Reading Assignment: Chapter 22

1 Depth-First Search

procedure Search(\(G = (V, E)\))

// Assume \(V = \{1, 2, \ldots, n\}\) //

time \(\leftarrow\) 0;
\(vn[1..n] \leftarrow 0;\) /* \(vn\) stands for visit number */

for \(i \leftarrow 1\) to \(n\)
    if \(vn[i] = 0\) then call \(dfs(i)\)

procedure \(dfs(v)\)

\(vn[v] \leftarrow time \leftarrow time + 1;\)

for each node \(w\) such that \((v, w) \in E\) do
    if \(vn[w] = 0\) then call \(dfs(w)\);

\(fn[v] \leftarrow time \leftarrow time + 1\) /* \(fn\) stands for finish number */
2 Topological Sort

- Problem: given a directed graph $G = (V, E)$, obtain a linear ordering of the vertices such that for every edge $(u, v) \in E$, $u$ appears in the ordering before $v$.

- Solution:
  - Use depth-first search, with an initially empty list $L$.
  - At the end of procedure $dfs(v)$, insert $v$ to the front of $L$.
  - $L$ gives a topological sort of the vertices.
3 Strongly Connected Components

- A directed graph is *strongly connected* if for every two nodes $u$ and $v$ there is a path from $u$ to $v$ and one from $v$ to $u$.

- Decide if a graph $G$ is strongly connected:
  - $G$ is strongly connected iff (i) every node is reachable from node 1 and (ii) node 1 is reachable from every node.
  - The two conditions can be checked by applying $dfs(1)$ to $G$ and to $G^T$, where $G^T$ is the graph obtained from $G$ by reversing the edges.

- A subgraph $G'$ of a directed graph $G$ is said to be a *strongly connected component* of $G$ if $G'$ is strongly connected and is not contained in any other strongly connected subgraph.

- An interesting problem is to find all strongly connected components of a directed graph. (Note that each node belongs to exactly one component.)
• Algorithm:

1. Apply depth-first search to $G$ and compute $fn[u]$ for each node.
2. Compute $G^T$.
3. Apply depth-first search to $G^T$:
   
   $$visited[1..n] \leftarrow 0$$

   for each vertex $u$ in decreasing order of $fn[u]$ do
   
   if $visited[u] = 0$ then call $dfs(u)$

4. The vertices on each tree in the depth-first forest of the preceding step form a strongly connected component.
4 Articulation Points and Biconnected Components

4.0.1 Definitions

- Let $G$ be a connected, undirected graph.
- An articulation point of $G$ is a vertex whose removal disconnects $G$.
- A bridge of $G$ is an edge whose removal disconnects $G$.
- A graph with at least two edges is biconnected if it contains no articulation points.
- A biconnected component of $G$ is a maximal biconnected subgraph.
- Each non-bridge edge belongs to exactly one biconnected component. (See Figure 23.10 on page 495 of the textbook.)
4.0.2 Identifying All Articulation Points

- Let $G_\pi$ be any depth-first tree of $G$.
- An edge in $G$ is a back edge iff it is not in $G_\pi$.
- The root of $G_\pi$ is an articulation of $G$ iff it has at least two children.
- A non-root vertex $v$ in $G_\pi$ is an articulation point of $G$ iff $v$ has a child $w$ in $G_\pi$ such that no vertex in subtree($w$) is connected to a proper ancestor of $v$ by a back edge. (subtree($w$) denotes the subtree rooted at $w$ in $G_\pi$.)

- Define

$$\text{low}[w] = \min \left\{ \frac{vn[w]}{vn[x]} : x \text{ is joined to some vertex in subtree}(w) \text{ by a back edge} \right\}$$

- A non-root vertex $v$ in $G_\pi$ is an articulation point of $G$ iff $v$ has a child $w$ such that $\text{low}[w] \geq vn[v]$. 

• Note that

\[ low[v] = \min \begin{cases} 
vn[v] \\
vn[w] : w \text{ is connected to } v \\
low[w] : w \text{ is a child of } v 
\end{cases} \]

• Computing \( low[v] \) for each vertex \( v \):

\begin{algorithm}
procedure Art(v, u)
/* visit \( v \) from \( u \) */
\begin{align*}
low[v] &\leftarrow vn[v] \leftarrow time \leftarrow time + 1; \\
\text{for each vertex } w \neq u \text{ such that } (v, w) \in E \text{ do} \\
\quad \text{if } vn[w] = 0 \text{ then} \\
\quad \quad \text{call } Art(w, v) \\
\quad \quad low[v] \leftarrow \min\{low[v], low[w]\} \\
\quad \text{else} \\
\quad \quad low[v] \leftarrow \min\{low[v], vn[w]\} \\
\quad \text{endif}
\end{align*}
endfor
\end{algorithm}

• Initial call: \( Art(1, 0) \).