Depth-First Search and Topological Sort

August 9, 2008

This time we consider the Graph-DFS algorithm as depicted in Figure 1. It performs a depth-first search of \( G \) and creates a forest, by repeatedly calling the recursive DFS procedure on a fresh unmarked node whenever the call to the DFS\((G, v)\) procedure returns to the top-level and not all nodes have been marked. A node \( v \) is considered unmarked as long as \( d[v] = \bot \), and it is marked once \( d[v] \) is set to a finite time. Our focus in this note will be the connection between DFS and Topologically sorting a DAG.

\[
\text{Graph-DFS}(G): \\
1 \quad \text{time} \leftarrow 0 \\
2 \quad \text{for every } v \in V \text{ do } d[v] \leftarrow \bot; f[v] \leftarrow \bot; /* nodes start out unmarked */ \\
3 \quad \text{for every } v \in V \text{ do } \\
4 \quad \text{if } v \text{ not yet marked then } \\
5 \quad \text{DFS}(G, v)
\]

Figure 1: Graph-DFS

We say that node \( w \) is reachable from \( v \) in \( G \), denoted \( v \rightarrow w \), if there is a path from \( v \) to \( w \) in \( G \). We now consider a few of the properties of Graph-DFS, stated in terms of the discovery times \( d[v] \) and finishing times \( f[v] \) for the nodes. Clearly, at the end of the Graph-DFS algorithm we have that \( 1 \leq d[v] < f[v] \leq 2|V| \) for all \( v \in V \). Moreover, since the time variable is incremented before every use, the times assigned to the \( d \) and \( f \) events are all distinct.

We start with the following lemma that captures inclusion or exclusion property of \( d-f \) intervals of different nodes: For any two nodes, either one node’s \( d-f \) interval contains the other’s, or the two intervals do not intersect. So \( d \) and \( f \) act as correctly balanced parentheses (SogRayim), with \( f[v] \) the closing parenthesis matching \( d[v] \):
DFS(G, s):

11 \hspace{1em} \text{time} \gets \text{time} + 1
12 \hspace{1em} \text{d}[s] \gets \text{time} \quad \text{/* discovery time for } s \text{ */}
13 \hspace{1em} \text{visit } s
14 \hspace{1em} \text{for every } w \in \text{Adj}[v] \text{ do}
15 \hspace{2em} \text{if } \text{d}[w] = \bot \text{ then} \quad \text{/* w unmarked */}
16 \hspace{2em} \text{DFS}(G, w)
17 \hspace{1em} \text{DFS\_tree} \gets \text{DFS\_tree} \cup \{(s, w)\}
18 \hspace{1em} \text{time} \gets \text{time} + 1
19 \hspace{1em} \text{f}[s] \gets \text{time} \quad \text{/* finishing time for } s \text{ */}

Figure 2: DFS procedure (with timestamping)

**Lemma 1** If \( d[v] < d[w] < f[v] \) then

1. \( d[v] < d[w] < f[w] < f[v] \), and
2. there is a path \( v \leadsto w \) in \( G \).

**Proof:** This immediately follows from Theorem 23.6 in the book. \( \square \)

We now turn to discuss DFS on directed acyclic graphs.

**Definition 1** \( G \) is a DAG if it is a directed graph that contains no cycles.

\( (\text{DAG} = \text{Directed, Acyclic Graph.}) \)

We can now show:

**Lemma 2** Let \( G \) be a DAG. If \( d[v] < d[w] \) and \( v \leadsto w \) in \( G \), then
\( d[v] < d[w] < f[w] < f[v] \).

**Proof:** By Lemma 1 it suffices to show that \( d[w] < f[v] \). We prove the claim for all nodes \( v, w \) in \( G \) by induction on the distance \( k = \delta(v, w) \) between \( v \) and \( w \). The base case is \( \delta(v, w) = 1 \). By assumption, \( d[v] < d[w] \). Since \( w \in \text{Adj}[v] \), the loop on line 14-15 will test for \( d[w] = \bot \) during the execution of \( \text{DFS}(G, v) \). Since this occurs before lines 18-19 are reached, it happens when \( \text{time} + 1 < f[v] \). If \( d[w] \neq \bot \) at that point, then \( d[w] \) was set after \( d[v] \) and so \( d[w] < f[v] \), as desired. If \( d[w] = \bot \) there,
Lemma 3

If $G$ is a DAG and $v \leadsto w$ in $G$, then every execution of Graph-DFS satisfies that $f[w] < f[v]$.


- Assume by way of contradiction that $d[w] < d[v]$. In this case we have that $d[w] < d[v] < f[w]$ and so $w \leadsto v$ by Lemma 1. But since $v \leadsto w$ is given, we obtain that there is a cycle $w \leadsto v \leadsto w$ in $G$, contradicting the assumption that $G$ is a DAG.

- Finally, assume that $d[v] < d[w]$. Recall that $v \leadsto w$ in $G$ by assumption. It now follows by Lemma 2 that $f[w] < f[v]$, as desired.
Definition 2 A topological sort of a directed graph $G$ is an ordering of the nodes of $V$ satisfying that for all $(v, w) \in E$ the node $v$ appears before $w$ in the ordering.

It is immediate to check that if $G$ contains a cycle then it can not be topologically sorted, since if both $v \Rightarrow w$ and $w \Rightarrow v$ hold in $G$, then each of the two nodes must appear before the other in each and every any topological sort of $G$. Lemma 3 immediately implies

**Corollary 1** If $G$ is a DAG and $f[v] < f[w]$ holds in some run of Graph-DFS on $G$, then there is no path $v \Rightarrow w$ on $G$.

Based on Corollary 1, we obtain that a DAG can be topologically sorted by running Graph-DFS and ordering the nodes according to decreasing $f[s]$ order. The complexity of topological sort when performed in this manner is $O(|V| + |E|)$.