Course: Introduction to Graph Algorithms

Professor: C Pandu Rangan

Department: Computer Science and Engineering

Institute: IISc

Week: 06

Lecture 23 Prims Algorithm

Namaskara in this session we begin our discussions on algorithms for finding minimum spanning trees. In the past sessions we have seen the algorithms for single source shortest path problem and all pair shortest path problem. They were all working on directed graphs, all our minimum spanning tree algorithms are going to work on undirected and connected graphs. So let G be a connected undirected graph. This is edge weighted, each edge has got a weight and weight of an edge is denoted by w of e

$$w(e)$$

As usual w of e could be arbitrary it could be positive negative 0 any value we put no restriction on the weight and the weights are arbitrary real numbers for all practical purposes we can take them to be arbitrary integers.

A tree is a connected acyclic graph, the basic definition of a tree. A tree is said to be a spanning tree for a graph if vertex sets are identical you can see that V and V dash the vertex set is same.

$$V' = V$$

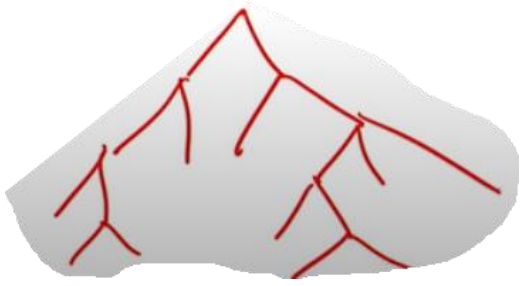
The edge set is a subset of E, vertex set is same so you are given a G let us say a, b, d e f let us say this is the graph G with the same set I am going to use a subset of edges to build a tree okay. So, for example at a I can connect with b, b with c, b with e, b with d, e with these edges this is a subset of edges therefore this is a spanning tree you can see that edges of T are edges of G, so it is a spanning tree.

Now weight of a spanning tree how do we define a weight or what is known as the cost of a spanning tree is some of the weights of the edges found in that tree, at all the edge weights you get the cost of the spanning tree, minimum cost spanning tree usual definition there may be several spanning trees we are interested in that spanning tree which has got a minimum cost. So weight of T is less than or equal to weight of T dash for any spanning tree T dash okay

$$w(T) \leq w(T')$$

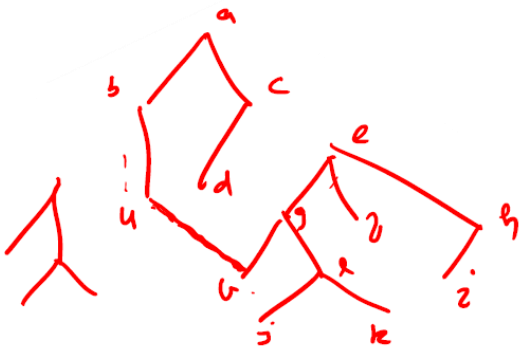
You take any spanning tree it is cost will be equal or more that means the tree that we have is minimum cost spanning tree or minimum spanning tree minimum spanning some books will use minimum cost spanning tree most books use minimum spanning tree we follow the terminology of Minimum Spanning Tree. So we have to find an algorithm efficient algorithms for finding the minimum cost spanning tree there is one interesting property of the MST if all edge weights are distinct there will be only one minimum cost spanning tree otherwise there can be several minimum cost spanning trees. If the edge weights are distinct there will be only one minimum spanning tree which is yeah only one minimum cross the spanning tree. So we assume without any loss of generality that all the edge weights are distinct.

Now we will see certain properties of spanning trees. In general a tree is a connected acyclic graph, that means it is minimally connected you remove any edge the tree breaks into pieces it becomes disconnected, and there are no cycles because there are no cycles you take any two vertices in a tree it is connected therefore there is a path and there is only one path. If you have multiple paths they may enclose a cycle, therefore between every pair of vertices there exists only one path in T, there exists a path and that will be the only one path okay. You cannot have two different paths, if they have that will enclose a cycle which is contradiction to a cyclic property, that is the reason why it is minimally connected, every tree has got n minus 1 edges where n is the number of vertices, a tree with n vertices there are several configurations of the tree but all the trees will have exactly n minus 1 edges. a tree and suppose we add a nontree edge so here is a picture.



The tree need not be binary suppose I add a non-tree edge this is u this is v let us say, between u and v there is a unique path that path plus this edge is going to form a cycle, in

general if there is a path and if you add an edge it forms a cycle right. So here is a cycle this is a cycle that is formed. I can give a label and remove this. For example, if this vertices are called a b c d e f g h i j k l something like that so you to b to a to c to e to g to v to u this is a cycle. The tree has a path and you have added an edge the path and an edge will form a cycle, unique cycle there will be only one cycle that is created.



Now in this cycle if you remove any edge again another spanning tree gets created because the cycle is broken. For example, if you remove the edge ce okay let me see whether I could do that if I remove uv is included uv is added this another spanning tree why this is a tree now you see there is no disconnection it is acyclic. there was a cycle and an edge in that cycle is removed because an edge in the cycle is removed it does not get disconnected and the graph again becomes acyclic. So this is acyclic connected graph therefore this is a tree, this is a spanning tree because it has got all the vertices. So you can take a spanning tree T add an edge e, remove an edge e dash you call that as T dash.

$$T + e - e' = T'$$

What is that edge e dash? e dash is not any edge, e dash is an edge in the cycle created by adding e to T, when you add e to T a cycle is created e dash must be one of the edges of that cycle if you remove any other edge the cycle still will remain that is not the idea. So, we need to remove an edge of the cycle any edge in that cycle can be removed, the cycle is broken the connectivity is there. So, you get T dash T dash is also a spanning tree. If T is a spanning tree the T dash constructed in this way is also a spanning tree.

Now how are the weights related it is very simple weight of T dash equal to weight of T plus weight of e minus weight of e dash you have removed an edge you have added an edge.

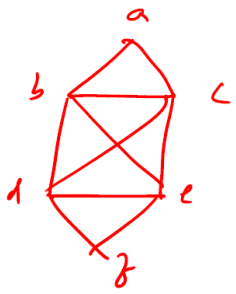
$$w(T') = w(T) + w(e) - w(e')$$

So this weight is added and this weight is removed this is how the weights are related. So from one spanning tree you can obtain another spanning tree, in this process add an edge and remove an edge in the cycle that is formed you have another spanning tree. The cost

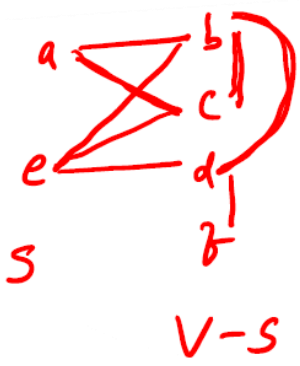
of the new spanning tree and the old spanning tree is related in this way okay. These preliminaries are good enough for us to begin our discussions on minimum spanning tree algorithm.

The first algorithm that we are going to look at is designed by Prim. So we will take a closer look at Prim's algorithm, the design of the Prim's algorithm is motivated or inspired by certain interesting properties of the edges of minimum cost spanning tree, okay. So we will take a look at the properties and then we go to the design and development of Prim's algorithm. A general property and this property allows us to build the minimum spanning tree in variety of ways okay. Therefore, this is a crucial and important property and by the way we are since the vertex set is same as the vertex set of the input graph, all we have to do is that build the edge set for the spanning tree.

Okay therefore we will focus on the edge set of the spanning tree, how do I build that edge set there are n minus 1 edges that we have to pick, how do we pick those n minus 1 edges that exit being same alright. So here is an interesting property, we define a cut as a split of the graph, a cut is a partition of the vertex set into 2 parts a set S and its complement V minus S. When you split the vertex set and if you redraw the graph putting S in one side and V minus S in another side, okay this is the original graph. Let us take again the same graph let us say a, b, c, d, e, f.



Let us put a and e in one side other vertices this is S what is V minus S, b, c d f this is V minus S, let us draw the graph now a to b there is an edge a to c there is an edge and b to c there is an edge okay b to e there is an edge. b to d there is an h and d to f there is an h, d to e there is an h, c to e there is an h. So in this way we are going to redraw the graph. putting S vertices in one side, V minus S vertices in another side.

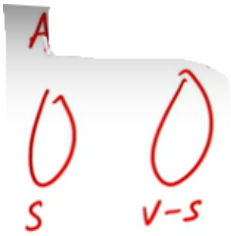


When we do like this, we get what is known as crossing edges and internal edges, bc is an internal edge, bd is an internal edge, ac is a crossing edge, why it is a crossing edge? It goes from S to V minus S, it is undirected graph, you can say that it is going from V minus S to S, we will generally say that it is between S and V minus S, all right.

Crossing edge means it will have one end vertex in S, another end vertex in V minus S. other vertices are all internal vertices. So let A be a set of edges, let A be a set of edges, we say a cut respects A if no edge of A is a crossing edge of the cut. A cut respects A if no edge of A is a crossing edge in other words edges of A are all distributed as internal edges either it will be internal to S or it may be internal to V minus S but edges of A are distributed as internal edges no crossing edge is in A, right such a cut is called respecting cut.

There can be several respecting cuts for a given A and there are cuts which may not respect A alright. So these are all basic definitions and these definitions and structural properties are useful. Let A be a set of edges of some minimum spanning tree okay. Let us assume that we have built the minimum spanning tree by identifying some of their edges, so A is the current set of edges in my hand. So A is a set of edges of some minimum spanning tree assume that S V minus S is a cut respecting and let E be a crossing edge with minimum weight okay.

So I have a set of edges A, I have a partition S V minus S.



Edges of A are all internal to S and V minus S it is a respecting cut no crossing edges in A such a cut. Of course this will have several crossing edges let e be a crossing edge with minimum weight each crossing edge has got a weight choose e to be that edge with the

minimum weight that is called the minimum weight crossing edge then A union e is a subset of edge of some minimum spanning tree. In other words I can extend A by including e in fact e is called safe edge e is called safe edge to add to A. It is safe to add you still have the property that A union e is a subset of some minimum spanning tree. that therefore here is a concrete way to extend A.

$$A \cup \{e\}$$

One concrete way is find a minimum crossing edge, minimum weight crossing edge for a cut that is respecting A then add that edge to A you have a subset of minimum spanning tree edges. A moments reflection will tell you that if I could do that n minus 1 times and keep on adding initially let me have A equal to empty

$$A = \emptyset$$

Keep adding edges I am not mindlessly adding edges I am adding edges but still I have a set of edges which are a part of some minimum spanning tree. If I could do that n minus 1 times cardinality of A will be n minus 1

$$|A| = n - 1$$

and this will be a minimum spanning tree that is it an algorithm is now possible therefore this mathematical property is very important. So based on this mathematical property, here is the template, a model for all algorithms for minimum spanning tree. We will apply this template for Prim's algorithm.

Initially A is empty, n minus 1 times I am going to add edges to A what kind of edge I add safe edges why it is safe A will still be a subset of a minimum spanning tree. A will keep growing but still it will be a part of some minimum When A becomes n minus 1 it completes a minimum spanning tree because a minimum spanning tree can have only n minus 1 edges here is a set of n minus 1 edges which are a part of a minimum spanning tree that means they define minimum spanning tree that is it. Therefore the algorithm is find a cut respecting A, find a minimum weight crossing edge e. A is A union e go back now A is updated for the updated A find a cut respecting A in the new cut find a minimum weight crossing edge, add that to A. Now A will have 2 edges again go back for the 2 edge A find a respecting cut and so on do this n minus 1 times you have the edge set of a minimum spanning tree.

$A = \emptyset$

For $i = 1 \; to \; n - 1$

Find a cut $(S, V - S)$ respecting A.

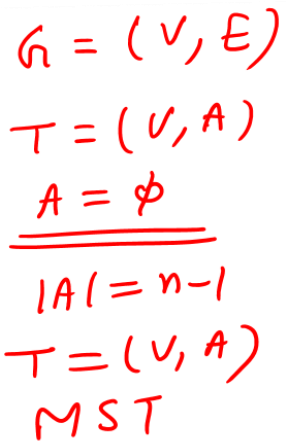
Find a minimum weight crossing edge e of the cut

$(S, V - S)$

$A = A \cup \{e\}$

Return $T = (V, A)$ \\ T is a MST of G

Okay what is the vertex set it is the vertex set of the graph that we do not have to worry about it that is the reason why minimum spanning tree algorithms will focus on building the minimum spanning tree edge by edge, vertex set is fixed okay. So, initially G equal to V E, T equals V A, A is empty set, then A gets one edge after another in n minus 1 iteration finally cardinality of A will be n minus 1 at that point T equal to VA will be a minimum spanning tree.



That is all our plan we will see how this can be implemented okay there are choices. So we have to make clever choices, all right any cut respecting A we can choose. Therefore we have to choose the cut in a clever way that construction of the cut finding the minimum weight crossing edge they all can be done efficiently. If you work with a complicated cut searching in that working with that could become very inefficient.

Therefore our focus will be on how are we going to get a respecting cut and once we have a respecting cut for A and after adding an edge to A. How are we going to find a new respecting cut for the new A? Thinking along these lines we are going to build an algorithm and finally those kind of analysis and discussions will lead to the description of Prim's algorithm. We will see the details in our next session. Thank you.