Course: Introduction to Graph Algorithms

Professor: C Pandu Rangan

Department: Computer Science and Engineering

Institute: IISc

Week: 08

Lecture 33 Strong Connected Components

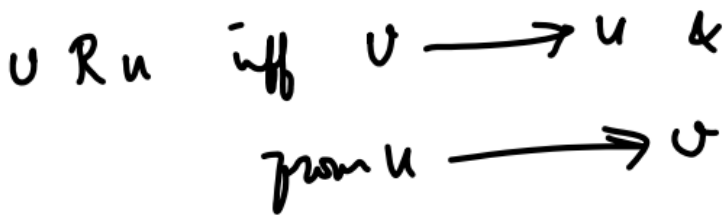
Namaskara, in this session we will begin our discussions on finding strongly connected components of a directed graph. So given a directed graph G directed graph G equal to V comma E,

$$G = V, E$$

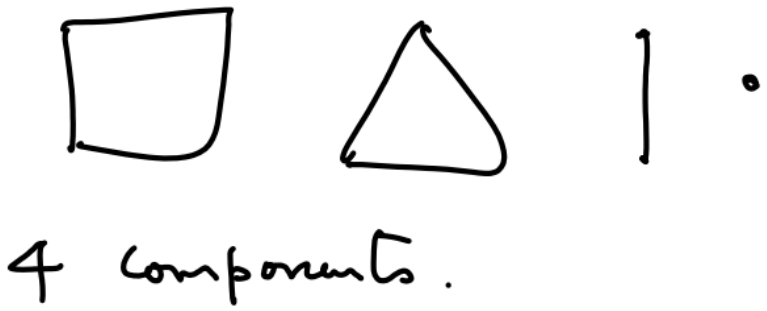
we say the graph is strongly connected okay. G is said to be strongly connected if and only if, definition for every pair of vertices uv, ordered pair of vertices uv, there is a path from u to v. This means for the path for the pair vu there is a path from v to u right. Therefore this implies for every pair of vertices vu there is a path from v to u and from u to v. okay because consider the ordered pair vu, the definition implies that there must be a path from v to u. So there must be a path from u to v and there must be a path from v to u and if this is the case for every pair then the graph is said to be strongly connected.

This is a kind of a generalization of reachability in an undirected graph. in an undirected graph a graph is to be connected if there is a path between every pair of vertices. Generalizes to directed path because we are working in a directed graph. If there is a path from u to v there is automatically a path from v to u in undirected graph that may not be the case in directed case.

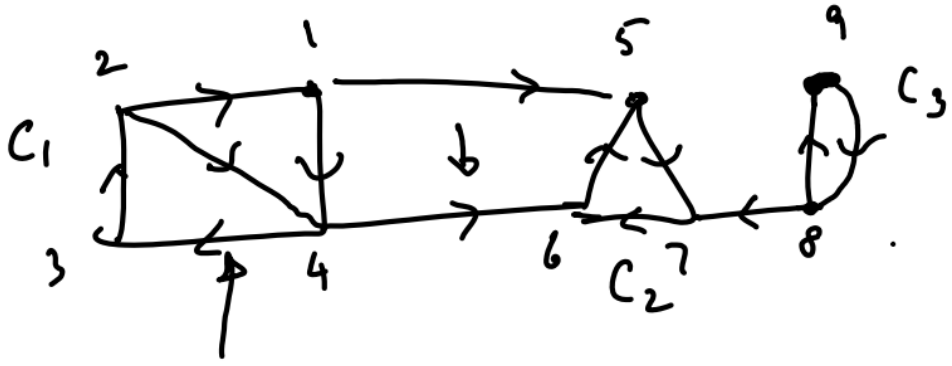
There may be a path from v to u in the opposite direction there may not be a path at all. Therefore the definition insists that in both directions we want then only it is called strongly connected. In the undirected graph if the graph is not connected it is broken into disconnected pieces and each one is called the connected component. The same thing is going to happen here, it is a minor difference, we will see what the difference is. So we use this definition to define a relation okay, a vertex v is related to u okay if and only if there is a path from v to u and from u to v.



This relation is an equivalence relation, right because it is an equivalence relation it is going to induce a partition, this relation induces a partition on the vertex set. So each such partition is called a strongly connected component, okay. In undirected graph connected components are not only vertex disjoint there is also no connection whatsoever between the components no edge. So, for example this is a undirected graph, it is a disconnected graph and this has got 4 components.



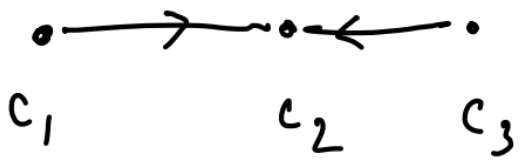
This has got 4 components, you can see that there is no edge between the different components alright. However, in the directed graph there could be edges between 2 strongly connected components but those edges will be one way kind of an edge. If that is a two way kind of an edge that is if there is one edge and then the whole thing will become a bigger connected component okay. For example, this is a directed graph and in this directed graph you have 3 strongly connected component.



This is one strongly connected component between every pair of vertices you have a directed here also the case everywhere, however you cannot go from here to here, there is no way you can reach, sorry from here to here you can from the other way. Let us say

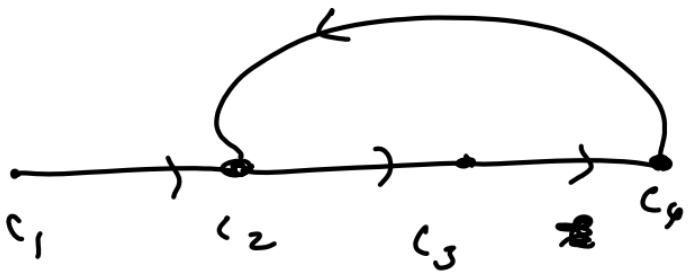
if this vertex is 1, 2, 3, 4, 5, 6, 7, 8, 9, you cannot go from, no path from 5 to 1. 5 and 1 are not related, they are in different components, from 1 you can go to 5 but from 5 you cannot go to 1, therefore they are not related okay that is because this link is a one way kind of a link. It this does not allow you to go on both directions that is the reason why this is one strongly connected component, this is another strongly connected component and this is yet another strongly connected component. This particular example has got 3 strongly connected components. There is an interesting way to look at this whole graph under connected components which is allowing us to work out algorithms and prove properties and so on okay. Let us condense each connected component to a single vertex right assume that all these vertices have been merged to a single vertex that is called the condensation. If this is C1, C2, all vertices in C1 is condensed to a vertex called C1.

All vertices 6, 7, 5, 6, 7 they are condensed to C2, 8 and 9 that is condensed to C3. From C1 there is an edge to C2, from C3 there is an edge to C2.



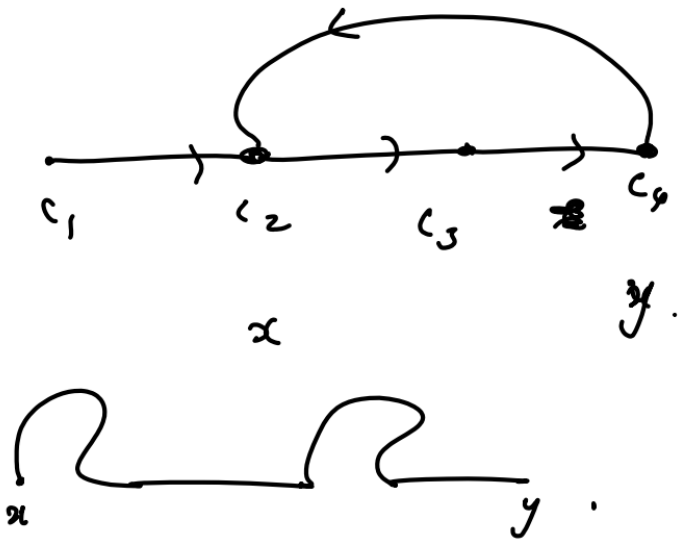
So each condensation compresses one connected component by to a vertex and then these vertices would be connected, if there are links between the vertices of different component. There could be more than one connection, for example there might be an edge even like this, multiple edges might be there, so from C1 you can go to C2, but from C2 you can never come back to C1. Look at the edges which are here, so within a component you will be able to go back and forth, but between two different components you have only a one way flow possible.

Because it is one way, right, it is going to be a directed graph it will be an acyclic graph right why it is an acyclic graph, if there is a cycle okay that would mean all the vertices are mutually reachable in each of this component in fact they would form a giant component okay. Consider for example C1 to C2 to C3 to C4 and C4, there is an edge like this. C1, C2, C3, C4 are individual one.



Suppose you draw the condensation graph. In the condensation graph, if there were a cycle. Now you can see that all vertices in C1, C2, C3 form one big connected component okay.

So take any two vertices in let us say one in C2 and one in C4, right take x is in this and the xy in this is there a path from x to y of course there is a path because from x you go to the vertex which is providing the bridge. from there use the bridge and go to C3 again in C3 you can visit everywhere go to that vertex which is providing the bridge. So in this way your path could be from x for a while in this, then in this for a while, you have a path x to y and y to x is same it is in the reverse direction.



So you can see that you have a path from x to y or from y to x. Therefore all these things will form one big connected component C1, I mean C2 won't be a separate component C3 won't be a separate component and so on that is the reason why the condensed graph is acyclic okay.

Directed acyclic graph are called DAG. We have already seen that DAGs have got some interesting properties and so on. So this condensed graph will also, this is the original graph and this is the condensed graph. Condensed graph is a DAG, directed acyclic graph okay. How are you going to find the connected components? First of all the whole graph is strongly connected okay.

If the whole graph is strongly connected, the whole thing is in one piece. So how do I know whether the whole graph is a single connected component or not? If I apply the definition then, that will call for enormous number of checks. For example, I want to know whether there is a path from v to E, you take two vertices u, v you belong to start depth first search from u if there is a path from u to v surely v is reachable and if v is reachable then DFS will reach v, the DFS will reach v because there is a path okay. You

have started from a vertex you wanted to go to v if it is done then you are happy. If v is not reachable you have taken a vertex v and another vertex v and then if v is not reachable then the graph is not strongly connected because for every pair there must be a path. Here is a pair for which there are no paths, so you have found out, you can declare that the graph is disconnected, the graph is not strongly connected but for the pair you have taken okay.

How do you check whether v is reachable, u is reachable from v, v is reachable from u, have checked is u is reachable from v, is there a path from v to u, start a DFS from v and check if u is reached. So there are n choose 2 pairs, for each pair you have to run DFS. So 2 times n choose 2 that is equal to n into n minus 1 times DFS is invoked to check if G is strongly connected or not.

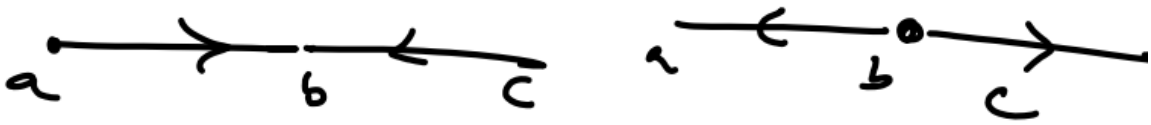
$2\binom{n}{2} = n(n-1)$ times DFS is invoked

Each DFS is going to cost n m time n plus m time, so the total cost order n square m that is because n into n minus 1 times m, is very huge, if the graph is dense, m itself is another n square, so the whole thing will become n power 4 algorithm alright. But there is a very simple way there is a new characterization.

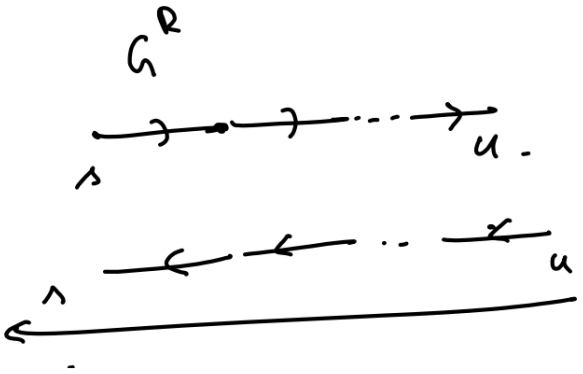
Total cost $O(n^2m)$ $n(n-1)(m)$

Let G equal to VE be a directed graph. G is strongly connected, if and only if for any fixed s belong to v for all v belong to v and v not equal to s, there is a path from s to v and v to s, so there are n minus 1 vertices, on behalf of each vertex you have to just check whether v is reachable from u or not and whether other vertices are reachable from other vertices can you reach s, okay. How to check whether there is a path from s to v for all v not equal to s perform DFS from s, this DFS will reach all vertices reachable from s. If DFS can reach all vertices then one part is proved from s you have. Now from all other vertices to S do they have a path, from other vertices to s, from s to all other vertices you have path, it is very easy to do, all you have to consider is that you do not have to check n times, there is no need to check n times.

Consider a G reverse, that is each edge is reversed its direction. So, for example a b if this is a directed graph it is reversal is going to be a b b c you reversed all the edges in the graph, that is called G reverse.

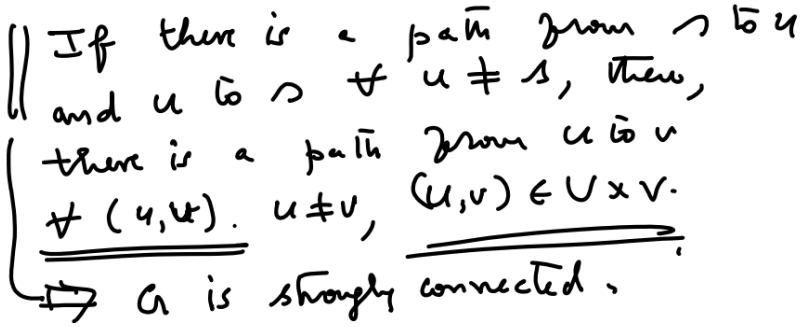


Now in G reverse, this is a notation G power R, suppose s suppose there is a path s to u this implies there is a path from u to s in the original graph, this is the reverse, u to s there is a path, this is in G reverse, in G reverse, no sorry from s to u, reverse of a reverse is the original graph.



So if you reverse it you get s there is a path from u to s. So if there is a path from s to u in G reverse then there is a path from u to s in the graph G. So I have a path from s to u and u to s if this happens for all vertices then G is strongly connected. okay so I have reduced the whole thing to 2 depth first search. My earlier solution was having n into n minus 1 DFS but now I have only 2 DFS the improvement n times n minus 1 DFS calls to 2 DFS is really interesting.

Earlier for every pair of vertices I have to do but now it is enough if I do for one vertex to other pairs, one vertex to other vertex. Now for this you do not have to keep invoking every time, just start DFS from here see whether every vertex is listed if there is something is missing that is a contradiction, that shows that this is not strongly connected. So how do you prove this it is very simple if there is a path from s to u and there is a path from u to s for all if there is a path and u to s for all u not equal to s then there is a path from u to v for all ordered pair uv, u equal to v and uv belongs to v cross v. That means if these conditions are satisfied this implies G is strongly connected.



The proof is very simple, there is a path from u to s and there is a path from s to v, they have a path from, in this there will be a path from u to v that is it.

The argument is very simple, therefore given a directed graph G equal to V, E we can determine if G is a strongly connected graph or not by performing 2 DFS one over G and another over G reverse, that is all okay. So, you can see how elegantly we are able to determine whether a graph is strongly connected or not. Suppose the graph is not strongly connected then we have to find a strongly connected components. If G is not strongly connected how do we find the strong connected components, this is the question that we have to answer.

We continue our discussions on finding the strongly connected components in the next session. Thank you.