FALL 1998: **COMP 7/8713: Advanced Algorithms**

Sample Solution

Problem

Given a connected, undirected graph $G(V, E)$ with n vertices and m edges, design an $O(n+m)$-time algorithm to determine whether or not the graph has a cycle of odd length.

Basic Idea

Perform DFS and label vertices -1 or 1 in such a way that all vertices adhjacent to a -1 vertex are labeled 1 and vice versa. If an odd cycle exists in the graph, then it must have 2 adjacent vertices labeled the same. The following algorithm is first called as DFS-VISIT$(G, 1, 1)$. It is a simple modification of DFS-VISIT from p478 of [CLR].

Algorithm

DFS-VISIT(G, u, b)
Comment: Assume that $label[u] = b$

$color[u] \leftarrow$ GRAY
$d[u] \leftarrow time \leftarrow time + 1$
for each vertex $v \in Adj[u]$ **do**
  if $color[v] =$ WHITE **then**
    $\pi[v] \leftarrow u$
    $label[v] \leftarrow -b$ ▷ New statement
    DFS-VISIT$(G, v, -b)$
  else if $label[u] = label[v]$ **then** ▷ New statement
    Print "Odd Cycle Exists"; **Stop** ▷ New statement
$color[u] \leftarrow$ BLACK
$f[u] \leftarrow time \leftarrow time + 1$

Proof of Correctness

Claim 1 If $e = (u, v)$ is a **tree edge** of the DFS tree, then $label[u] \neq label[v]$.

Claim 2 If the above algorithm encounters an edge $e = (u, v)$ with $label[u] = label[v]$, then e is a **back edge** of the DFS tree, and this edge along with the unique path in the tree from u to v forms an odd cycle.

Claim 3 If there exists an edge $e = (u, v)$ with $label[u] = label[v]$, then the algorithm will find it.

Claim 4 If there exists an odd cycle in G, then there must be two adjacent vertices with the same *label*, i.e., there must be an edge $e = (u, v)$ with $label[u] = label[v]$.

Analysis and Lower Bound

Time Complexity is the same as that of DFS, which is $O(m + n)$. Clearly the time complexity cannot be improved, since $O(m+n)$ is needed simply to read in the input. Thus the time complexity is $\Theta(m + n)$.