Notes for Recitation 7

1 A Protocol for College Admission

Next, we are going to talk about a generalization of the stable marriage problem. Recall that we have some horses and we'd like to pair them with stables so that there is no incentive for two horses to swap stables. Oh wait, that's a different problem.

The problem we're going to talk about is a generalization of the one done in lecture. In the new problem, there are N students $s_1, s_2, \ldots, s_N$ and M universities $u_1, u_2, \ldots, u_M$. University u_i has n_i slots for students, and we're guaranteed that $\sum_{i=1}^{M} n_i = N$. Each student ranks all universities (no ties) and each university ranks all students (no ties).

Design an algorithm to assign students to universities with the following properties

1. Every student is assigned to one university.
2. $\forall i$, u_i gets assigned n_i students.
3. There does not exist s_i, s_j, u_k, u_ℓ where s_i is assigned to u_k, s_j is assigned to u_ℓ, s_j prefers u_k to u_ℓ, and u_k prefers s_j to s_i.
4. It is student-optimal. This means that of all possible assignments satisfying the first three properties, the students get their top choice of university amongst these assignments.

The algorithm will be a slight modification of the mating algorithm given in lecture. For your convenience, we have provided a copy of the mating algorithm on the next page.

Each Day:

- Morning:
  - Each girl stands on her balcony
  - Each boy stands under the balcony of his favorite girl whom he has not yet crossed off his list and serenades. If there are no girls left on his list, he stays home and does 6.042 homework.
- Afternoon:
  - Girls who have at least one suitor say to their favorite from among the suitors that day: "Maybe, come back tomorrow."
  - To the others, they say "No, I will never marry you!"
- Evening:
  - Any boy who hears "No" crosses that girl off his list.

Termination Condition: If there is a day when every girl has at most one suitor, we stop and each girl marries her current suitor (if any).

Solution. Each Day:

- Morning:
  - Each university asks which students are interested in applying.
  - Each student applies to his/her favorite university that has not yet rejected him/her. If there are no universities left on the student's list, the student takes some time off to think about life, the future, and the unchangeable past.
- Afternoon:
  - Universities u_i do the following:
  - u_i tells its favorite n_i applicants "Maybe, we are still processing your application." If u_i has less than n_i applicants, it tells all of its applicants this message.
  - If u_i has more than n_i applicants, it tells the remaining ones "Sorry, there were a large number of very qualified students applying this year, yet we can only accept a very limited number. We regret to inform you that you were not accepted. Thank you for applying to our university."
- Evening:
  - Any student who hears "Sorry, ..." from some university, crosses off that university from his/her list.

Termination Condition: If there is a day when each university u_i has at most n_i applicants, we stop and each university accepts all of its applicants (if any). ■

1. Before we can say anything about our algorithm, we need to show that it terminates. Show that the algorithm terminates after $NM + 1$ days.

   Solution. On each day, if the algorithm has not terminated, then some university u_i has more than n_i applicants. It follows that in the afternoon, at least one student s_j hears "Sorry, ...", and thus in the evening s_j crosses off u_i from his/her list. As there are N students and M universities, it follows that the algorithm must terminate after $NM + 1$ days, as otherwise there would be no university left for any student to cross off. ■

2. Next, we will show that the four properties stated earlier are true of our algorithm. To start, let's show the following: if during some day a university u_j has at least n_j applicants, then when the algorithm terminates it accepts exactly n_j students.

   Solution. At this day, each of the students applying to u_j has u_j as their favorite university that has not yet rejected him/her. Therefore, if u_j tells a student "Maybe, ...", that student will come back the next day. Since there are at least n_j applicants, it follows that u_j will tell its favorite n_j "Maybe, ...". It follows by induction that every day after this day, u_j will have at least n_j applicants. Thus, this holds when the algorithm terminates. Since when the algorithm terminates there are at most n_j applicants, it follows that exactly n_j students are assigned to u_j. ■

3. Next, show that every student is assigned to one university.

   Solution. It is clear that no student can apply to more than one university at once since a student applies to at most one university on any given day, so this means the students can be assigned to at most one university. So we just need to show that each student is assigned to at least one university. We argue by contradiction.

   Suppose not, and let s_j be a student not assigned to any university. Then, since the algorithm terminates, and when the algorithm terminates each university u_i accepts at most n_i students, it follows that some university u_i accepts less than n_i students. By the previous problem, it follows that in every day, u_i had less than n_i applicants. But then consider the day that s_j applied to u_i. Since there were less than n_i applicants to u_i that day, it follows that u_i would have told s_j "Maybe, ..." in that day, and thus in every future day. Thus, s_j would be assigned u_i when the algorithm terminates. This is a contradiction. ■

4. Next, show that for all i, u_i gets assigned n_i students.

   Solution. Since the algorithm terminates, on some day each u_i gets assigned at most n_i students. Suppose some u_i got assigned strictly less than n_i students. Since $\sum_{i=1}^{M} u_i = N$, this means that some student is not assigned. This contradicts the previous problem. ■

5. Before continuing, we need to establish the following property. Suppose that on some day a university u_j has at least n_j applicants. Define the *rank* of an applicant s_i with respect to a university u_j as s_i's location on u_j's preference list. So, for example, u_j's favorite student has rank 1. Show that the rank of u_j's least favorite applicant that it says "Maybe, ..." to cannot decrease (e.g., going from 1000 to 1005 is decreasing) on any future day. Note that u_j's least favorite applicant might change from one day to the next.

   Solution. On the next day, there are two cases: u_j either says "Maybe, ..." to its least favorite applicant s_i from the previous day, or it says "Sorry, ..." to s_i. In the first case, this means that all of the $n_j - 1$ applicants u_j liked more than s_i on the previous day will also be told "Maybe, ...", and so s_i will again be u_j's least favorite applicant it did not reject. Thus, the rank of its least favorite applicant did not decrease. In the second case, this means that there were at least n_j applicants that u_j preferred to s_i, and thus the rank of its least favorite applicant it said "Maybe, ..." to did not decrease. As shown above, on any future day u_j has at least n_j applicants, and so by applying this analysis again, we conclude that the rank of u_j's least favorite applicant that it says "Maybe, ..." to cannot decrease on any future day. ■

6. Next, show there does not exist s_i, s_j, u_k, and u_ℓ where s_i is assigned to u_k, s_j is assigned to u_ℓ, s_j prefers u_k to u_ℓ, and u_k prefers s_j to s_i. Note that this is analogous to a "rogue couple" considered in lecture.

Solution. Assume, towards a contradiction, that such an s_i, s_j, u_k and u_ℓ existed. Since s_j prefers u_k to u_ℓ, but is assigned to u_ℓ, on some day u_k told s_j "Sorry, ...". On that day, there must have been more than n_k applicants to u_k. If s_i was also an applicant to u_k on that day, then s_i would have also been rejected since u_k prefers s_j to s_i, and thus s_i could not have been assigned to u_k. On the other hand, if at any later day s_i were to apply to u_k, it would have been rejected since s_i's rank is less than s_j's with respect to u_k, since by the previous problem we know that the rank of the least favorite applicant that u_k says "Maybe, ..." to, cannot decrease. Thus, it is impossible for s_i to be assigned to u_k, which is a contradiction. ■

7. Finally, we show in a very precise sense that this algorithm is *student-optimal*. As in lecture, define the *realm of possibility* of a student to be the set of all universities u, for which there exists some assignment satisfying the first three properties above, in which the student is assigned to u. Of all universities in the realm of possibility of a student we say that the student's favorite is *optimal* for that student.

   Show that each student is assigned to its optimal university.

   Solution. We argue by contradiction. Consider the first (in time) student s_i not assigned to its optimal university, and suppose for s_i this university is u_k. Then on the date s_i is rejected from u_k, there was another student s_j which u_k preferred to s_i. Since u_k is in the realm of possibility of s_i, there is an assignment M of students to universities assigning s_i to u_k with the properties above. Suppose M assigns s_j to u_ℓ. Suppose u_{opt} is s_j's optimal university. Then, since s_i was the first student not assigned its optimal university, s_j prefers u_k to u_{opt}, though u_k may equal u_{opt}. On the other hand, s_j prefers u_{opt} to u_ℓ, since u_{opt} is its favorite university in its realm of possibility, and u_ℓ occurs in its realm of possibility. It follows that s_j prefers u_k to u_ℓ. But then in the assignment M we have found an s_i, s_j, u_k, and u_ℓ with s_i assigned to u_k, s_j assigned to u_ℓ, s_j prefers u_k to u_ℓ, and u_k prefers s_j to s_i. This is a contradiction to the property of M established in the previous problem. ■

6.042J / 18.062J Mathematics for Computer Science
Fall 2010