In this note we review some of the properties of depth first search (DFS) and describe a linear-time DFS-based algorithm for computing strongly connected components.

The high idea behind the DFS traversal is very simple: explore edges that lead to unvisited vertices; if a "dead end" is reached, backtrack. Below is the pseudo-code for the DFS routines.

\[
\text{DFS}(G) \\
\text{counter} = 0 \\
\text{For } u \in V[G] \\
\quad \text{Mark } u \text{ as not visited} \\
\text{For } u \in V[G] \\
\quad \text{If } u \text{ was not visited} \\
\quad\quad \text{parent}[u] = \text{nil} \\
\quad \text{DFS-visit}(u)
\]

\[
\text{DFS-visit}(u) \\
\quad \text{Mark } u \text{ as visited} \\
\quad \text{counter} = \text{counter} + 1 \\
\quad \text{d}[u] = \text{counter} \\
\text{For } v \text{ adjacent to } u \\
\quad \text{If } v \text{ was not visited} \\
\quad\quad \text{parent}[v] = u \\
\quad \text{DFS-visit}(v) \\
\quad \text{counter} = \text{counter} + 1 \\
\quad \text{f}[u] = \text{counter}
\]

Define the DFS forest as \(\{(\text{parent}[u], u) \mid \forall u \in V \text{ parent}[u] \neq \text{nil}\}\). In class we proved many interesting properties about \(f[u]\) and \(d[u]\), the finishing and discovery time of a vertex \(u\). The following lemmas hold for both directed and undirected graphs.

**Lemma 1** (nesting property). Let \(u\) be a vertex in the graph, and \(v_1, \ldots, v_k\) the children of \(u\) in the DFS forest. The intervals \([d[v_i], f[v_i]]\) are pair-wise disjoint and span \((d[u], f[u])\).

**Lemma 2** (descendant property). Let \(u\) and \(v\) be vertices in the graph. Vertex \(v\) is a descendant of \(u\) in the DFS forest if and only if \(d[u] < d[v] < f[v] < f[u]\).

**Lemma 3** (unvisited path property). Suppose that right before making the call DFS-visit\((u)\) there was a path \(u \sim v\) using unvisited vertices. Then \(v\) will become a descendant of \(u\) in the DFS forest.

Let us use the above properties to argue the correctness of the following elegant algorithm for computing the strongly connected components (SCC) of a directed graph.

\[
\text{Find strongly connected components}(G) \\
i) \text{call DFS}(G) \text{ to compute finishing times } f[u] \text{ for every vertex } u. \\
ii) \text{call DFS}(G^{rev}) \text{ processing vertices in the main loop in decreasing order of } f[u]. \\
iii) \text{output each tree in the second DFS forest as a separate component.}
\]

Because of Lemma 3 we know that each SCC is contained within a single tree in the DFS forest. This is good, because we do not want a component to be split across different trees.
But note that a given tree may, in principle, contain more than one SCC; we must show that at most one is contained.

**Lemma 4.** Every tree in the second DFS forest corresponds to exactly one SCC.

*Proof.* Let $T$ be a tree in the DFS forest. Suppose that $r$, the root of $T$, belongs to the strongly connected component $C$. Thus $C \subseteq V[T]$, we need to argue that $C = V[T]$.

How can DFS exit component $C$? By taking an edge $(u, v) \in E[G^{rev}]$ such that $u \in C$ and $v$ belongs to some other component $C'$. However, the edge will not be used if component $C'$ had already been visited before the the call $\text{DFS-visit}(r)$ was made. One way this could happen is if

$$\max_{w \in C'} f[w] > \max_{w \in C} f[w]. \tag{1}$$

Let us see what happens in the first DFS execution. Let $x \in C'$ and $y \in C$ be the vertices in $C'$ and $C$ with smallest discovery time. By Lemmas 2 and 3 it follows that $f[x] = \max_{w \in C'} f[w]$ and $f[y] = \max_{w \in C} f[w]$. Therefore, condition (1) holds provided

$$f[x] > f[y]. \tag{2}$$

Consider the following two cases:

*case 1:* $d[x] < d[y]$. Because $C'$ and $C$ are strongly connected when the call $\text{DFS-visit}(x)$ was made there was a path $x \leadsto u \rightarrow v \leadsto y$ using unvisited nodes. Applying Lemmas 2 and 3 we get that $f[y] < f[x]$.

*case 2:* $d[y] < d[x]$. Suppose $f[x] < f[y]$, by Lemma 2 vertex $x$ must be a descendant of $y$; this means there is path $y \leadsto x$, but we just argued that there is a path $x \leadsto y$. This contradicts the fact that $x$ and $y$ are in different strongly connected components. It must be that $f[y] < f[x]$.

Hence DFS never leaves $C$ and the Lemma follows. \hfill $\square$