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 adjacent 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 $\text{label}[u] = b$
1 \hspace{1em} \text{color}[u] \leftarrow \text{GRAY}
2 \hspace{1em} d[u] \leftarrow \text{time} \leftarrow \text{time} + 1
3 \hspace{1em} \textbf{for} \ \text{each vertex} \ v \in \text{Adj}[u] \ \textbf{do}
4 \hspace{2em} \text{if} \ \text{color}[v] = \text{WHITE} \ \textbf{then}
5 \hspace{3em} \pi[v] \leftarrow u
6 \hspace{3em} \text{label}[v] \leftarrow -b \ \triangleright \text{New statement}
7 \hspace{1em} \textbf{else if} \ \text{label}[u] = \text{label}[v] \ \textbf{then} \ \triangleright \text{New statement}
8 \hspace{2em} \text{DFS-Visit}(G, v, -b)
9 \hspace{1em} \text{if} \ \text{color}[u] = \text{label}[v] \ \textbf{then} \ \triangleright \text{New statement}
10 \hspace{1em} \text{Print} \ \text{“Odd Cycle Exists”; Stop}
11 \hspace{1em} \text{color}[u] \leftarrow \text{BLACK}
12 \hspace{1em} f[u] \leftarrow \text{time} \leftarrow \text{time} + 1

Proof of Correctness

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

Claim 2 If the above algorithm encounters an edge $e = (u, v)$ with $\text{label}[u] = \text{label}[v]$, 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 $\text{label}[u] = \text{label}[v]$, 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 $\text{label}[u] = \text{label}[v]$.

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 $\Theta(m + n)$. 