CS 344: BFS and DFS Algorithms

BFS(G=(V,E),s)

for all $u \in V - \{s\}$ **do** $d(u) := \infty$

$d(s) := 0$

$Q := \{s\}$

while $Q \neq \emptyset$

  do remove u from Q

    for all $v \in Adj(u)$

      do if $d(v) := \infty$

        then $d(v) := d(u) + 1$; $\pi(v) := u$; put v onto Q

Remarks. Q is a FIFO queue. π is the predecessor array. BFS gives a tree. If G is connected, the tree is a spanning tree. Time complexity is $O(|V| + |E|)$.

DSF(G=(V,E))

for all $u \in V$ **do** $\mathit{color}(v) := \mathit{white}$

$\mathit{time} = 0$

for all $u \in V$ **do if** $\mathit{color}(u) = \mathit{white}$ **then** DFS-VISIT(u)

DSF-VISIT(u)

$\mathit{color}(u) := \mathit{gray}$

$d(u) := \mathit{time}$; $\mathit{time} := \mathit{time} + 1$

for all $v \in Adj(u)$

  do if $\mathit{color}(v) := \mathit{white}$ **then** DFS-VISIT(v)

$\mathit{color}(u) := \mathit{black}$; $f(u) := \mathit{time}$; $\mathit{time} := \mathit{time} + 1$

Remarks. Results in a forest of trees. Time complexity is $O(|V|+|E|)$. $d(u)$ is the first time u is visited, $f(u)$ is the last time u is visited. For every u, $d(u) < f(u)$.

Four types of edge:

tree edge: edges in tree.

back edge: from descendant to ancestor.

forward edge: nontree edge from ancestor to descendant.

cross edge: remainder.

An edge (u, v) is tree edge or forward edge if and only if $d(u) < d(v) < f(v) < f(u)$; it is a back edge if and only if $d(v) < d(u) < f(u) < f(v)$; it is a cross edge if and only if $d(v) < f(v) < d(u) < f(u)$.