

Advanced Graph Theory
Dr. Rajiv Misra
Department of Computer Science and Engineering
Indian Institute of Technology, Patna

Lecture – 10
Stable Matchings and Faster Bipartite Matching

Stable matching and faster bipartite matching.

(Refer Slide Time: 00:17)

Preface

Recap of Previous Lecture:

- In the previous lecture, we have discussed Weighted Bipartite Matching, Transversal, Equality subgraph and Hungarian Algorithm.

Content of this Lecture:

- In this lecture, we will discuss Stable Matchings, Gale-Shapley Algorithm and Faster Bipartite Matching *i.e.* Hopcroft-Karp algorithm

Advanced Graph Theory Stable Matching and Faster Bipartite

Recap of previous lecture, we have discussed weighted bipartite matching, transversal equality sub graph and Hungarian algorithm. Before this particular lecture, we have also covered the maximum bipartite matching. Content of this lecture, in this lecture we will discuss stable matching, then we will take up the algorithm for it that is given by Gale and Shapley algorithm. We will discuss also advance topic on maximum cardinality bipartite matching and another algorithm for it that is called faster bipartite matching and this algorithm is also well known as Hopcroft Karp algorithm stable matchings.

(Refer Slide Time: 01:13)

Stable Matchings

- Instead of optimizing total weight for a matching, we may try to optimize preferences. Given n men and n women; we want to establish n "stable" marriages.
- If man x and woman a are paired with other partners, but x prefers a to his current partner and a prefers x to her current partner, then they might leave their current partners and switch to each other. In this situation we say that the unmatched pair (x, a) is an **unstable pair**.
- **Example:**

For a: $x > y$
For x: $a > b$
 (a, x) — unstable pair

Advanced Graph Theory Stable Matching and Faster Bipartite

So, instead of optimizing total weights for a matching we may also try to optimize using preferences. Hence the matchings which are based on the preferences optimizations are called stable matching. For example, if there are n man and n woman and we want to establish an stable marriages, then this is an ideal solution that is called stable matchings.

Now, if a man x and a woman a are paired with the other partners, but x prefers a and to his current partner and a prefers x to her current partner, then they may leave their current partners and a prefers x to her current partner, then they might leave their current partners and switch to the to each other. In this scenario we say that the unmatched pair x a is an unstable pair.

Let us take an example to understand this concept of unstable pair. So, here we are given a pair a y and b x and we say that this particular pair is unstable. If a prefers x and b prefers y , then this particular pair will become unstable or we can also say that if a prefers x and x also prefers a , then this particular pair will be unstable pair. So, that is shown here in this particular example that is for a prefers x in compare to y . Similarly for x prefers a in compare to b . So, if a pairing a y and b x is given then this is unstable pair. So, having shown what is unstable pair.

(Refer Slide Time: 03:51)

Definition

- A **perfect matching is a stable matching** if it yields **no unstable unmatched pair**.
- Example:** Given men x, y, z, w , women a, b, c, d , and preferences listed below, the matching $\{xa, yb, zd, wc\}$ is a stable matching.

x	y	z	w	Men {x, y, z, w}	Women {a, b, c, d}
•	•	•	•	✓ x: a > b > c > d	✓ a: z > x > y > w
				✓ y: a > c > b > d	✓ b: y > w > x > z
•	•	•	•	✓ z: c > d > a > b	✓ c: w > x > y > z
a	b	c	d	✓ w: c > b > a > d	✓ d: x > y > z > w

Problem setting for stable matching (SM)

Advanced Graph Theory Stable Matching and Faster Bipartite

Now, we define a stable matching, a perfect matching is a stable matching if it yields no unstable unmatched pairs.

So, unstable, unmatched pairs in the sense the matched pairs are unstable. Hence they are called unstable matched pairs; that means, the current pairs will basically prefer each other, which are not in the current pairing. So, another example which we will give you is about, let us say that we are given mens x, y, z, w , women a, b, c, d and in this particular problem setting, we are also given the preferences of the other mates, other partners for all the other partners.

For example, x will give you a preferences from for a, b, c, d , for example, here we will say that a given x gives a higher preference to a , a is greater than b , is greater than c , greater than d , similarly y also gives a preferences in this particular ranking or order that is y prefers a is greater than c , is greater than b , is greater than d , similarly z and w . On the other side also in this particular problem setting, that is women also has to give the ranking of their preferences for all the men which are listed. For example, a will give a preference to z , z is greater than x , is greater than y , greater than w , similarly b, c and d . So, this becomes a problem setting for stable matching.

(Refer Slide Time: 06:13)

Algorithm: Gale-Shapley Proposal Algorithm 3.2.17

Input: Preference rankings by each of n men and n women. $K_{n,n}$

Idea: Produce a stable matching using proposals by maintaining information about who has proposed to whom and who has rejected whom.

Iteration:

- ✓ (i) Each man proposes to the highest woman on his preference list who has not previously rejected him.
- ✓ (ii) If each woman receives exactly one proposal, stop and use the resulting matching.
- ✓ (iii) Otherwise, every woman receiving more than one proposal rejects all of them except the one that is highest on her preference list.
- ✓ (iv) Every woman receiving a proposal says "maybe" to the most attractive proposal received.

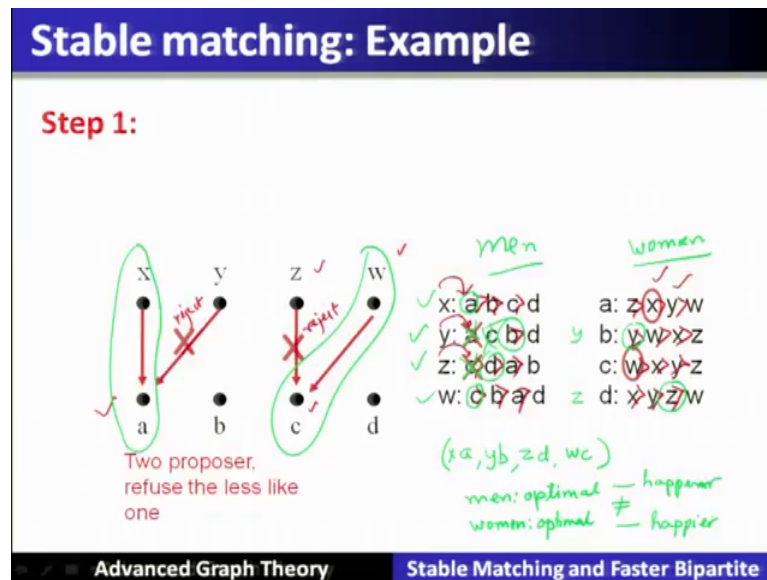
Advanced Graph Theory Stable Matching and Faster Bipartite

We will see now the algorithm which is well known as Gale Shapley proposal algorithm and which will give the stable matching as an output. So, the input of this algorithm is the preference ranking by each n men and n women. So, the graph if we see it look like $K_{n,n}$, there is a complete bipartite graph, where one partite set represents n men side and another partite set will represent n women.

Here the idea is to produce a stable matching using proposals by maintaining information about who has proposed to whom and who has rejected whom. There are two important notions, proposal and the rejection. The algorithm will iterate for each man proposes to the highest woman on his preference list, who has not previously rejected him. So, it will start from the step number 1.

Step number 2 says that if each woman receives exactly 1 proposal then it will stop and use the resulting matching. Otherwise every woman receiving more than 1 proposal will reject all of them except the one that is highest on her preference list that is a third step. Every woman receiving a proposal says maybe to the most attractive proposal received, attractive in the sense it will be on the preference list placed at the higher order.

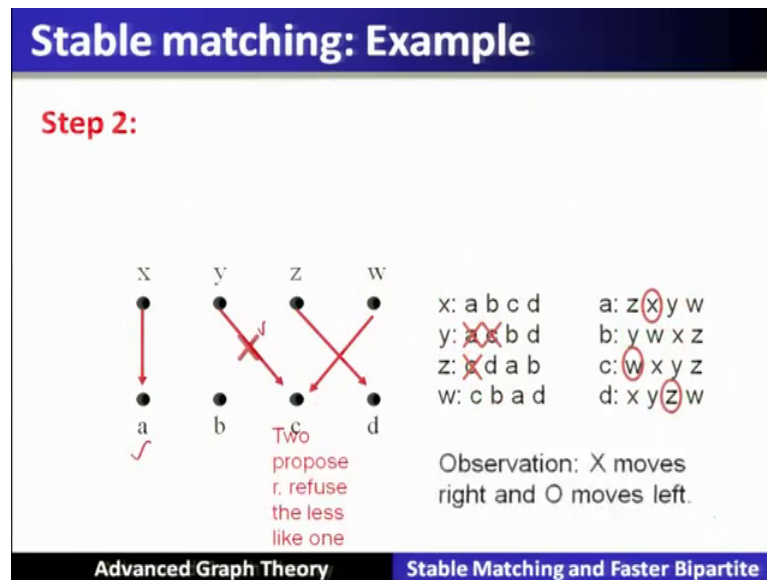
(Refer Slide Time: 08:13)



Now, all this 4 steps of this algorithm, let us trace on the same example, these preference orders are in this particular manner. Now here if we represent the first step if we see the step number 1. So, every man will make a proposal to the highest ranked woman in his preference list. So, a will send, x will send the proposal to a, similarly y will send the proposal to again a, z will send its proposal to c, and w will send the proposal to c again. So, here we can see in this particular figure that a will receive the proposal from x and y, similarly c will receive the proposal from z and w.

So, step number 2, what step number 2 will do. So, the woman a, since woman has received more than 1 proposals, she will accept the one which is at the higher preference compared to the other one. So, it will accept x and rejects y. So, y will be rejected. So, a will send the rejection and this will be the may be condition. Similarly here on the z side, on the c side. So, c will accept w because it is higher rank proposal compare to z. So, z will be rejected here in this particular case. So, what will happen after this.

(Refer Slide Time: 10:17)



So, now x and a, that is fixed. Then, another thing which is fixed is y and c. Let us workout here in this particular picture, the entire algorithm and then we will summarize it. So, here in the iteration number 1 or the step number 1 with the first iteration what we does is a as accepted x. So, this is accepted in step number 1 and also c has accepted w. So, c has accepted w and y will get a rejection. So, y will get a rejection and z will also get a rejection from c and y will get a rejection from a. Now, we will see that. So, here c and a they are being accepted.

Now, let us go to the next iteration because y and z they have got the rejections from their respective this one a and c because there is a clash in their proposals more than 1 proposals were received by a and c. So, they have to accept 1 and the reject the other proposals. So, now the next step will be that the y will send the proposal to c, y will send the proposal to c and z will send the proposal to d. Now since c will receive only 1 proposal, y will send to c, c has already c yeah.

So, y will send to c, c is already has higher preference. So, it will basically send the rejection. So, now, y will send to b, the proposal. So, b has received only 1 proposal. So, b will accept the proposal of y. So, y is also fixed and this will be fixed as. Similarly as far as d is concerned on receiving z. So, d has only 1 proposal received. So, z will be accepted here. So, d will be basically approved here in this particular scenario.

So, now with this scenario, the stable matching pairs which we will basically collect is x a, then y b, then z d, then w c. Now here let us assume that x y z w they are men and a b c d they are woman. So, what we will see here in this particular algorithm that we have started looking the men. So, this particular algorithm is men optimal. If we run the algorithm keeping woman on the left side and then men in the right side.

So, it will be men woman oriented and woman optimal algorithm will be resultant out of Gale Shapley algorithm, both are not equal in this particular problem setting. So, that is why if a if it is a man optimal. So, men this particular algorithm will make this men happier. Similarly if it is woman optimal then the women's will be made happier in this particular algorithm. So, all these steps are basically traced up.

(Refer Slide Time: 14:35)

Example Stable Matching

$M = \{(1, 4), (2, 3), (3, 2), (4, 1)\}$

1: 2 ④ 1 3	1: 2 1 ④ 3
2: ③ 1 4 2	2: 4 ③ 1 2
3: ② 3 1 4	3: 1 4 3 ②
4: 4 ① 3 2	4: 2 ① 4 3
Men's preferences	Women's preferences

Advanced Graph Theory
Stable Matching and Faster Bipartite

(Refer Slide Time: 14:47)

Example Stable Marriage Instance

1: 2 4 1 3	1: 2 1 4 3
2: 3 1 4 2	2: 4 3 1 2
3: 2 3 1 4	3: 1 4 3 2
4: 4 1 3 2	4: 2 1 4 3

Men's preferences Women's preferences

Advanced Graph Theory Stable Matching and Faster Bipartite

Now, we will see another example of a stable matching algorithm let us see that the preferences of 1 is 2 and the preference of 2 is 3, preference of 3 is 2, then preference of 4 is 4. So, here in this case 4 is receiving only 1 proposal. So, 4 will accept it the proposal from 4 and it will also be enclosed over here. Similarly as far as 3 is concerned, 3 will receive only 1 proposal that is of 2. So, from proposal to this particular pair will be finalized as far as 2 is concerned there is a conflict. So, 2 has a choice between 1 and 3. So, 3 is having higher preference. So, as far as 3 is concerned, 3 will be getting first preference and 2 is concerned, 2 will reject the proposal by 1. So, this gets rejected.

Now, in the next round. So, when 1 will send to 4, as far as 4 is concerned, 4 has already sent, may be to 4, but when it receives a higher preference, then it will basically allot or it will accept the proposal made by 1. So, this is accepted and this particular proposal from 4 is rejected. So, if it gets rejected, then again there is a possibility that 4 will now send its proposal to 1. 1 so far has not get accepted any proposal. So, after receiving from 4, it will send the accept acceptance. Hence this particular pairing is finalized by this particular algorithm. So, whatever we have worked out is also shown here in this particular final example, same things are happening.

(Refer Slide Time: 16:49)

Analysis of the Gale-Shapley Algorithm

- The algorithm terminates after no more than n^2 iterations with a stable marriage output.
- The stable matching produced by the algorithm is always **gender-optimal**: each man gets the highest rank woman on his list under any stable marriage. One can obtain the **woman-optimal** matching by making women propose to men
- A man (woman) optimal matching is unique for a given set of participant preferences
- The stable marriage problem has practical applications such as matching medical-school graduates with hospitals for residency training

Advanced Graph Theory Stable Matching and Faster Bipartite

Now, let us analyze the Gale Shapley algorithm. This algorithm terminates after no more than n^2 iterations with the stable matching as an output. So, the stable matching produced by the algorithm is always gender optimal, I have already explained this. So, each man, each man gets the highest rank woman on his list under any stable marriage, one can obtain the woman optimal matching by making woman propose to the men. A man or woman optimal matching is unique for a given set of participant preference. The stable marriage problem has a practical application such as matching the medical school graduates with the hospital for residency training and so on it has many other applications in many other fields.

(Refer Slide Time: 17:45)

Stable Marriage Formalisation

- Set n men $S_M = \{m_1, m_2, \dots, m_n\}$
- Set n women $S_W = \{w_1, w_2, \dots, w_n\}$
- Each man ranks the women in S_W in strict order of preference.
- Each woman ranks the men in S_M in strict order of preference.
- A matching M is a bijection between the men and women.
- We say a (man, woman) pair (m, w) **blocks** M if:
 - m prefers w to his partner in M , and
 - w prefers m to her partner in M .
- A matching that admits no blocking pair is said to be **stable**
 - Can't improve by making an arrangement outside the matching.
- Stable Marriage was first formalised by **David Gale** and **Lloyd Shapley** in 1962.

Advanced Graph Theory Stable Matching and Faster Bipartite

Now, let us see more details about stable marriage particular problem. Here we can see that each man ranks woman in S_W in the strict order of preferences. So, also woman ranks men in the strict order of preferences the matching m is bijection between men and woman. So, man m prefers woman to his partner in the matching m , w prefers m to her partner in matching m , a matching that admits no blocking pair said to be stable. Hence Gale Shapley stable marriage was first formalized by Gale and Shapley 1962.

(Refer Slide Time: 18:36)

Structure of Stable Marriage

- **The Gale-Shapley (GS) algorithm has two possible orientations:**
 - **man-oriented** GS (MEGS) algorithm : where the men propose to the women.
 - **women-oriented** GS (WEGS) algorithm : where the women propose to the men.
- **Optimality properties of each algorithm:**
 - MEGS algorithm - **man-optimal** stable matching M_o - simultaneously the best possible stable matching for all men.
 - WEGS algorithm - **woman-optimal** stable matching M_w - simultaneously the best possible stable matching for all women.
- Deletions that occur during an execution of either algorithm result in a set of reduced preference lists on termination of each algorithm:
 - The **MGS-lists** for the MEGS algorithm.
 - The **WGS-lists** for the WEGS algorithm.
- The intersection of the MGS-lists and the WGS-lists is called the **GS-lists**.
- The GS-lists have many important structural properties.

Advanced Graph Theory Stable Matching and Faster Bipartite

The structure of the stable marriage algorithm, here we can see that the optimality property of each algorithm, man oriented stable matching, simultaneously the best possible stable matching for all men. Similarly woman optimal also the best possible stable matching for all woman. Now, deletion that occur during the execution of either algorithm result in a set of reduced preferences list on the termination of each algorithm. Now, if you take the intersection of man oriented list and woman oriented list, then the intersection will be called as g_s list, the g_s list have many important structural properties and depending upon the application this particular properties are exploited or being utilized to solve various particular problem.

(Refer Slide Time: 19:41)

Faster Bipartite Matching: Hopcroft and Karp [1973]

- The running time of an algorithm for finding maximum matchings in bipartite graphs can be improved by seeking augmenting paths in a clever order; when short augmenting paths are available, we needn't explore many edges to find one.
- Using a **Breadth-First Search** simultaneously from all the unsaturated vertices of X , we can find many paths of the same length with one examination of the edge set.
- **Hopcroft and Karp [1973]** proved that subsequent augmentations must use longer paths, so the searches can be grouped in phases finding paths of the same lengths. They combined these ideas to show that few phases are needed, enabling maximum matchings in n -vertex bipartite graphs to be found in **$O(n^{2.5})$ time.**

Advanced Graph Theory Stable Matching and Faster Bipartite

Faster bipartite matching, faster bipartite matching algorithm is given by Hopcroft and Karp algorithm in 1973. Now, this particular algorithm will find out the maximum cardinality, faster than the algorithm which we have seen earlier in previous lecture, that is the algorithm which we have seen was called augmenting path algorithm, APA algorithm and that algorithm if you recall would take of the order n^3 time in the worst case, where n is the number of elements or the nodes in the graph in a bipartite graph [vocalized-noise].

This particular algorithm, which is basically doing the same task that is finding out the maximal cardinality bipartite matching, will find out the maximum matching a bit faster

and the total time which this particular algorithm will take to compute the maximal cardinality matching is of the order $n^2.5$.

So, hence that is why this algorithm is called a faster bipartite matching. So, this is the advanced topic. So, we are going to give you the glimpse how what basically makes this algorithm faster compare to the augmenting path algorithm. So, the running time of the algorithm for finding maximum cardinality matching in a bipartite graph can be improved by seeking the augmenting path in a clever order. When short augmenting paths are available, we need not explore many edges to find them and that will be done through the use of a breadth first search. So, breadth first search goes and explores the shorter path before it goes and explores the longer paths.

So, hence using that particular property, breadth first search simultaneously from all unsaturated vertices of a 1 partite set x will find many paths of the same length in 1 examination of the entire edge set. Hopcroft and Karp in 1973 proved that subsequent augmentations must use longer paths, using the property of the breadth first search. So, the searches can be grouped in a phases for finding paths of the same length. They combine these ideas to show that a fewer phases are needed, thus enabling the maximum matching of a bipartite graph to be found in of the order 2.5 time that is faster than the previous one.

(Refer Slide Time: 22:54)

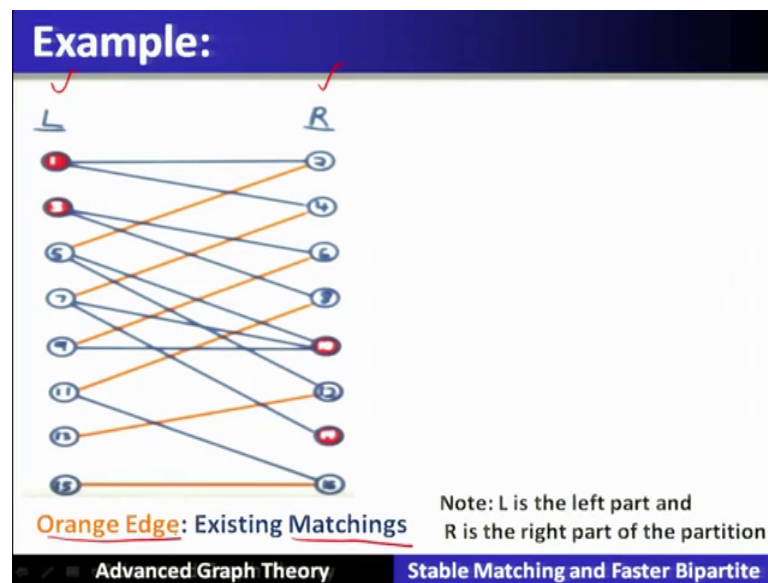
Hopcroft-Karp Algorithm

1. Initialize $M = \emptyset$ // M is matching
2. Repeat
 - Build alternating level graph rooted at unmatched vertices in L using breadth first search (BFS) // L is the left part of the partition and R is the right part

Advanced Graph Theory Stable Matching and Faster Bipartite

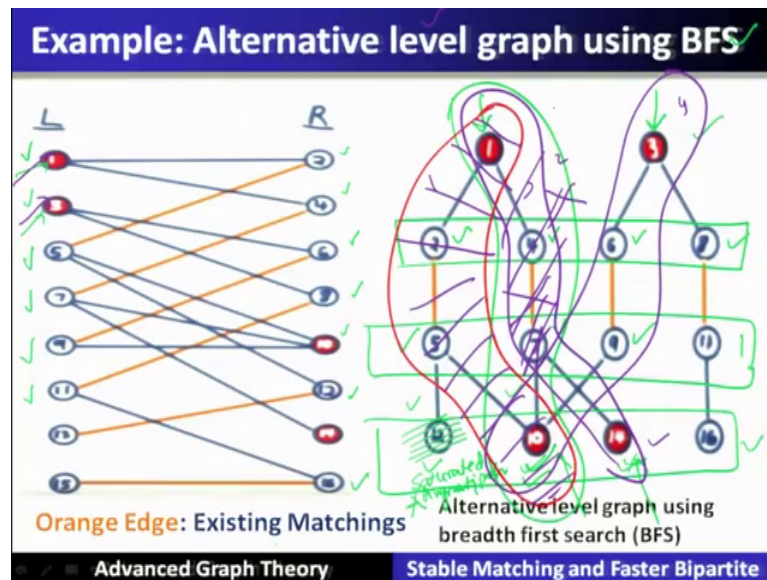
Let us see the algorithm working and we will then analyze the more details about the Hopcroft Karp algorithm. So, initialize the matching set m as empty, why because we have to begin it, then we start the iterations. First we have to build the alternating level graphed rooted at unmatched vertices in L using breadth first search.

(Refer Slide Time: 23:35)



Take this particular example. So, this particular problem setting, we will see that L is nothing, but the edges which we want to be saturating. So, let us call it as l instead of saying x , here we are little bit changing the terminologies, here we call it as l and on the other side instead of calling y , we say that it is r and let us say that we are given the matchings which are shown in the orange colors. So, matchings which we are now given, definitely this is not basically the maximal matching. So, we are basically extending this particular matching in the steps.

(Refer Slide Time: 24:17)



So, if we take the node number 1, if we take the node number 1, that node number 1 has 2 paths. One will go to 2, the other one go to 4. Another node which is unsaturated by any of these matched edges node number 3. So, 3 will go to a node number c, which is also unsaturated and node number this 8, which is also unsaturated. So, simultaneously when we invoke the algorithm that will basically invoke from both this particular unsaturated nodes. In both these nodes the BFS will run. So, after running the BFS in this particular level, 1 will reach to 2 and 4, similarly 3 will reach to 6 and 8 on the other side it will go via the unmatched edges.

Now, then what it will do? It will pick up using the matched edge and it will reach again to 1. So, 2 can take this particular edge matched edge and can reach to 5. Similarly 4 can reach to 7 using matched edge, 6 will take the matched edge and can reach to 9 and 8 can take a matched edge to reach to 11 and that too in the next level of BFS. Now, after reaching on 1, then it has to pick the unmatched edge and can reach to again to the r, to basically touch unsaturated node and that will be an augmenting path.

Let us see that 5, 5 has 2 alt options. 5 can reach to this particular node which is unsaturated. 5 can also reach to this node which is saturated. So, 5 has 2 options. So, it can reach to 12 or it can reach to 10. 12 is already saturated. So, it is useless, useless in the sense, it is not going to identify augmenting path. This will identify augmenting path in

this particular level. As far as nod 3 is concerned, when it is at position 9 it can reach to nod number 10, which is unsaturated this is nod number 10 which is unsaturated.

Similarly, nod number 11 is concerned, has only one option by unsaturated edge it can reach to 16. Now, we can see that at this particular level of inspection this is unsaturated nod, this is also unsaturated nod, you can reach this unsaturated nod via this path or you can reach this particular nod via another path, you can reach via this particular path. So, there are 3 different possible path and hence they are called alternative level graph, when this BFS is run parallely simultaneously from these 2 vertices which are unsaturated. Similarly this is also nod is unsaturated.

So, there is only 1 path possible and this can be reached from nod number 1. So, there are how many paths are there? 1, 2, 3 4, 4 different alternative paths are there, but we have to find out the vertex disjoint path. So, vertex disjoint path means if we select this particular path, let us assume then the other 2 paths cannot be basically used, why because the vertices are common. So, just see that only 1 out of them, only 1 path will be used up when we talk about the vertex disjoint paths

(Refer Slide Time: 29:11)

Hopcroft-Karp Algorithm

1. Initialize $M = \emptyset$
2. Repeat $\leftarrow O(|V|)^{1/2}$ "phases" (Handwritten: $\sqrt{n} = n^{.5}$)
 - Build alternating level graph rooted at unmatched vertices in L using breadth first search (BFS)
 - Augment M with a maximal set of vertex disjoint shortest-length paths.
3. Return M .

Handwritten notes on slide: $O(|E|)$ (next to step 2), $O(E \cdot \sqrt{n})$, $O(n^2 \cdot n^{.5})$, $O(n^{2.5})$, and $O(E)$ (under the graph).

Advanced Graph Theory | Stable Matching and Faster Bipartite

So, here we are finding up an augmenting path m with a maximal set of vertex disjoint shortest length paths. Let us see the same scenario that particular path which we have identified is also selected as the vertex disjoint shortest length path here in this particular algorithm. So, once this particular path is identified then we will extant this particular

length and we will find out the other augmenting path and so on. So, let us see the number of iterations. So, this building of the path will require order E edges. So, it has to be built through the edges. So, number of edges are to be scanned. So, hence of the order E , this particular step will take. Now, this repetition how many times it is going to repeat here that is called the phases.

So, it is proved by the Hopcroft that the total number of phases required to complete this particular matching will require only of the order n raise power or under root n that is nothing, but n raise power.5. Hence this particular approach will take 2.5, why because. So, E times multiplied by root n . So, E you know that is n square in the worst case. So, n raise power 2 times, n raise power.5, that comes out to be n raise power 2.5 is basically the time complexity analysis of this particular algorithm.

(Refer Slide Time: 31:29)

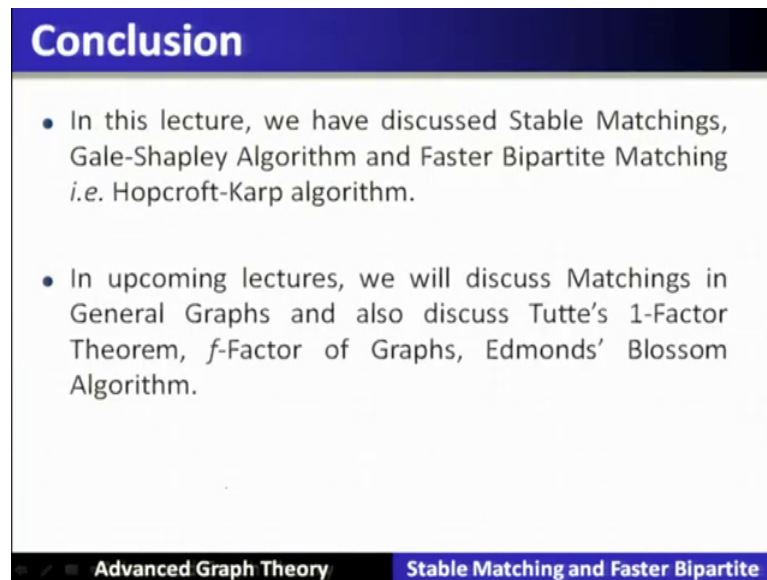
Theorem 3.2.22

- Given a bipartite graph $G=(V,E)$, Hopcroft-karp finds a maximum matching in time $O(|E| |V|^{1/2})$ with V vertices and E edges.

Advanced Graph Theory Stable Matching and Faster Bipartite

So, there is a theorem which says that given a bipartite graph Hopcroft Karp algorithm finds a maximal cardinality matching in a time of the order E times root V , with V is the set of vertices and E is the set of edges.

(Refer Slide Time: 31:50)



Conclusion

- In this lecture, we have discussed Stable Matchings, Gale-Shapley Algorithm and Faster Bipartite Matching *i.e.* Hopcroft-Karp algorithm.
- In upcoming lectures, we will discuss Matchings in General Graphs and also discuss Tutte's 1-Factor Theorem, f -Factor of Graphs, Edmonds' Blossom Algorithm.

Advanced Graph Theory Stable Matching and Faster Bipartite

Conclusion, In this lecture we have discussed stable matching given by Gale Shapley algorithm and a faster bipartite matching which is given by Hopcroft Karp algorithm. In upcoming lectures we will discuss matching in a general graphs and also discuss tuttes one factor theorem, F factor of a graph, Edmonds, blossom, matching algorithm for general graphs.

Thank you.