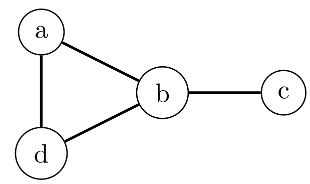
CMSC 451
Summer 2006

Cut vertices/edges

Julián Mestre

Let G be a connected undirected graph on n vertices and m edges. Vertex u is said to be a cut vertex if its removal disconnects G. Similarly edge (u, v) is a cut edge if removing (u, v) disconnects the graph. For example, in the picture on the right b is a cut vertex and (b, c) is a cut edge—removing either leaves vertex c isolated from the rest of the graph.



Suppose we run DFS on G. Because the graph is connected, the DFS forest consists of a single tree T. The first thing to notice is that non-tree edges are not cut edges; this is because the tree keeps the graph connected even if all non-tree edges are removed. This is important because the size of the search space for cut edges reduces from m to just $n - 1$.

Let T_u be the subtree of T rooted at u, i.e., T_u includes u and all of u's descendants. Using the discovery times of the DFS define for every vertex u

$$\texttt{low}[u] = \min \begin{cases} \texttt{d}[w] \mid (x, w) \text{ is a non-tree edge and } x \in T_u \\ \texttt{d}[u] \end{cases} \tag{1}$$

Recall that in the DFS forest of an undirected graph all non-tree edges connect two nodes such that one is the descendant of the other. Intuitively, $\texttt{low}[u]$ is trying to capture how high up the tree we can go from u by taking zero or more edges down the tree followed by a single non-tree edge going up to reach a node w. In order to keep $\texttt{low}[u]$ well defined (in the case that no non-tree edge exists) we also consider the option of staying at u. Because the discovery time of the vertices decreases as we go up the tree we take the minimum of the discovery times of the vertices before mentioned.

Armed with the low values it is very easy to identify a cut edge.

Lemma 1. *A tree edge* $(u, \texttt{parent}[u])$ *is a cut edge if and only if* $\texttt{low}[u] = \texttt{d}[u]$.

Proof. Removing the edge $(u, \texttt{parent}[u])$ can potentially disconnect T_u from the rest of the graph. The only way to remain connected is through a non-tree edge (x, w) where $x \in T_u$ and $w \notin T_u$. In that case we know that $\texttt{d}[u] > \texttt{d}[w] \geq \texttt{low}[u]$. On the other hand, if no such edge exists we have $\texttt{d}[u] = \texttt{low}[u]$. □

The characterization of cut vertices is not as clean as that of cut edges. First note that removing a leaf of the DFS tree cannot disconnect the graph, as the rest of tree connects the remaining vertices. For other vertices we consider two cases.

Lemma 2. *The root of the tree is a cut vertex if and only if it has more than one child.*

Proof. If the root has only one child, like in the case of a leaf, removing the root does not disconnect the graph. However, if it has more than one child, because there are no edges between the subtrees of the children, removing the root disconnects the graph. □

Lemma 3. *An internal vertex* u *is a cut vertex if and only if* u *has a child* x *with* $\texttt{low}[x] \geq \texttt{d}[u]$.

Proof. Let x be a child of u. If the removal of u disconnects T_x from the rest of the graph then u is a cut vertex. This happens when the non-tree edges out of T_x cannot take us higher than u up on the tree, or equivalently if $\texttt{low}[x] \geq \texttt{d}[u]$. On the other hand, if the subtree of *every* child is connected to the rest of the graph then u is not a cut vertex. □