IEOR 151 – Lecture 9
Residency Matching

1 National Resident Matching Program (NRMP)

1.1 Model

There are m residency programs and n applicants. We will denote the i-th applicant as v_i and the j-th program as p_j. Let s_i be the number of open positions in p_i, let a_i be the number of programs listed by the i-th applicant, and let b_i be the number of applicants listed by the i-th program. We will denote the preferences as a list from most to least preferred. For instance, the preferences of the v_i are given by $(p_{v_i[1]}, p_{v_i[2]}, \dots, p_{v_i[a_i]})$ and the preferences of the p_i are given by $(v_{p_i[1]}, v_{p_i[2]}, \dots, v_{p_i[b_i]})$. An important feature of this model is that preferences are unique in the sense that no program or applicant can be preferred at the same level. The problem is to match applicants to positions, so that no applicant matches to more than one position and so that preferences are maximized in an appropriate sense.

1.2 Matching Algorithm

Let $I = (1, 2, \dots, n)$ be a list, then the matching algorithm is given by the following:

1. While I is nonempty, remove the first element of I
   (a) For $j = 1 \dots a_i$, attempt to place v_i into $p_{v_i[j]}$
      - If $p_{v_i[j]}$ has an empty spot and $p_{v_i[j]}$ has v_i in its list of preferences, then denote $v_i \leftrightarrow p_{v_i[j]}$ a tentative match.
      - If $p_{v_i[j]}$ does not have an empty spot, but there is a "least-preferred" tentative match $v_k \leftrightarrow p_{v_i[j]}$ such that v_i is preferred over v_k by $p_{v_i[j]}$; then remove the tentative match $v_k \leftrightarrow p_{v_i[j]}$, add the tentative match $v_i \leftrightarrow p_{v_i[j]}$, and append k to I.
2. Once I is empty, finalize the matches.

1.3 Example

An example modified slightly from literature provided by the NRMP can help clarify the algorithm. Suppose the applicants' preferences are given by:

Anderson	**Chen**	**Davis**	**Eastman**	**Ford**
1. General	1. City	1. City	1. City	1. City
2. City	2. Mercy	2. General	2. Mercy	2. General
		3. Mercy	3. General	3. Mercy

Suppose that each program only has two open positions, and that the preferences of the programs are given by

Mercy	**City**	**General**
1. Davis	1. Anderson	1. Eastman
2. Ford	2. Chen	2. Davis
	3. Eastman	3. Anderson
	4. Davis	4. Chen
	5. Ford	

Then the final match results are given by

Mercy	**City**	**General**
Ford	Chen	Anderson
	~~Davis~~	Davis
	Eastman	

1.4 Game-Theoretic Considerations

The first thing to note is that this algorithm generates a stable matching. The intuition for why this is the case is that the algorithm tries to first match to more preferred programs/residents. The second thing to note is that this algorithm is strategy-proof for the residents, under the assumption that residents do not collude. This means that the best game-theoretic strategy for a resident is to submit their true preference list of residency programs. The intuition is again because the algorithm tries to match more preferred elements.

1.5 Optimality

It is important to note that the algorithm is not symmetric with respect to the roles of residencies and residents. For comparison, suppose we were to change the algorithm so that we switch the role of the hospitals and residents. Then, the resulting matching would be:

Mercy	**City**	**General**
~~Davis~~	Anderson	Eastman
Ford	Chen	Davis

Compare this matching to the previous result: In the previous result, the residents achieved higher preferences; in this result, the hospitals achieved higher preferences. The general result is that the group that proposes a match will achieve higher preferences than the group that accepts a match. Because the NRMP is structured so that residents "propose" a match, the residents achieve higher preferences than the residency programs.

2 Deferred Acceptance Algorithm

The algorithm we described for the NRMP is a simplification of the true algorithm used in practice, which also takes couples into account; however, the algorithm given is a special case of the deferred acceptance algorithm. The general features of the matching market that is solved by this algorithm is that (i) the market is two-sided, and (ii) one side of the market accepts one or more proposals, and (iii) proposals are made by the other side of the market. Historically, this algorithm was developed as a solution to the *stable marriage problem*, which is a two-sided matching market in which one side accepts at most one proposal from the other side.