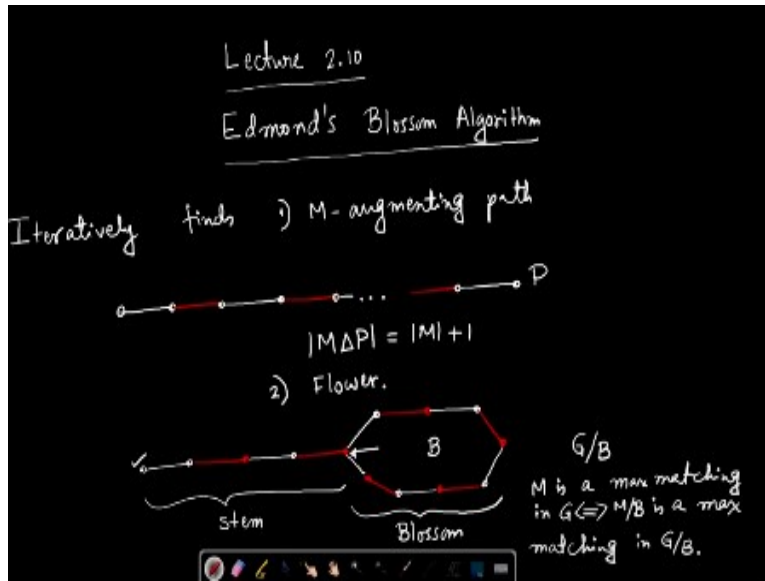


**Selected Topics in Algorithm**  
**Prof. Palash Dey**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture No # 10**  
**Module No # 02**  
**Edmond – Karp Algorithm (Contd.)**

Welcome in the last lecture we started discussing the maximum matching problem in a non-bipartite graph. And we saw the description of Edmond's Blossom algorithm so in today's lecture we will prove its correctness.

(Refer Slide Time: 00:45)



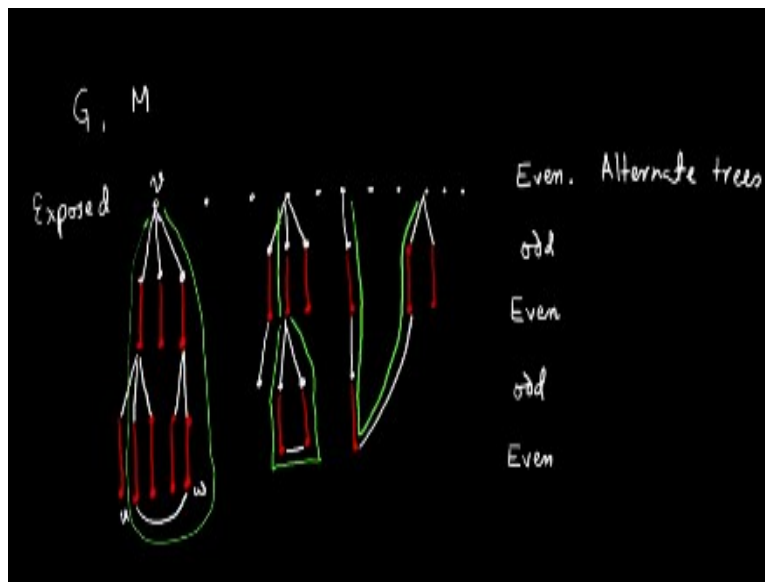
So let us briefly recall Edmond's Blossom algorithm it iteratively finds either an m augmenting path. That is how it looks like it starts with an exposed vertex; a vertex which is not matched and it is a path where matching edges and non-matching edges alternate and it starts and ends at exposed vertex. So suppose, this red edges are matching edges and so on and it ends at a, exposed vertex.

So the first and last vertex must be exposed so it is an M augmenting path and if there is such a path then we can use this part to increase the size of a matching M we can get another matching M prime. So namely what is the new matching if this path is P what we do is that M symmetric difference P is a new matching and its size is one more than the size of M. Or it finds a blossom it finds a flower.

What is a flower? It has a, what is called stem which is an alternating path and then we have an odd cycle like this the first part is called the steam of the flower and this odd cycle is called the blossom. And we take symmetric difference of stem with the current matching and that way this vertex will become exposed and the first vertex of steam which was an exposed vertex will now become matched.

And once we find a blossom we can contract this blossom so if this blossom is B instead of looking at G I look at  $G/B$ . And it is I can do because the M is a maximum matching in G if and only if M is maximum matching in G if and only if  $M/B$  is maximum matching in  $G/B$ . So the search continues in  $G/B$  that graph.

**(Refer Slide Time: 06:20)**



And now let us recall how, does the algorithm proceeds suppose I have a graph G and M is a current matching. So currently I have a set of vertices which are exposed vertices so they are all marked as even and we will be building an alternate tree. Now if there is an unlabeled so we will pick one even vertex at a time so let us call this let us pick this vertex v and if there is an edge to an unlabeled vertex all these vertices are marked as odd.

And these neighbors this matching neighbors because its partners are marked as even. So these are the roots so you see that you know this all the vertices marked even are at a, even distance from the root and all the vertices marked odd or at the odd distance. So again so I have got 3 new even vertices which are again put in the set of even vertices which will in turn will find will grow such even vertices. Find edges to unlabeled, if there are edges to unlabeled vertices then they will be marked odd.

And whenever I have an edge to unlabeled vertex they must be matched I look at their partners they are marked as even. So this way various trees are grown from each even vertex we do this processing. Now if it turns out that from an even vertex now if there is an edge to an even vertex in the same tree then we have or maybe let me put this way.

Suppose there is an edge between u and w between 2 even vertices and they are in the same tree then you see that this I have found what I am looking for I am I was looking for a blossom and I got a blossom. So in this case this is the odd cycle that I am looking for. Let us see in this example see if from an even vertex there is an edge to another even vertex of the same tree then I have got a flower. So here is the flower this is the stem and this is the blossom.

On the other hand if there is an, people if the from an even vertex there is an edge to another even vertex but of different tree then I have found what is called augmenting path so here is the augmenting path. So while constructing this alternate tree because the number of vertices is finite I will find either an m augmenting path or blossom or a flower.

Now if I get a flower I take the symmetric difference of the current matching with stem and go mod the blossom and repeat the same process. If I get an augmenting path then I augment the current matching using this augmenting path and new increase the size of the current matching by one. I get another matching of size strictly more than one more than the current matching. This way the algorithm terminates this search terminates when I find an M-augmenting path.

And we have seen that how if I find an M augmenting path in  $G/B$  then how using that I can get a, M augmenting path and in G. So the in some sense the algorithm iteratively finds an m augmenting path and augments it is a, iterative algorithm. So at the end now let us go to

correctness of the algorithm why at the end the when the algorithm terminates it outputs a maximum matching.

**(Refer Slide Time: 12:43)**

Correctness:- From  $G$  suppose we have performed several contractions (using blossoms) and conclude that  $M'$  is a maximum matching in  $G'=(V', E')$ .

Consider  $U = \text{odd}$ .

- There is no edge between any two even vertices.
- There is no edge between an even vertex and any unlabelled vertex.

Whenever we have an algorithm we need to show it is correct for all possible input it outputs the desired outcome. So from  $G$  suppose we have performed several contractions using blossoms and obtained and concluded that all right  $M$  prime is maximum matching in  $G'=(V', E')$ . Now to show we need to show that you know indeed this algorithm is correct so whenever it terminates it; it is correct.

So to show this let us consider  $u$  to be the set of vertices which are marked odd. So first observe that none of the 3 possibilities were applicable anymore in the description of Edmond's algorithm that is where it terminates. So there is so some observations there is no an edge between any 2 even vertex otherwise we could have found either a flower or an augmenting path.

Also there is no edge between even vertices this is vertices between an even vertex and any unlabeled vertex. Otherwise the first condition will apply we will mark that vertex even and so on we will proceed.

**(Refer Slide Time: 17:13)**

each even vertex is an (odd) component in  $G' \setminus \text{Odd}$ .

$$o(G' \setminus \text{Odd}) \geq |\text{Even}|.$$


$$|M'| \geq |\text{Odd}| + \frac{1}{2} (|V'| - |\text{Odd}| - |\text{Even}|)$$

$$= \frac{1}{2} (|V'| + |\text{Odd}| - |\text{Even}|)$$

$$\geq \frac{1}{2} (|V'| + |\text{Odd}| - o(G' \setminus \text{Odd}))$$

$$= \frac{1}{2} (|V'| + |U| - o(G' \setminus U))$$

$\Rightarrow M'$  is a maximum matching in  $G'$



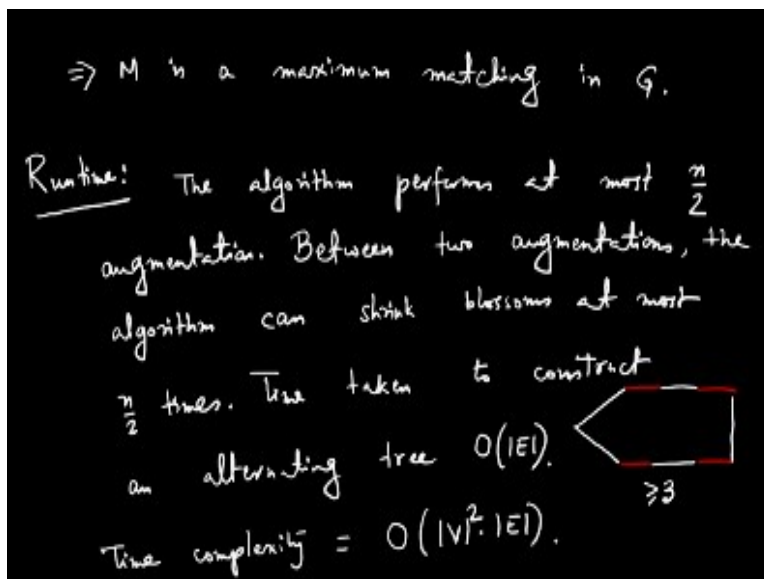
So now what we have it we have that each even vertex is an odd component namely singleton component. But so all the edges so here is this even set, here is the odd set, and here is the unlabeled. Now there could be edges within unlabeled there could be edges within odd there could be edges from even an odd of course there could be edges from odd to unlabeled. And hence if I remove these odd vertices then all even vertices become singleton.

So in particular they become odd connected component, each even vertex is an odd component in  $G'$  minus odd. So number of odd components in  $G'$  minus odd is greater than equal to number of even vertices. On the other hand what is the size of the matching  $M'$ , this is odd plus because each is odd plus let us see half cardinality  $V'$  minus odd minus even.

See that each even vertex is matched with an odd vertex each odd vertex is matched yes each odd vertex is matched with an even vertex, or some unlabeled vertex and the rest of the rest of the vertices is this. So and they are matched so size of matching  $m$  is at least this, but then what is this term this is of  $P'$  plus odd minus even.

And this is cardinality even is at is less than equal to number of odd components this is greater than equal to half  $V'$  plus odd minus number of odd components in  $G'$  minus odd. So this odd this set of odd set of vertices label or this is our vertex set  $u$  this is half  $V' + u - o(G') - u$ . And recall this is a this for every  $u$  this quantity is an upper bound on the size of maximum matching so hence  $M'$  is maximum matching in  $G'$ .

(Refer Slide Time: 22:07)



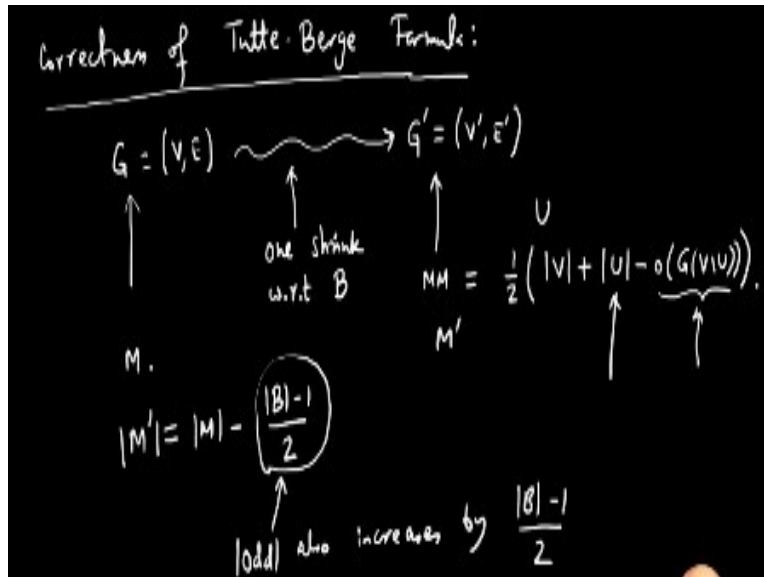
Which in turn implies that M is a maximum matching in G because earlier we proved the lemma that M is a maximum matching in G if and only if  $M/B$  is a maximum matching in  $G/B$ . Now let us see what is the running time? Running time of this algorithm; so first observe that the algorithm performs at most  $\frac{n}{2}$  augmentations. Because each augmentation increases the size of the matching by 1 and hence and the size of maximum matching could be at most  $\frac{n}{2}$ .

And between each augmentation how many times it can shrink the blossom? So between 2 augmentations it can the algorithm can shrink at most shrink blossoms at most  $\frac{n}{2}$  times. Why? Because recall blossom is a, odd cycle it looks like this so whenever I am shrinking at least 3 vertices are involved in this blossom so 3 vertices I am getting rid of and one vertex says is I am getting.

So and there are n vertices so the number of shrinks could be at most  $\frac{n}{2}$  times because each shrinking drops the size of the number of vertices by 2. Now time takes to now between one of this shrink at some either it will keep; on shrinking blossoms and it will output that the current matching is the maximum matching or at the intermediate step it will find the augmenting path M.

Augmenting path recall that will happen when from an even vertex there is another edge to an even vertex in the same alternating tree. So time taken to construct an alternating tree is order of size of E because each edge is visited at most once. So, time complexity is  $O(|V|^2|E|)$ .

(Refer Slide Time: 27:32)



Now using this we can also verify the correctness of Tutte-Berge formula. You know when Tutte-Berge again from  $G=(V, E)$  suppose the algorithm when it terminates after couple of strings it is looking at  $G'=(V', E')$  and in  $G'$  Tutte-Berge formula holds. Because I have found a matching whose size is or the, whose size matches with the, upper bound imposed by the Tutte-Berge formula.

There exist a  $u$  such that the size of maximum matching is of  $v + u$  minus number of odd components in  $G[V \setminus U]$ . So this I have to show that the Tutte-Berge formula is or applies, here also. So let us see what happens when you unshrink another so let us unshrink one at a time so for without loss of generality we can assume that there is only one shrink here. And one shrink with respect to a blossom  $B$  so if we unshrink  $B$  we get the previous graph and here we show that the Tutte-Berge formula again holds.

That means the size of matching again matches with this upper bound and if we can show for one step we apply this step for each shrinking and we get that in the initial graph the Tutte-Berge formula holds. So let  $B$  be the blossom and you see that you know this suppose this matching is  $M$  prime

and this is  $M$ . Now a size of  $M'$  is size of  $M - \frac{|B|-1}{2}$ . So size of matching from  $M'$  to  $M$  grows by this much  $\frac{|B|-1}{2}$  and you can show that this is how this much the size of number of odd sets also increase.

So the cardinality of odd set and this cardinality of offset also increases by  $\frac{|B|-1}{2}$  when I go from  $G'$ , to  $G$ . And hence both matching and this cardinality of  $U$  increases by 2 but the number of odd components which is even that remains same so the Tutte-Berge formula holds so let stop here thank you.