COMPSCI 330: Design and Analysis of Algorithms **October 17, 2017**

Lecture 12: Graph Algorithms

Lecturer: Rong Ge *Scribe: Will Long*

12.1 Types of Edges

Given a graph $G = (V, E)$, we can use depth-first search to construct a tree on G. An edge $(u, v) \in E$ is in the tree if DFS finds either vertex u or v for the first time when exploring (u, v). In addition to these tree edges, there are three other edge types that are determined by a DFS tree: forward edges, cross edges, and back edges. A forward edge is a non-tree edge from a vertex to one of its descendants. A cross edge is an edge from a vertex u to a vertex v such that the subtrees rooted at u and v are distinct. A back edge is an edge from a vertex to one of its ancestors. The graphic below depicts the four types of edges for a DFS tree that was initialized from vertex s. Solid lines indicate tree edges.

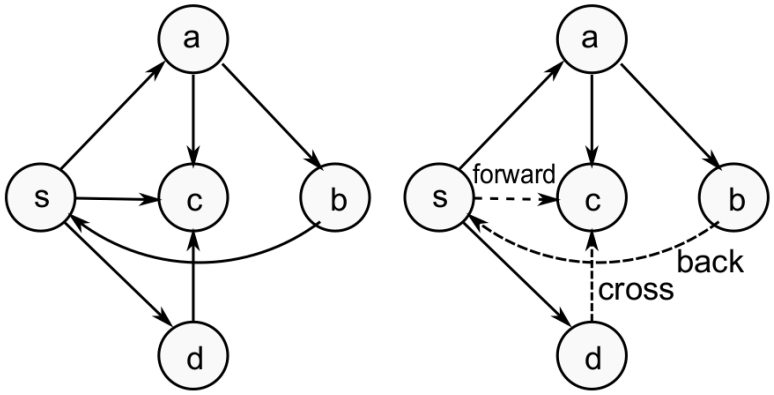

Figure 12.1: The Four Edge Types

For DFS trees, edges can also be classified using the pre-order and post-order of their vertices. Recall that in DFS, the pre-order of a vertex is when it is pushed into the stack, and the post-order is when it is popped off the stack. For a given edge (u, v), we have the following pre/post-orders for each type:

Edge Type (u, v)	**Pre/Post-Order**
Tree/forward	$\text{pre}(u) < \text{pre}(v) < \text{post}(v) < \text{post}(u)$
Back	$\text{pre}(v) < \text{pre}(u) < \text{post}(u) < \text{post}(v)$
Cross	$\text{pre}(v) < \text{post}(v) < \text{pre}(u) < \text{post}(u)$

We will now show two applications of DFS: cycle-finding and topological sort.