



CSE201 –
Evaluation only.

Created with Aspose.Slides for .NET Standard 2.0 23.1.
Copyright 2004-2023 Aspose Pty Ltd.

Data Structure and Algorithms

Articulation Points, Bridges &
Biconnected Components

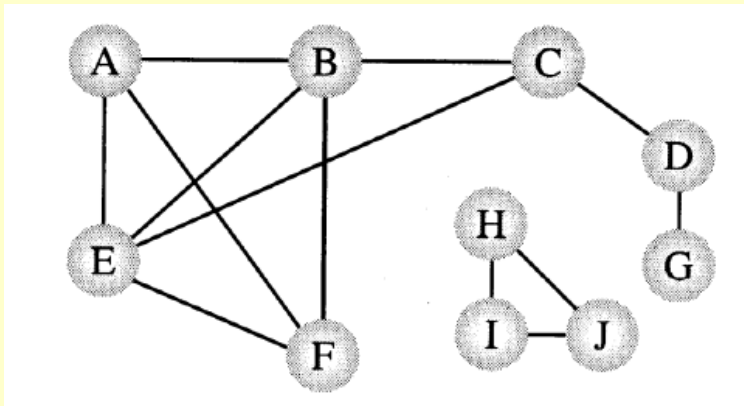
Connectivity/Biconnectivity for Undirected Graph

A node and **all the nodes reachable** from it compose a **connected component**. A graph is called **connected** if it has only one connected component.

Evaluation only.

Created with Aspose.Slides for .NET Standard 2.0 23.1.

Since the function **visit()** of DFS visits every node that is reachable and has not already been visited, the **DFS can easily be modified** to print out the connected components of a graph.

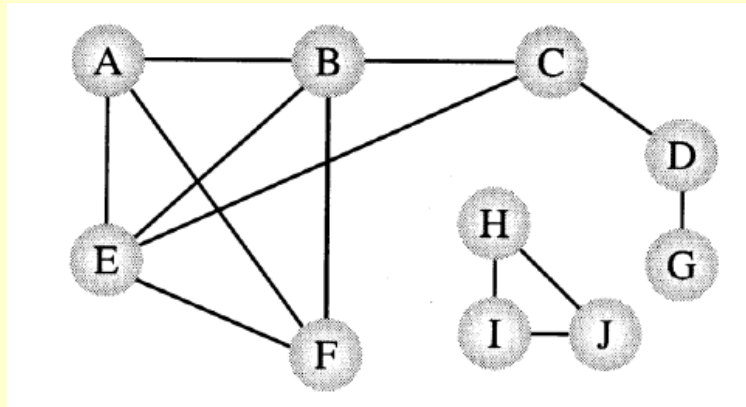


Two connected components

Connectivity/Biconnectivity

In actual uses of graphs, such as networks, we need to establish not only that every node is connected to every other node, but also there are **at least two independent paths between any two nodes**. A maximum set of nodes for which there are two different paths is called **biconnected**.

Evaluation only
Created with Aspose Slides for .NET Standard 2.0 23.1.
Copyright 2004-2023 Aspose Pty Ltd.



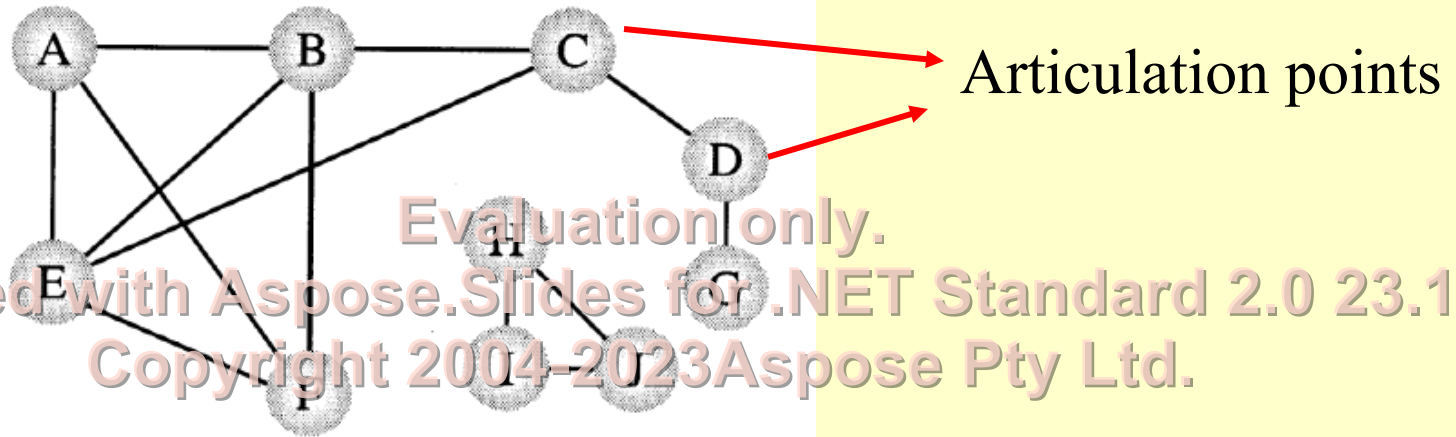
$\{H, I, J\}$ and $\{A, B, C, E, F\}$ are biconnected.

Connectivity/Biconnectivity

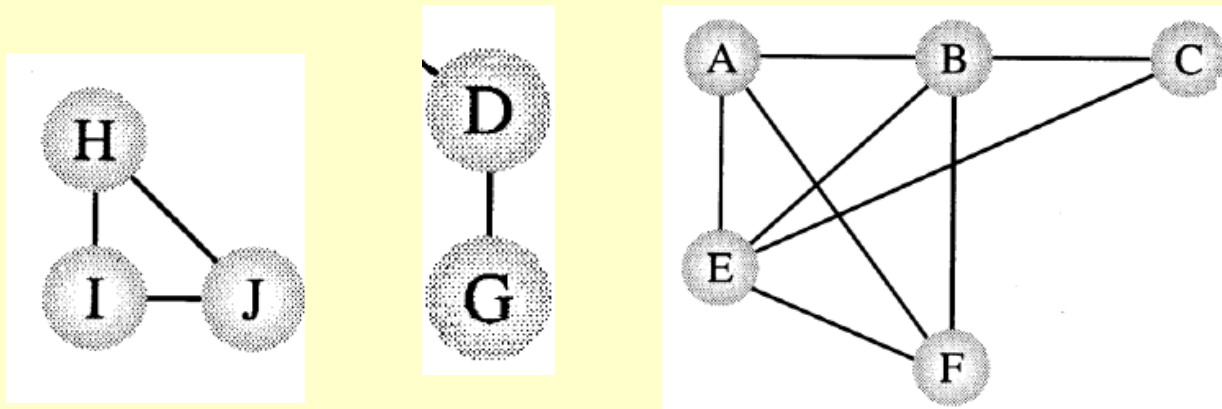
Another way to define this concept is that there are **no single points of failure**, no nodes that when deleted along with any adjoining arcs, would split the graph into two or more separate connected components. Such a node is called an **articulation point**.

If a graph contains no articulation points, then it is **biconnected**. If a graph does contain articulation points, then it is useful to **split the graph** into the pieces where each piece is a maximal biconnected subgraph called a **biconnected component**.

Connectivity/Biconnectivity



Three biconnected components



Finding Articulations

- Problem:
 - Given any graph $G = (V, E)$, find all the articulation points.

Evaluation only.

Created with Aspose.Slides for .NET Standard 2.0 23.1.

- Possible strategy:

- For all vertices v in V :

Remove v and its incident edges

Test connectivity using a DFS.

- Execution time: $\Theta(n(n+m))$.

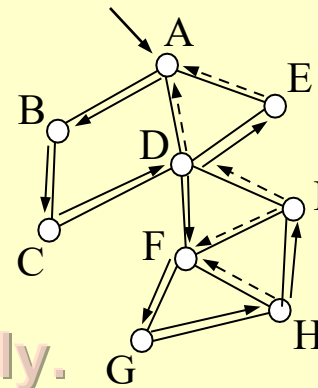
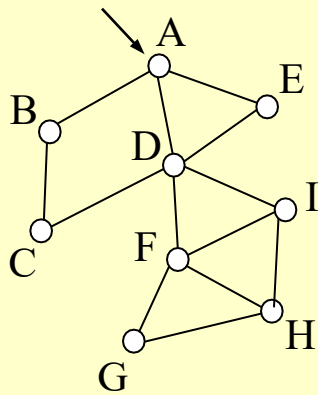
- Can we do better?

Finding Articulation Points

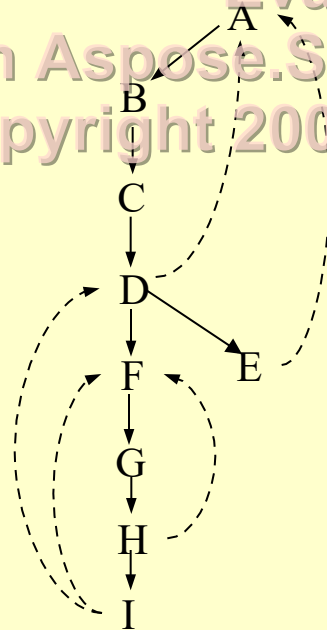
- A DFS tree can be used to discover articulation points in $\Theta(n + m)$ time.

Evaluation only.

Created with Aspose.Slides for .NET Standard 2.0 23.1.
Copyright 2004-2023 Aspose Pty Ltd.

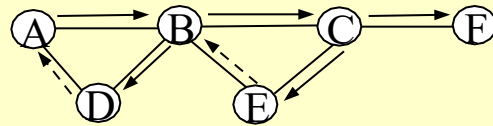


Evaluation only.
 Created with Aspose.Slides for .NET Standard 2.0 23.1.
 Copyright 2004-2023 Aspose Pty Ltd.



Can you characterize D ?

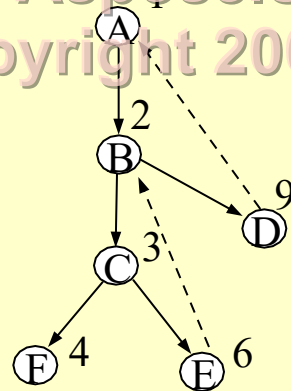
Depth First Search number



$G = (V, E)$

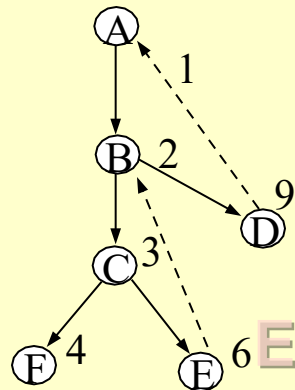
Evaluation only.

Created with Aspose.Slides for .NET Standard 2.0 23.1.
Copyright 2004-2023 Aspose Pty Ltd.



A	B	C	D	E	F
1	2	3	9	6	4

Any relation between Discovery time and articulation point ?



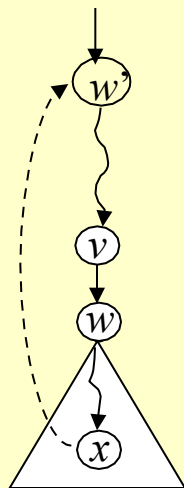
Assume that $(a,b) \Leftrightarrow a \rightarrow b$

Tree edge : (a,b) $a < b$

Back edge : (a,b) $a > b$

Evaluation only.

Created with Aspose.Slides for .NET Standard 2.0 23.1.
Copyright 2004-2023Aspose Pty Ltd.



If there is a back edge from x
to a proper ancestor of v ,
then v is reachable from x .

Finding Articulation Points

- A DFS tree can be used to discover articulation points in $\Theta(n + m)$ time.
 - We start with a program that computes a DFS tree labeling the vertices with their discovery times.
 - We also compute a function called $\text{low}(v)$ that can be used to characterize each vertex as an articulation or non-articulation point.
 - The root of the DFS tree will be treated as a special case:
 - The root has a $d[]$ value of 1.

Finding Articulation Points

- The root of the DFS tree is an articulation point if and only if it has two or more children.
 - Suppose the root has two or more children.
 - Recall that back edges never link vertices between two different subtrees.
 - So, the subtrees are only linked through the root vertex and its removal will cause two or more connected components (i.e. the root is an articulation point).
 - Suppose the root is an articulation point.
 - This means that its removal would produce two or more connected components each previously connected to this root vertex.
 - So, the root has two or more children.

Definition of $low(v)$

- Definition. The value of $low(v)$ is the discovery time of the vertex closest to the root and reachable from v by following zero or more tree edges downward, and then at most one back edge.
- We can efficiently compute Low by performing a postorder traversal of the depth-first spanning tree.

$$low[v] = \min\{$$

$d[v],$

lowest $d[w]$ among all back edges (v,w)

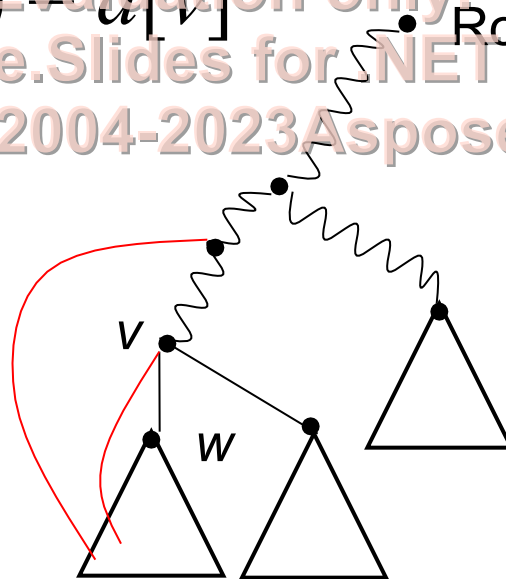
lowest $low[w]$ among all tree edges (v,w)

$\}$

- In English: $low(v) < d[v]$ indicates if there is another way to reach v which is not via its parent

Low(v)

- Observe that if there is a back edge from somewhere below v to above v in the tree, then $\text{low}(v) < d[v]$
- Otherwise $\text{low}(v) = d[v]$



Finding Articulation Points

- Let v be a non-root vertex of the DFS tree T .
- Then v is an articulation point of G if and only if there is a child w of v with $low(w) \geq d[v]$.

Evaluation only.

Created with Aspose.Slides for .NET Standard 2.0 23.1.
Copyright 2004-2023 Aspose Pty Ltd.

Articulation Points: Pseudocode

Data: color[V], time, prev[V], d[V], f[V], low[V]

DFS(G) // where prog starts

{

for each vertex $u \in V$

{

color[u] = WHITE;

prev[u] = NIL;

low[u] = inf;

f[u] = inf; d[u] = inf;

}

time = 0;

for each vertex $u \in V$

if (color[u] == WHITE)

DFS_Visit(u);

}

Evaluation only.

Created with Aspose.Slides for .NET Standard 2.0 23.1.

Copyright 2004-2023 Aspose Pty Ltd.

Articulation Points: Pseudocode

```
DFS_Visit(v)
{
    color[v]=GREY; time=time+1; d[v] = time;
    low[v]= d[v];
    for each w ∈ Adj[v]{
        if (color[w] == WHITE) {
            prev[w]=u;
            DFS_Visit(w);
            if low[w] >= d[v]
                record that vertex v is an articulation
            if (low[w] < low[v]) low[v] := low[w];
        }
        else if w is not the parent of v then
            //--- (v,w) is a BACK edge
            if (d[w] < low[v]) low[v] := d[w];
    }
    color[v] = BLACK;   time = time+1;   f[v] = time;
}
```

Created with Aspose.Slides for .NET Standard 2.0 23.1.
Copyright 2004-2023 Aspose Pty Ltd.

Special Case

- When “v” is a root of the DFS tree, you have to check it manually.

Evaluation only.

Created with Aspose.Slides for .NET Standard 2.0 23.1.
Copyright 2004-2023 Aspose Pty Ltd.

Source

- Mark Allen Weiss – Data Structure and Algorithm Analysis in C
 - Articulation Point
- Exercise: Evaluation only.
 - Cormen – Exercise 22-2
 - What is bridge? How can it be detected?

Created with Aspose.Slides for .NET Standard 2.0 23.1.
Copyright 2004-2023 Aspose Pty Ltd.