Computation of the connectivity number $\kappa(G)$

$G = (V, E)$ is a graph with $n := |V|$, $m := |E|$.

(i) Check whether $\kappa(G) \geq 1$ holds (G is connected):

in linear time $O(n + m)$ time by applying Depth First Search (DFS).

(ii) Check whether $\kappa(G) \geq 2$ holds (G is 2-connected):

in linear time $O(n + m)$ time by applying Depth First Search (DFS).

(iii) Check whether $\kappa(G) \geq 3$ holds (G is 3-connected):

in linear time $O(n + m)$ time, see e.g J.E.Hopcroft and R.E.Tarjan, Dividing a graph into triconnected components, *SIAM J. on Computing* **2**, 1973, 135–158.

(iv) compute $\kappa(G)$

in $O(n^4\sqrt{m})$ time applying Menger's theorem and the push-relabel algorithm (PRA) for the max-flow problem (run PRA for every pair of vertices, time complexity $O(n^2\sqrt{m})$ per pair).

Faster: in $O(\sqrt{n}m^2)$ time by S. Even and R.E.Tarjan, Network flow and testing graph connectivity, *SIAM J. on Computing* **4**, 1975, 507–518.

Computation of the edge-connectivity number $\lambda(G)$

$G = (V, E)$ is a graph with $n := |V|$, $m := |E|$.

Apply Mengers's theorem:
$\lambda(G)$ equals the minimum cut in G

A minimum cut in G can be computed in $O(mn + n^2 \log n)$
by the Stoer-and-Wagner algorithm (SVA)
M. Stoer and F. Wagner, A simple min-cut algorithm, *Journal of the ACM* **44**, 1997, 585–591.

SVA uses *t*he maximum adjacency order (MA order) and
has been discussed in Combinatorial Optimization 1

see also `https://en.wikipedia.org/wiki/Stoer-Wagner_algorithm`

DFS: definitions and notations

Apply $DFS(G, s)$ for a connected graph $G = (V, E)$ with $n := |V|$, $m := |E|$, $s \in V$.

Definition 1.

The edges $\{parent[v], v\} \in E$, for $v \in V$, are called **tree-edges***. Set $E_{DFS} := \big\{\{parent[v], v\} \colon v \in V \setminus \{s\}\big\}$ and $T := (V, E_{DFS})$.*

Observe: T is a spanning tree in G and is called the **DFS-tree**.
T is rooted at s with tree-order $\preceq$:

$$v \preceq w \text{ iff } v \text{ lies on the unique } s\text{-}w\text{-path in } T.$$

Definition 2.

If $v \preceq w$, then w is a **descendant** *of v and v is an* **ancestor** *of w.*

For $v \in V$, T_v is the tree formed by the descendants of v, rooted at v.

$\{v, w\} \in E \setminus E(T)$ is called a **backward edge** *starting at v, if $DFSNum[w] < DFSNum[v]$ at the moment where the DFS explores $\{v, w\}$ for the first time starting at v (in the line* `FOR` *$w \in N(v)$ of the pseudocode).*
Otherwise, $\{v, w\}$ is a **forward edge***.*

The set of backward edges is denoted by E_B.

DFS: definitions and notations

Observe: All non-tree edges are backward edges, i.e. $E_B = E \setminus E_{DFS}$. For every $\{v, w\} \in B$, $w \prec v$ holds, i.e. w is contained in the s-v-path in T.

Direction: direct each $\{u, v\} \in E(G)$ from the vertex at which DFS explores it first. **Notation:** (u, v) if the direction is from u to v.

For $v \in V(G)$ set $LowPoint[v] :=$
$\min \Big\{ \{ DFSNum[v] \} \cup \{ DFSNum[z] \colon v \preceq x\,, (x, z) \in E_B \} \Big\}$.

Equivalently: $LowPoint[v] := \min(\{ DFSNum[v] \} \cup A_v)$, where
$A_v := \{ DFSNum[z] \colon (v, z) \in E_B \} \cup \{ LowPoint[w] \colon (v, w) \in E(T) \}$.

Definition 3.
A tree-edge $(v, w) \in E(T)$ is called a **leading edge** *iff* $LowPoint[w] \geq DFSNum[v]$.

Equivalently: (v, w) is a leading edge iff every backward egde starting at T_w ends either at T_w or at v.

Using DFS to recognize blocks and cut-vertices of a graph

Let G be an arbitrary (not necessarily connected) graph.

Lemma 1.

Let T be the output of DFSb. Let H be a connected component of G and s be the root of $T[V(H)]$. Let $v \in V(H) \setminus \{s\}$, $u = parent[v]$ and $(v, w) \in E(H)$. The following statements are equivalent:

(i) *(u, v) and (v, w) lie in the same block of G.*

(ii) *(v, w) is not a leading edge.*

Lemma 2.

Let T be the output of DFSb. Let H be a connected component of G and s be the root of $T[V(H)]$. The following statements hold:

(i) *All tree edges incident with s are leading edges. s is a cut-vertex iff it is incident to at least two leading edges.*

(ii) *A vertex $v \in V(H) \setminus \{s\}$ is a cut-vertex iff there exists at least one leading edge (v, w).*

Using DFS to recognize blocks and cut-vertices of a graph (contd.)

Theorem 1 ((Tarjan 1972)).

The blocks and the cut-vertices of G can be determined in linear time, i.e. in $O(n+m)$, where $n := |V|$ and $m := |E|$.

Corollary 2.

For a given graph G with $n := |G| > 3$ and $m := |E(G)|$ it can be tested in $O(n+m)$ time whether $\kappa(G) \geq 2$.

Proof: Observe that G is 2-connected iff $bc(G)$ is a singleton.