Algorithms for Big Data
Prof. John Ebenezer Augustine
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture – 04
Karger's Mincut Algorithm

We now come to the second half of this lecture in which we move on from data structure to talk about a Randomized Algorithm. In particular we are going to look at a randomized algorithm for the Mincut problem, and this was proposed by a David Karger.
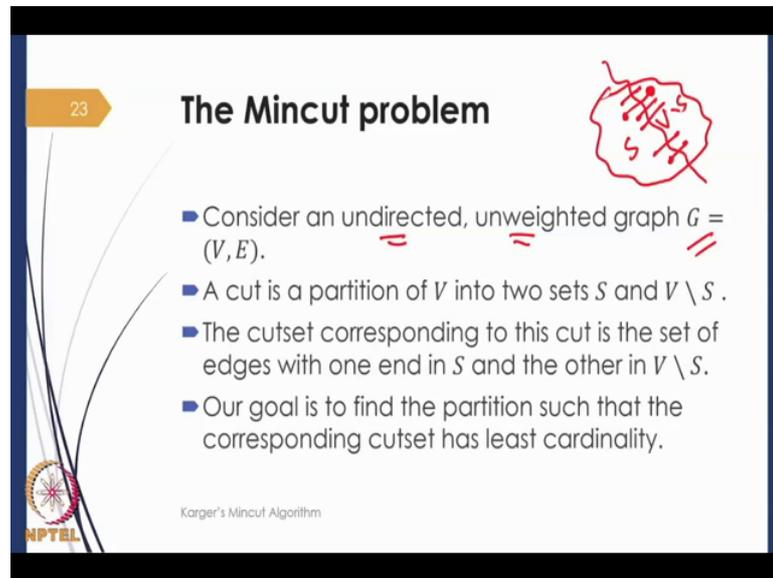
(Refer Slide Time: 00:38)

And for this very fundamental problem, let me motivate this problem first. Here, in this slide we have three graphs, and if these were network graphs and you are concerned about the robustness of the network, how easy it gets to connected things like that. Quite obviously, you will notice that this graph is very bad, this graph is ok, and this graph is good. And what makes you spot this? Because you can see in this graph that a single edge, if it is deleted can disconnect the graph into two components, and this is the nature of real world issues that motivate the mincut problems.

(Refer Slide Time: 01:47)

And chances are you have seen this problem in the context of flow based solution, but in today's lecture we are going to talk about a very simple elegant randomized algorithm for this problem. So, let me define the problem. we have input as graph g, it is an undirected graph and is unweighted graph. A cut is a simply partition of the vertex set into two sets. If this is your set of vertices a partition into set S and V minus S is a cut.

The cut set corresponding to this cut will be the set of edges with one end in S and the other end in across the cut in V minus S. So, all these edges that go across the cut comprise to the cut set. Now goal is to find the partition, the cut such that the corresponding cut set has least cardinality. And so naturally this is called the minimum cut or mincut problem.

(Refer Slide Time: 03:16)
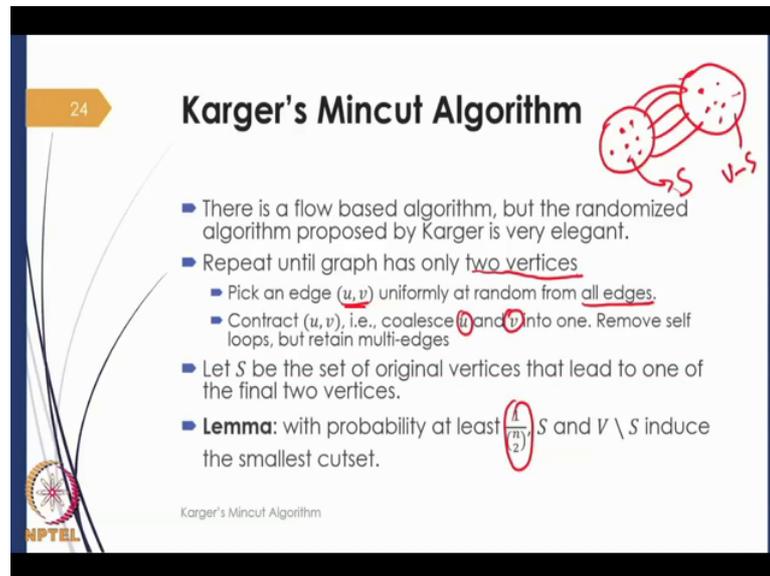
So, as I mentioned earlier you may have seen a flow based algorithm, but in today's lecture we are talking about a randomized algorithm and here is the algorithm course. We start with the full graph and we repeat these next two steps, until finally we left with just two vertices. So, each iteration, we pick an edge (u, v) uniformly at random from all the edges that are currently in the graph. And once we pick an edge, we contract that edge. And would mean by that, we coalesce the two vertices. So, the edge (u, v) is connecting vertex u and vertex v, we coalesce them into a single vertex. So, any edge between u and v, now becomes a self-loop.

So, we simply remove those self-loops, but then there could be multi edges. We will see how such multi edges could form, from an example that we will look at shortly. We will be retained multi edges loop. The self-loops we will discard, but the multi edges we retained. As we coalesce, we keep track of this super nodes and the original nodes that were actually comprise. So, each super node was formed by smaller super nodes or smaller original nodes, we keep track of all of that.

Finally, we will be left with the two vertices after n minus 2 contractions. And we will be left with two super nodes if you will, with several possible, several multi edges going across them. But each of these super nodes actually is made up of several original nodes.

So, let us pick one of these super nodes and the set of nodes that they actually are contained in this super node is simply S. Which means this would be V minus S, and this is our candidate cut.

The claim now is that with a good nonzero probability, in fact 1 by polynomial n probability, S and V minus S, actually induce smallest cut set. This seems quite magical, let us look at an example that might give us some intuition as to why this is true and in the subsequent segment, we will actually analyze it carefully.

(Refer Slide Time: 06:17)

Karger's Mincut Algorithm. Source: https://en.wikipedia.org/wiki/Karger's_algorithm
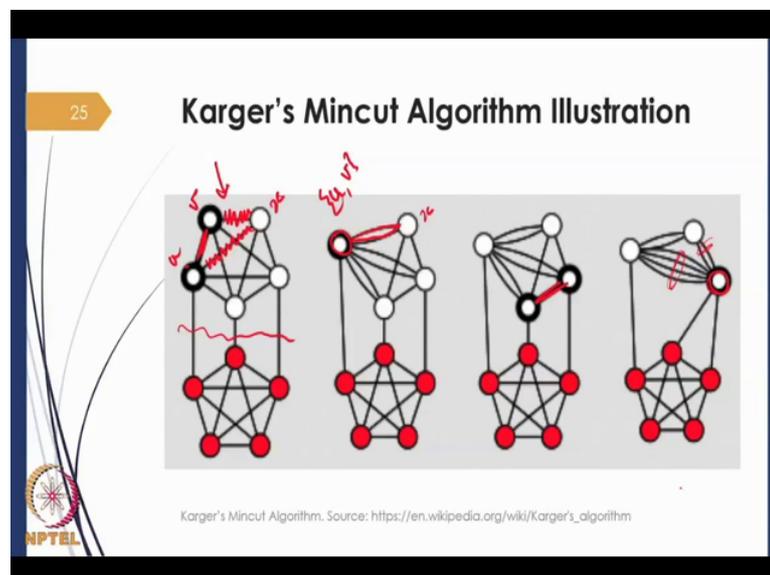
Let us look at how the algorithm operates, and this is our original graph. And as you can probably see by this eyeballing it, this is going to be their minimum cut and this execution that we are going to see now illustrated here, actually is going to discovered that particular a mincut. So, let us see how that works. There are several edges; the algorithm picks one of these edges uniformly at random. So, let's, here the edges pick this one. And it is connecting to verities v, we call it even v. Now, this edge has to be contracted and we get a super node and this super node includes both u and v.

Of course, the self-loop is thrown away, but notice that there are multi edges that are created. For example, there is an edge between v and let us say call this as x, there is also

an edge between u and x. Now between the super node that includes u and v, we have to include both of those edges, and then we proceed.

Similarly, all other multi edges also notice, are being retained. So, we now again randomly choose one of the edges and the notices that these multi edges you cannot discard them, so each one of them multi edges is also candidate for being picked. In this case let say this edge is picked. We contract those two vertices, we get this super node. And notice that we get four multi edges over here and two multi edges over here, we retain all of them and we proceed with this, in this manner.

(Refer Slide Time: 08:33)

Karger's Mincut Algorithm. Source: https://en.wikipedia.org/wiki/Karger's_algorithm
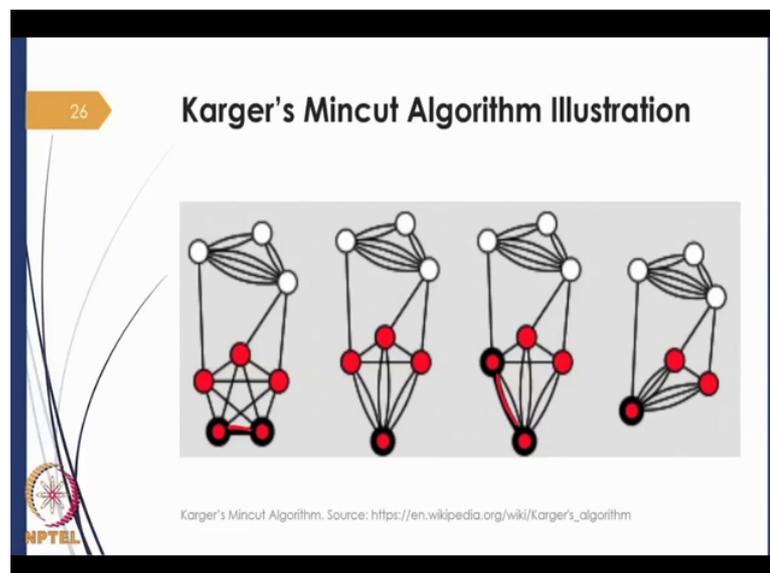
Again we, this edge uniformly at random contracted we get this new structure. And look here one of the multi edges here is being picked, again contraction takes place, self-loops are thrown away and this procedure is continued; now here we are retaining all the multi edges.

(Refer Slide Time: 09:00)
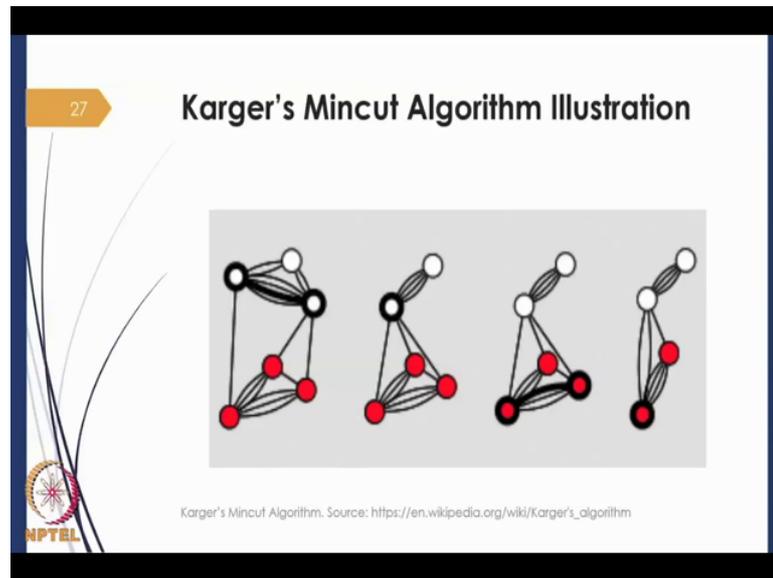
We continue on and finally we will be left with a scenario like this, where you have two vertices and a few edges going across them.

(Refer Slide Time: 09:07)
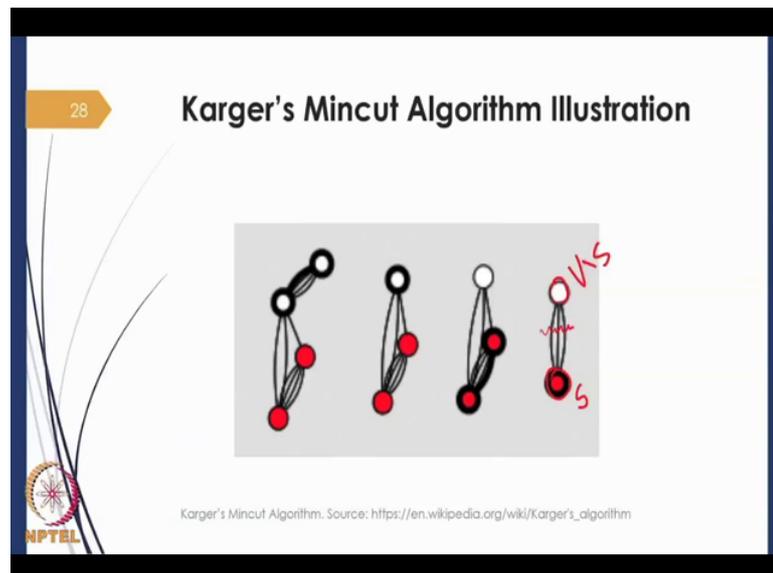
And each of these super vertices, super nodes is actually representing larger set of vertices in the original graph. So, let's, we have to keep track of them. So, let us say this

super node comprises the set S and this super node comprises the set V minus S. Our claim of course is that, so this cut induce by S and V minus S, will be a minimum cut with probability at least one over n choose two. That is the proof of this claim; we will study that in the subsequent segment.