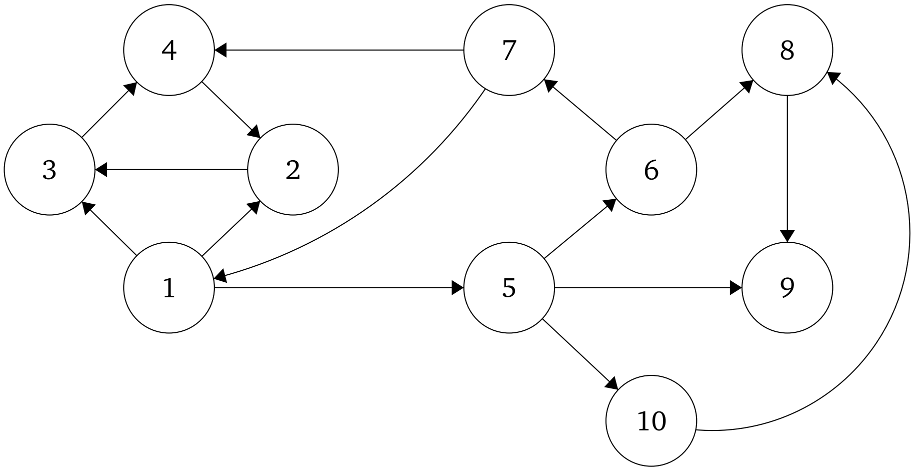
Recitation 10 – *DFS*

Parallel and Sequential Data Structures and Algorithms, 15-210 (Spring 2015)

March 17th, 2015

DFS Trees and Numberings

1. Draw the DFS tree of the following graph, visiting vertices in increasing order. Identify each edge as a tree, back, forward, or cross edge. Also determine the discovery and finishing times for each vertex.



2. When DFS is run on an undirected graph, is it possible to see either a foward or cross edge?

3. Let d_v and f_v be the discovery and finishing times for a vertex v, respectively. In each of the following scenarios, determine the ancestor/descendent relationship (if any) within the DFS tree between two vertices u, v.

   - $d_v < d_u < f_u < f_v$

   - $d_v < f_v < d_u < f_u$

4. You run DFS on a directed graph, and discover that there are no back edges. Does this graph contain a cycle?

Generalized DFS on Directed Graphs

Below is an implementation of DFS given in terms of abstract functions `revisit`, `discover`, and `finish`. We can define some piece of "state" information (for which we use the variable Σ) that will be passed around and modified by these functions during DFS.

```
fun DFS $(G, \Sigma_0, s) =$
  let
    fun DFS' $p$ $((X, \Sigma), v) =$
      if $v \in X$ then $(X,$ revisit$(\Sigma, v, p))$ else
      let
        val $\Sigma' =$ discover$(\Sigma, v, p)$
        val $X' = X \cup \{v\}$
        val $(X'', \Sigma'') =$ iter (DFS' $v$) $(X', \Sigma')$ $(N_G^+(v))$
        val $\Sigma''' =$ finish$(\Sigma', \Sigma'', v, p)$
      in
        $(X'', \Sigma''')$
      end
  in
    DFS' $s$ $((\emptyset, \Sigma_0), s)$
  end
```

Here's an example. Our state is a triple (D, F, t) where D and F are tables recording the discovery and finishing times of each vertex, respectively, and t is the current DFS time.

```
val $\Sigma_0 = (\emptyset, \emptyset, 0)$
fun revisit$(\Sigma, \_, \_) = \Sigma$
fun discover$((D, F, t), v, \_) = (D \cup \{v \mapsto t\}, F, t + 1)$
fun finish$(\_, (D, F, t), v, \_) = (D, F \cup \{v \mapsto t\}, t + 1)$
```

1. Consider an edge (p, v), and suppose that DFS just called `revisit`(Σ, v, p). Is (p, v) a tree edge? How about forward, back, or cross?

2. How many times will `discover` and `finish` be called, per vertex? How many times will `discover` and `revisit` be called in total?

3. Give definitions for Σ_0, `revisit`, `discover`, and `finish` which implement cycle detection on directed graphs. You should assume that the graph is first augmented with a new source vertex s and edges (s, v) for every other v in the graph. *Hint:* remember the relationship between cycles and back edges.

4. Will this cycle detection algorithm correctly detect cycles in undirected graphs? (Assume undirected graphs are represented as directed graphs with edges in both directions.)