# MATH 566: Lecture 27 (11/19/2024)

Today: * stable marriage problem
* non-bipartite cardinality matching

## Stable Marriage Problem

A special case of weighted bipartite matching, but we work with rankings rather than $c_{ij}$'s.

We have two groups (group1, group2) of $n$ persons each. Each person provides a ranking of the $n$ people in the other group, e.g., from most to least preferred.

**Def** A pair $[p_1, p_2]$ of persons with $p_i \in$ group$_i$, $i=1,2$, is **unstable** if they are not married to each other, but they prefer each other to their spouses.

The goal of stable marriage problem is to identify a perfect matching that is **stable**, i.e., it has no unstable pairs.

**Q.** Does a stable perfect matching always exist? YES!

We describe an $O(n^2)$ algorithm to construct such a matching. This is a **propose-and-reject** algorithm, which is a greedy algorithm.

Input: Two $n \times n$ matrices of rankings by the two groups. The rankings are integers in $[1,n]$, e.g., each person produces a permutation of $1, 2, \ldots, n$.

**The set-up** The algorithm maintains a LIST of unassigned group 1 members, and each person in LIST has a current_candidate, to whom they propose next.

→ from group 2.

**The algorithm**

* At each step, pick a person from LIST, say Bill.
* Bill proposes to his current_candidate, say, Helin.
* If Helin is unassigned, the proposal is accepted, and [Bill, Helin] are engaged.
* If Helin is engaged already to France ∈ group 1, Helin selects their preferred partner (between Bill and France), and rejects the other person.
* Rejected person is added back to the LIST, and they designate the next person in their preferred list as their current_candidate.
* Continue until LIST is empty, and then all engaged couples are married.

LIST can be handled as a FIFO or LIFO queue — it does not make a big difference in complexity here.

## Correctness

If Dave ∈ group1 prefers Laura ∈ group2 over his spouse (as determined by the algorithm), he would've proposed to Laura earlier, and Laura would have rejected Dave in favor of someone she likes more. Since no person in group2 switches to a partner they prefer less, Laura prefers her spouse to Dave. Hence [Dave, Laura] is not an unstable pair.

Hence the matching found is stable. □

→ Laura must prefer Dave to her spouse as well to create an unstable pair

**Def** A pair $[P_1, P_2]$, $P_i \in \text{group}_i$, $i=1,2$, are called **stable partners** if they are matched in some stable matching.

Note: There could be many stable (perfect) matchings.

**AMO Lemma 12.4** In the propose-and-reject (P&R) algorithm, a person of group2 never rejects a stable partner.

**Proof** Let $M^*$ be the stable matching given by the P&R algo. Suppose lemma is false, i.e., group2 persons reject stable partners. Let the first such rejection be when Jona rejects Dave.

Let $M^\circ$ be a stable matching in which [Dave, Jona] is a pair.

We will show $M^\circ$ cannot be stable.

Let the rejection (in $M^*$) happen because Jona is engaged to Steve, and Jona prefers Steve to Dave. Since this is the first such rejection, no other group 2 person rejected a stable partner before. Hence Steve prefers Jona to all other stable partners.

In $M^\circ$, let [Steve, Sue] be matched.

Look at pair {Steve, Jona}:

- Steve prefers Jona to Sue
- Jona prefers Steve to Dave

Pairs in $M^\circ$:

[Dave, Jona]
[Steve, Sue]

$\Rightarrow$ {Steve, Jona} is an unstable pair, and hence $M^\circ$ is not a stable matching — a contradiction! □

We can argue that the P&R algo creates a **group-1 optimal** matching, in which every group 1 person is married to their best stable partner. This result follows from the observation that group 1 persons propose to group 2 candidates in decreasing order of preference, and no group 2 person ever rejects a stable partner.

This is a somewhat surprising result — if each group 1 person is given their best stable partner independently, we get a stable matching!

On the other hand, we can also show that the P&R algorithm produces a **group-2 minimal** matching, where each person group 2 gets their least preferred stable partner.

We could reverse the roles of groups 1 and 2 to get a matching that favors group 2 more.

## Complexity of P&R Algorithm

Each group 2 person either
(1) gets their first proposal, or → n times in total
(2) rejects a group 1 candidate. → at most (n-1) times for each group 2 person

Step (2) is bottleneck, taking $O(n^2)$ time. The data (input) is $O(n^2)$ to start with (two $n \times n$ matrices). Hence the P&R algorithm is optimal, since handling data itself takes $O(n^2)$ time.

So, we cannot design an algorithm that runs in time strictly smaller than $n^2$ (in the O-sense).

# 4. Non-bipartite Cardinality Matching

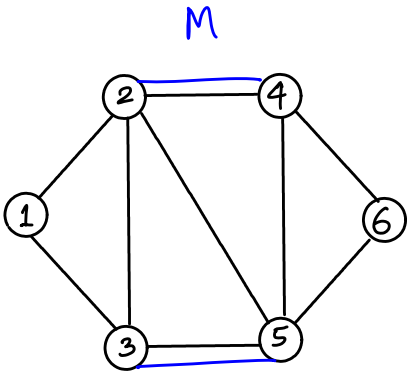
$G=(N,A)$ is an undirected network. We want to match as many nodes as possible uniquely (using edges in A).

A **matching** M of $G=(N,A)$ is a subset of the arcs such that no two arcs in the subset are incident to the same node. Arcs in M are matched arcs, rest are unmatched arcs.

If $|N|=n$, the **size** of a matching is at most $\lfloor \frac{n}{2} \rfloor$.

(size = # arcs in the matching)

## Example




$M=\{(2,4),(3,5)\}$ is a matching of size $|M|=2$. Note that nodes 1, 6 are unmatched here.

We will use "augmenting paths" to increase the cardinality of a matching — more in the next lecture.