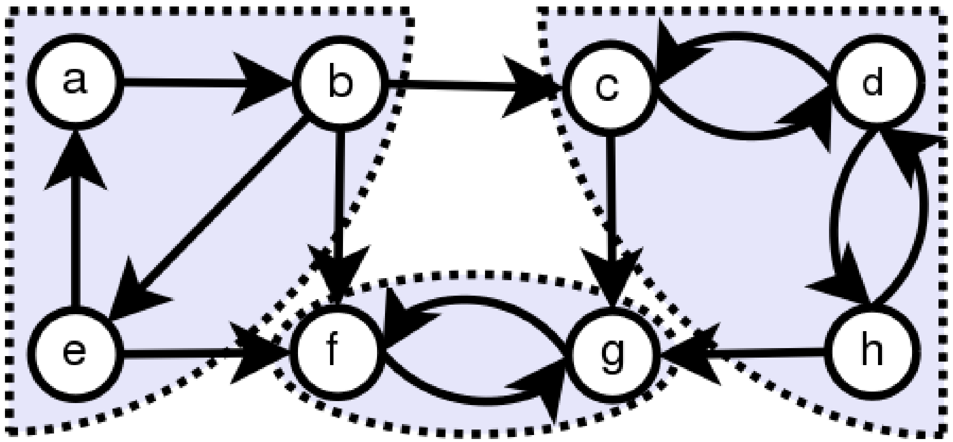
Strongly Connected Components



PROJECT WORK BY: ALGORITHM TEAM

NURDAULET DAUDOV

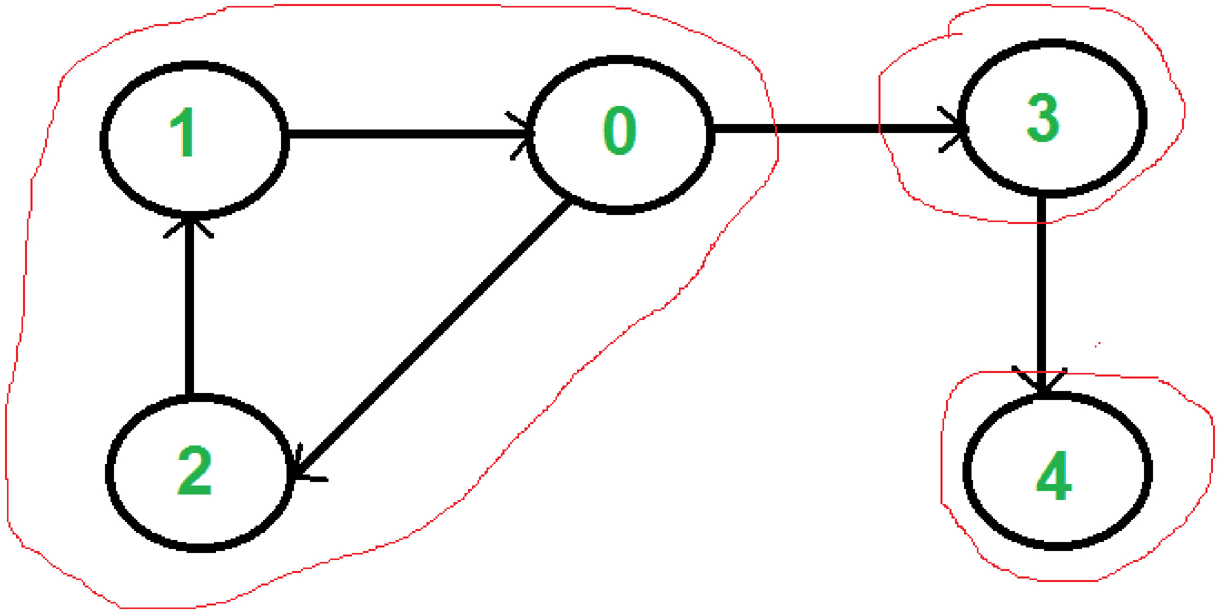
MEDET KURGANBAYEV

ALI TULEBERGENOV

ASKAROV ADLET

ZHOMART ABDRAKHMAN

A directed graph is strongly connected if there is a path between all pairs of vertices. A strongly connected component (**SCC**) of a directed graph is a maximal strongly connected subgraph. For example, there are 3 SCCs in the following graph.



Kosaraju’s and Tarjan’s algorithm

We can find our SSC in 2 ways, using kosaraju’s or tarjan’s
Kosaraju’s is based on DFS, it does DFS two times. The basic concept of this algorithm is that if we are able to arrive at vertex **v** initially starting from vertex **u**, then we should be able to arrive at vertex **u** starting from vertex **v**, and if this is the situation, we can say and conclude that vertices **u** and **v** are strongly connected, and they are in the strongly connected sub-graph.

Tarjan’s is based on DFS too, but It does only one DFS. Tarjan’s is based on following facts :
1. DFS search produces a DFS tree/forest
2. Strongly Connected Components form subtrees of the DFS tree.
3. If we can find the head of such subtrees, we can print/store all the nodes in that subtree (including head) and that will be one SCC.
4. There is no back edge from one SCC to another (There can be cross edges, but cross edges will not be used while processing the graph).

Both of them are in linear time complexity.

Tarjan’s and Kosaraju’s Complexity

Time Complexity: The Tarjan procedure is called once for each node; the forall statement considers each edge at most once. The algorithm's running time is therefore linear in the number of edges and nodes in G, i.e. O (| V | + | E |).

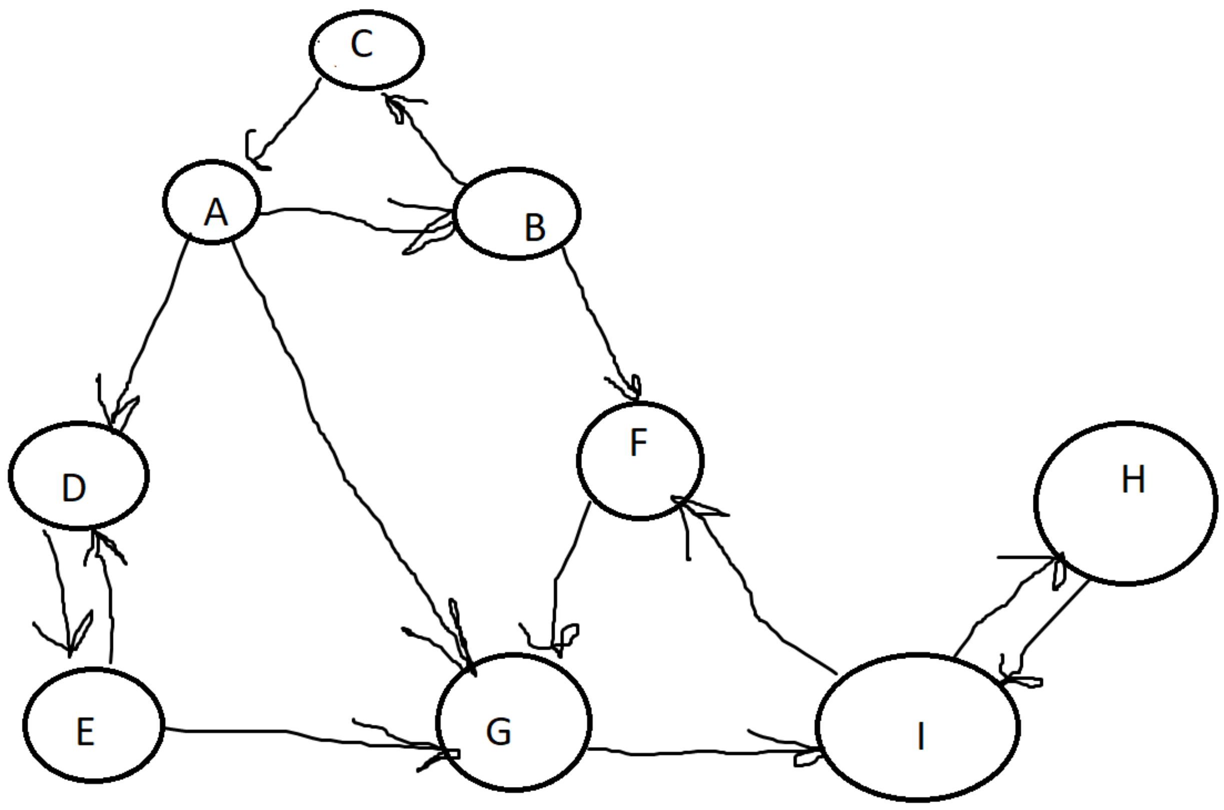
Space Complexity: The Tarjan procedure requires two words of supplementary data per vertex for the index and lowlink fields, along with one bit for onStack and another for determining when index is undefined. In addition, one word is required on each stack frame to hold v and another for the current position in the edge list

----

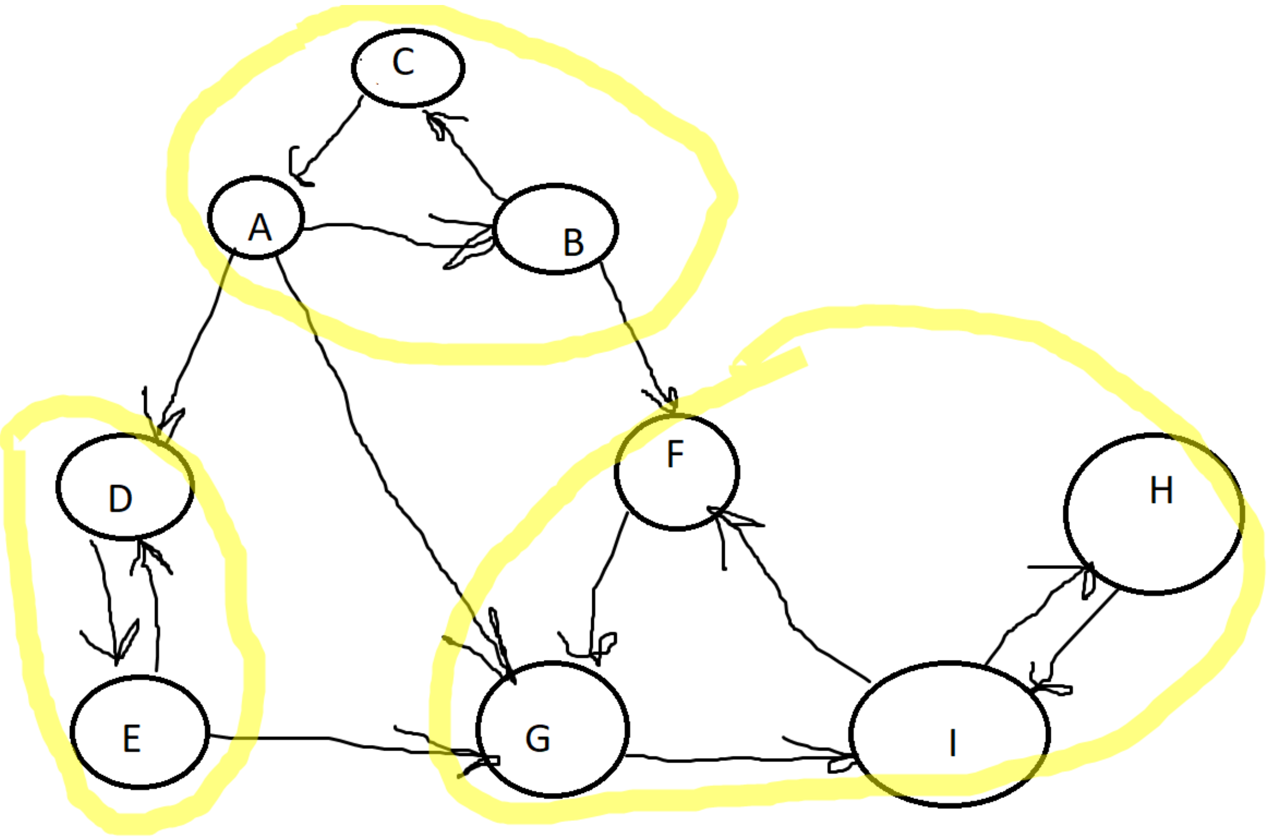
Provided the graph is described using an adjacency list, Kosaraju's algorithm performs two complete traversals of the graph and so runs in Θ(V+E) (linear) time, which is asymptotically optimal because there is a matching lower bound (any algorithm must examine all vertices and edges). It is the conceptually simplest efficient algorithm, but is not as efficient in practice as Tarjan's strongly connected components algorithm and the path-based strong component algorithm, which perform only one traversal of the graph.

If the graph is represented as an adjacency matrix, the algorithm requires $O(V^2)$ time.

Here is graph, lets find strongly connected components



Solution



Usage of SSC

For example:

The use of strongly connected components that one could use it to find groups of people who are more closely related in a huge set of data. Think of facebook or any social network, how they recommend people that might be your friends...