Graph Connectivity

Instructor: Meng-Fen Chiang

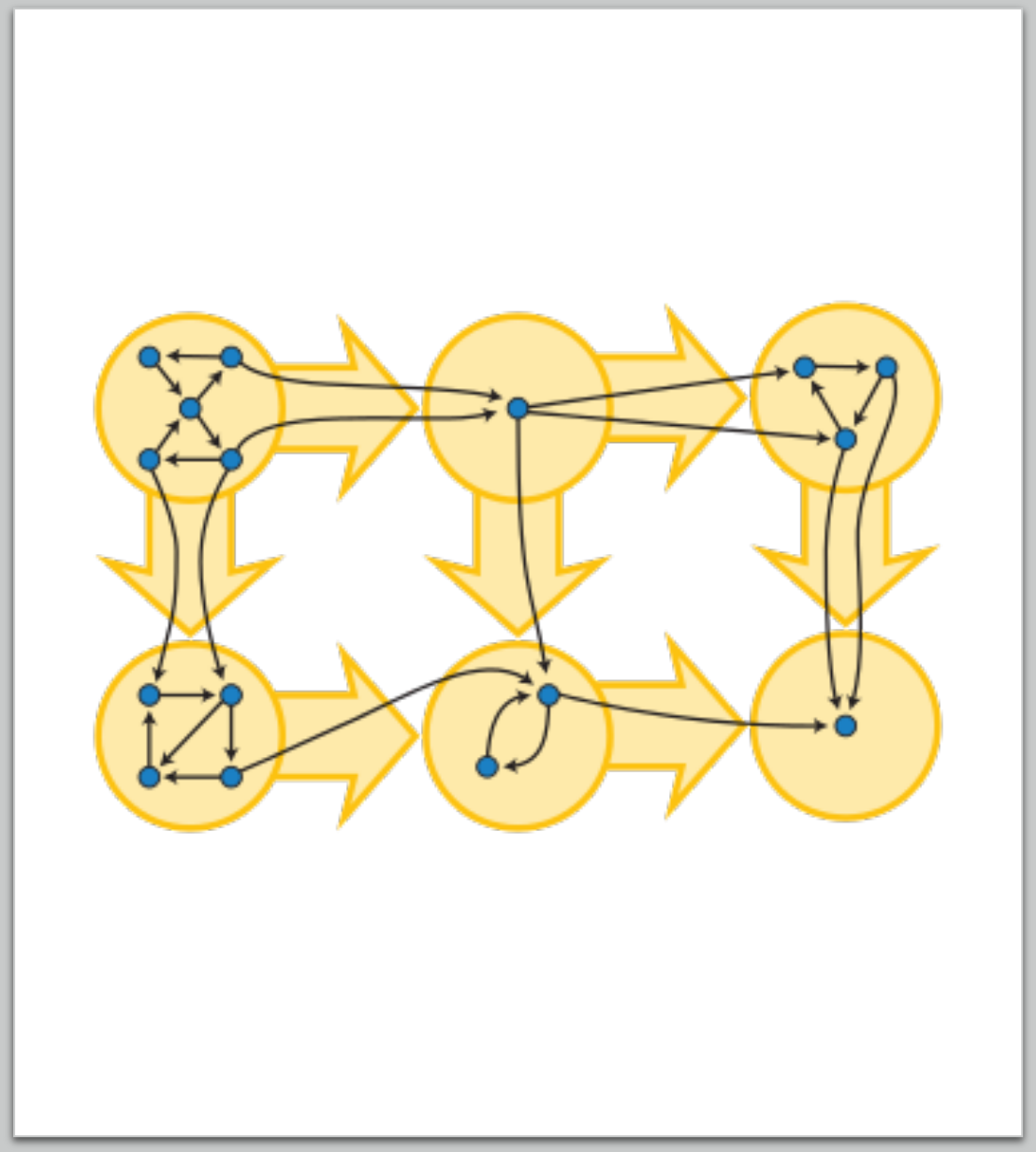
COMPCSI220: WEEK 10

Slides adapted from Mark Wilson, Georgy Gimel'farb, Simone Linz, Tanya Gvozdeva, and Kaiqi Zhao

THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

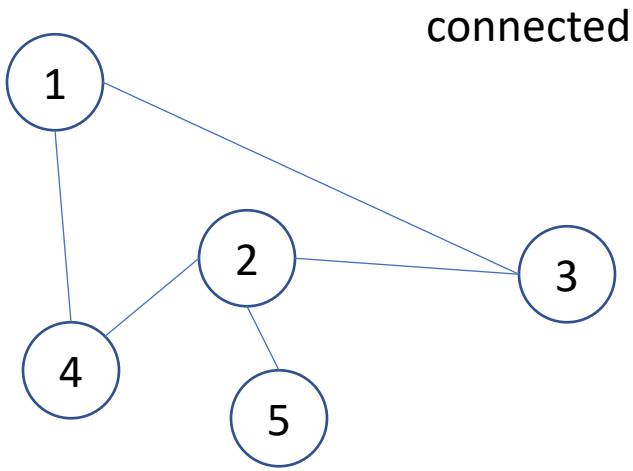
OUTLINE

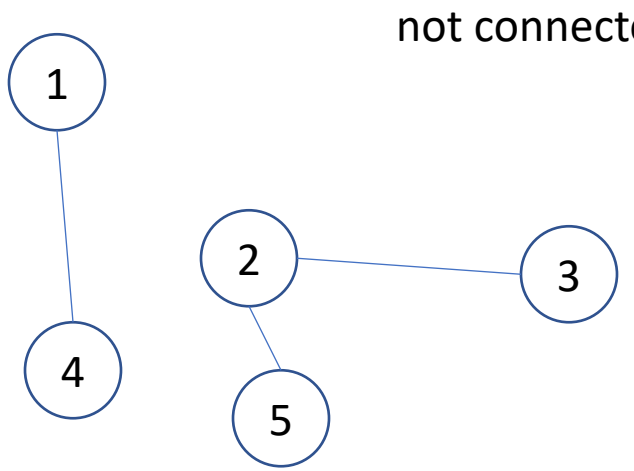
- Graph and Digraph Connectivity
- Terminology
  - Connected Components (CCs) in a Graph
  - Strongly Connected Components (SCCs) in a Digraph
- Finding the SCCs in a Digraph



Graph Connectivity

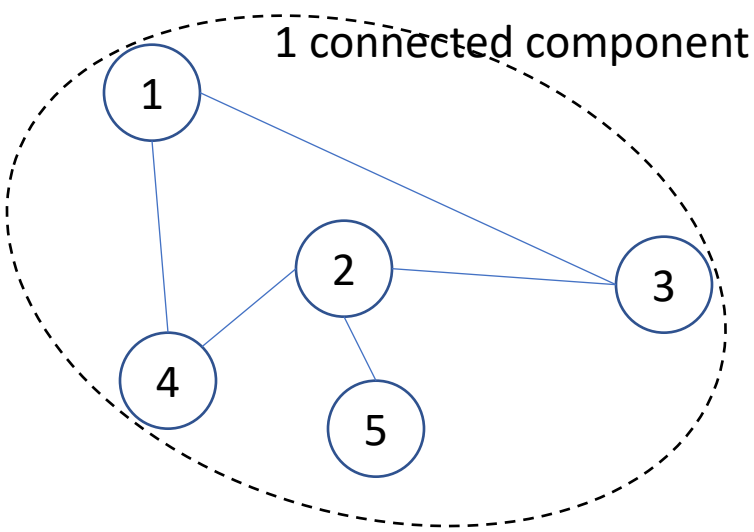
- **Definition**. A graph G is connected if for each vertex $u \in V(G)$, there is a path to any other vertex $v \in V(G)$
- **Definition**: A graph G is disconnected if it is not connected and the maximal induced connected subgraphs are called the components of G.

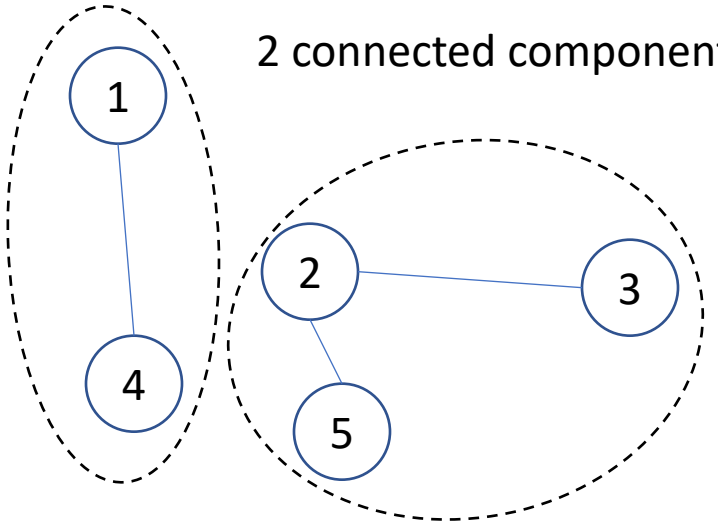




Connected Components

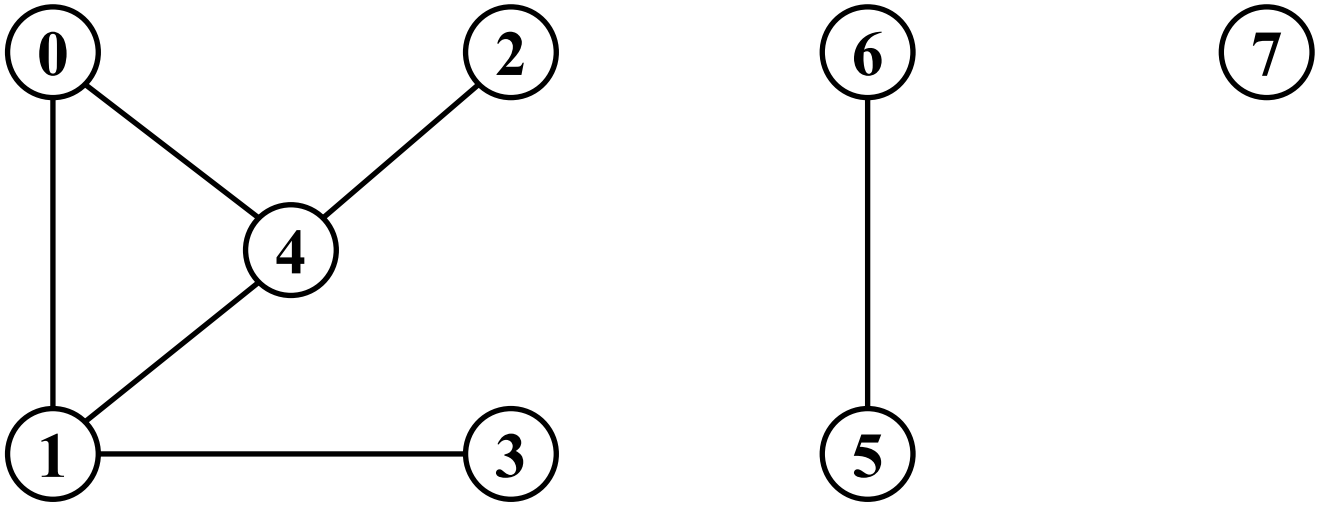
- Any graph G consists of one or more subgraphs G_i such that
  - Each G_i is connected, and
  - If there is a path between $u, v \in V(G)$, then u and v are in the same subgraph G_i
  - The subgraphs G_i are called the connected components of G





Finding Connected Components

- **Theorem**. Let G be an undirected graph and suppose that **DFS** or **BFS** is run on G. Then the connected components of G are precisely the subgraphs spanned by the trees in the search forest.



3 connected components

Digraph Connectivity

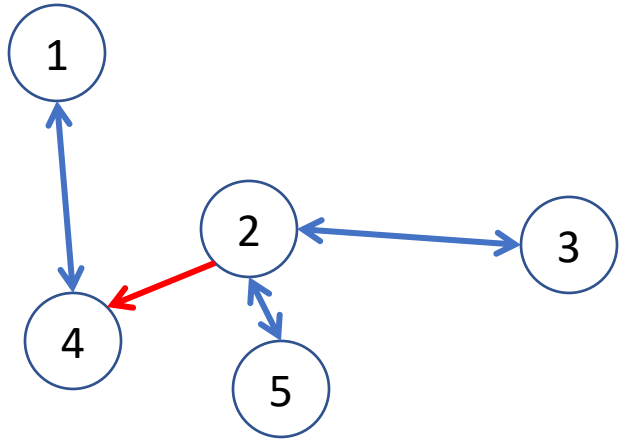
- **Definition**. A digraph G is strongly connected if for each pair of vertices $u, v \in V(G)$ are mutually reachable - there is a path from u to v, and a path from v to u. A digraph G is weakly connected if its underlying graph is connected.

- **Definition**. The maximal sub-digraphs induced by mutually reachable nodes of a digraph are called the strong components of G.

- A strongly connected digraph of order at least two contains at least one cycle!

Example: Digraphs Connectivity

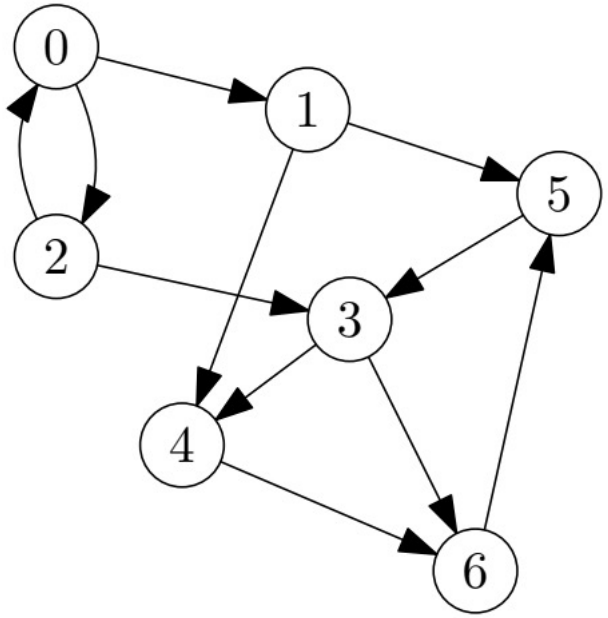
- Let's consider digraphs. Is this a connected digraph?



There is no path from 1 and 4 to 2, 3, and 5.
Answer: **weakly connected**

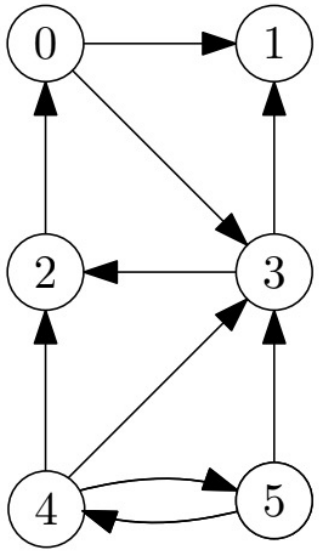
Example 26.8

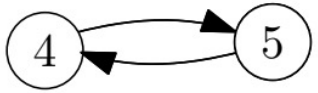
- This digraph appears to be “all in one piece” as its underlying graph is connected. But can you get from any node to any other node following the direction of the arcs?

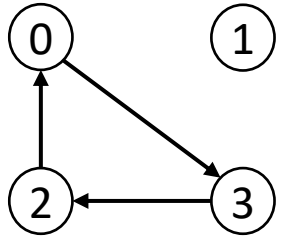


Example 26.12

- This digraph has three strong components. Find and draw the two missing ones.

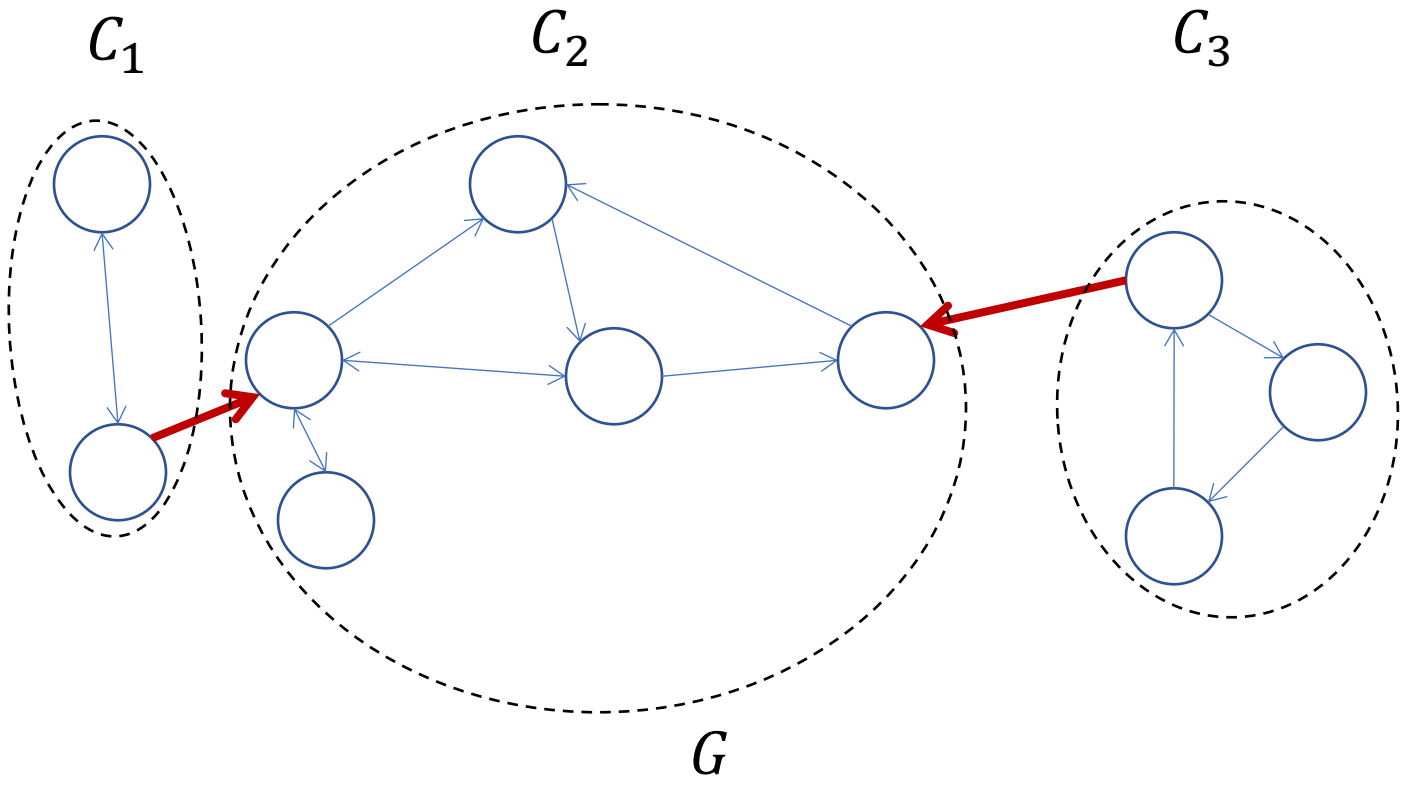






- Note that there are arcs of the digraph not included in the strongly connected components.

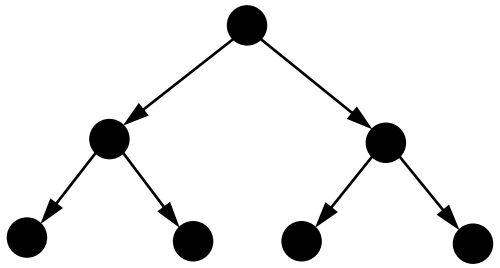
Example: Strongly Connected Components of a Digraph



$$V(G) = V(C_1) \cup V(C_2) \cup V(C_3)$$

Facts about Digraph Connectivity

- The method for finding components of a graph using **DFS** and **BFS** fail at finding strong components of digraphs.

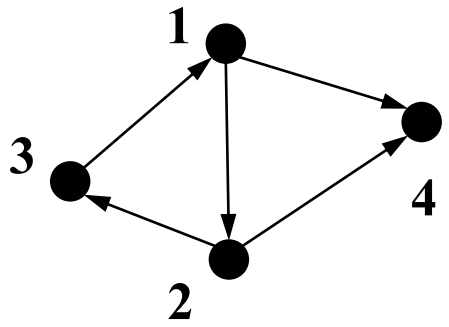


- Not strongly connected
- Yet BFS would return a single root (a single tree in the search forest).

Facts about Digraph Connectivity

- But we can use DFS/BFS to determine whether a digraph is strongly connected!

- Run **DFS**/**BFS** for each vertex and check whether or not each resulting search forest contains a single tree that spans the entire graph.

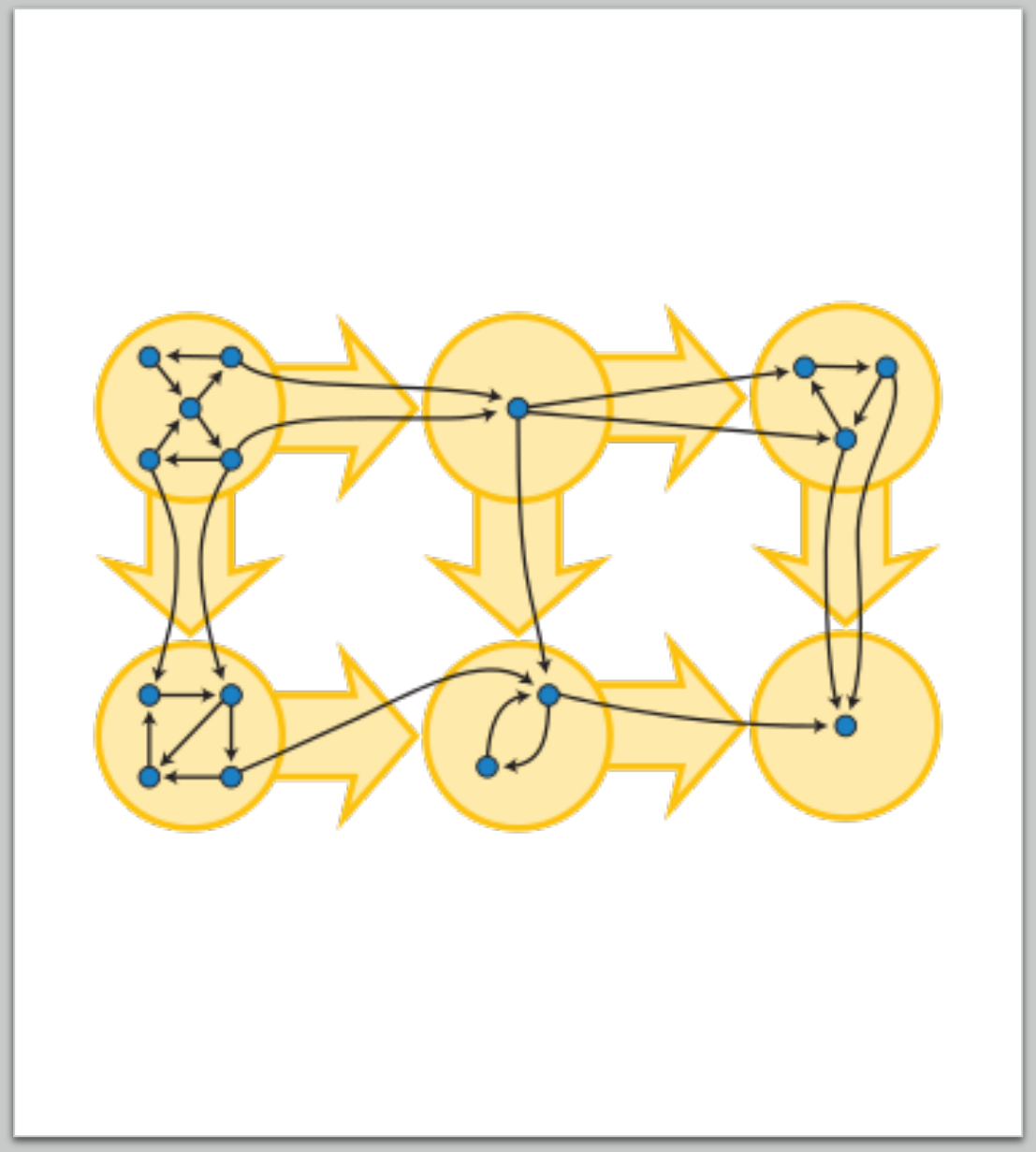


- DFS starts at 1: OK!
- DFS starts at 4: not OK!

THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

OUTLINE

- Graph and Digraph Connectivity
- Terminology
  - Connected Components (CCs) in a Graph
  - Strongly Connected Components (SCCs) in a Digraph
- Finding the SCCs in a Digraph



Graph and Digraph Connectivity

	(Undirected) Graph	Di-graph
Connectivity	An undirected graph G is connected if for each pair of vertices $u, v \in V(G)$, there is a path between them.	Strongly connected - for each pair of vertices are mutually reachable Weakly connected - if its underlying graph is connected.
Components	Components - the maximal induced connected subgraphs.	Strong components - the maximal sub-digraphs induced by mutually reachable nodes
Finding components	BFS / DFS	???

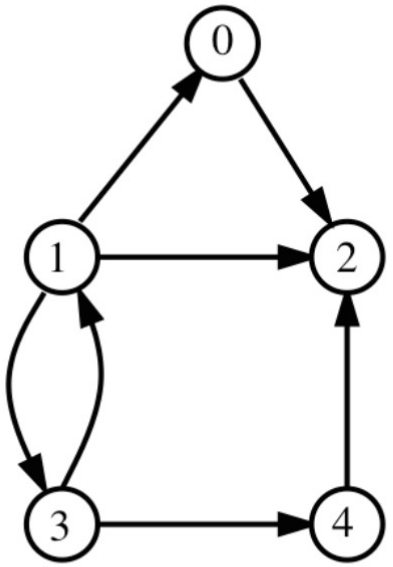
Observations: Strongly Connected Components

- Observation 1
  - Consider a digraph G and its reverse digraph G_r.
  - Then the strongly connected components are the same subsets of vertices in both digraphs.

- Observation 2
  - Now consider "shrinking" the connected components down to a single vertex in both digraphs. This gives us H from G and H_r from G_r.

Finding Strong Components in Digraphs

- We need to make sure each tree in the search forest corresponds to one strong component. The order of choosing starting nodes of DFSVisit matters!

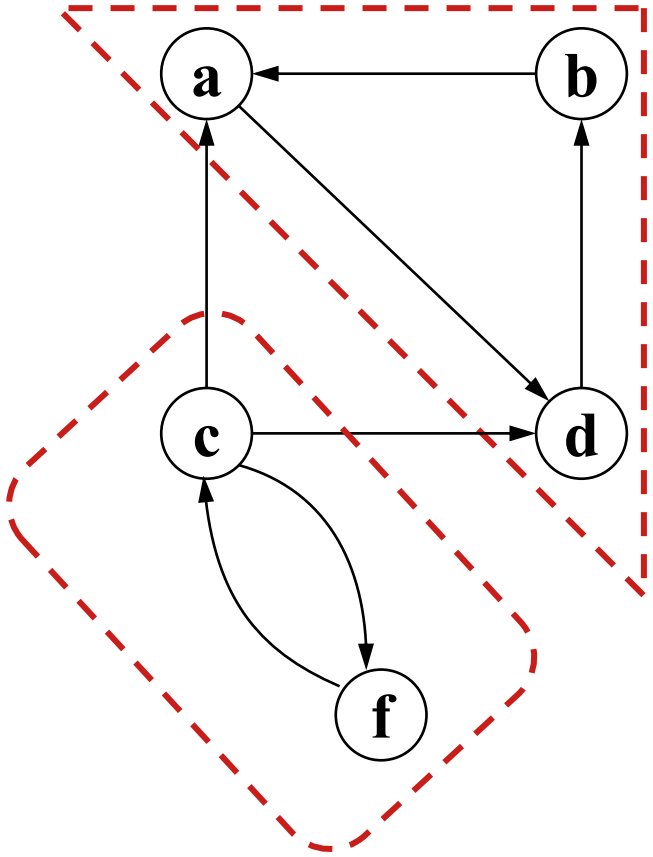


G

If run DFSVisit on node 1 first, then we get a single tree: $\{(1, 0), (0, 2), (1, 3), (3, 4)\}$, which does NOT correspond to a strong component

If run DFSVisit on nodes in the order of 2, 0, 4, 1, 3
then we get the strong components: $\{2\}, \{0\}, \{4\}, \{1, 3\}$

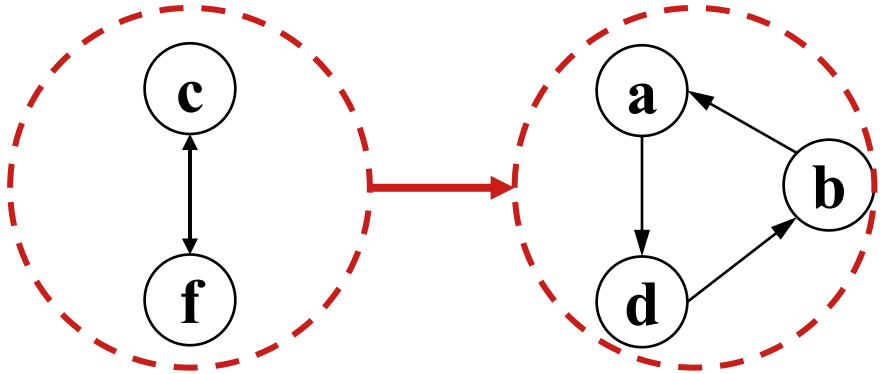
Motivation: Component Graph



G

Idea - component graph:

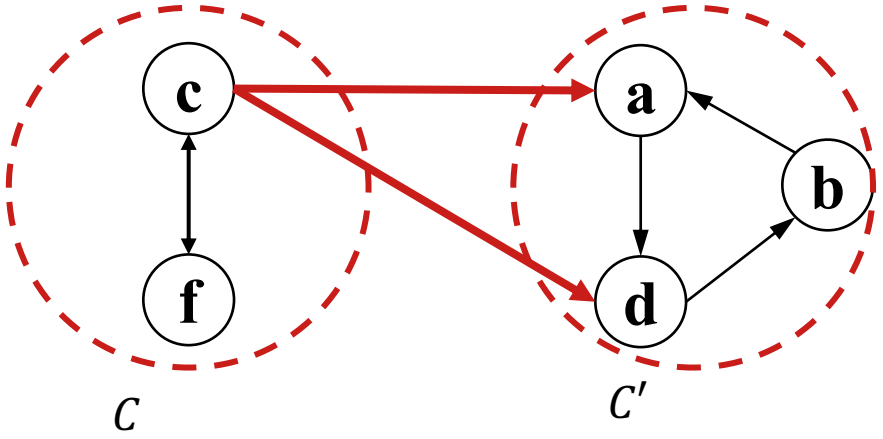
- Let each strong component be a vertex
- Maintain an arc from one strong component C to the other C' if there is an arc (u, v), where $u \in C, v \in C'$



Facts: Component Graph

Let C and C' be two distinct strong components

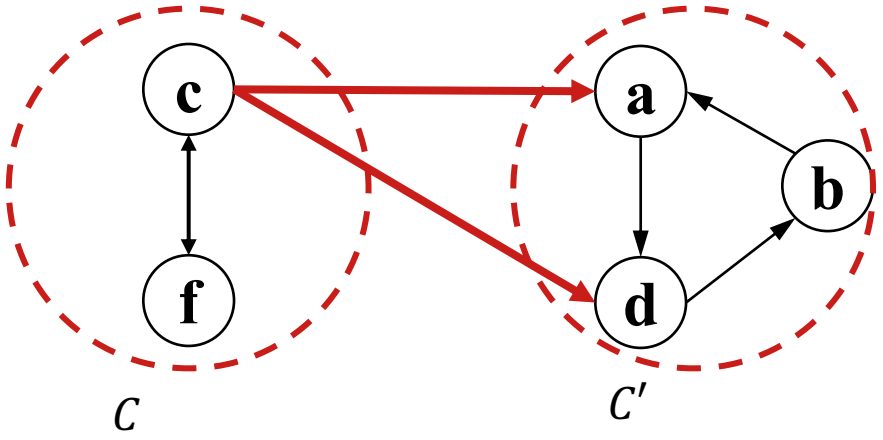
- If there is an arc from C to C', then there shouldn't be an arc from C' to C
- A component graph is a DAG



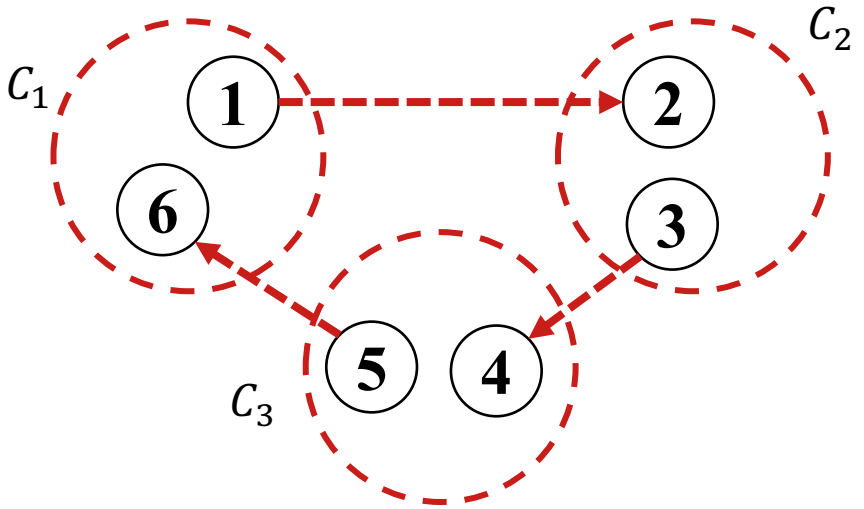
Facts: Component Graph

Let C and C' be two distinct strong components

- If there is an arc from C to C', then there shouldn't be an arc from C' to C
- A component graph is a DAG



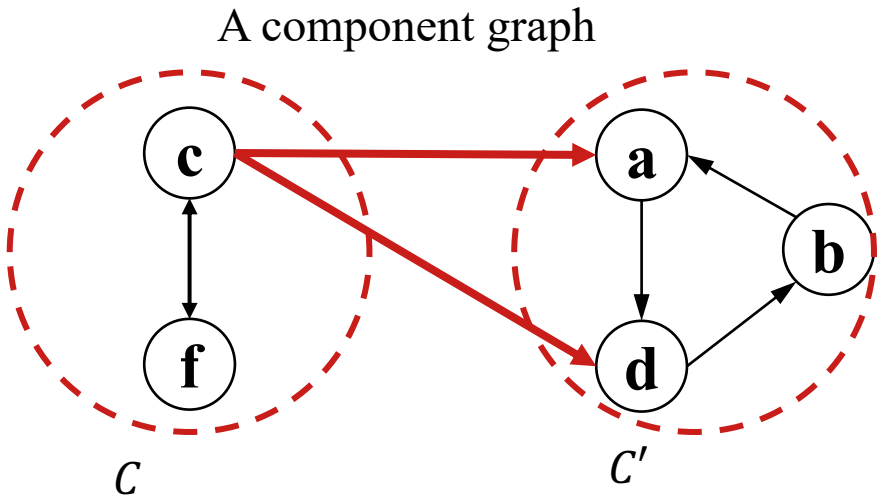
A component graph



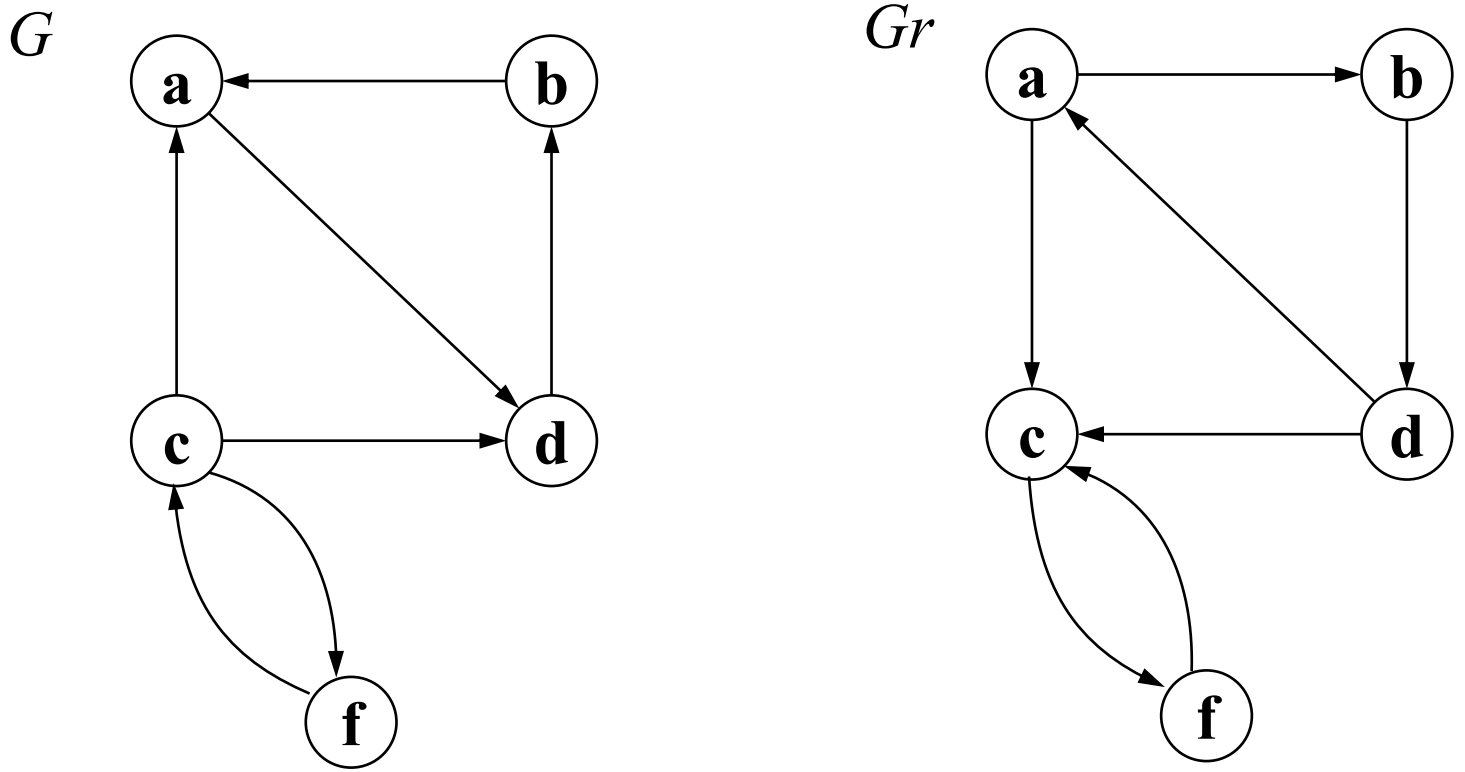
This is not a component graph

Motivation: Component Graph

- If we run **DFS** on a node from C' above first, then we should get a search forest with two trees corresponding to the two strong components!
- But we don't know the component graph beforehand! So, we need an alternative way.

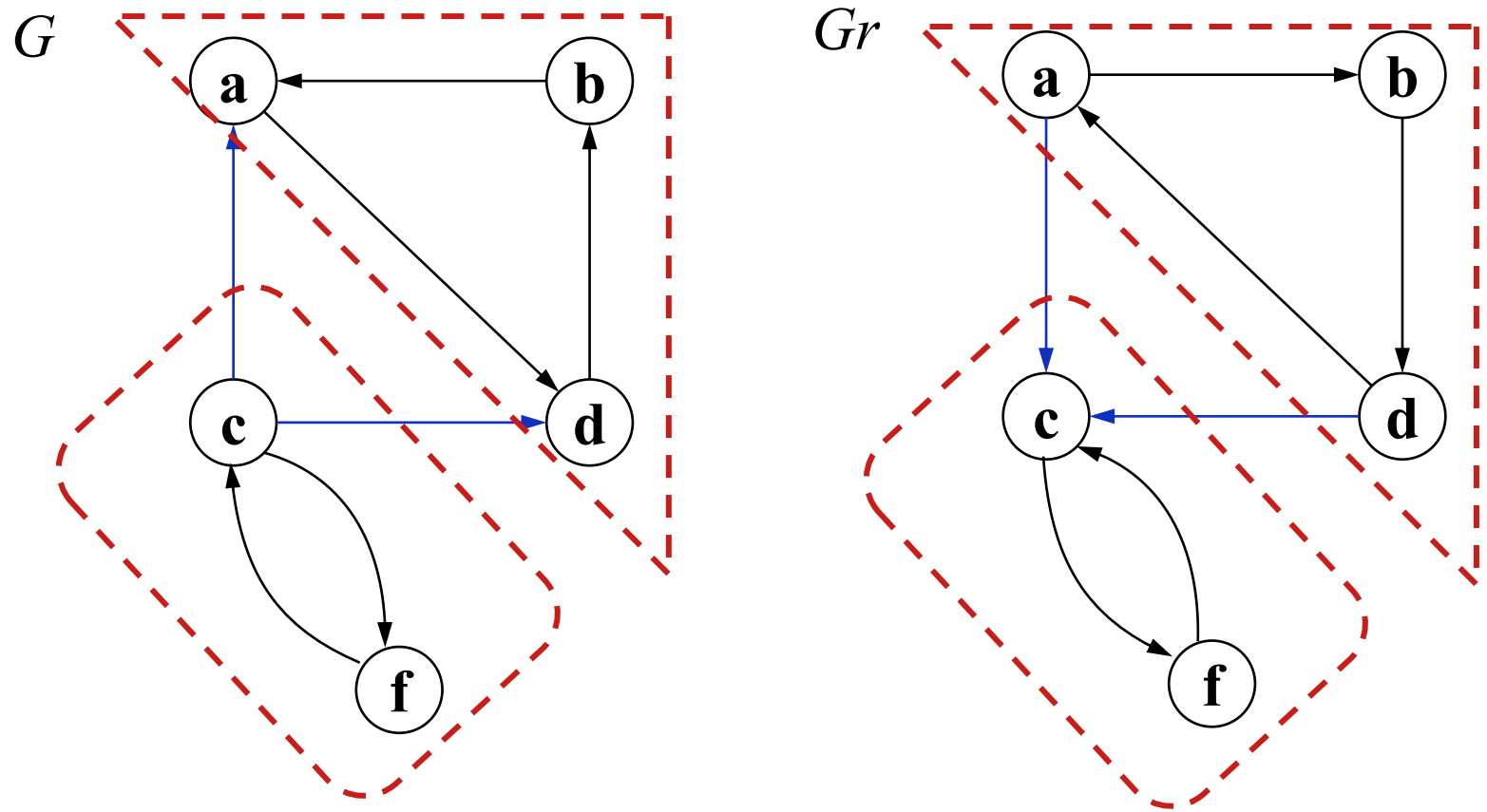


Finding Strong Components in a Reverse Digraph



The strongly connected components in G are the same as those in Gr .

Finding Strong Components in a Reverse Digraph

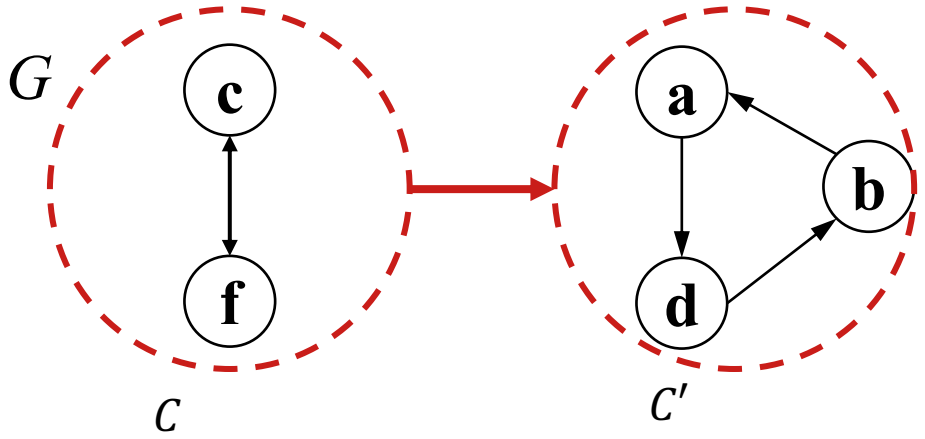


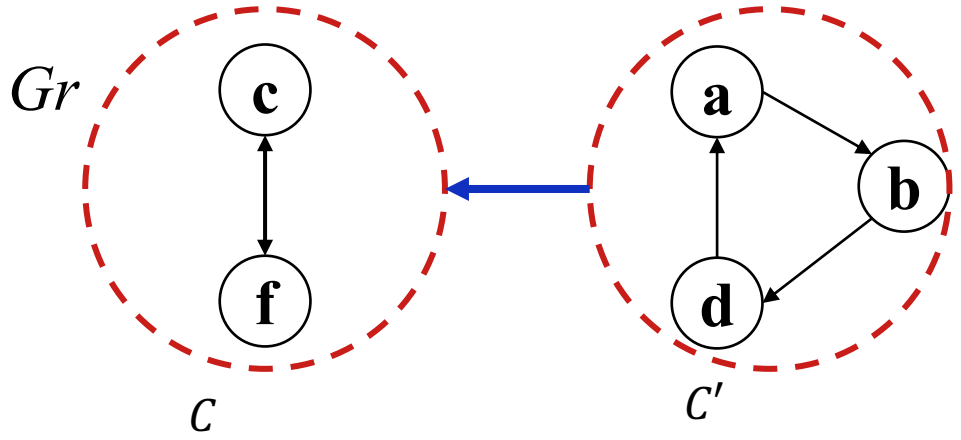
The component graph of the reverse graph is also a reverse of the component graph of the original graph.

Finding Strong Components in a Reverse Digraph

- **Observation**. if we run the **DFS** on the **reverse digraph** Gr, there are two cases in the resulting search forests:

1. Start DFSVisit on some node x in C': The starting node x in C' will be the last one finished (with greatest $done[x]$) among all nodes in both C and C'.
2. Start DFSVisit on some node y in C: All nodes in C will be finished before the second run of DFSVisit on some node x in C'. Thus, x still has the greatest $done[x]$





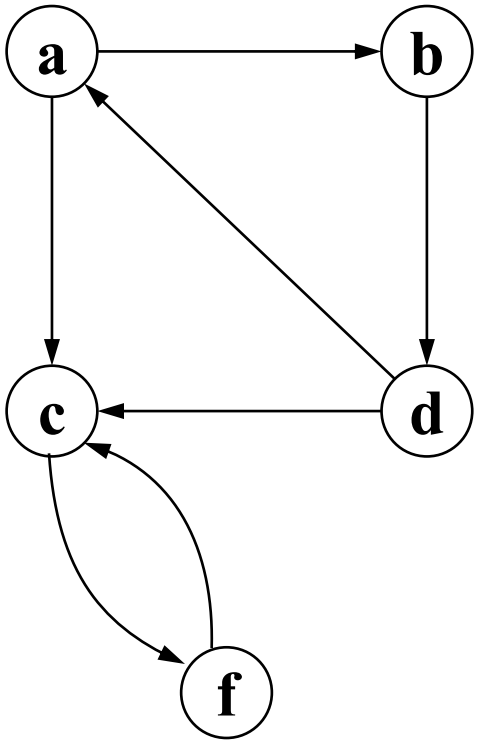
Finding Strong Components

- Let G be a digraph Gr be its reverse digraph obtained by reversing all arcs in G. The idea is to run DFSVisit on a node of G which finished last in the DFS of Gr.

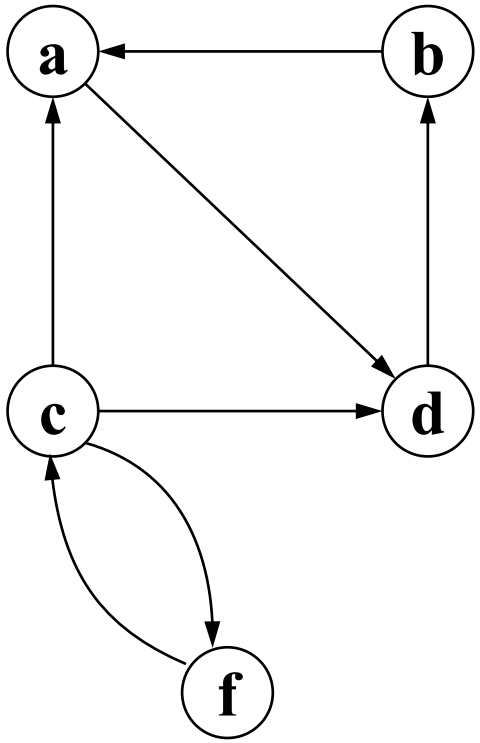
- **Phase 1**. Run DFS on Gr, to get depth-first forest Fr. For each node x in Gr, let $done[x]$ be the time at which x turned from grey to black.

- **Phase 2**. Run DFS on G; when choosing a new root, choose a white node x such that $done[x]$ is as large as possible. This gives a forest F.

Strong connected components example (1)

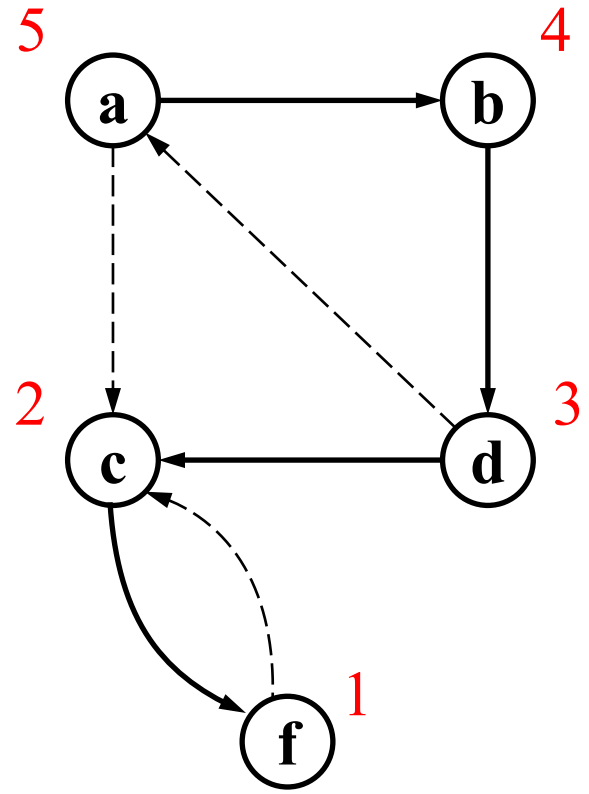


Gr

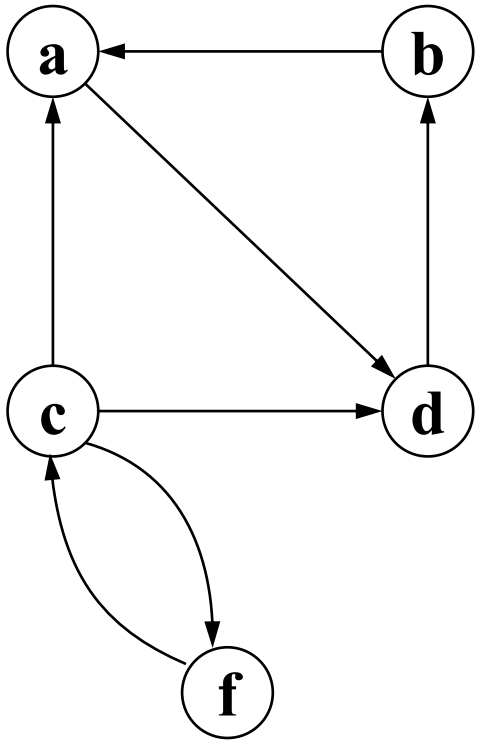


G

Strong connected components example (1)

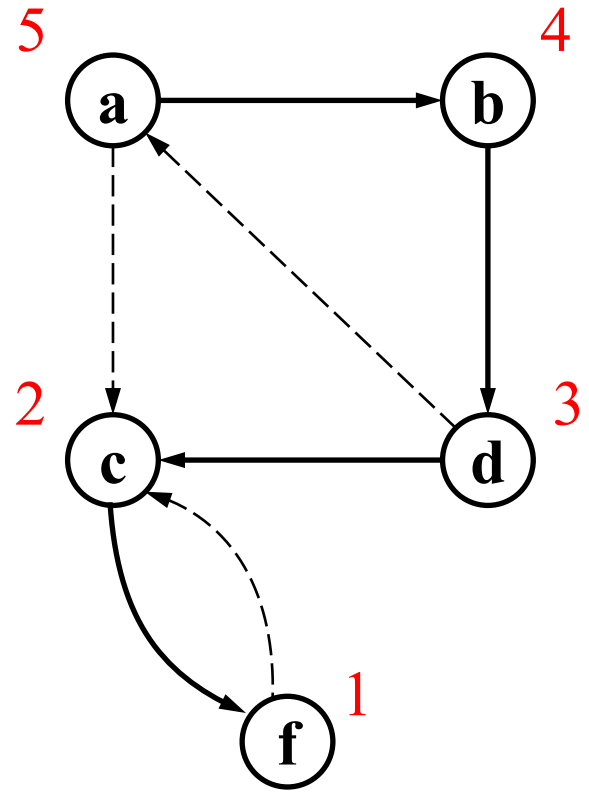


Gr

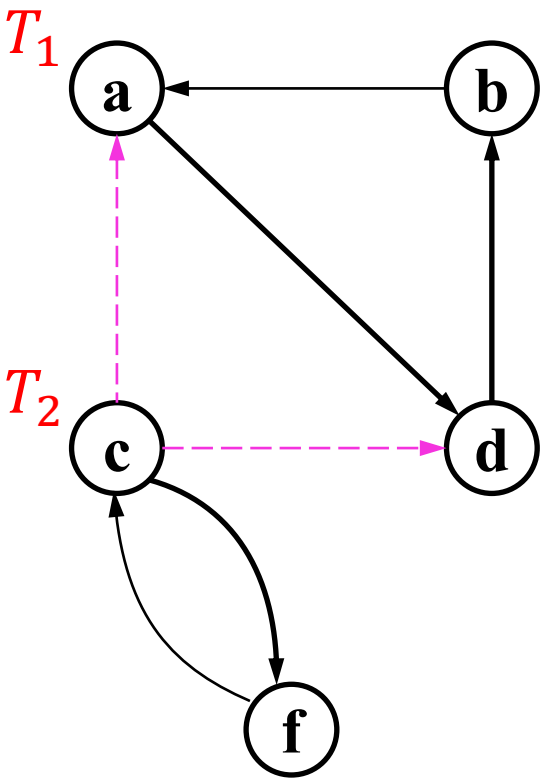


G

Strong connected components example (1)

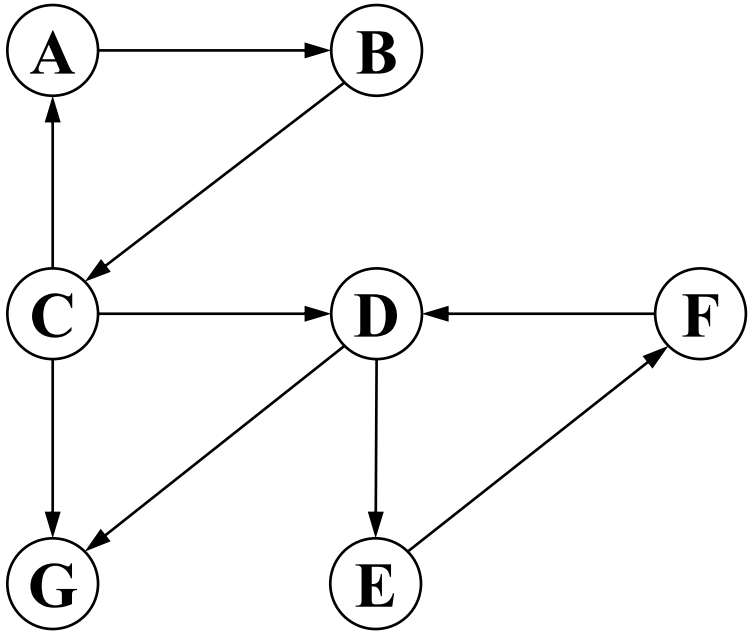


Gr

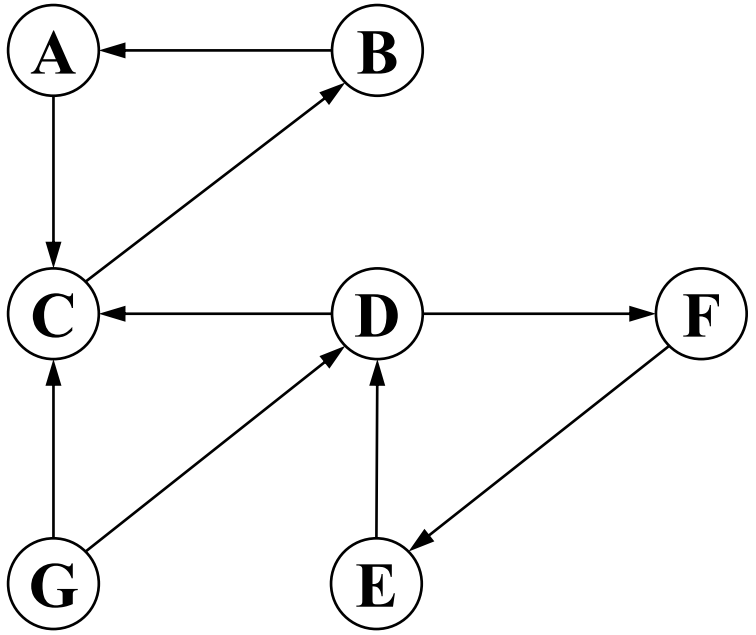


G

Strong connected components example (2)



Gr

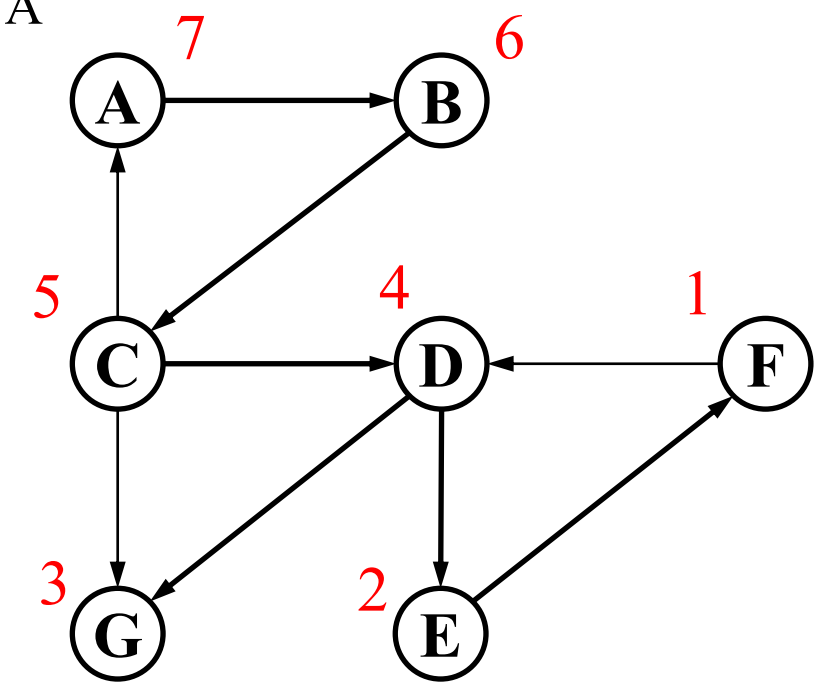


G

Strong connected components example (2)

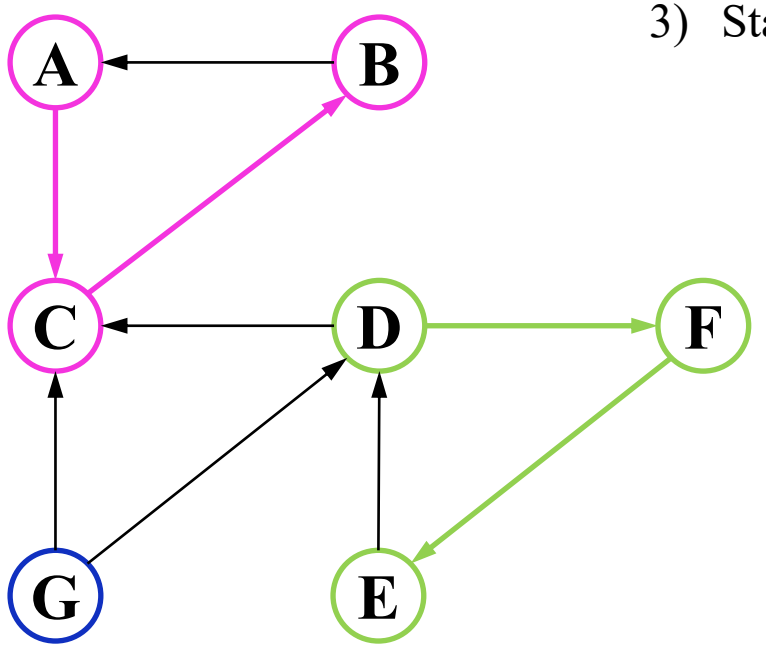
DFS
Start: A

done time



Gr

DFS:
1) Start: A (7)
2) Start: D (4)
3) Start: G (3)



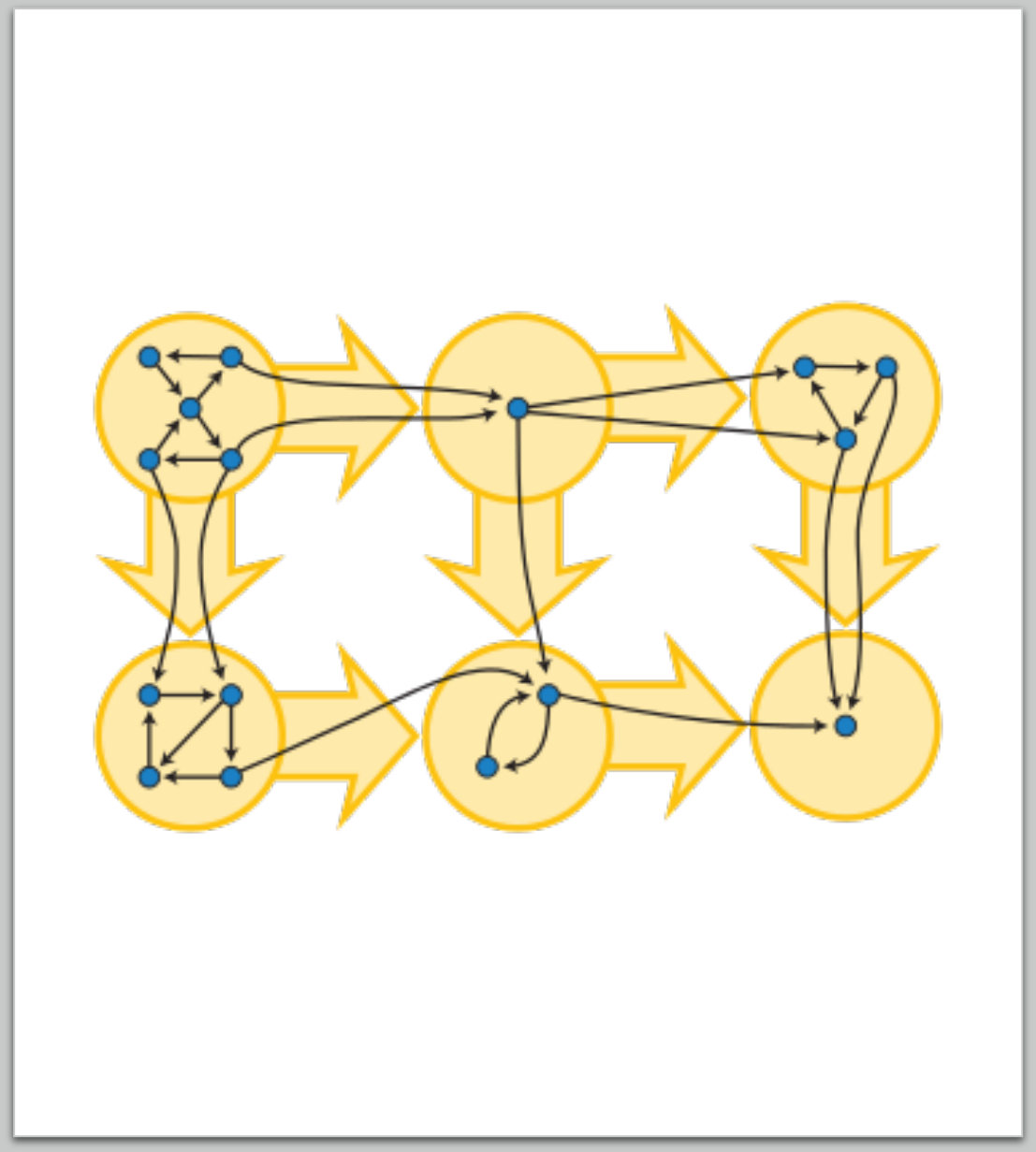
G

G has three strongly connected components.

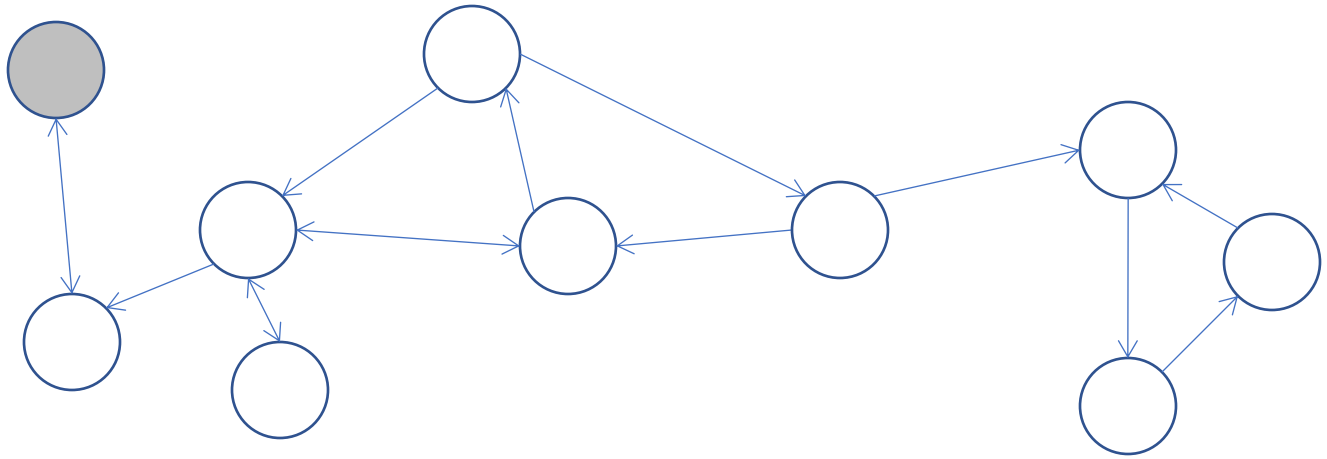
THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

OUTLINE

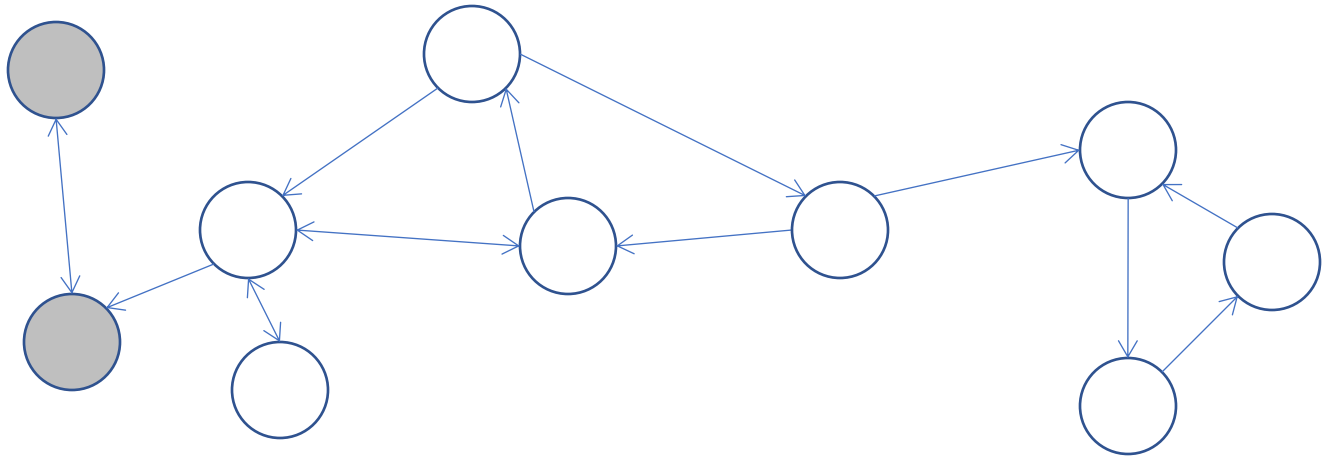
- Terminology
  - Connected Components (CCs) in a Graph
  - Strongly Connected Components (SCCs) in a Digraph
- Graph and Digraph Connectivity
- Finding the SCCs in a Digraph
- Illustrating Examples



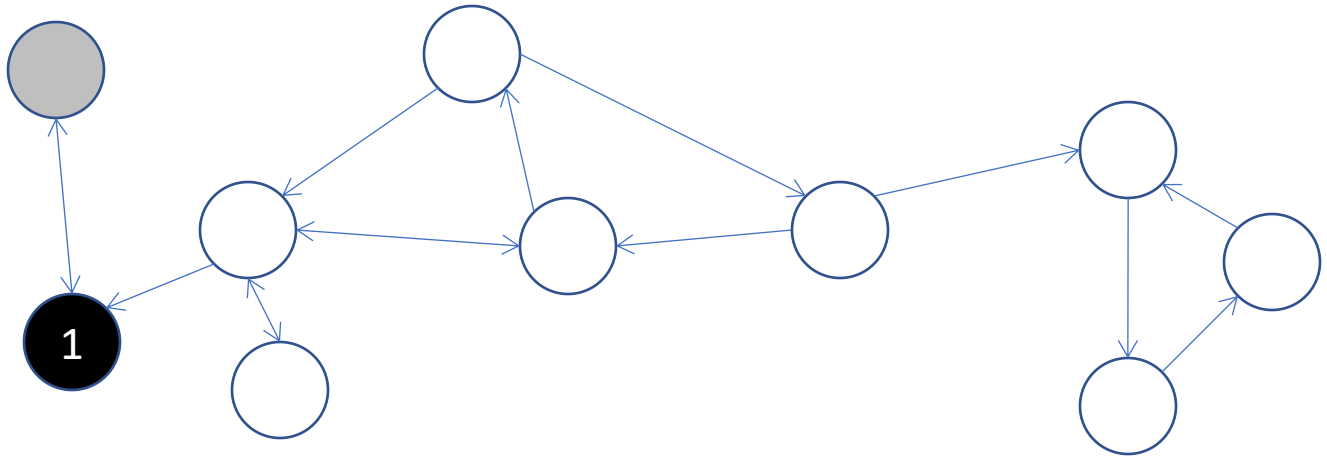
Phase 1: DFS on G_r



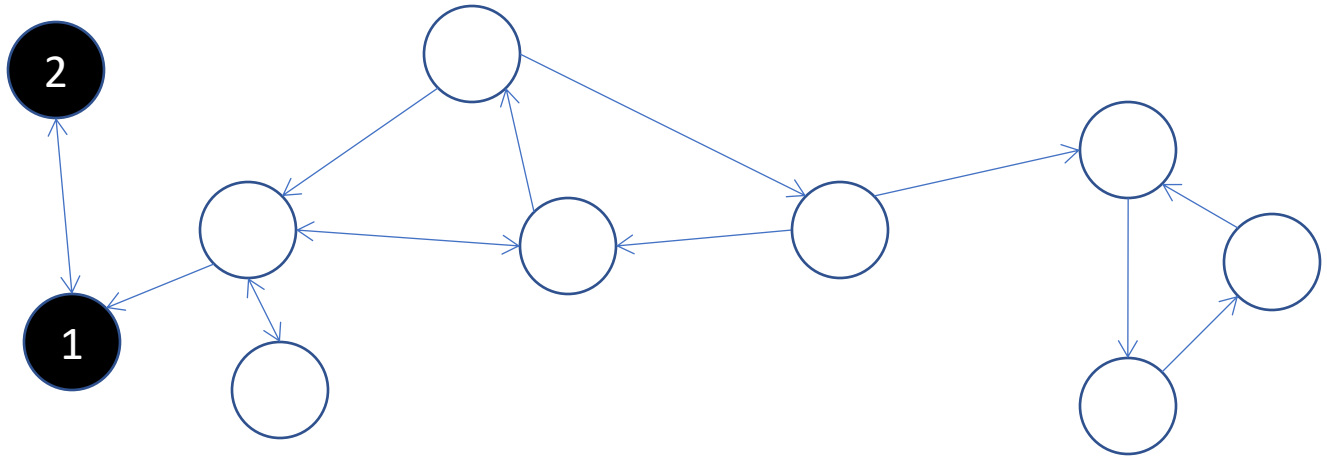
Phase 1: DFS on G_r



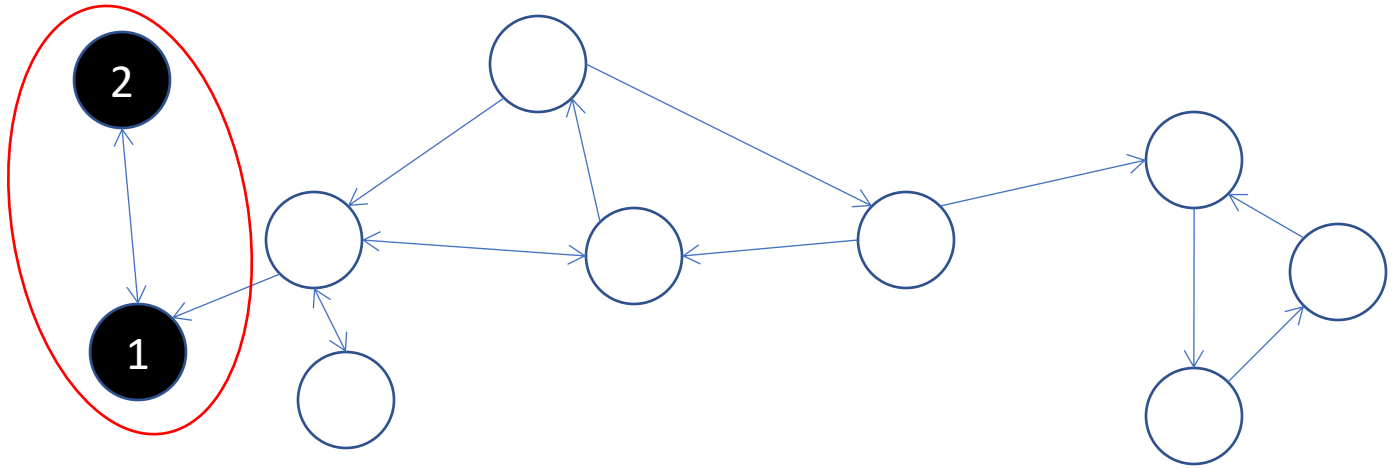
Phase 1: DFS on G_r



Phase 1: DFS on G_r

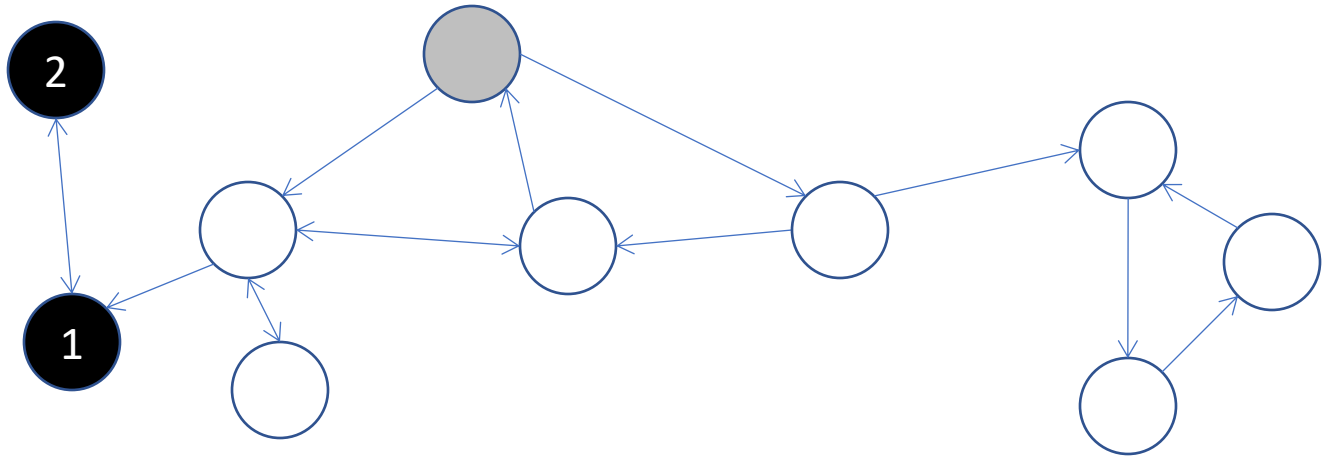


Phase 1: DFS on G_r

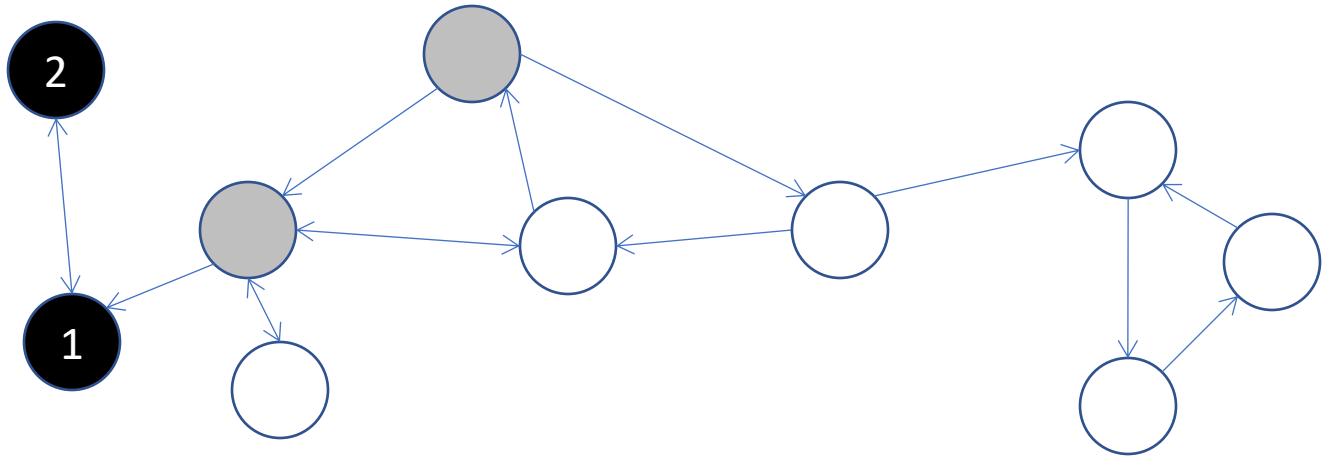


We have found a tree with root 2 (the last node to turn black)

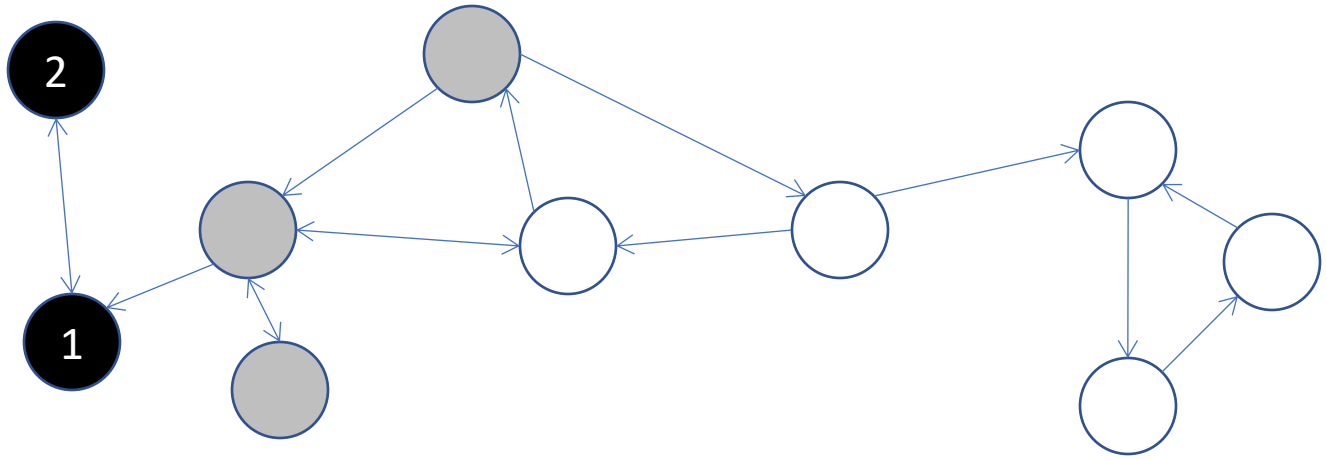
Phase 1: DFS on G_r



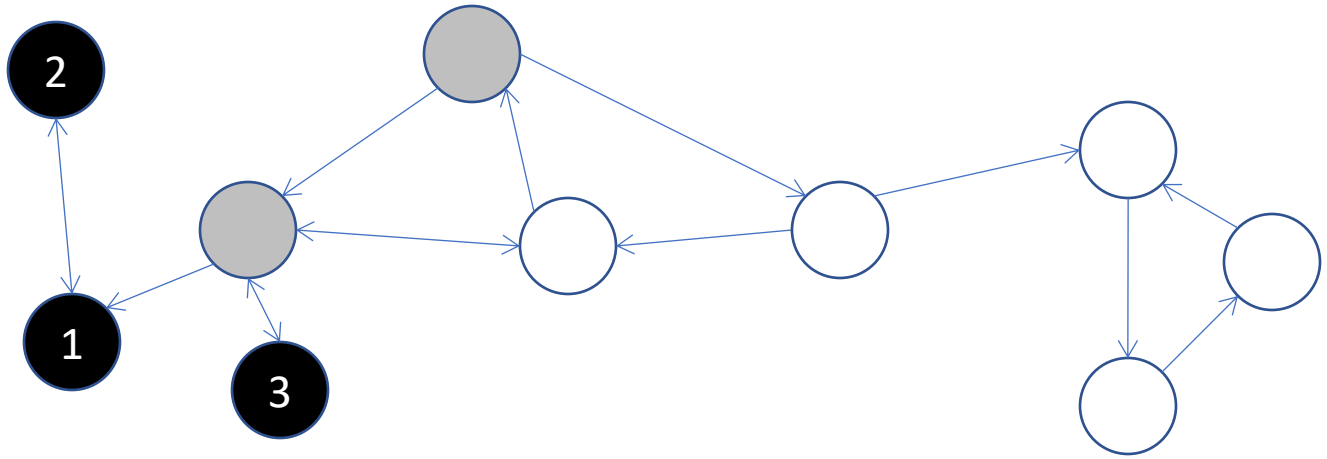
Phase 1: DFS on G_r



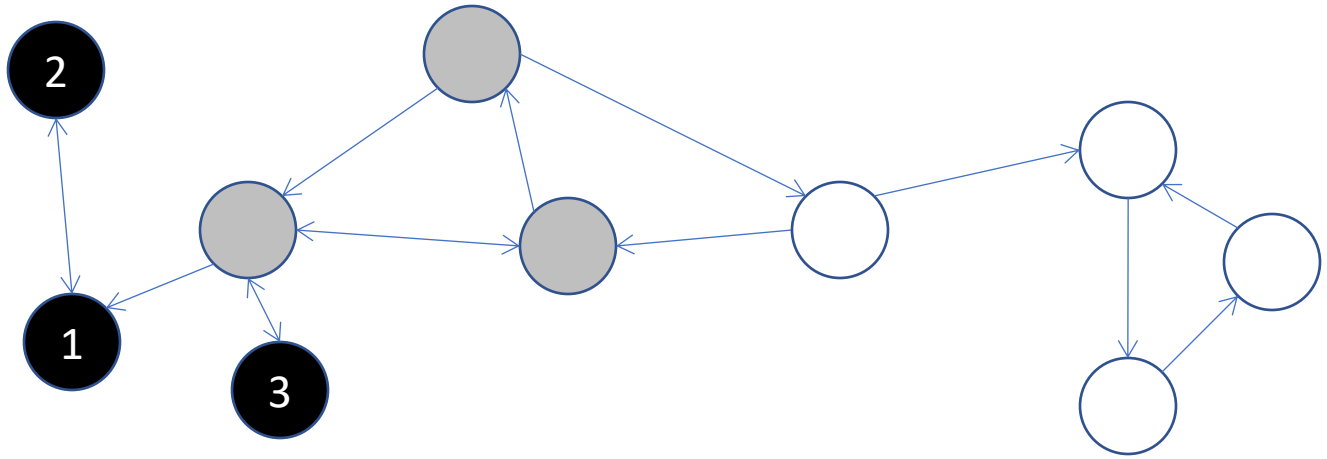
Phase 1: DFS on G_r



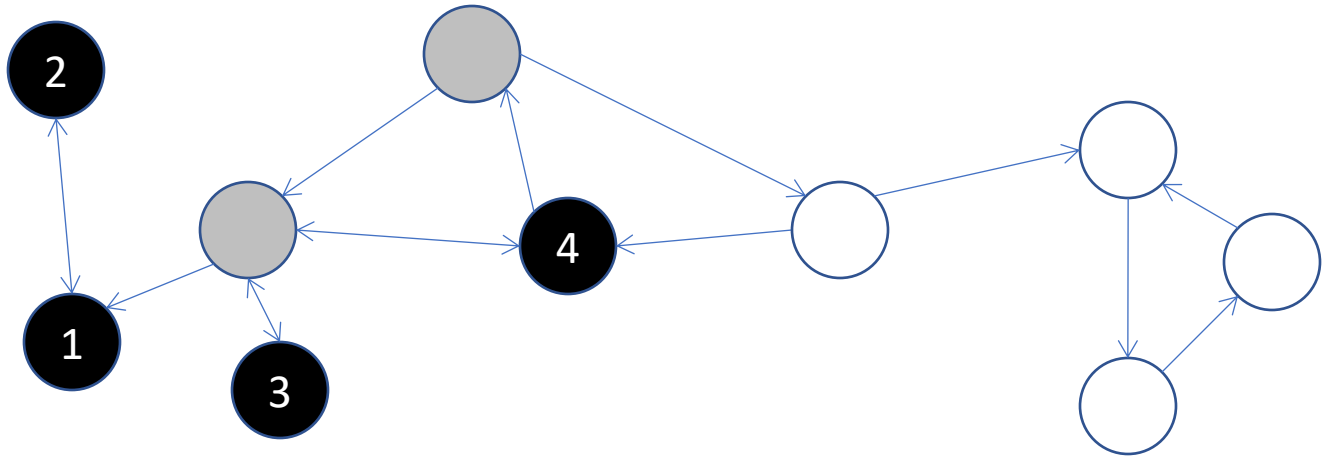
Phase 1: DFS on G_r



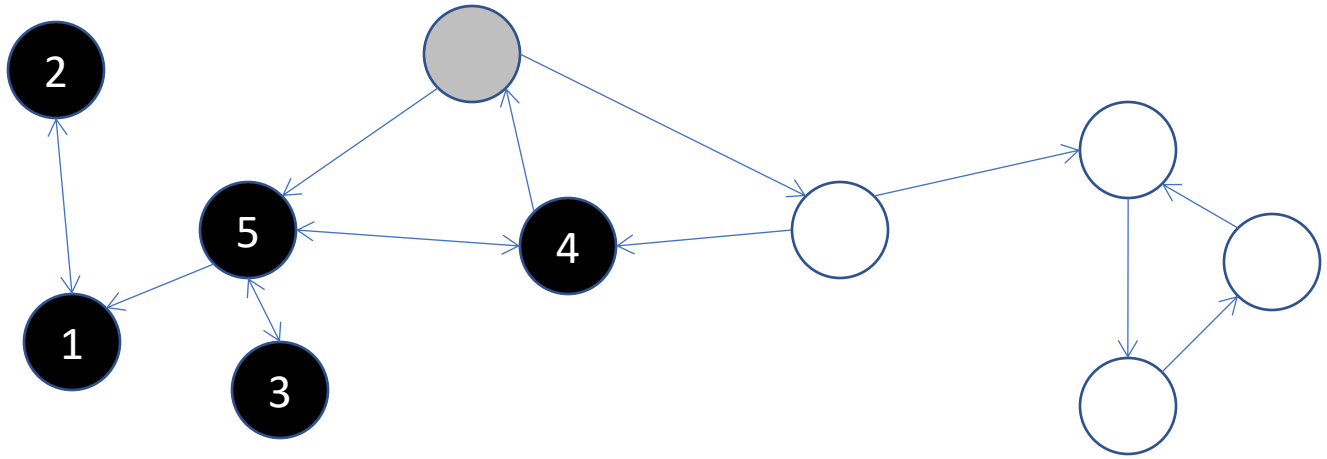
Phase 1: DFS on G_r



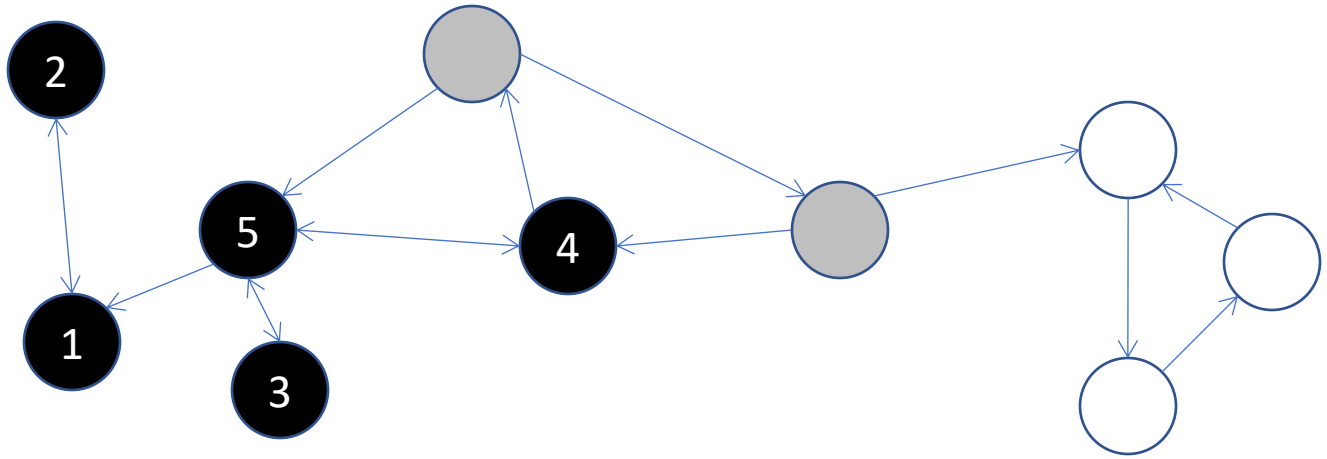
Phase 1: DFS on G_r



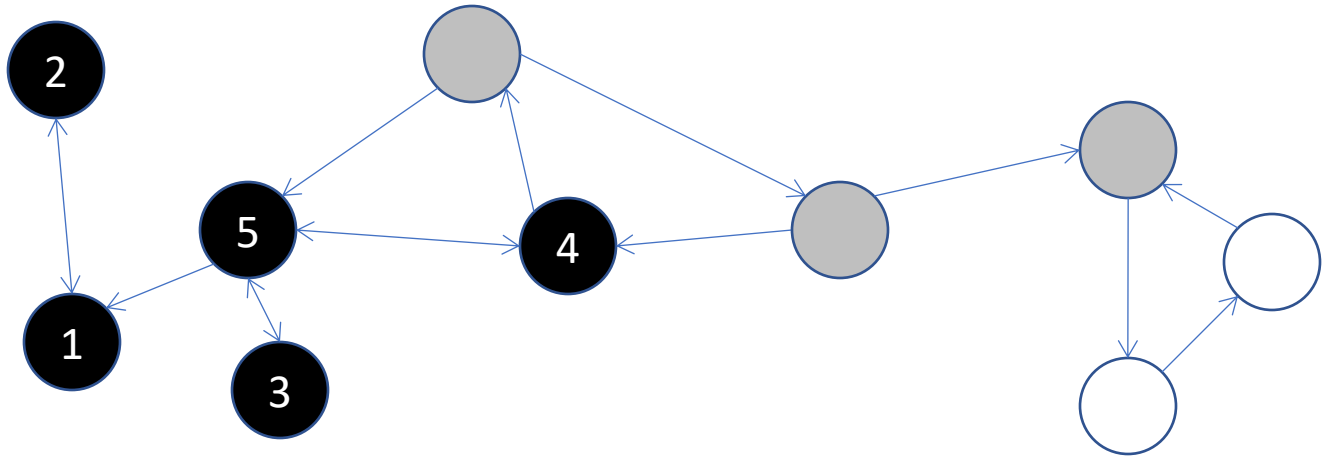
Phase 1: DFS on G_r



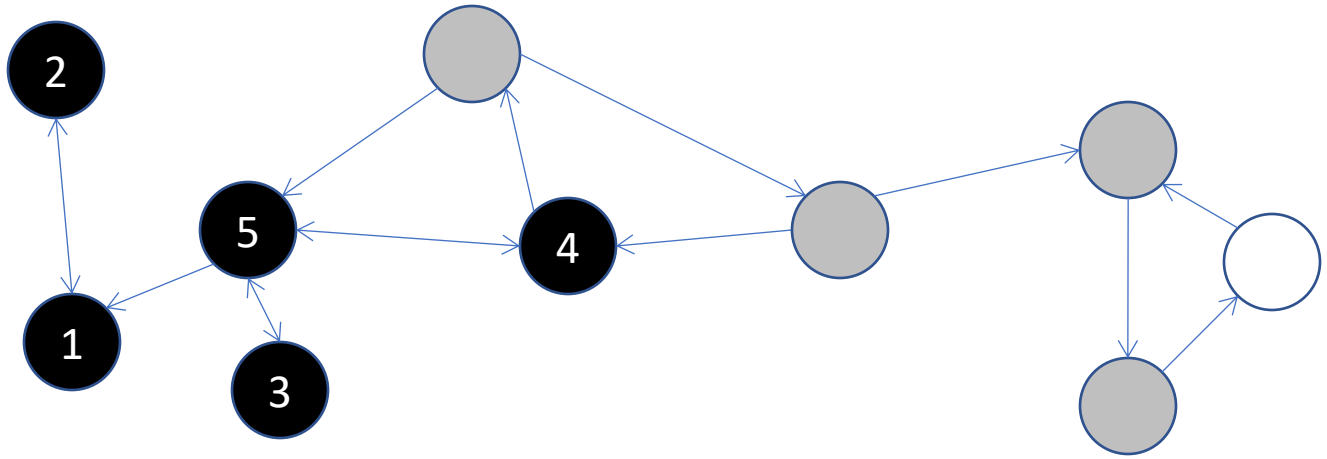
Phase 1: DFS on G_r



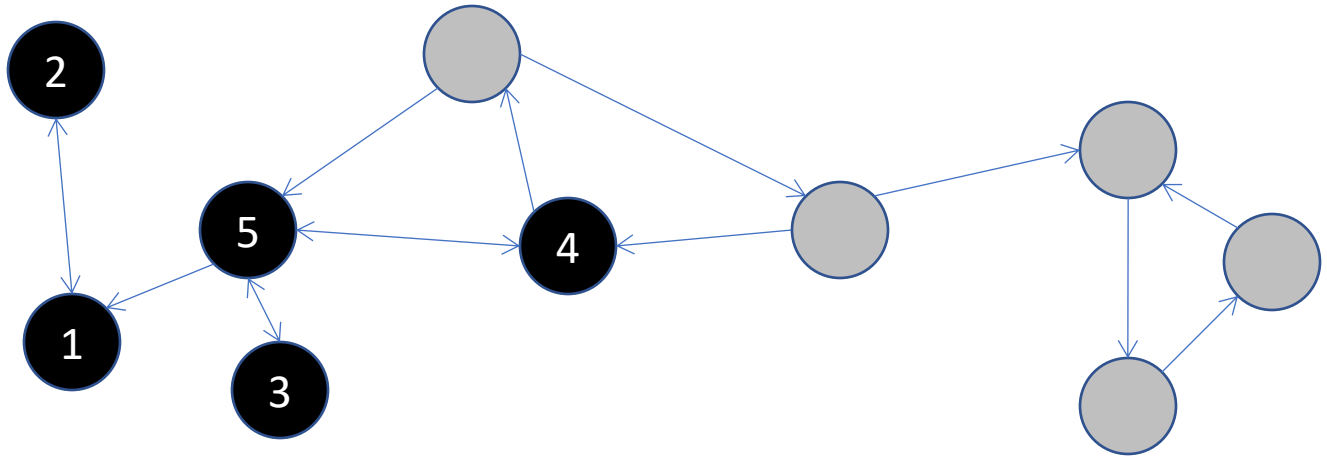
Phase 1: DFS on G_r



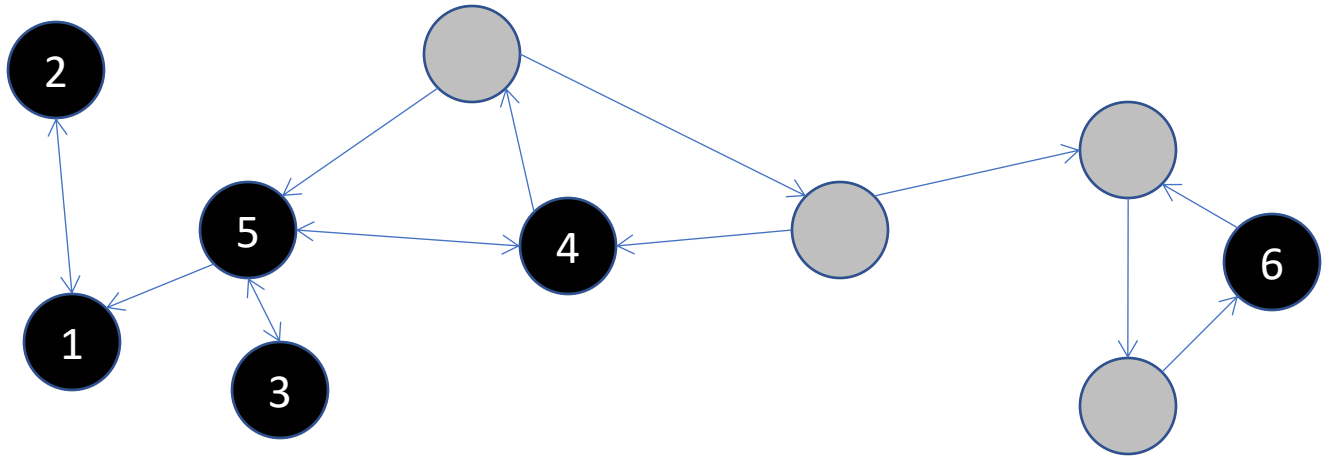
Phase 1: DFS on G_r



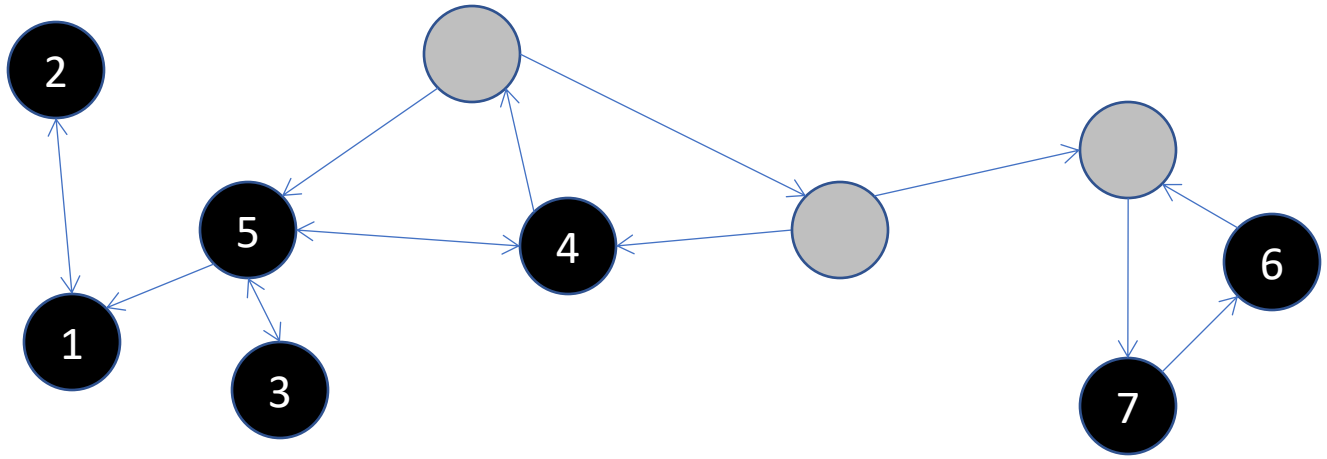
Phase 1: DFS on G_r



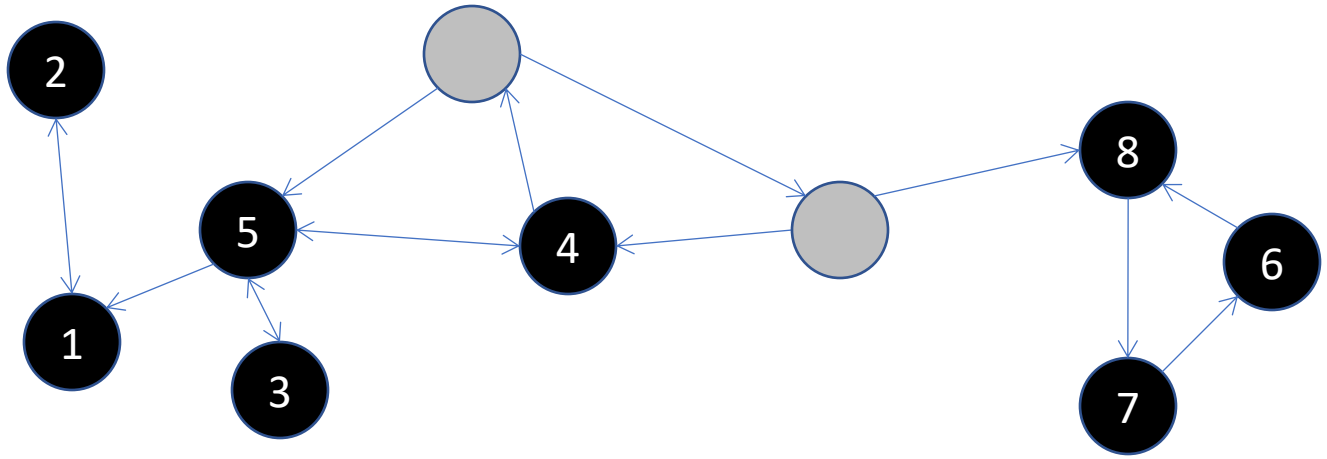
Phase 1: DFS on G_r



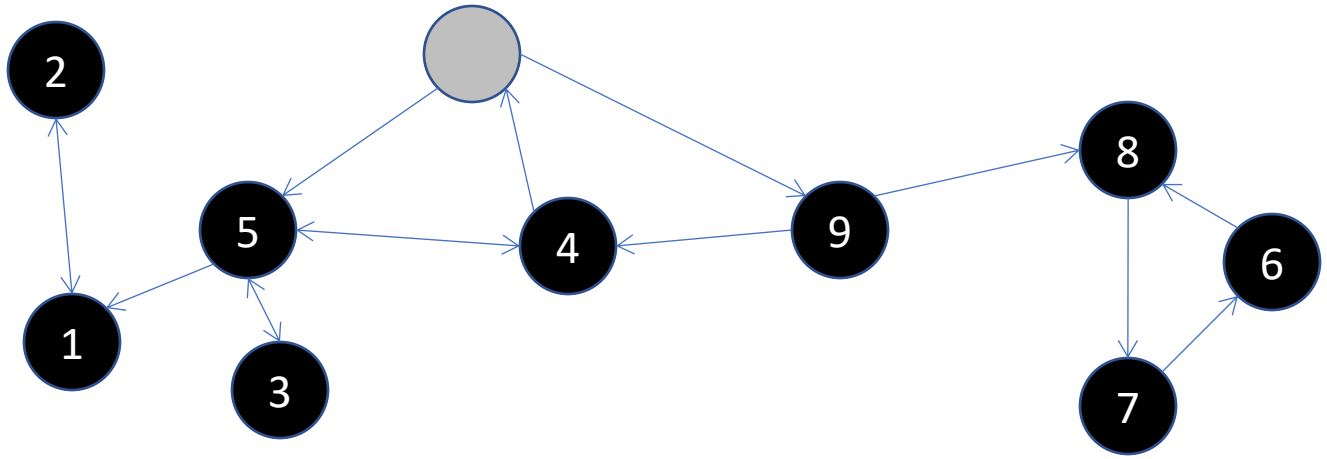
Phase 1: DFS on G_r



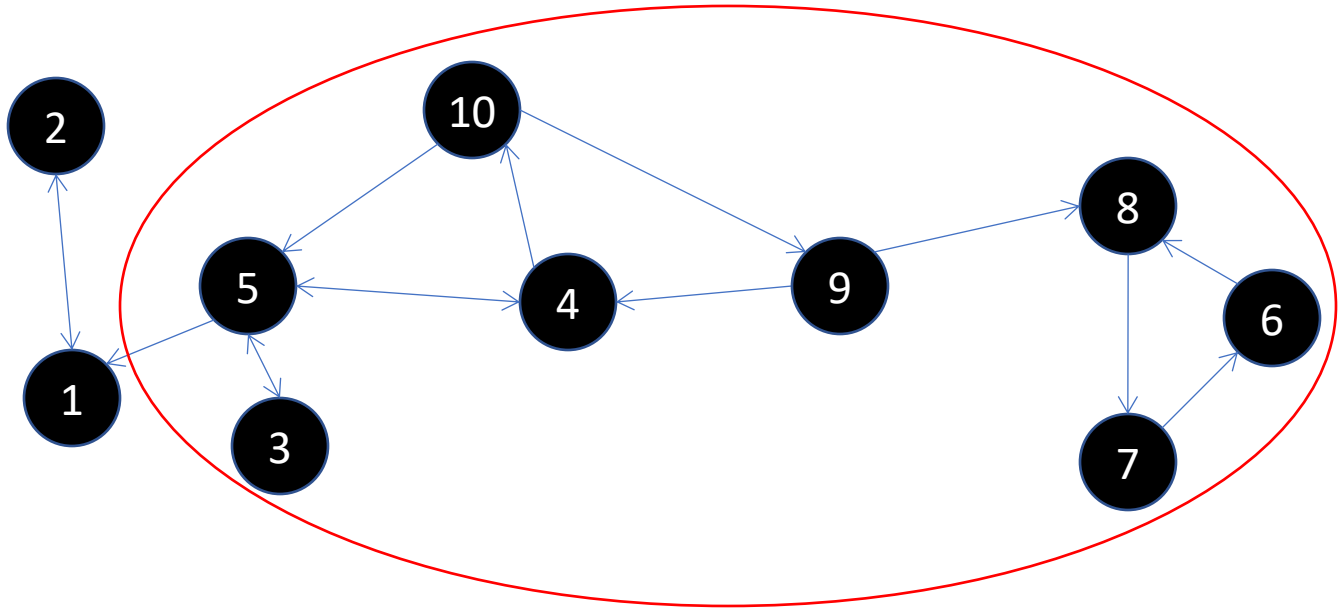
Phase 1: DFS on G_r



Phase 1: DFS on G_r

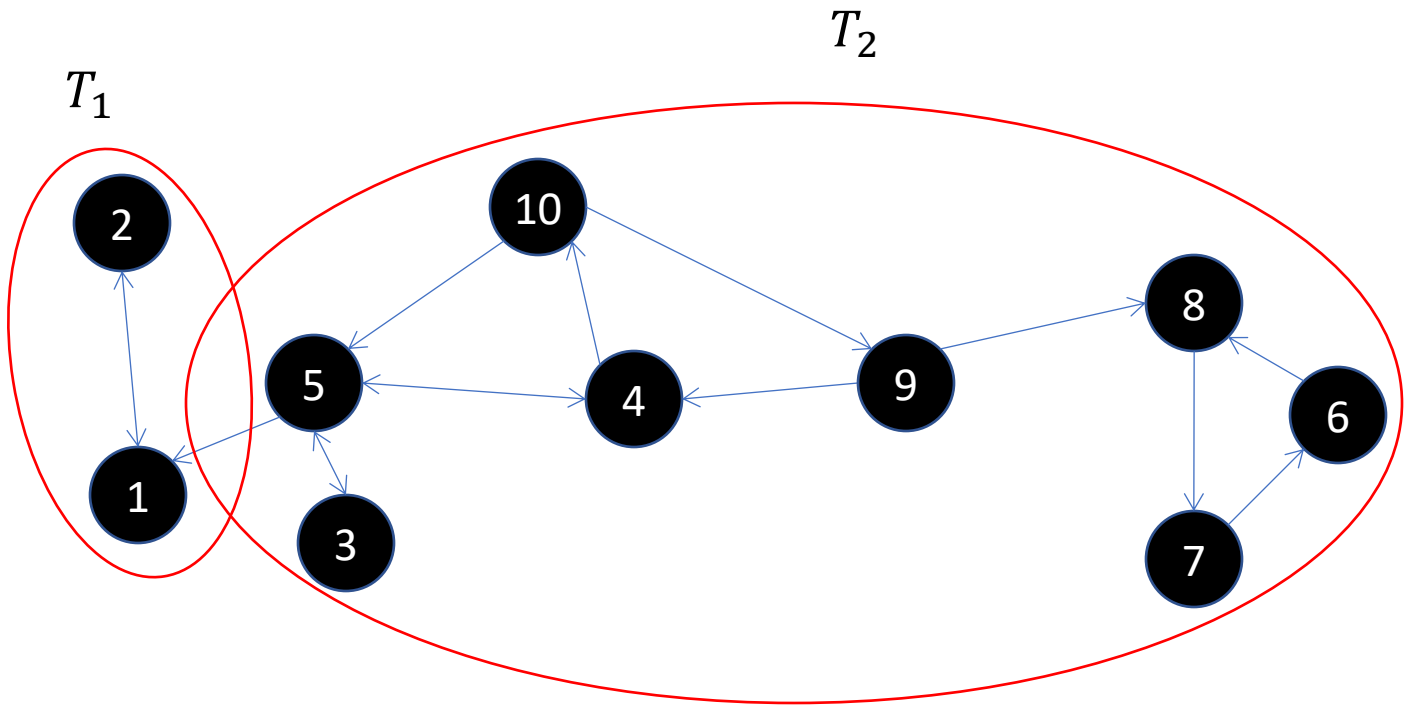


Phase 1: DFS on G_r



This is another tree with root 10 (the last node to turn black)

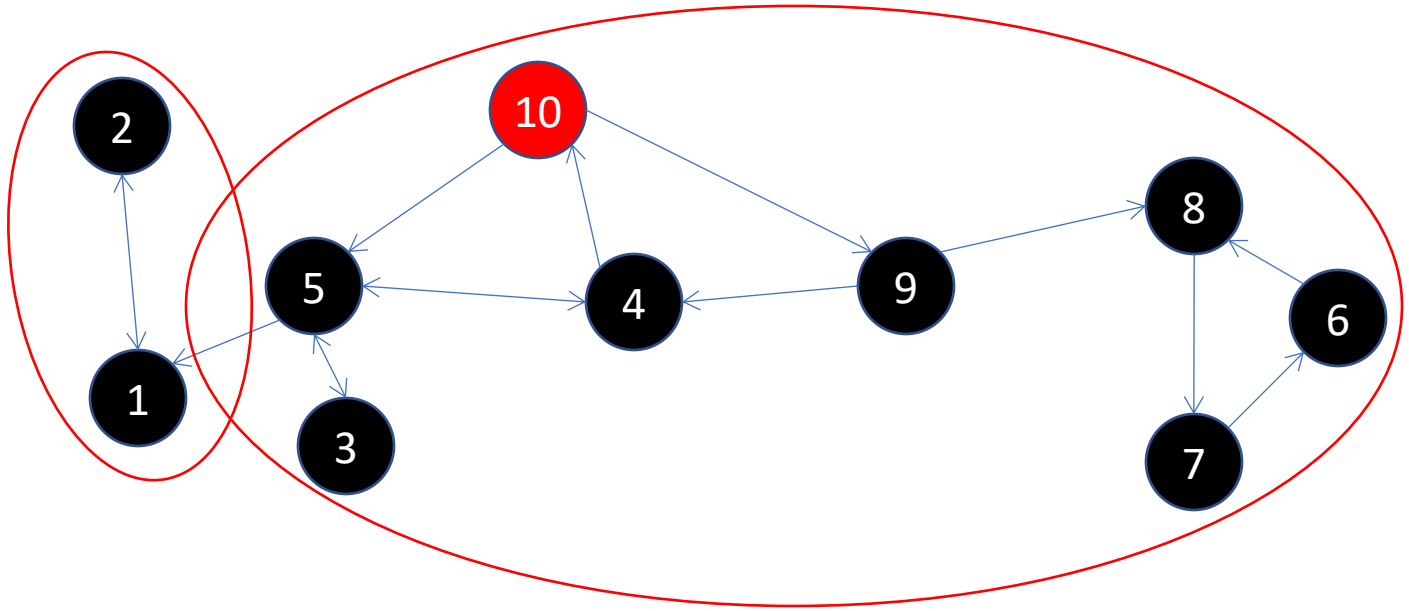
Phase 1: DFS on G_r



The DFS has returned two trees, T_1 with root 2 and T_2 with root 10.

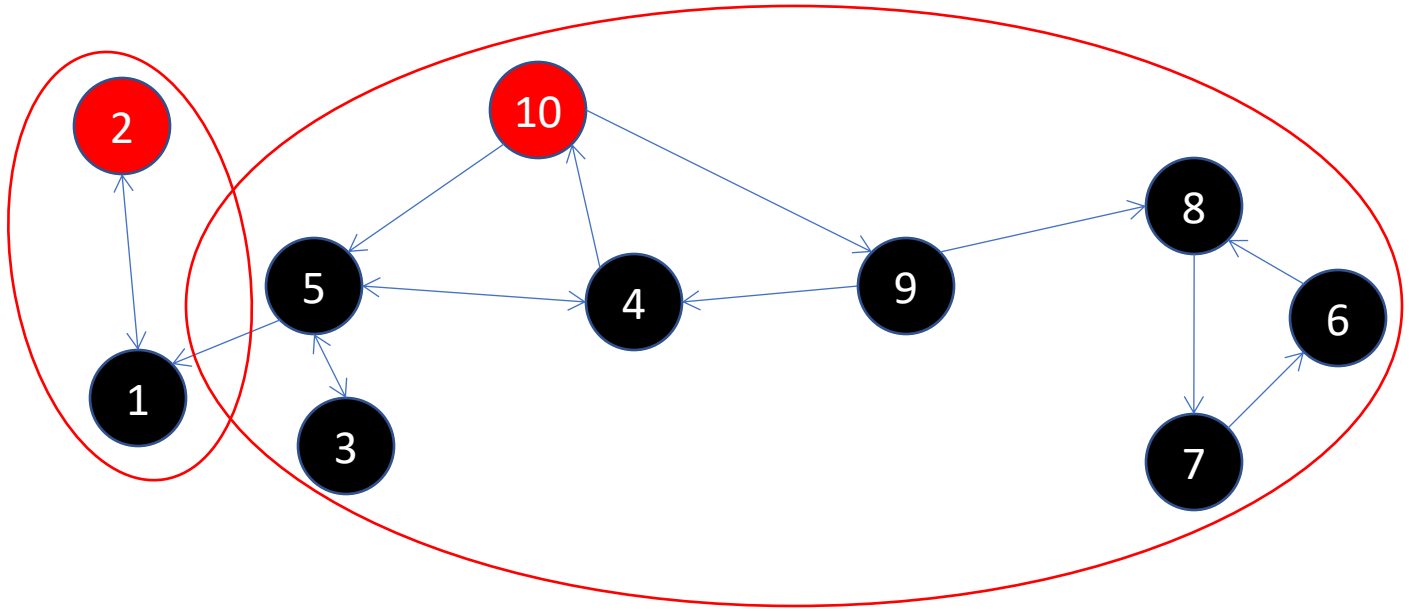
Finding the SCCs in a Digraph (Contd.)

- The vertex with the highest number (10) is the root of the last tree (T_2) of the search forest for G_r.
- Note the numbers in this tree (T_2) are all higher than those in other trees of the search forest.



Finding the SCCs in a Digraph (Contd.)

- The root in each tree of the search forest has the highest number in that tree
- The vertices in a tree have numbers between the number of the root and the number of the root of the previous tree in the search forest.



Finding the SCCs in a Digraph (Contd.)

- Now we run DFS on G_r by using the white node with the highest number as root whenever we start a new tree in the search forest for G.

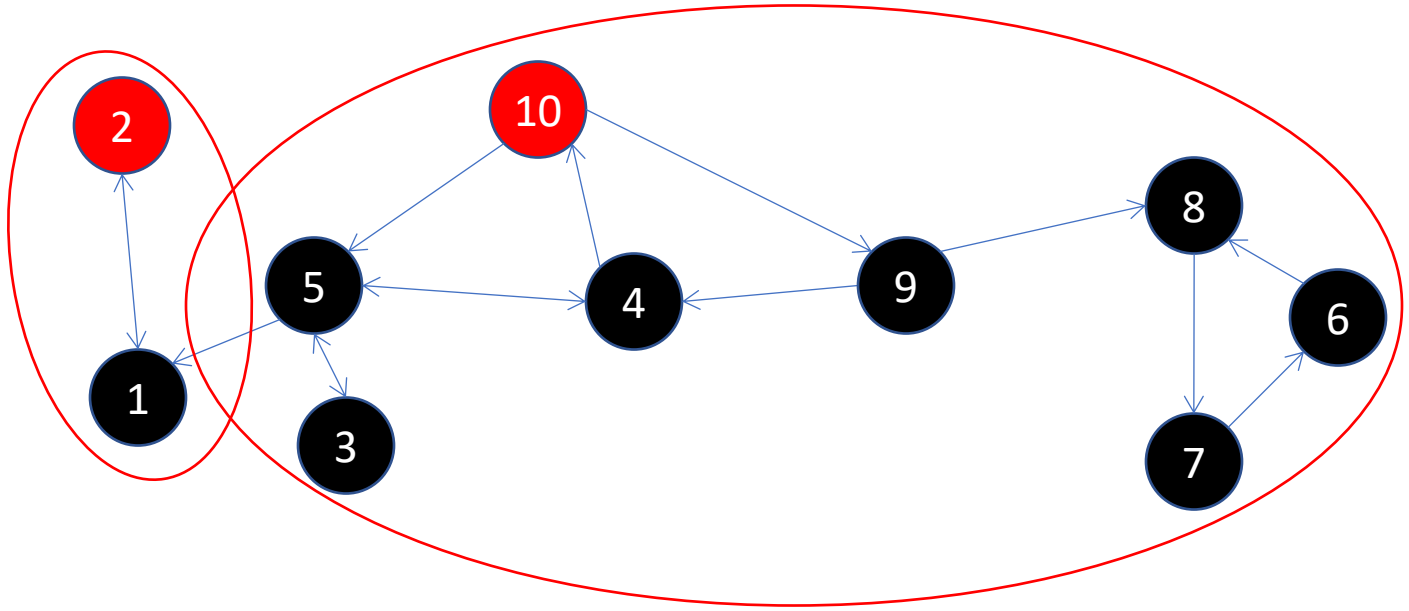
- This means we start with the root of the last tree in the search forest of G_r.

Theorem: If T_1 and T_2 are two distinct trees in the search forest F for G for some algorithm that follows our basic traversal algorithm:

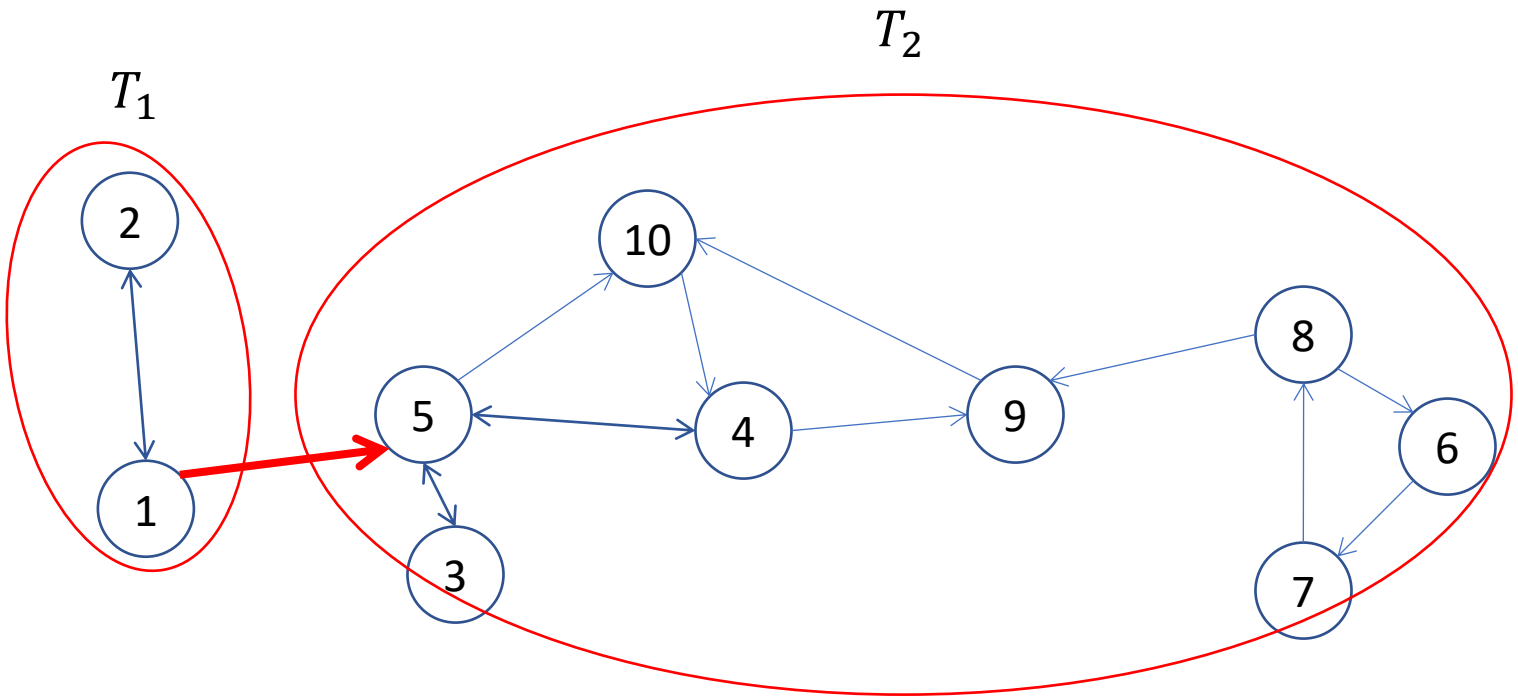
- T_1 is traversed before T_2 , then there are no arcs/edges from T_1 to T_2 .

Finding the SCCs in a Digraph (Contd.)

- No tree in G_r has an incoming arc from a predecessor tree



Finding the SCCs in a Digraph (Contd.)

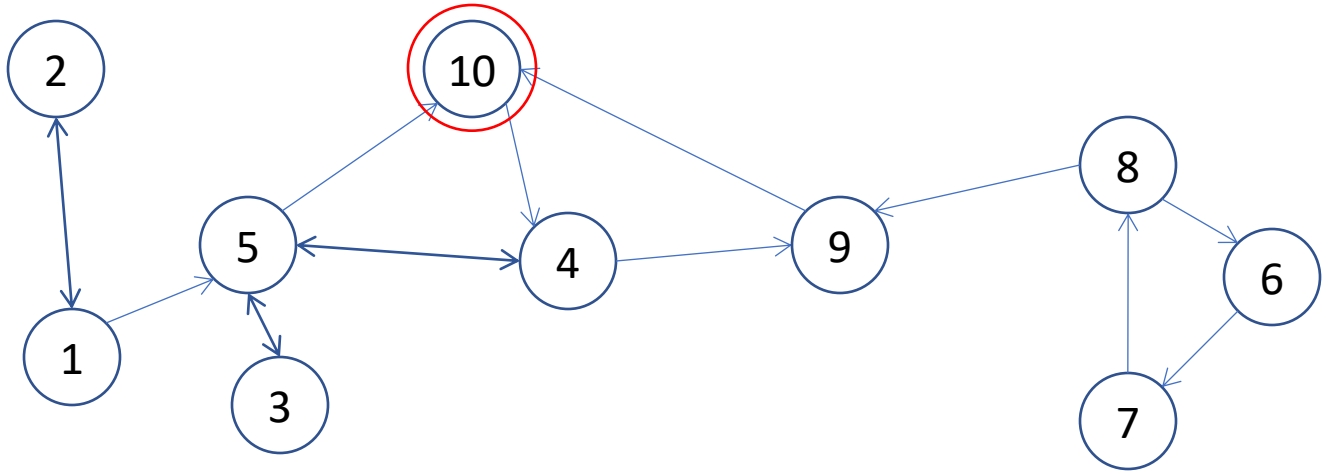


Finding the SCCs in a Digraph (Contd.)

- If the DFS on G does not reach every vertex in the current tree from G_r , then the tree contains more than one strongly connected component.

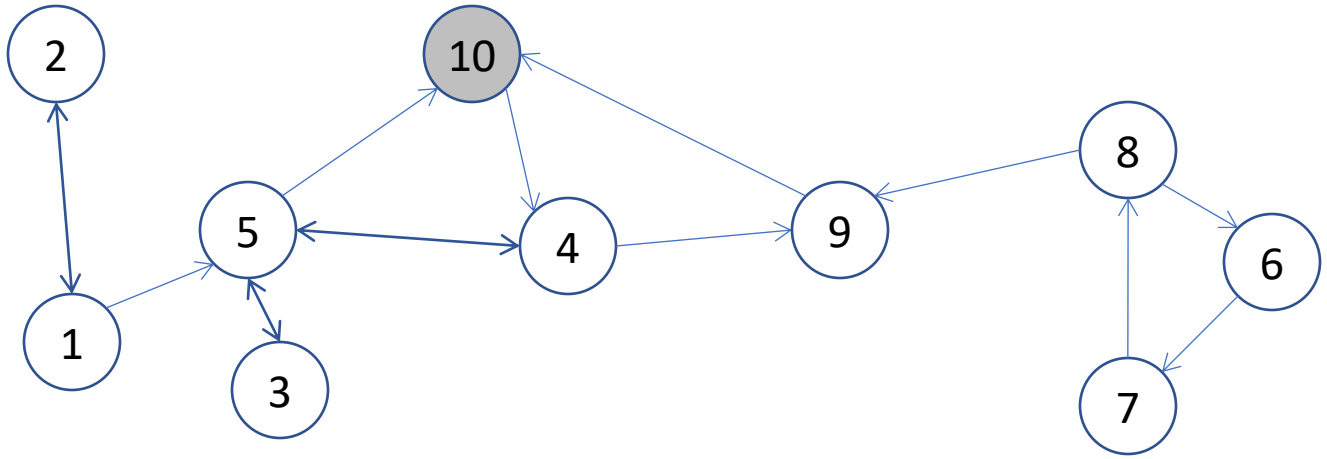
- In this case, we again start the next tree with the white node with the highest number.

Phase 2: DFS on G

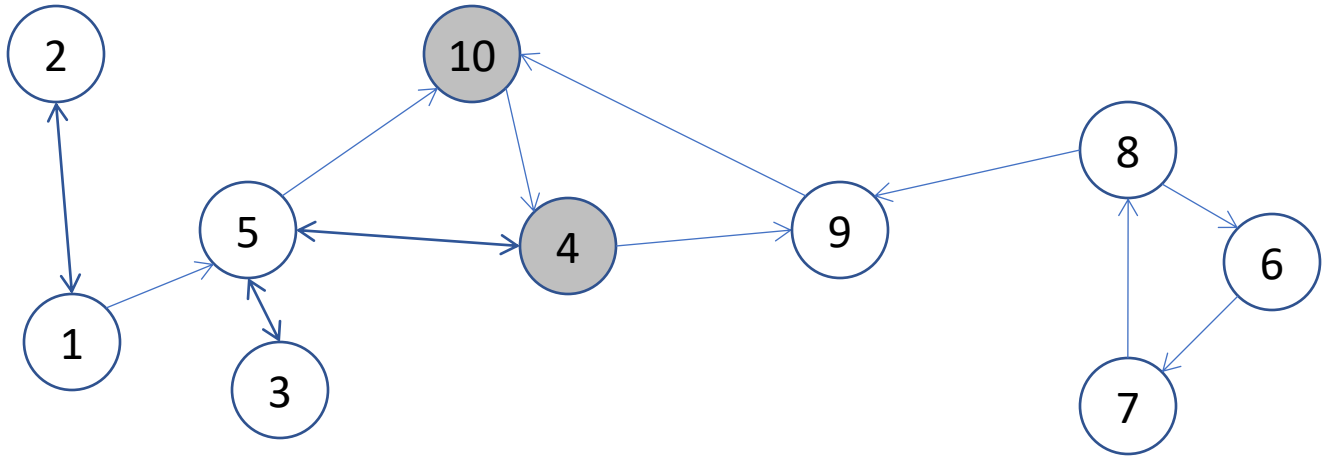


Let's start the DFS from the node with the highest number

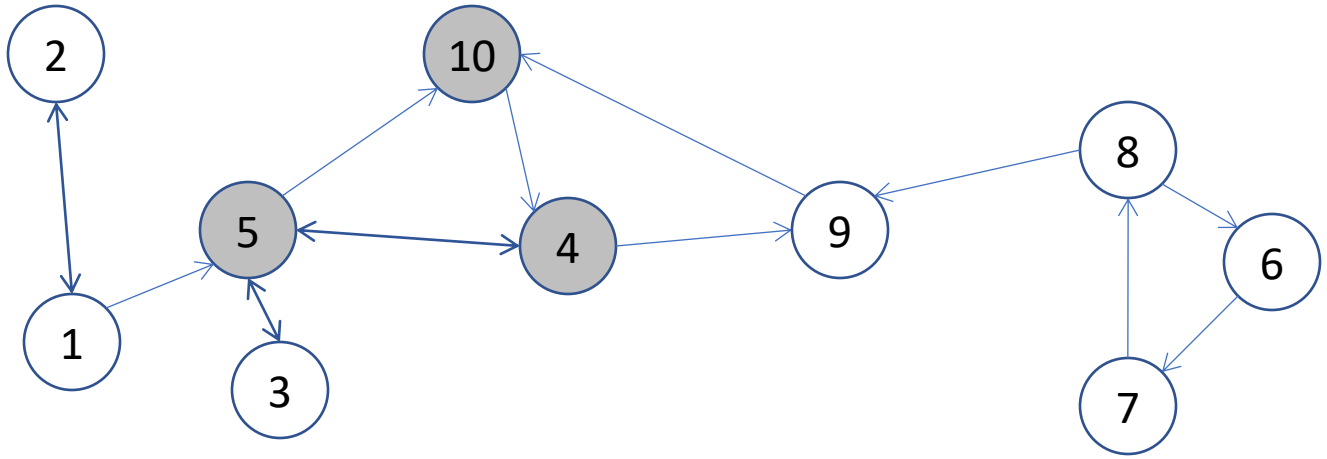
Phase 2: DFS on G



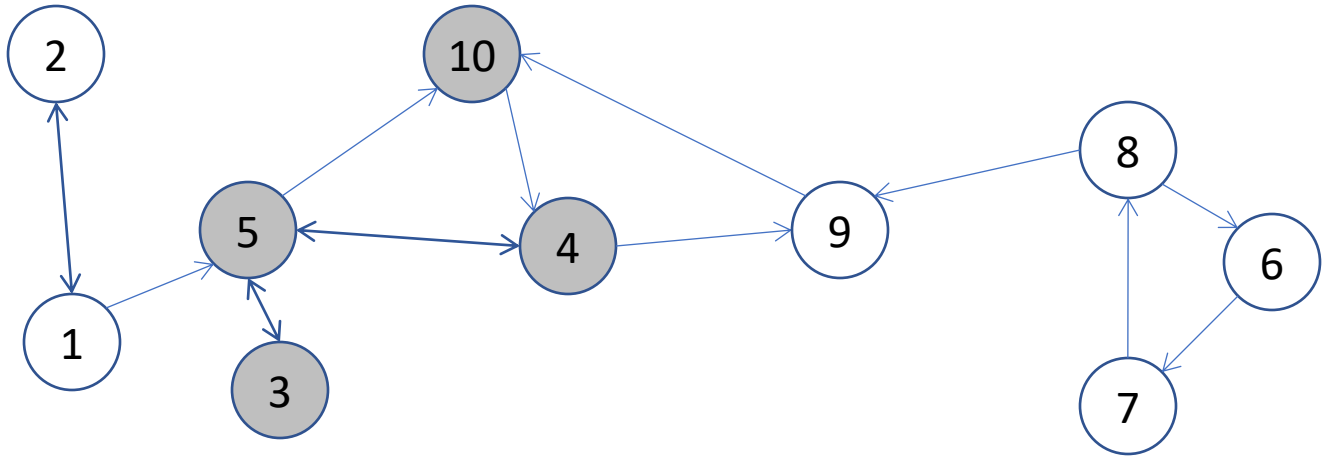
Phase 2: DFS on G



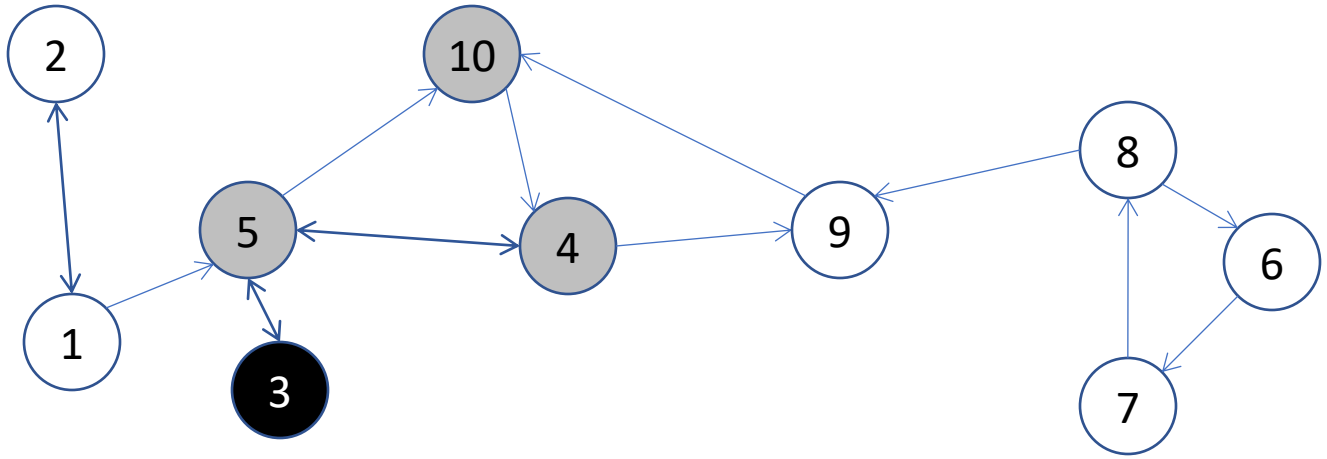
Phase 2: DFS on G



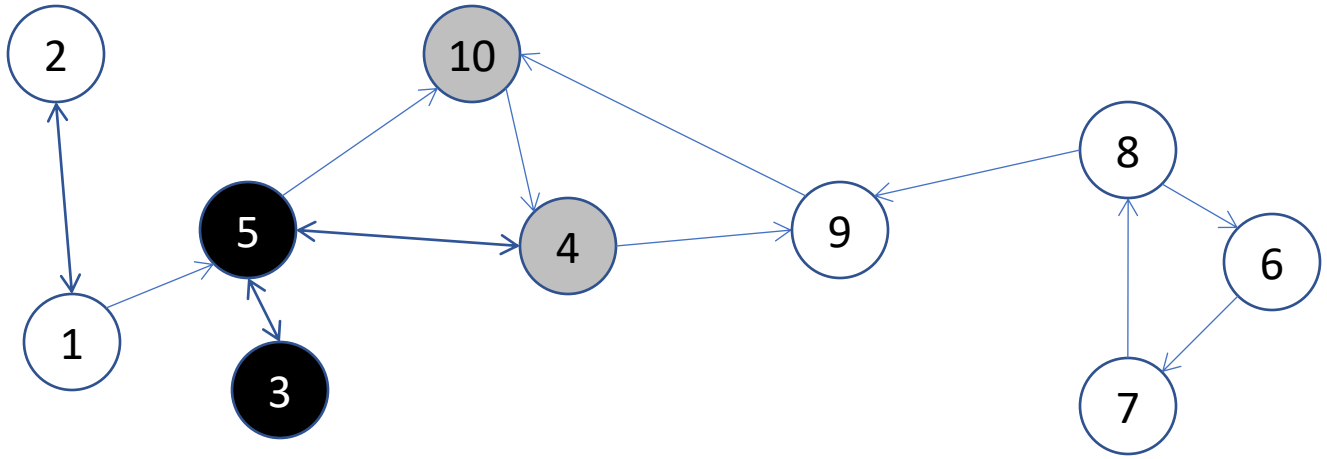
Phase 2: DFS on G



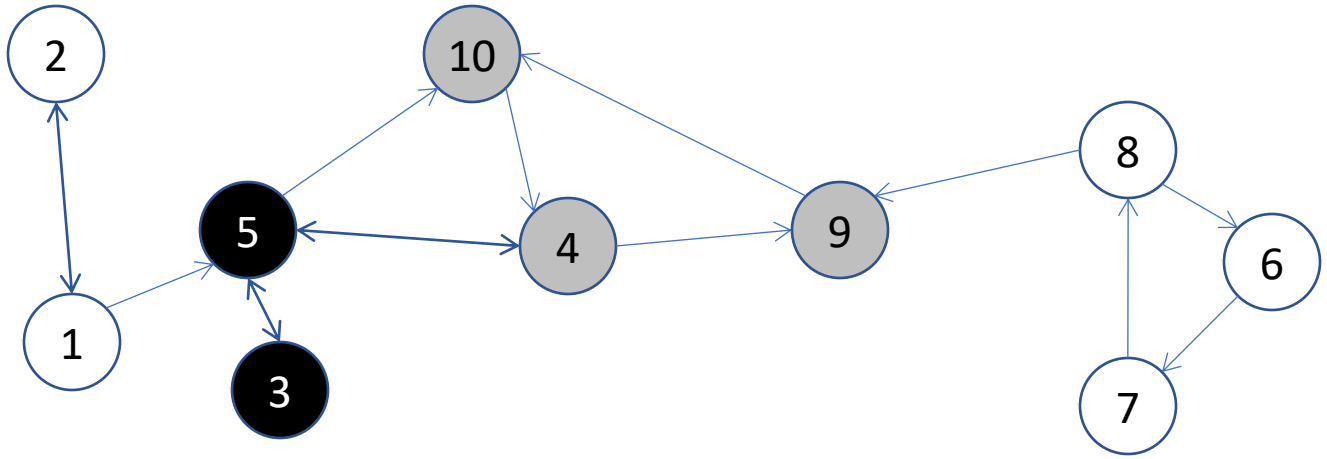
Phase 2: DFS on G



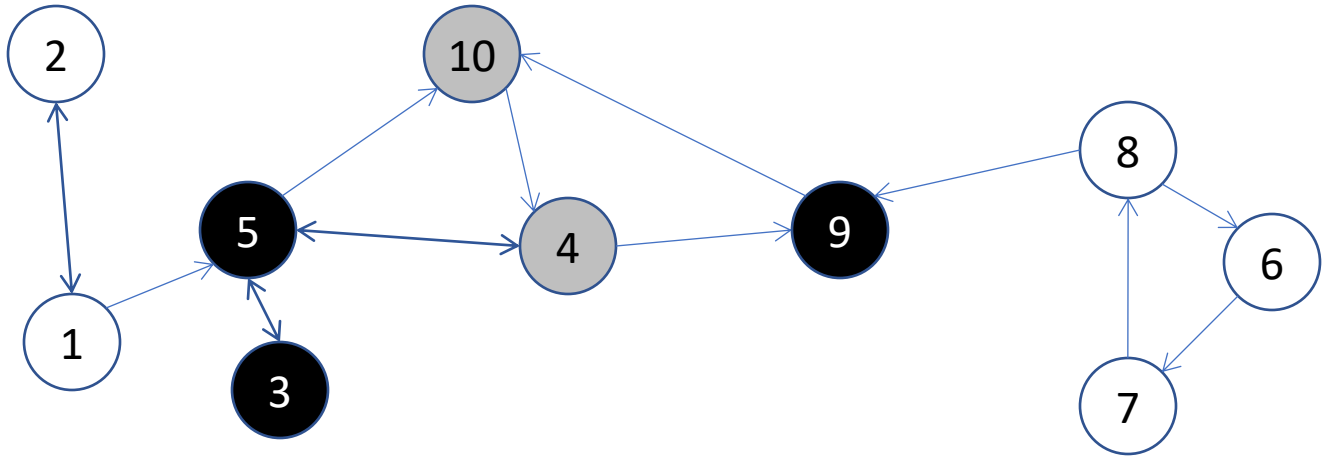
Phase 2: DFS on G



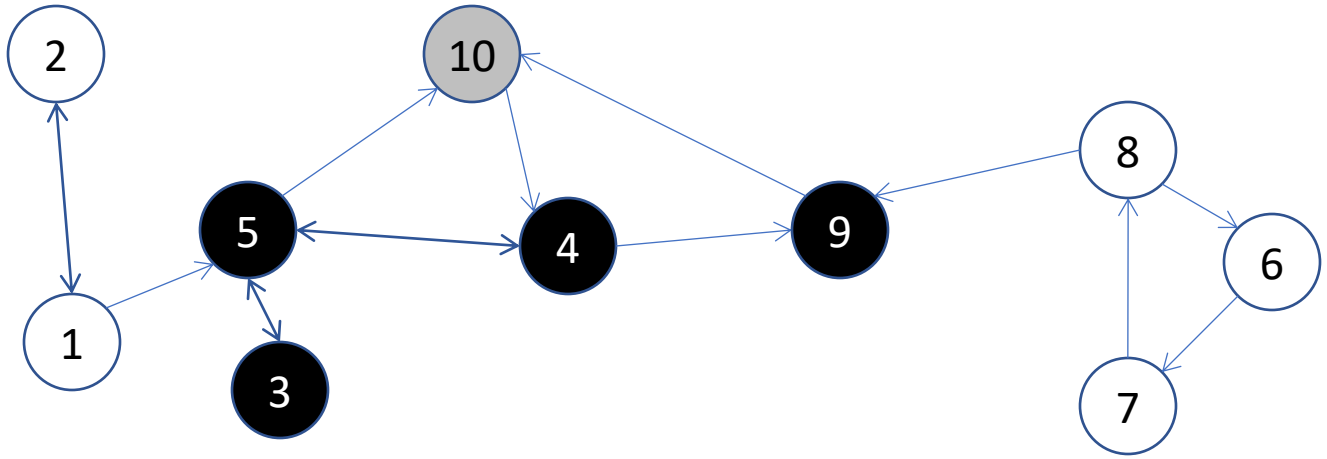
Phase 2: DFS on G



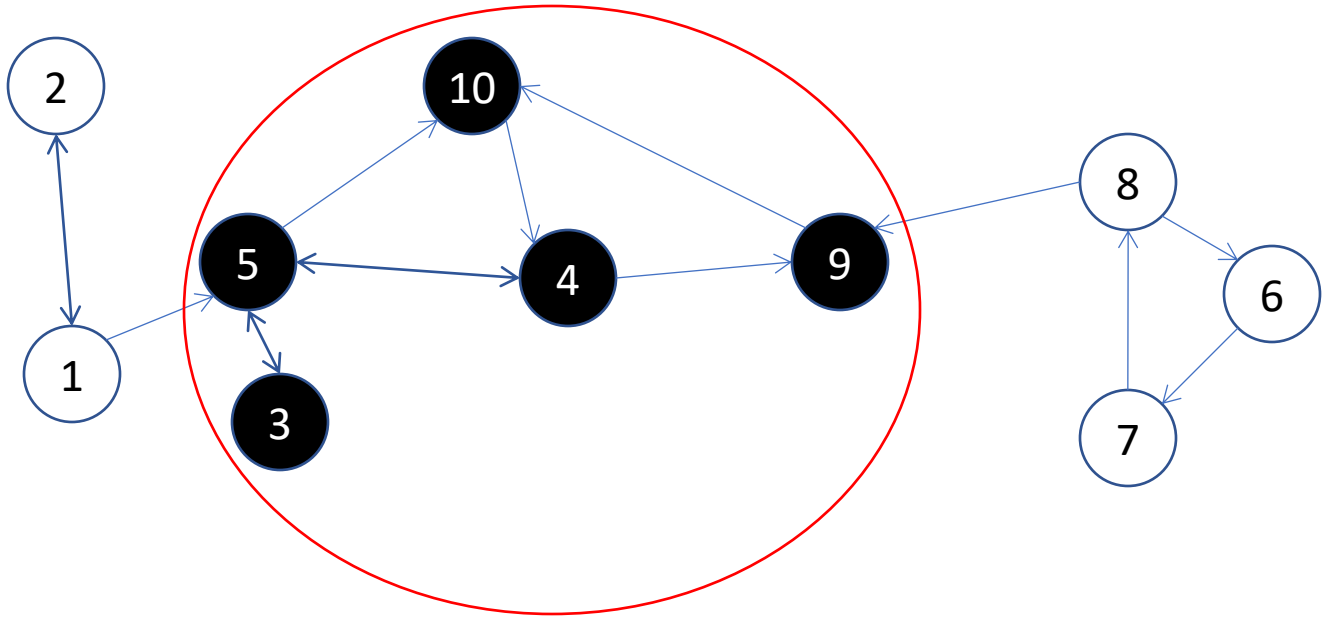
Phase 2: DFS on G



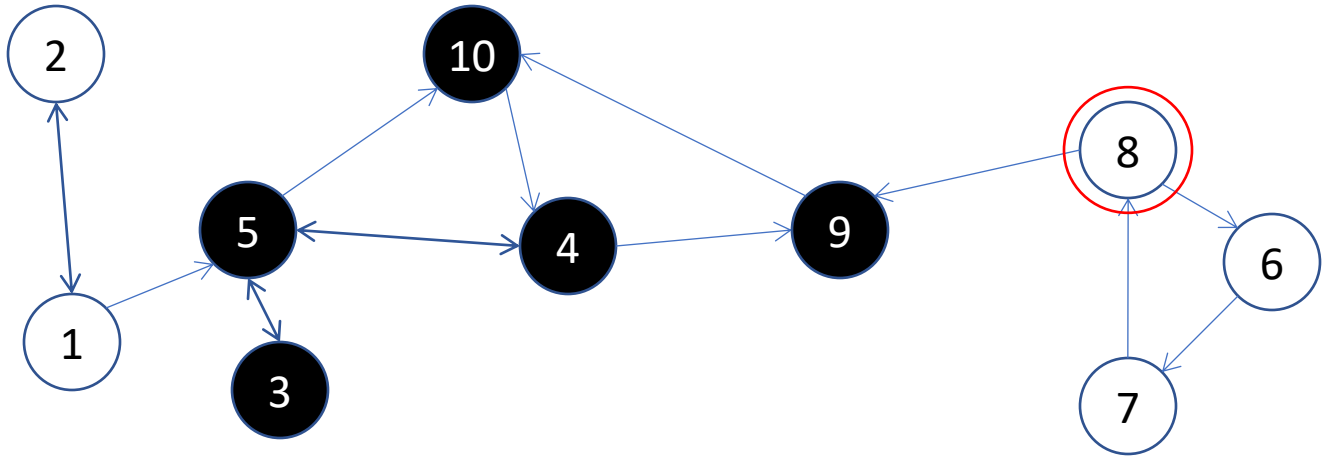
Phase 2: DFS on G



Phase 2: DFS on G

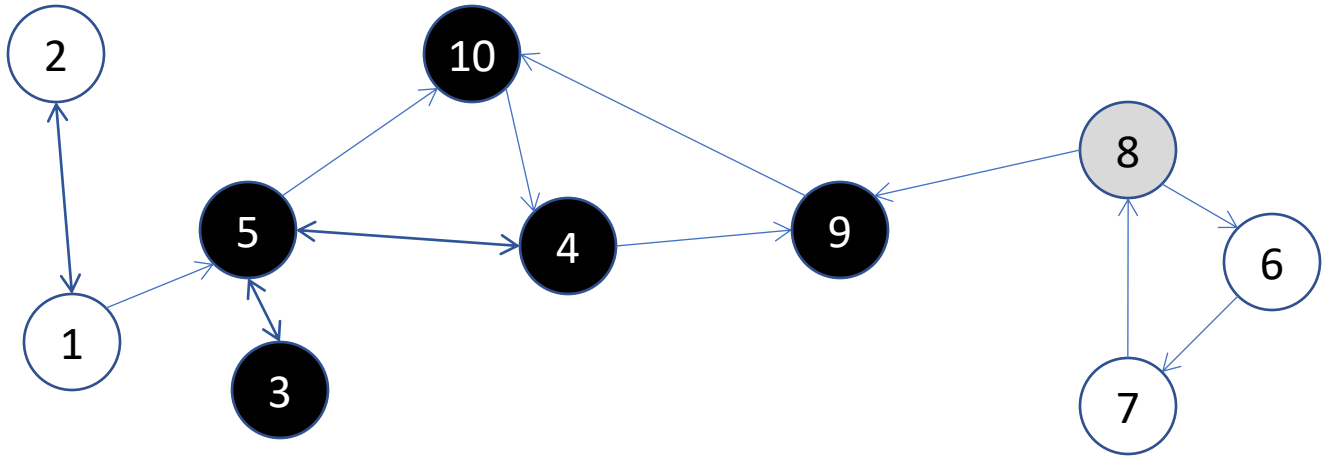


Phase 2: DFS on G

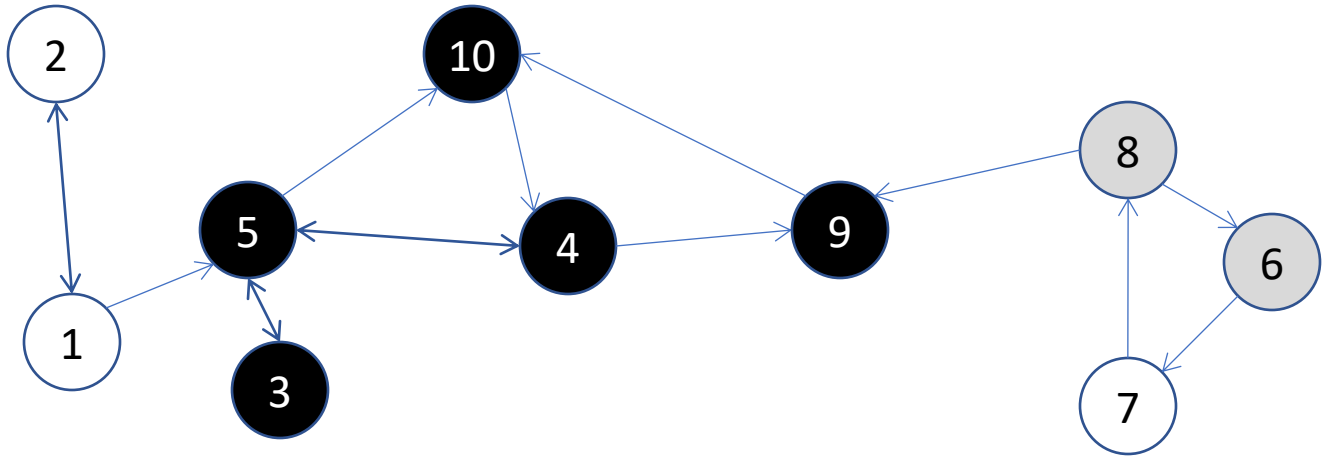


Next we select the white node with highest number

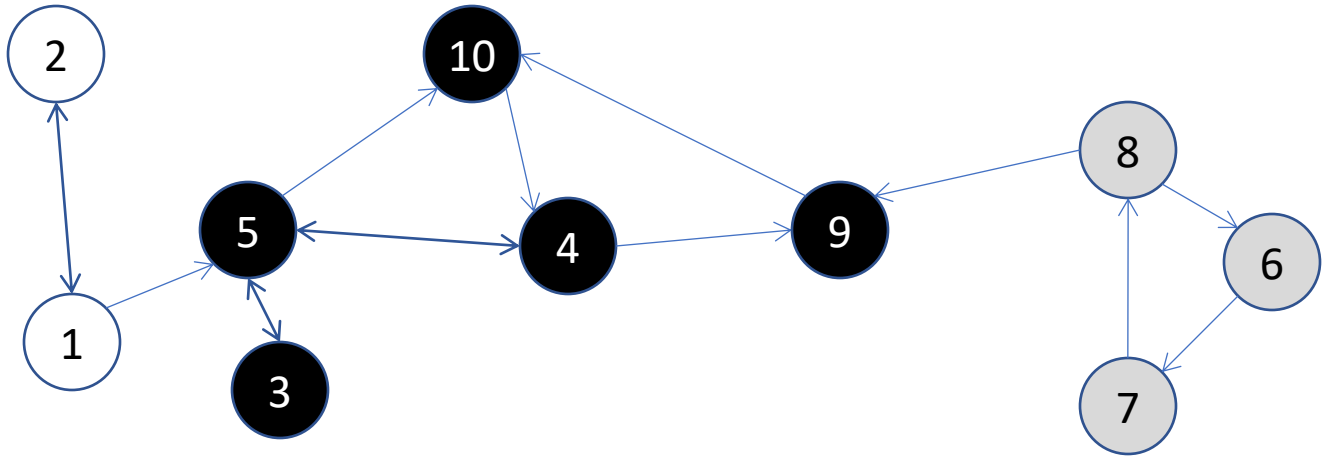
Phase 2: DFS on G



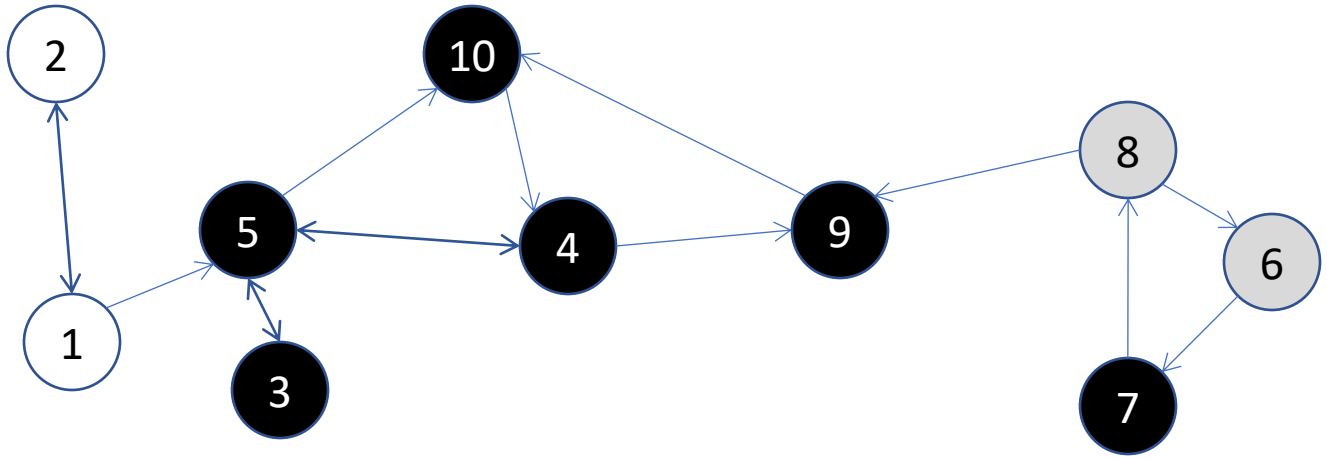
Phase 2: DFS on G



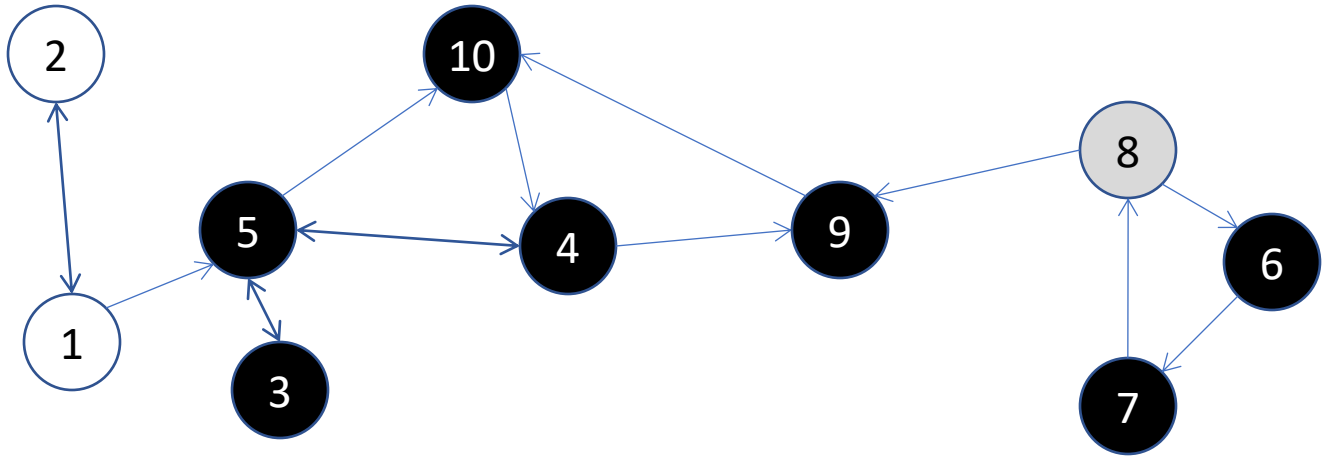
Phase 2: DFS on G



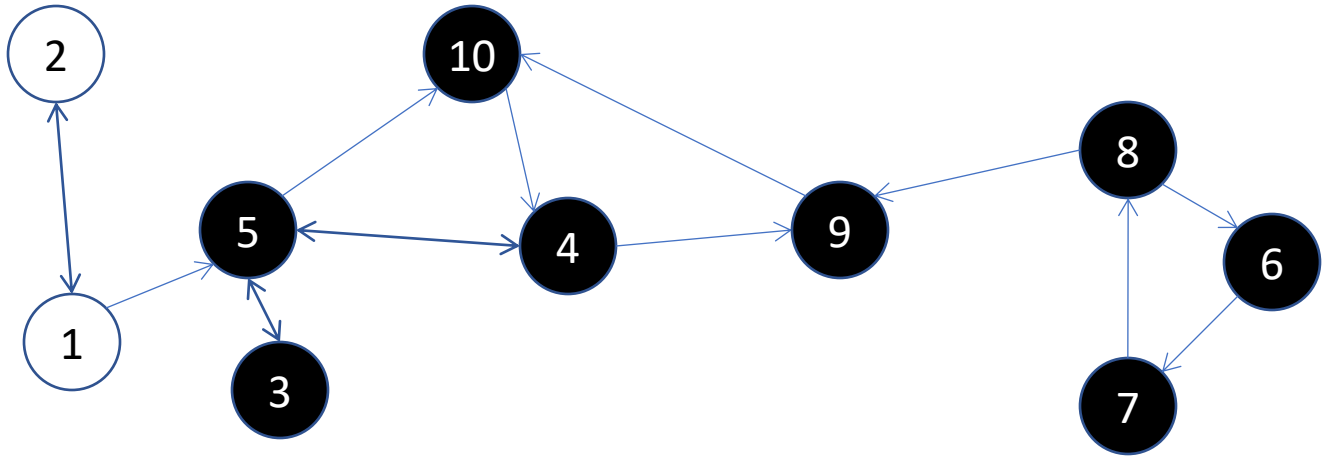
Phase 2: DFS on G



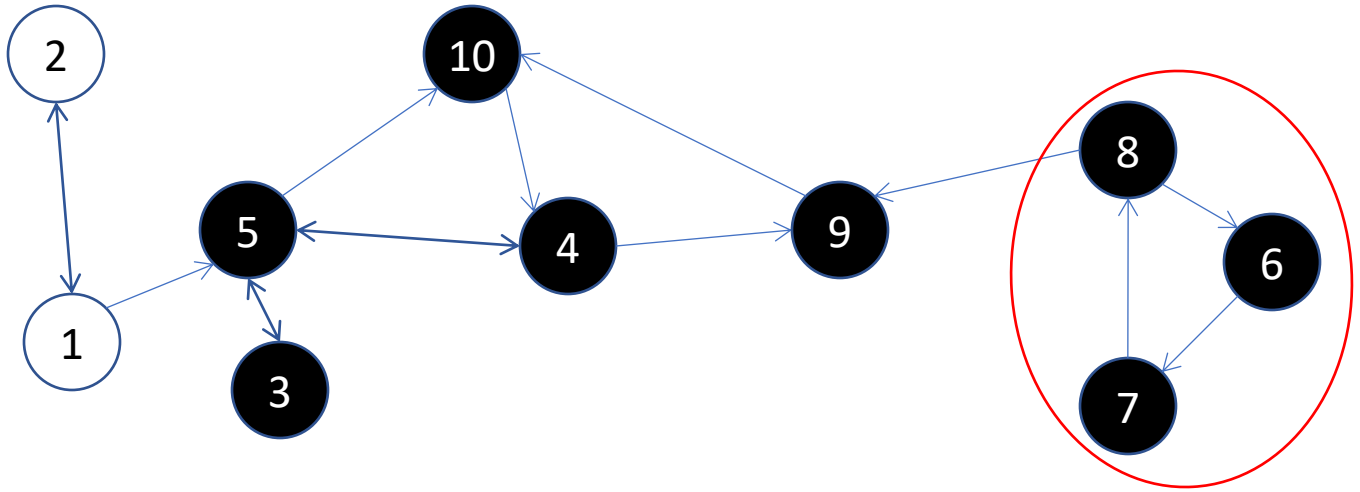
Phase 2: DFS on G



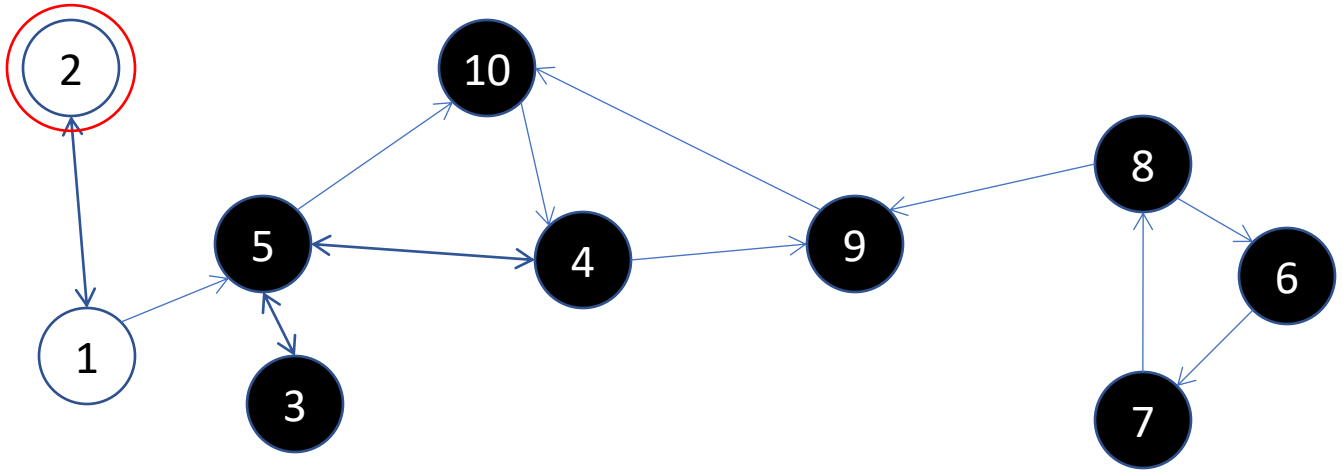
Phase 2: DFS on G



Phase 2: DFS on G

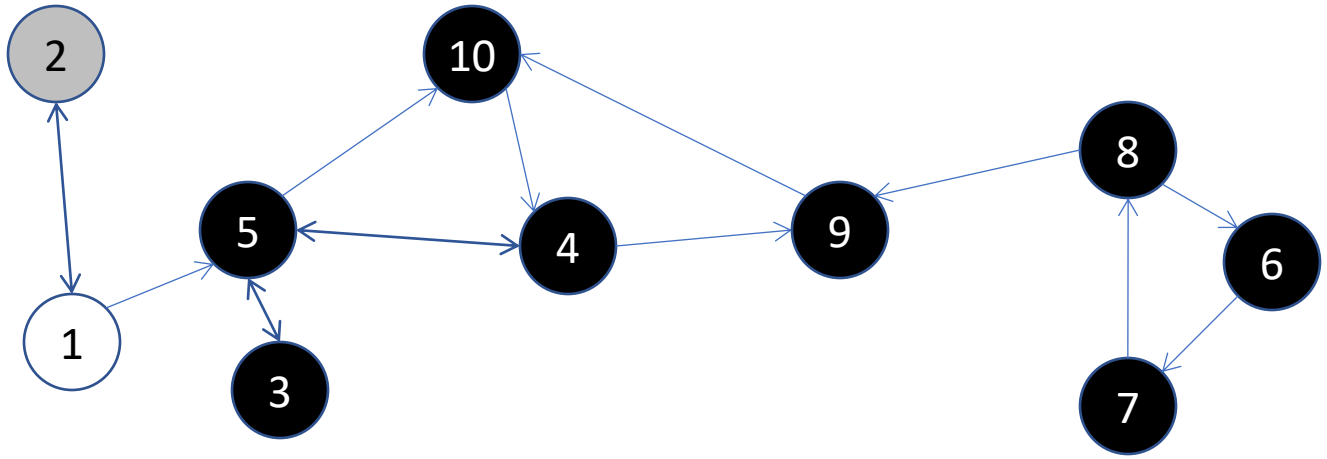


Phase 2: DFS on G

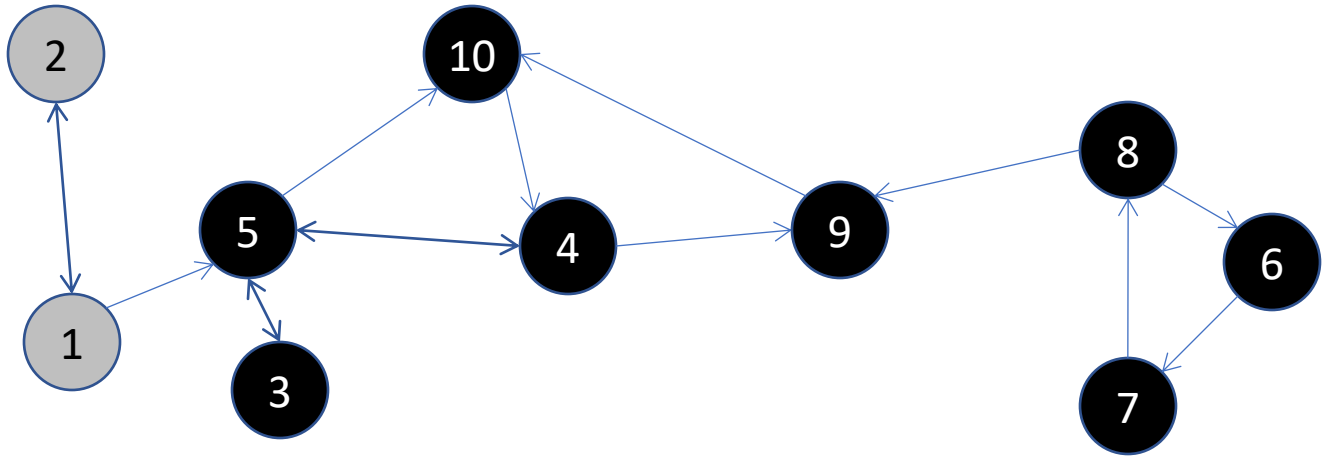


Again we select the white node with highest number

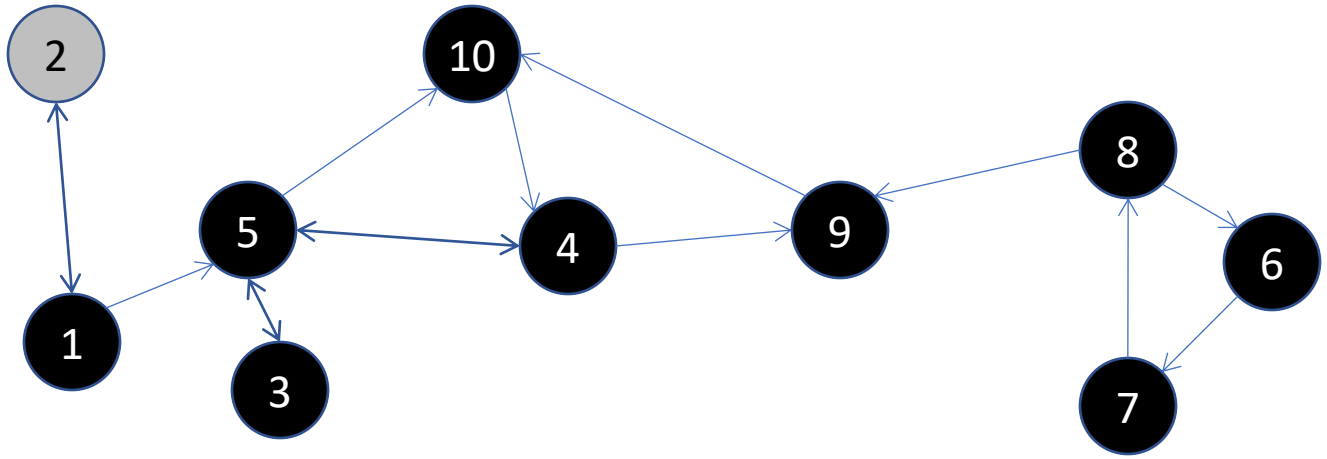
Phase 2: DFS on G



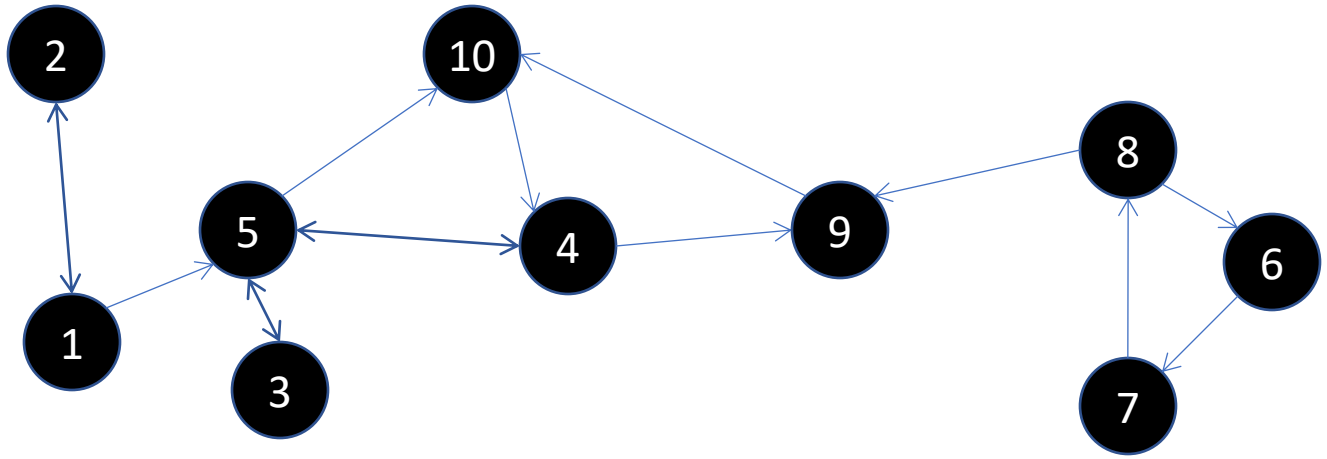
Phase 2: DFS on G



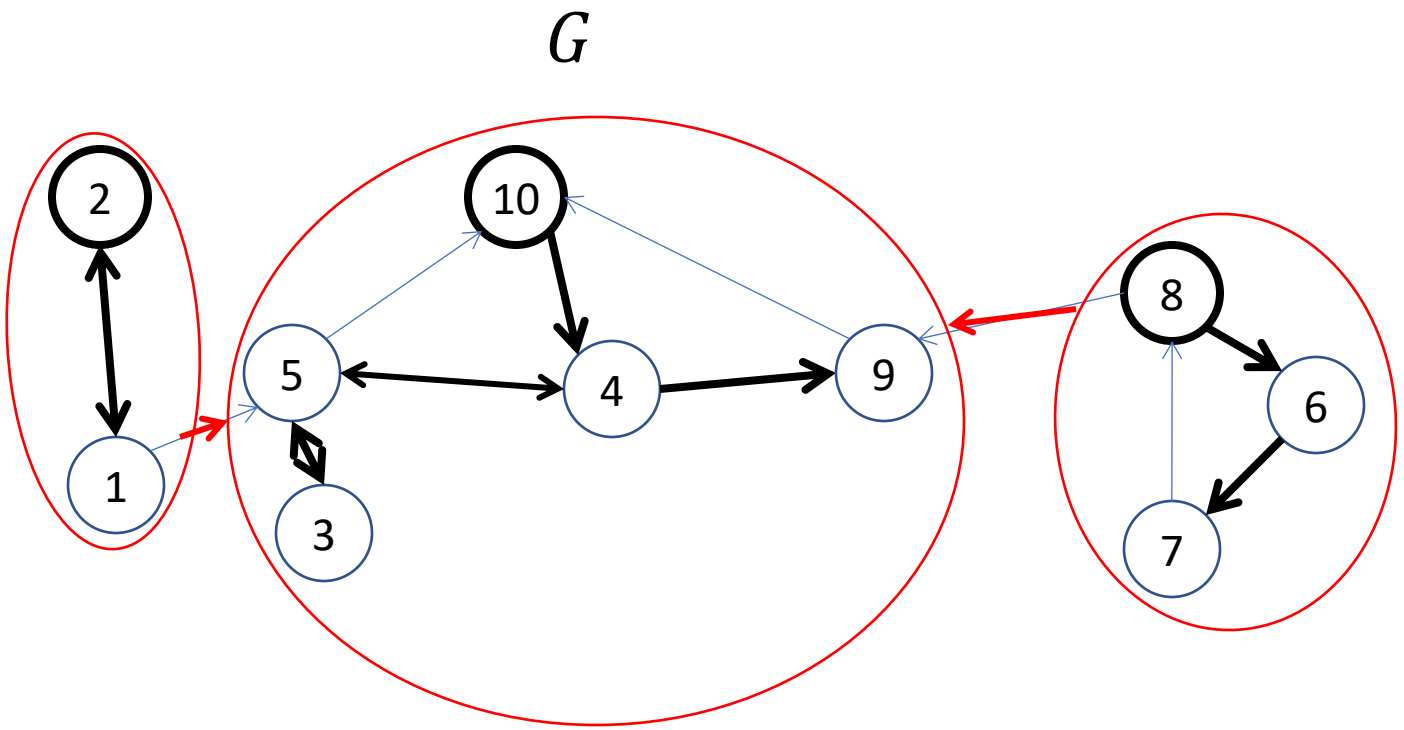
Phase 2: DFS on G



Phase 2: DFS on G



Strongly Connected Components of G



Trees in thick black with bold roots

Graph and Digraph Connectivity

	(Undirected) Graph	Di-graph
Connectivity	An undirected graph G is connected if for each pair of vertices $u, v \in V(G)$, there is a path between them.	Strongly connected - for each pair of vertices are mutually reachable Weakly connected - if its underlying graph is connected.
Components	Components - the maximal induced connected subgraphs.	Strong components - the maximal sub-digraphs induced by mutually reachable nodes
Finding components	BFS / DFS	Phase1: DFS on Gr Phase2: DFS on G

THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

SUMMARY

- Terminology
  - Connected Components (CCs) in a Graph
  - Strongly Connected Components (SCCs) in a Digraph
- Graph and Digraph Connectivity
- Finding the SCCs in a Digraph