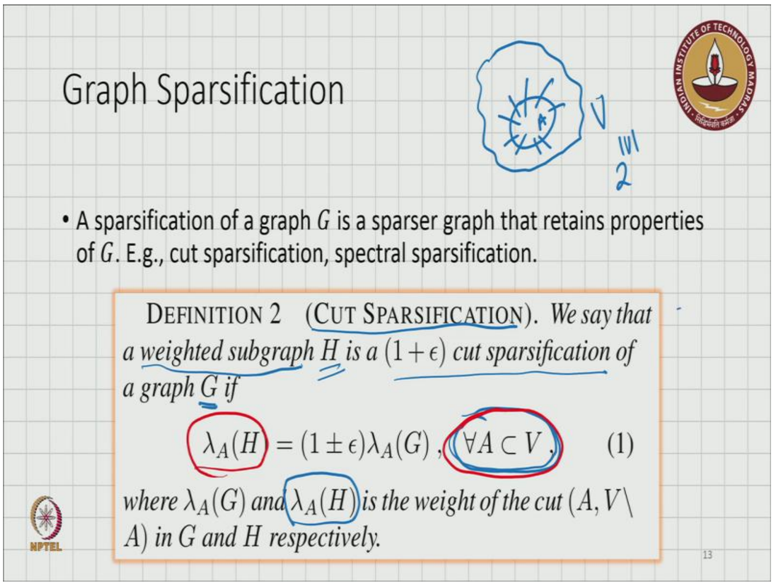
Algorithms for Big Data
Prof. John Ebenezer Augustine
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture – 46
Graph Streaming Algorithms: Graph Sparsification

Let us now look at slightly more advanced topic Graph Sparsification. This is in spirit similar to the notions like graph spanners. We want sub graph with far few a number of edges, but retaining some of the properties of the original graph and if not retaining the properties in exact value at least in an approximate sense.

(Refer Slide Time: 00:51)



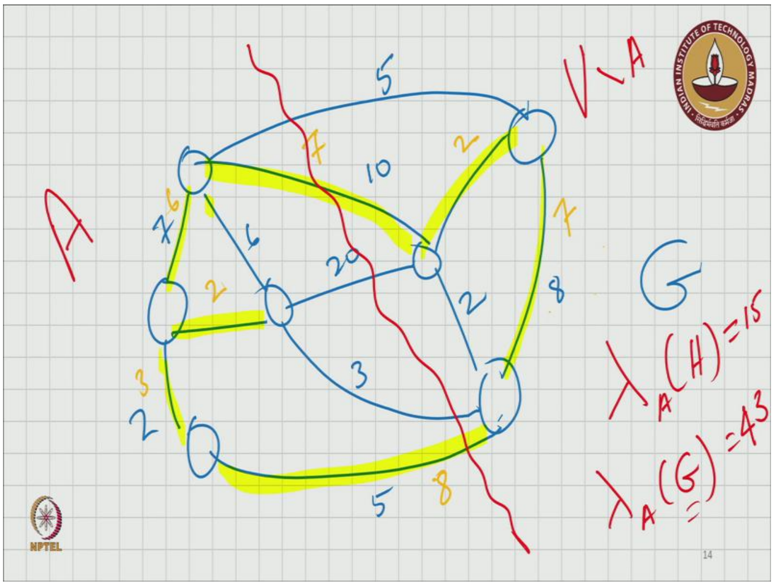
There are varieties of such sparsifications and the motivations of course, are quite obvious you have a very dense graph and you can take a lot of space, but you can sparisify the graph and still retain some of the properties. You can work with the sparsified graph which as a lot smaller foot print memory footprint and you will be able to scale you will be solve problems that you could not solve if the graph full of course, the price you will have to pay is some amount of approximation, but in most cases especially in big data applications we can leave it such approximations.

Now, let us look specifically at 1 form of sparsification called a cut sparsification and here we want to have a sparse graph that retains the cut properties of the original graph and let me also point out that we are talking about a weighted graphs and so, let say your original graph is to graph G, we say that the weighted sub graph H is a 1 plus epsilon cut sparsification of the original graph G if the following condition holds for all cuts. So, when you isolate a vertex that with as A subset of V what we are really talking about is a cut in the graph for the following reasons.

Suppose, you have a vertex set V this is the entire vertex set V when you isolate a set A what you are talking about implicitly is the cut the edge is going from a to outside A and that is the cut we talking about and for every possible cut and this this would mean that there are 2 raise to the cardinality of V number of cuts and the this property should holds for every such possible cut the lambda a of H is the weight of the cut between A and V minus a in A.

So, remember now we are actually talking about 2 graphs, G the original graph, G and the sub graph H. So, when you specify a cut in either 1 of these graphs there is an induced weight of the cut.

(Refer Slide Time: 04:05)

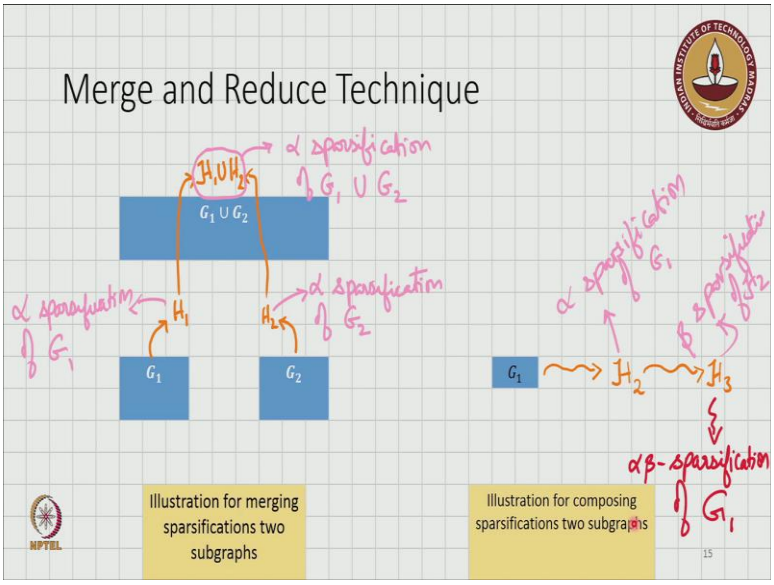


Let us look at that, now let us actually look at a specific example. So, let say we have a graph G let us say that the edge is drawn in blue are the graph G and let us highlight a few edges and will call them let say this is the graph and the highlighted edge is form the graph h. So, let us give some numbers. Let us say these are the edges the weights of the edges in the graph G the weight of the edges in H need not be the same weight as the original graph G.

Let us actually make provide numbers for that. So, let say the yellow edges have weights now let us consider cut and let us consider this cut. So, all the vertices on this side is a, and the vertices on this side are b minus a. Now, in this case let us look at. So, let see what lambda a of H and lambda a of g. These are the 2 weights of the cuts that we care about lambda a of H is the sum of the weights of the cut in the in graph h. So, that is going to be. So, in this cut that is going to be 7 and 8 that is equal to 15.

But if you look at graph G then the weight of the cut is going to be 5 plus 10, 15 plus 20 that is 35 plus 3 plus 38 plus 5 which is 43. So, you see that the 2 graphs have different weight of the weight for the same cut, but if we were to say that H is a sparsification cut sparsification in particular 1 plus epsilon cut sparsification then for any cut that we take the weight of the cut in H must be within 1 plus or minus epsilon of the weight of the cut in G and. So, what is that thus the weight of the cut in the sparsified graph should be should be in approximation of the weight of the cut in in the original graph G and this must be as i mentioned earlier true for all possible cuts. So, this is what we mean by graph sparsification.

(Refer Slide Time: 07:36)



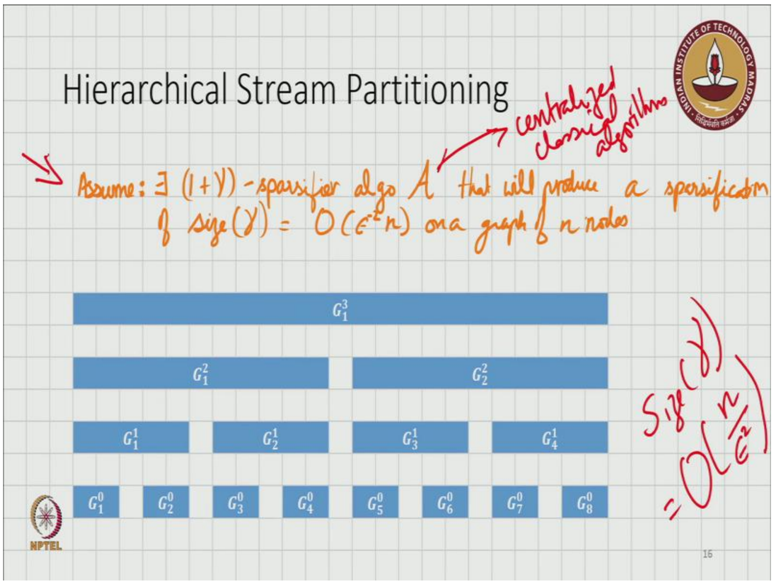
Let us look at technique to perform graph sparsification and in the streaming model now we are going to are mostly going to just talk about the idea here the details can be found in the in the survey paper on graph streaming which will be posted as well. So, let us focus in the idea here. Let say you have a way to obtain an alpha sparsification for 1 graph, say G 1 and now let say you also have a way to find an alpha sparsification for another graph G 2 and let say the 2 sparsifications are called H 1 and H 2.

Now, what you can see is that when you take the union of G 1 G 2. So, you look at the graph G 1 union G 2 you can get an alpha sparsification just by taking the union of H 1 and H 2. So, this is basically the point if you when you merge to alpha sparsifications you what you get is an alpha sparsification of the merged original graph. So, this is 1 nice property that we have which means that if we can somehow construct the alpha sparsification for G 1 and this somehow construct the alpha sparsification of G 2 and if we have H 1, H 2 then we can simply take the union of them and we will an alpha sparsification for G 1 union G 2.

This is 1 interesting property that we are going to take advantage of this second interesting property that we are going to take advantage of is composing multiple sparsifications. Let say you start with the graph G 1 and you construct an alpha sparsification of G 1 which let say in that and call that H 2 and then let say you further sparsify H 2 to get H 3 and this sparsification is a beta sparsification of H 2. So, you take H 2 you sparsified further using and you get beta sparsification of H 2 let say you that is becomes that call that H 3, now what we can say is that H 3 which is a beta sparsification of the H 2 and is actually in alpha beta sparsification of G 1. So, we lost as little bit of approximation here.

In this transformation from G 1 to H 2 that is alpha sparsification then we lost another beta factor when we got to H 3, H 3 as being an alpha beta sparsification of G 1. So, this illustrates how we can compose a sequence of sparsifications to get the original graph. So, these are 2 basic properties of sparsifications that we are going to take advantage of and now let us look at how we can do streaming.

(Refer Slide Time: 11:11)



But here in the interest of focusing of the streaming part rather actually constructing sparsifications we are going to make an assumption we are going to make an assumption this is an assumption based on the factor there are other works you know other papers talking about how such sparsifies a chemical structure.

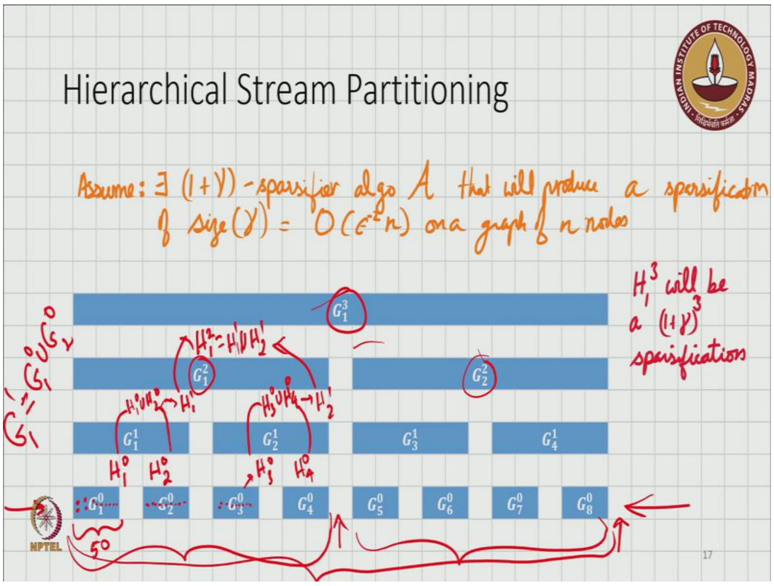
We are going to assume that there exists 1 plus gamma sparsifier algorithm a that will produce a sparsification of size gamma which is equal to O of epsilon to the negative 2 times n. So, that just to be clear we are talking about. So, size of the gamma sparsifier is equal to O of n over epsilon square on a graph of n nodes. So, this is we are going to assume that such a sparsifier exist. So, you have a way to perform sparsification, but the problem is you cannot simply apply this because you can only apply this in the streaming we were limited to the streaming mode this is the claim this is in this assumes that the entire graph is available for this algorithm a. So, this is a centralised algorithm centralised classical algorithm will say.

So, now what we need to do is a take advantage of these 2 properties these merge property and the composability property and I should construct an algorithm for the stream. So, here is the idea what we do is stream. So, this G 1 this very first 1 block it basically represents stream and you are getting a sequence of edges and as you are getting an let say this G 1 and each 1 of these for simplicity let say is 50 edges after the first 50 edges you are able to store the 50 edges in your local memory what you do is you apply the algorithm a on those 50 edges.

So, those 50 edges will be some part of the graph and you apply and you will you will get basically you will get sparsification and let us denote that as H 1 0 and then. So, now, you have H 1 0 then you start getting the edges in G 2 superscript 0 and when you collected all the edges in G 2 superscript 0 you again run the algorithm a and you will get sparsified version which is H which we denote this data as H 2 0.

Now, when you have these 2 sub graphs this sparsified graph H 1 0 and H 2 0, we know that these 2 can be merged and we will merge and we will get H 1 0 union H 2 0. So, this will be a sparsification of G 1 1 based on what we have already seen here the merge property that we have seen before.

(Refer Slide Time: 15:38)

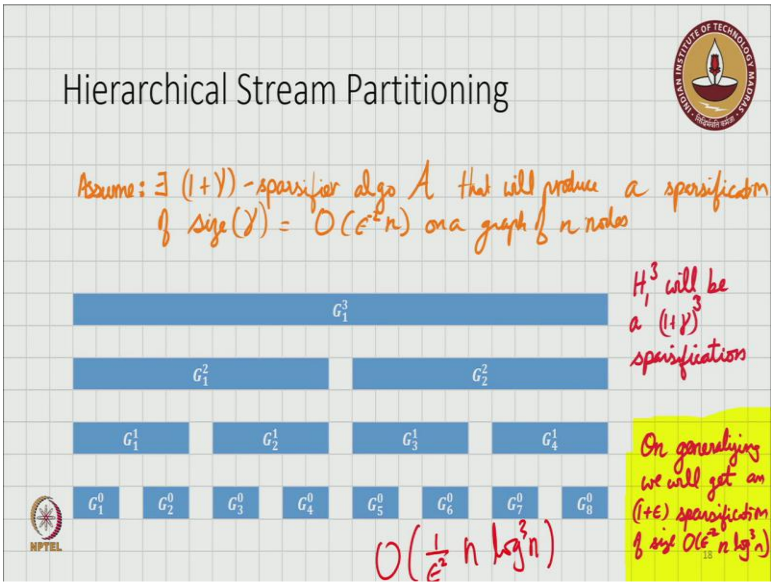


And then what we will do is we will be, looking at edges in G 3. So, then we will sparsify G 3, we will get H 3 0 then you will get 4 0 you will merge and you will get the merged sparsification of G 2 1 and that is going to be H 3 0 union H 4 0 and basically these we can call them as H 1 1 and this can be called as H 2 1 and now once we have H 1 1 and H 2 1 we can then again merge those 2 we will get H 1 2 which is nothing, but H 1 1 union H 2 1.

So, we are just repeatedly merging things and remember this H itself will be a sparsified sub graph sub graph. So, that will be able to fit into your memory the whole graph may not be fitted, but the sparsified sub graph the sparsifications will be able to fit in. So, and then we will continue. So, now, we have basically we have the sparse graph remember this G 1 1 in creates something that I forgot to mention basically G 1 1 is the G 1 1 is nothing, but the union of G 1 0 and G 2 0 and that is why we are able to take the union of those sparsifications and get the sparsification of the larger graph. So, your stream is always in this level, but for analysis purposes you can you can think of it as constructing the sparsification for larger and larger graphs.

So, similarly we will consider, at this point in time when the stream has reached this point we have basically constructed the sparsification for all of these edges which is represented by this G 1 2. So, here the sparsification continue and when we when we reach this point we would have computed this sparsification for all of these edges as well which is represented by G 2 2 and then we can put those 2 together and essentially find sparsification for the entire set which is represented by G 1 3. So, this is nice hierarchical way in which we can perform this sparsification.

(Refer Slide Time: 19:01)



And if we work out the details essentially what we get in this approach is that we will be able to get a 1 plus epsilon sparsification, where epsilon is some small positive constant and that sparsification will have size at most. If we write that out 1 over epsilon square O of 1 over epsilon square n log cubed n and the details are available in the survey, but for our purposes it will be good to understand at this conceptual level.