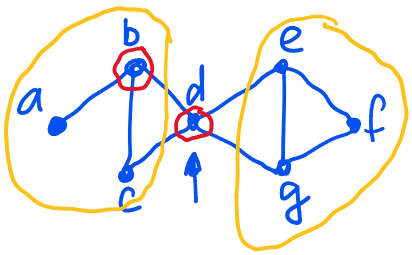
Lecture 30 Articulation Points and Biconnected Components

Grimaldi 12.5

An application of the depth-first search spanning tree.

Articulation Points



If the vertices are servers (cities) and edges are cables (roads), if the server (city) d goes down the network is disconnected. Adding a edge from a to g increases the reliability of this network.

A vertex of degree 1 is not an AP.

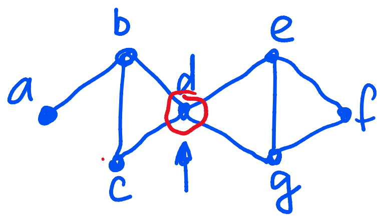
Cut vertex

Definition (Articulation Point)

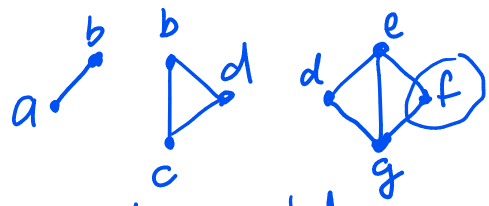
Let $G = (V, E)$ be a graph. A vertex v in G is an **articulation point** (AP) if removing v from G increases the number of connected components of G.

Lemma (12.3)

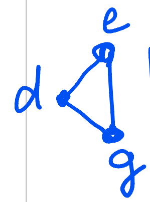
Let $G = (V, E)$ be a graph. A vertex $v \in V$ is an articulation point of G if and only if there are two vertices x and y in V such that $x \neq y \neq v$ and every path between x and y includes v.



All paths from a to f go through d therefore d is AP.



biconnected components BCs.



biconnected but not maximal.

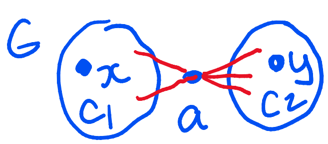
Definition (Biconnected Component)

Let $G = (V, E)$ be a graph. A subgraph of G with no articulation points is **biconnected**. 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 a is an A.P. in G. Then G−a has at least two connected components C_1 and C_2.



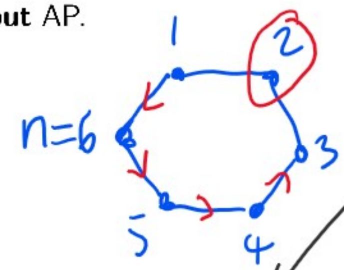
G cannot have a cycle which includes x and y because all paths from x to y go through a.

Exercise: Find a biconnected graph 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.



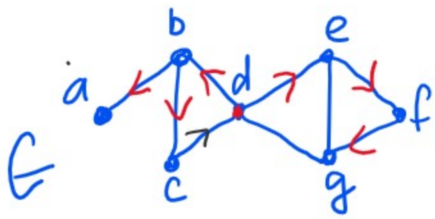
removing each vertex

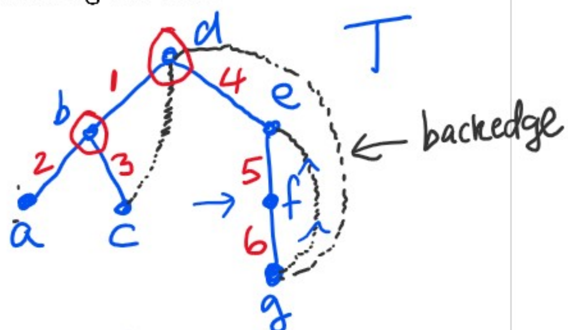
checking for connected.

The work done is prop. to $n\,(\underline{n-1})$ which 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.

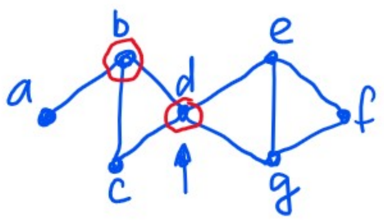


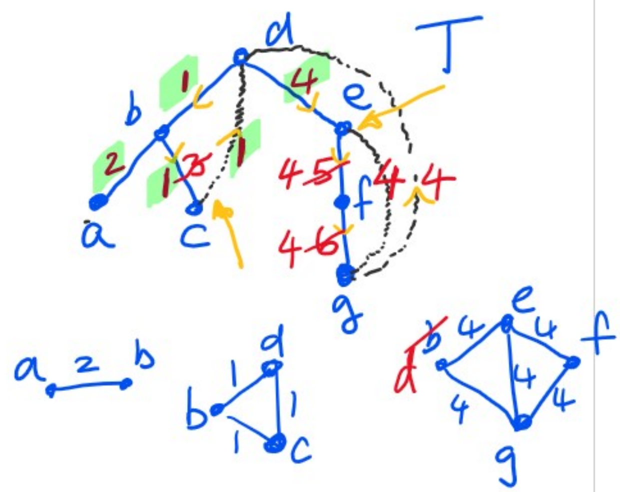


We will start at d.
Observe all edges in G not in T are called back edges.

- A leaf in T is not an AP : a, c, g. Why?
- The root is an AP iff it has $\underline{\geq 2}$ children.
- Other vertices in T are APs if a descendent has no backedge that goes around them.

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





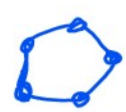
Redo example using vertex a as the root.

What are the articulation points? The APs are the vertices in T whose incident edges are not numbered the same.

What are the biconnected components?

The BCs are the maximal subgraphs of T (including backedges) 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  $n + n = 2n.$