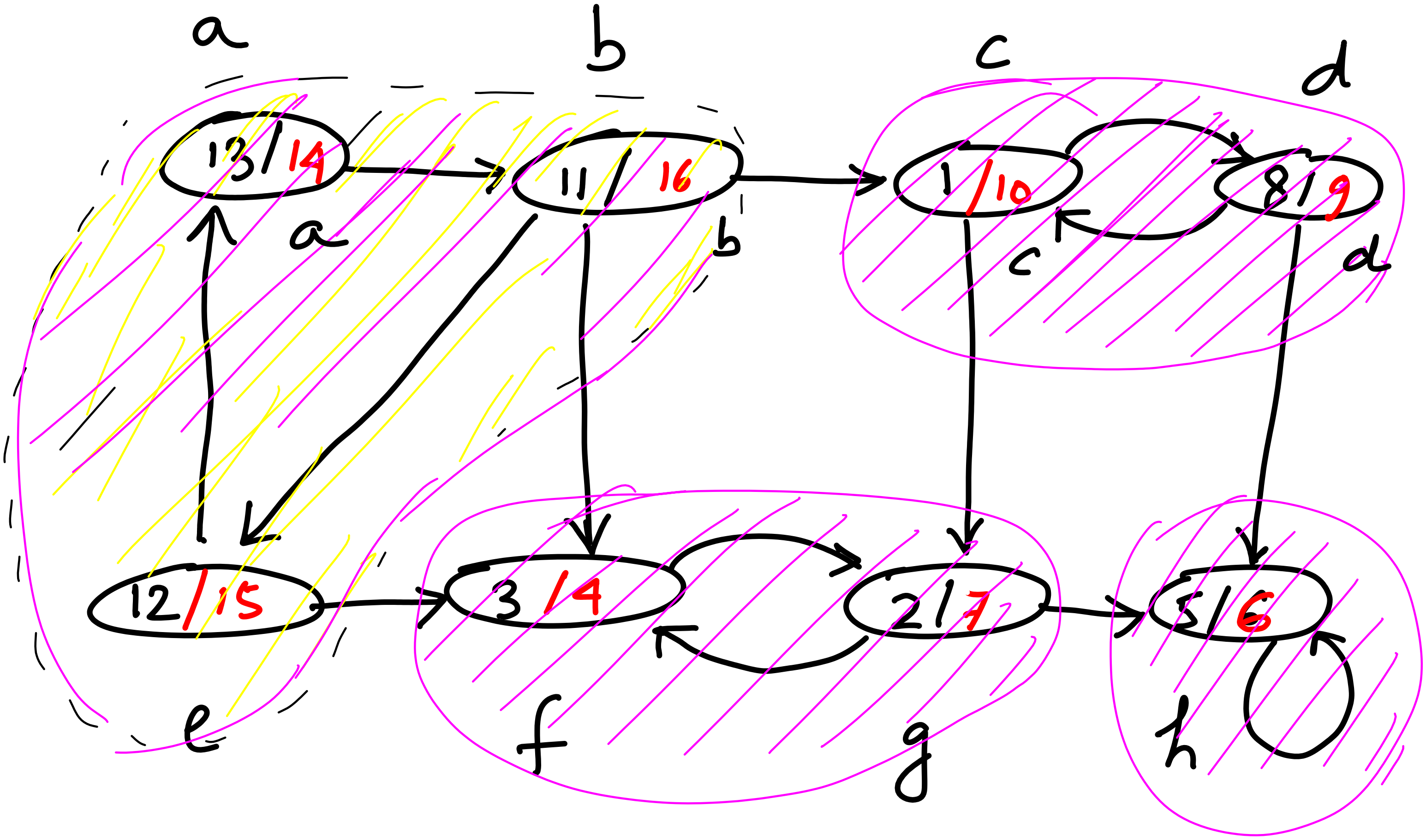
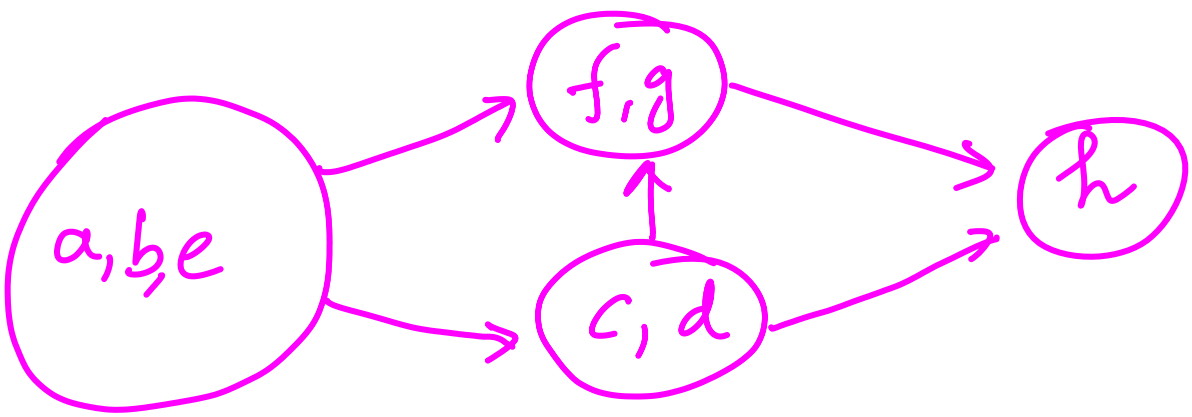
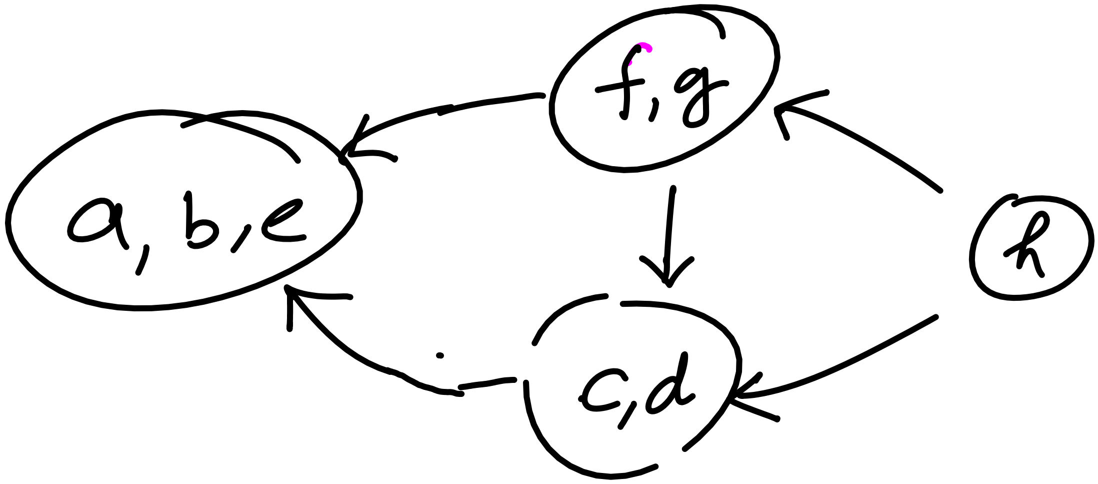
CSL 356 Lecture 38

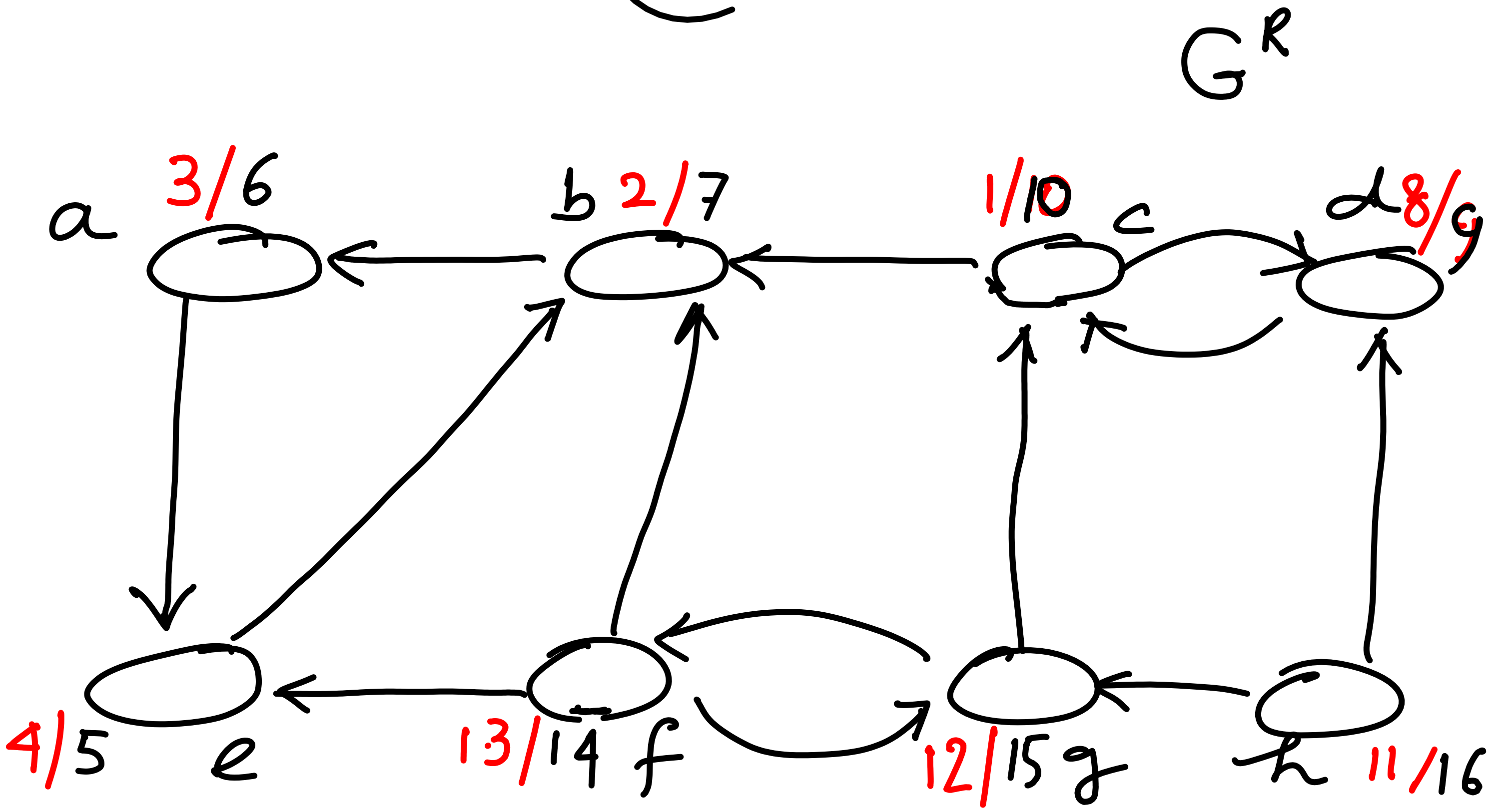




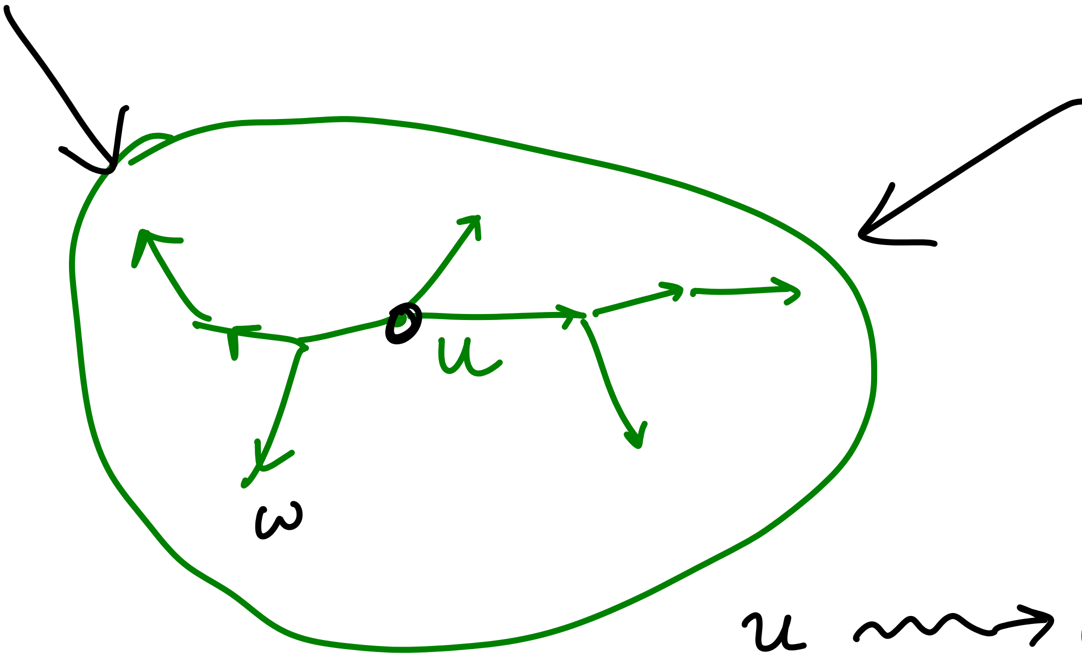
G^R	h	g	f	c	d	b	a	e
	16	15	14	10	9	7	6	5



f, g
a, b, e
c, d
h



G^R
a 3/6
b 2/7
1/10 c
d 8/9
4/5 e
13/14 f
12/15 g
h 11/16



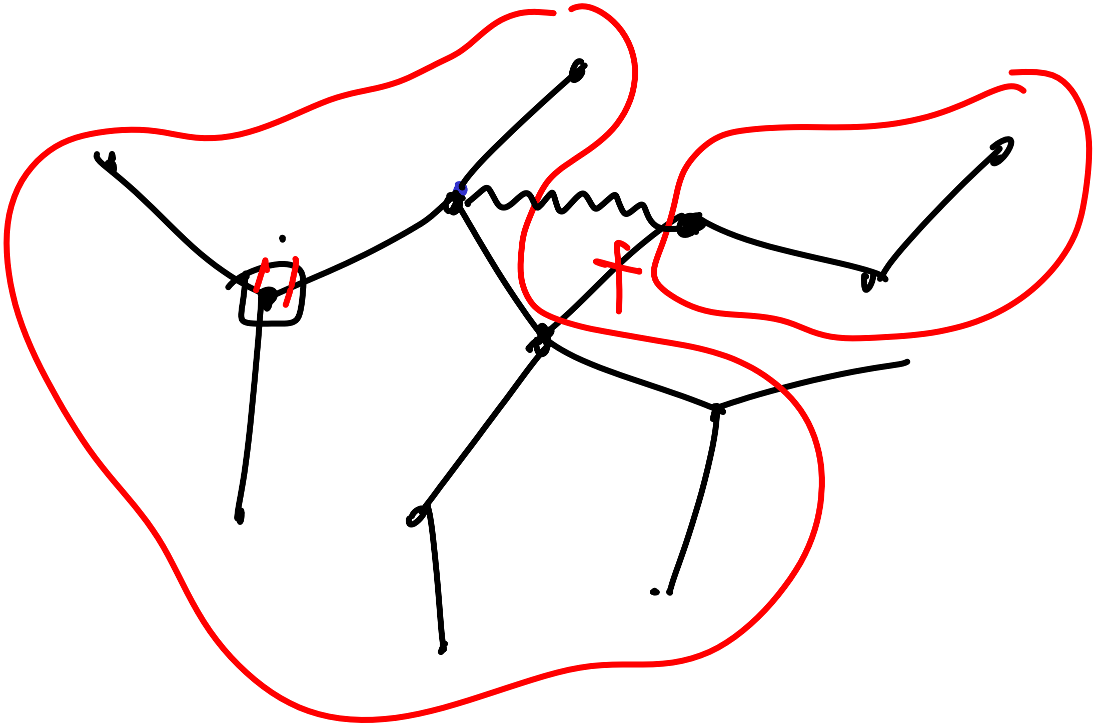
$u \underset{G}{\rightsquigarrow} w$ / $w \underset{G^R}{\rightsquigarrow} u$

why is $w \underset{G}{\rightsquigarrow} u$ or $u \underset{G^R}{\rightsquigarrow} w$

$\text{post}(u) \underset{G^R}{>} \text{post}(w)$ since u has highest postorder

either $u \underset{G^R}{\rightsquigarrow} w$

$w \underset{G^R}{\not\rightsquigarrow} u$



K- connected graph (undirected)

K-edge connected

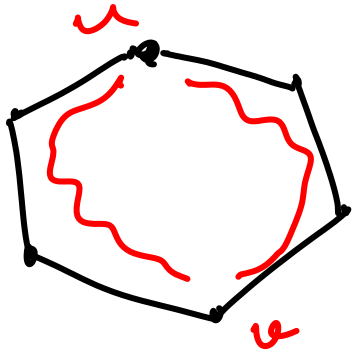
The n/w remains connected despite the removal of any K-1 edges in the graph

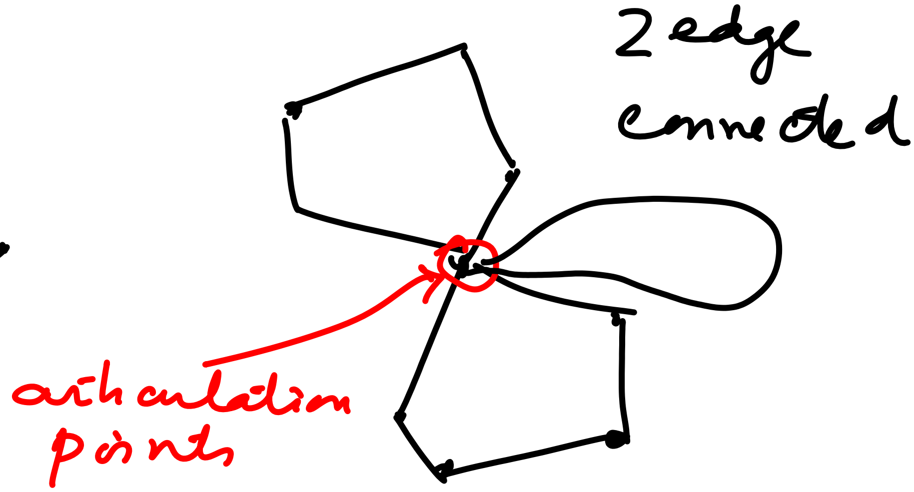
K- vertex connected

- - - - despite removal of any K-1 vertices in the graph

A tree is 1- connected

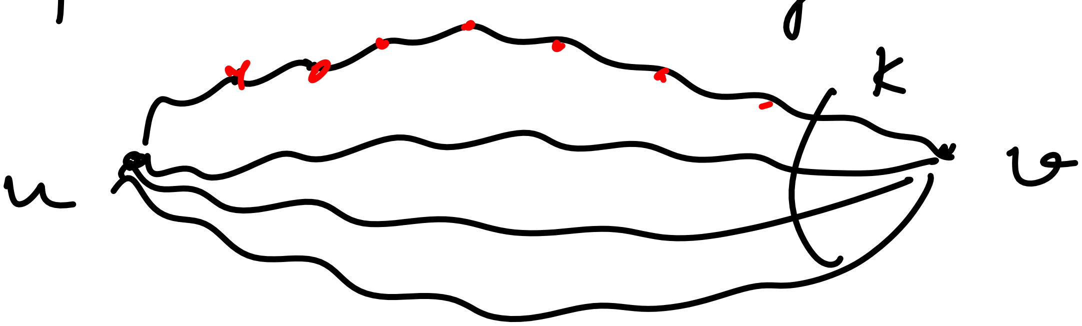
K = 2





2-vertex connected : biconnected

<u>Menger's theorem</u>: A K-vertex connected graph has at least K-vertex disjoint paths between any pair of vertices



Given a graph $G=(V,E)$ how do we determine if G is biconnected?

From the defn of 2-connectivity, we can run n instances of DFS

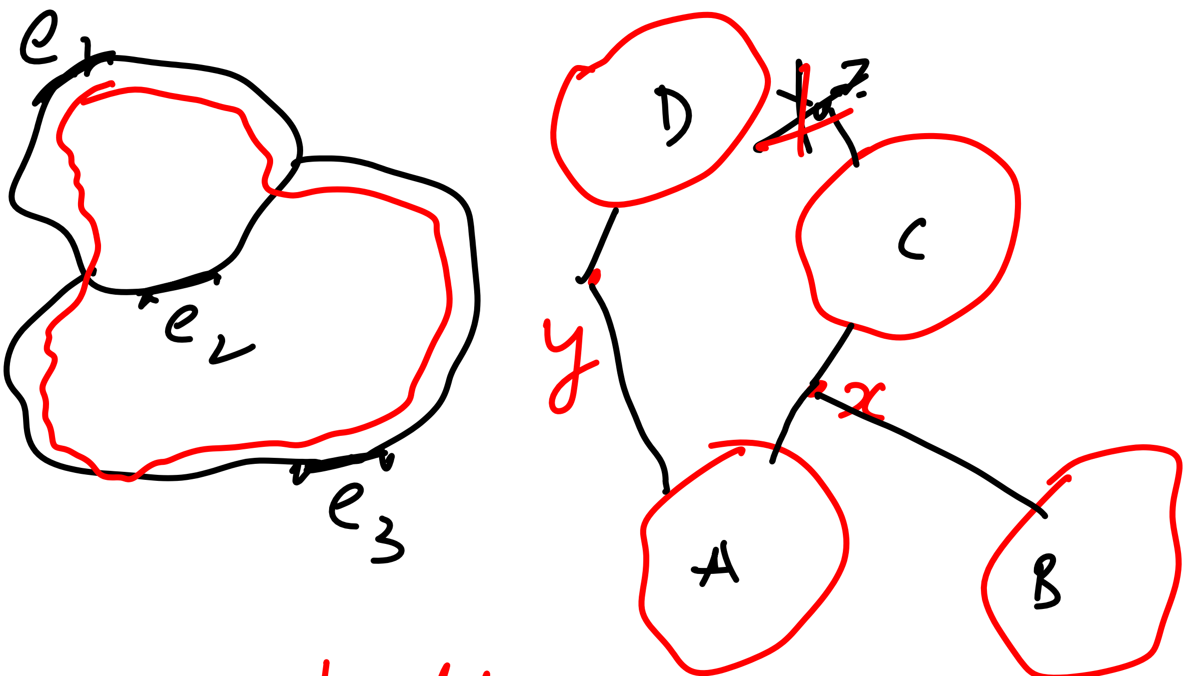
$\Rightarrow$ $O(n(n+m))$ running time

Question: Can we design a faster algorithm.

Consider the biconnected components of a graph:

e_1 is related e_2, $e_1, e_2 \in E$ iff there is a common cycle containing e_1 and e_2.

This relation is an equivalence relation and defines the maximal biconnected components.



x, y are articulation points

The component graph is a Tree

A DFS with some extra book keeping lets us identify the Biconnected components. Running time: $O(m+n)$