Strong connectivity in digraphs

Definition. Two vertices v and w are strongly connected if they are mutually reachable: that is, if there is a directed path from v to w and a directed path from w to v. A digraph is strongly connected if all its vertices are strongly connected to one another.

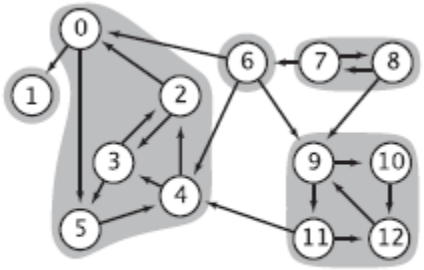
Strong connectivity in digraphs is an equivalence relation on the set of vertices, as it has the following properties:

Reflexive : Every vertex v is strongly connected to itself.

Symmetric : If v is strongly connected to w, then w is strongly connected to v.

Transitive : If v is strongly connected to w and w is strongly connected to x, then v is also strongly connected to x.

Equivalence classes form strongly connected components.



A digraph and its strong components

API

public class	SCC	
	SCC(Digraph G)	*preprocessing constructor*
boolean	stronglyConnected(int v, int w)	*are* v *and* w *strongly connected?*
int	count()	*number of strong components*
int	id(int v)	*component identifier for* v (*between* 0 *and* count()-1)

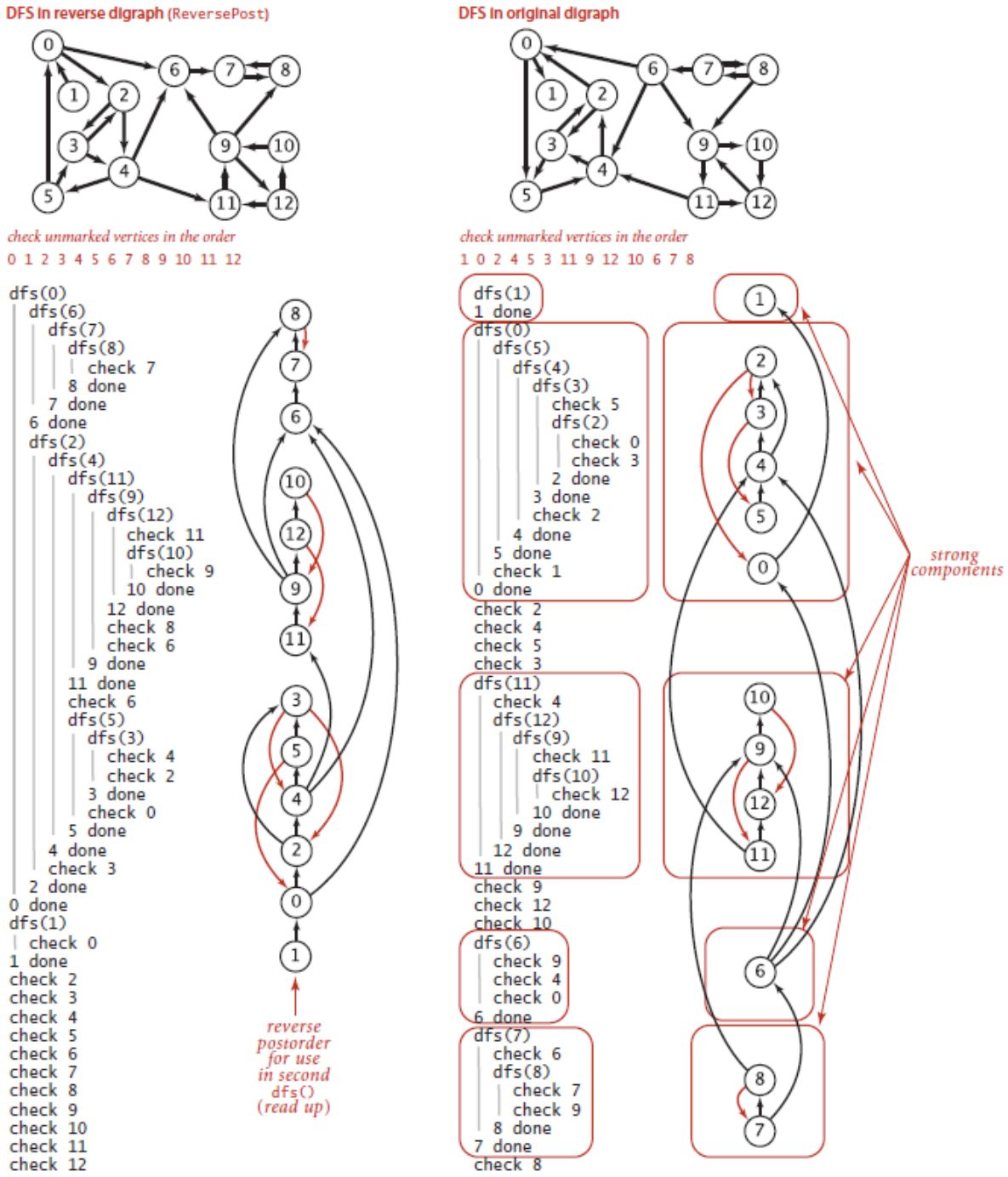
API for strong components

Kosaraju's algorithm

Run standard DFS on G and put vertices in stack as visited.

Use DFS to compute the reverse postorder of its G's reverse.

Pop element from the stack and use DFS on G, marking visited vertices. This gives strongly connected components.



Kosaraju's algorithm for finding strong components in digraphs

Theorem: Kosaraju's algorithm uses preprocessing time and space proportional to V+E to support constant-time strong connectivity queries in a digraph.