Second Level Algorithms

Prof. Palash Dey

Department of Computer Science and Engineering

Indian Institute of Technology, Kharagpur

Week – 08

Lecture 36

Welcome to the 36th lecture of the second-level algorithms course. In this lecture, we will see the celebrated Gale-Shapley algorithm to find a stable matching for a stable matching instance. So, let us begin. Before formally defining the algorithm, let us first understand the working principle of the algorithm using an example. So, let us start with a concrete example.

With, say, four men and four women, let's write down their preferences. So, suppose these are the preferences of men and women. It is an iterative algorithm—the Gale-Shapley algorithm. There are two versions. One is the men-proposing deferred acceptance algorithm, and the other is the women-proposing deferred acceptance algorithm.

They are symmetric with respect to men and women. So, for concreteness, let us discuss the men-proposing deferred acceptance algorithm. So, this is an iterative algorithm. The algorithm stops when we have a perfect matching. Initially, all men and all women are unmatched. In every iteration, we pick any unmatched man, and that man proposes to his most preferred woman whom he has not proposed to yet.

So, in the first iteration, suppose I pick an unmatched man M_1, and M_1 proposes to his most preferred woman, which is W_3. If she—because she is not matched and everybody prefers being matched—then W_3 accepts the proposal of M_1 and gets tentatively matched. It is not a final match but a tentative match with M_1. So, after the first iteration, M_1 and W_3 are matched. In the next iteration, we pick another unmatched man—let us take M_2—and M_2 again proposes to a woman whom he has not proposed to yet, which is W_2, his most preferred woman. Again, W_2 is unmatched, receives the proposal from M_2, and gets matched.

So, this is how the partial matching looks after two iterations of the main proposing deferred acceptance algorithm. Let us see what happens in the third iteration. The third iteration, we pick another unmatched man—let's say M_3—and M_3 again proposes to his most preferred woman whom he has not proposed to yet, which is W_2. Now, when M_3 proposes to W_2, W_2 was already tentatively matched with M_2. W_2 gets matched with the man whom she prefers more among these two, M_2 and M_3.

So, between M_2 and M_3, W_2 prefers M_3 more than M_2. So, the tentative matching of M_2 with W_2 gets revoked, and M_3 and W_2 get matched. So, this is how the partial matching looks after the third iteration of the main proposing deferred acceptance algorithm. So, in the fourth iteration, again, we will pick any unmatched man.

Let's pick M_2 again. M_2 will propose to his most preferred woman whom he has not proposed yet. So, M_2's most preferred woman is W_2, but W_2 M_2 has already proposed. So, M_2 now proposes to W_1 and W_1 was unmatched at that point. So, she accepts the proposal with ah from M_2 and M_2 and W_1 are get ah get tentatively matched.

In the next iteration we pick again an unmatched band that is only one M_4 and M_4 proposes to his most preferred woman which is W_4 and W_4 was unmatched at that point and hence she accepts the tentative proposal from M_4 and this is how the matching looks like. So now we see that these are perfect matching all men and all women are matched together. the Gale Shapley algorithm terminates at this step and outputs this perfect matching as a stable matching ok. So, we will see why it will always terminate, why when does it terminate the output is always a stable matching, but before that let formally write the Gale Shapley algorithm.

Pseudocode of main proposing deferred acceptance algorithm. Initially all men and women are unmatched that is the matching M that we will build iteratively is an empty set. Now, while M is not a perfect matching then let small m be a man who is unmatched in capital M. Let W be the most preferred woman of him whom he has not proposed yet small m makes a proposal to W if W is unmatched then M and W gets matched that is in the matching capital M we have a new pair namely M and W else if in this case W is matched but W prefers M over R current partner then from m we remove the pair capital M w comma w and add the pair

m comma w. Otherwise, w is matched and w prefers her partner under capital M than M and in this case no changes are made. And this we continue and so, the loop exit when we have a perfect matching and in this return the matching m so now we do the proof of correctness the first important question is why the algorithm will always terminate Or can we run into a case that we pick an unmatched man M, but he has proposed to all women and still not match. So, for example, is this step, let W be the most preferred woman of M whom he has not proposed yet.

Is this step well defined or not? So, these questions now we turn our attention to. So that comes under proof of correctness. The first observation is the following. In our example, we have seen that M_2 got matched with W_2, and then later it got unmatched again, then later it got matched again. So, throughout the run of the main proposing deferred acceptance algorithm, it can happen that a matched man becomes unmatched, but it can never happen that a matched woman becomes unmatched. So, that is our first crucial observation. Every woman, once matched, remains matched. Throughout the rest of the run of the algorithm.

Although she can change her tentative partner. But then this shows that this step is well-defined. So, a corollary of this observation is that If every woman or the iteration When every woman receives at least one proposal, it must be

The last iteration of the algorithm. OK, because in that iteration, every woman has received one proposal. That means every woman is matched, hence every man is matched because of our assumption that the number of men and women are the same. Hence, we have a perfect matching, and the algorithm terminates. In particular, There cannot be any unmatched man who has proposed Every woman, because that would mean that every woman has received a proposal, which would in turn mean that we have a perfect matching, and hence the particular man under consideration cannot be unmatched. Hence, the algorithm

Is well defined. By this, we mean that when we say, 'Let W be the most preferred woman of M whom he has not proposed to yet,' there indeed exists such a woman W. And because capital M is not a perfect matching, if we enter the while loop, of course, there exists a man M who is unmatched, OK. So, the algorithm is well defined, and now we will show that the algorithm always terminates—that means there is a stable matching—sorry, there is a perfect matching that is always reached. So, the next corollary

The algorithm always terminates within $n\times(n-1)+1$ iterations. Proof. If the algorithm has not terminated, I think we need a plus 1 here.

Within $n \times (n-1)+1$ iterations, then there exists at least one man by the pigeonhole principle, who has made n proposals. This implies that every woman has received at least one proposal.

However, the current matching must be a perfect matching since every woman and thus every man is matched. This however, contradicts our assumption that the algorithm has not terminated after or within $n \times (n-1)+1$ iterations ok. So, this shows that the algorithm always terminates moreover the number of iterations is at most $n \times (n-1)+1$. the number of iterations of the main proposing deferred algorithm is at most $n \times (n-1)+1$. So, the algorithm always terminates with a perfect matching in the next class we will prove why that perfect matching must be a stable matching ok. So, let us stop here. Thank you.