Lec31 Articulation Points Michael Monagan

April 12, 2021   12:10 PM

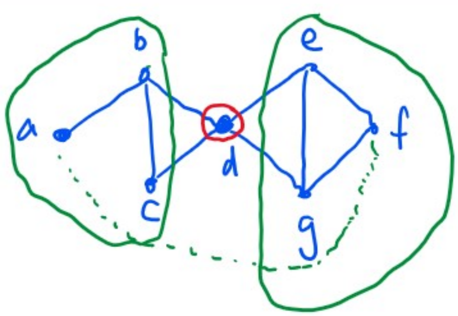
Lecture 31: Articulation Points and Biconnected Components

Grimaldi 12.5

An application of the depth-first search spanning tree.

Wednesday: Minimum Spanning Trees
Friday: Final exam info and a review

Articulation Points



If the vertices are servers (cities) and the edges are cables (roads), if the server (city) d goes down then the network is disconnected. Adding {a,f} increases the reliability of the network.

Suppose G is has 4 vertices. What is the minimum # edges

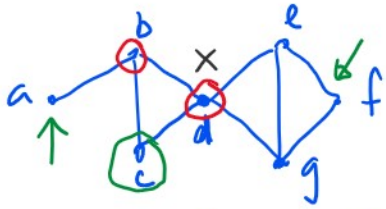
Definition (Articulation Point)

Let $G = (V, E)$ be a graph. A vertex v in G is an **articulation point** (AP) if removing v from G increases the number of connected components of G.

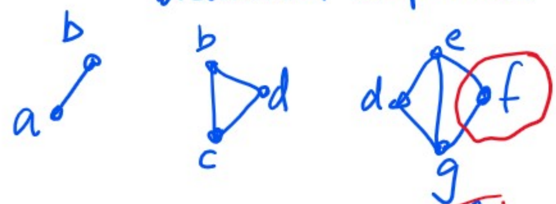
needed so that G is connected and has no A.P.s ?

Lemma (12.3)

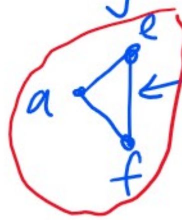
Let $G = (V, E)$ be a graph. A vertex $v \in V$ is an articulation point of G if and only if there are two vertices x and y in V such that $x \neq y \neq v$ and every path between x and y includes v.



Biconnected Components.



All paths from a to f go through d therefore d is an A.P. (cut vertex).



This is biconnected but not maximal.

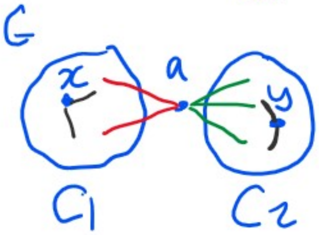
Definition (Biconnected Component)

Let $G = (V, E)$ be a graph. A subgraph of G is **biconnected** if it is connected and has no articulation points. A maximal biconnected subgraph of G is called a **biconnected component** of G.

Lemma

Let $G = (V, E)$ be a graph. If G has a Hamiltonian cycle then G must have no APs, equivalently, G must be biconnected.

Proof. Suppose vertex a is an A.P. in G. Then G−a has at least two connected components C_1 and C_2.



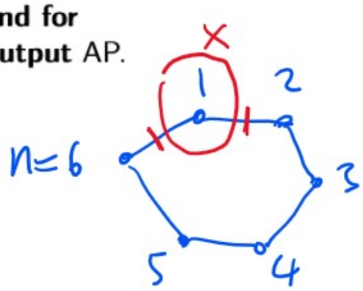
G cannot include a cycle which includes x and y because all paths from x to y go through a.

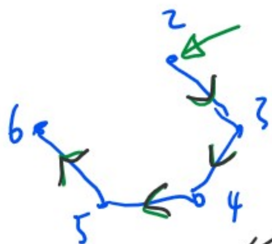
Exercise: Find a biconnected graph (no A.P.s) which does not have a Hamiltonian cycle.

How can we find the Articulation points in a **connected** graph G?

Algorithm 1.

set AP $= \phi$.
for each $v \in V$ do
  if removing v from G disconnects G **then**
    set AP $=$ AP $\cup\{v\}$.
  end if
end for
output AP.





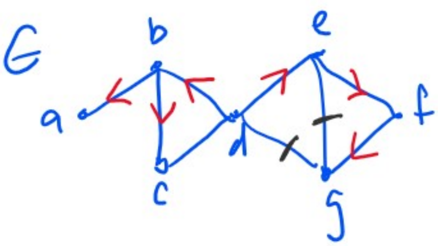
removing each vertex

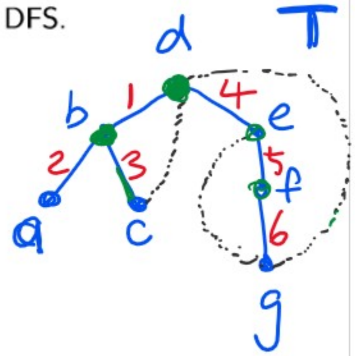
checking if $G-V$ is connected

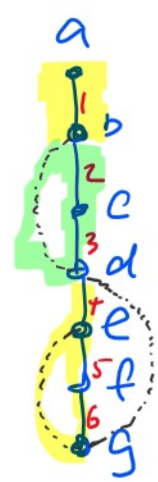
The work done is prop. to $n\cdot(n-1)$ is quadratic in n.

How can we find the APs and BCs in a graph G?

Step 1: Construct a DFS spanning tree T for G and number the edges in T in the order visited during the DFS.



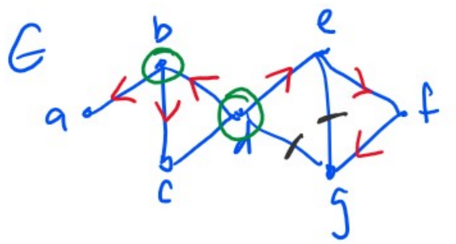


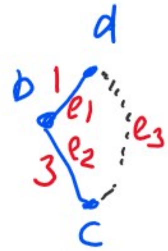


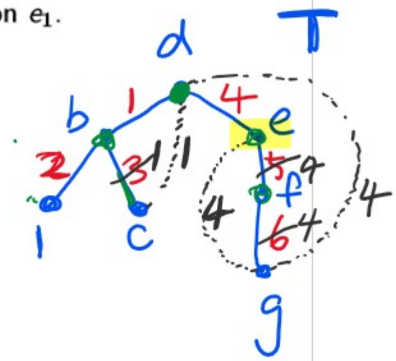
We will start at vertex d.
All edges in G not in T are called back edges. Observe.
- A leaf vertex in T is not an A.P. a, c, g ← are on a path in T.
- The root is an A.P. iff it has ≥ 2 children deg 1. degree 2.
- Other vertices in T are A.P.s if a descendant has no backedge going around it. E.g. b.

Step 2:

Traverse T in pre-order. If a vertex v has a backedge e_n from u to v, number all edges on the walk $v\, e_1\, x_1\, e_2 \ldots x_{n-2}\, e_{n-1}\, u\, e_n\, v$ from v down to u and back to v with the edge number on e_1.





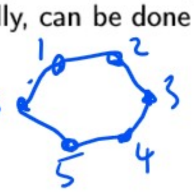


What are the articulation points? The A.P.s are the vertices whose incident edges are numbered differently: b, d

What are the biconnected components? The BCs are the maximal subgraphs of T (including the back edges) with the same edge numbers.

Why is this algorithm better than Algorithm 1?
If implemented carefully, can be done in time proportional to $|V| + |E|$.

C_n $n=6$



$n+n=\underline{\underline{2n}}$