We will discuss an algorithm for finding stable matchings (not the one you're probably familiar with). The "Instability Chaining Algorithm" is the most similar algorithm in the literature to the one actually used by the National Resident Matching Program, to assign new doctors to the hospitals where they do their residencies. The NRMP is mentioned in just about every work on stable matchings, but it turns out the theory of that market is quite a bit different than the standard version of the problem. We'll discuss some of those differences, and some of the history surrounding the algorithm.

1 Definitions and Things You Know

If you've had a class that mentioned stable matchings, there were two things you probably learned – the Gale Shapley proposal algorithm, and that the National Resident Matching Program actually uses an algorithm to generate a stable matching of new resident-doctors to hospitals. It turns out that the algorithm used in the NRMP isn't quite the proposal algorithm you're familiar with. Our plan for today is to introduce the algorithm (and prove correctness) in a simple case, describe why this doesn't actually capture the NRMP problem, and then outline how the real NRMP algorithm works.

Let's recall a definition of a stable matching instance we'll use to describe the algorithm.

We are given two groups (call them doctors and hospitals). Each doctor has a preference list over hospitals and not being matched at all (denoted $\varnothing$), and each hospital has its preferred list of doctors (or no match). A matching is a pairing of each doctor to at most one hospital, and each hospital to at most one doctor. A matching is individually rational if no agent can improve their match unilaterally (by choosing to be matched to no one rather than their current match) . A matching is stable if it is individually rational and there is no "blocking pair" – a pair d, h such that d prefers h to their current match and h prefers d to its current match.

The original stable matching algorithm (formalized by Gale and Shapley) involves one side offering/proposing to their most preferred remaining choice, while the other side rejects all but their best current offer. Eventually everyone is tentatively matched, and the result is a stable matching. [2] The intuition for the algorithm is that no proposer can form a blocking pair, since she only moves down her list when a proposee has rejected her, and no proposee can form a blocking pair since he only receives a proposal once the proposer has no better options. Thus we check the best possible matchings for proposers until we arrive at one that is stable.

Indeed not only does this produce a stable arrangement, it produces one with an extra optimality property. Call an agent "attainable" for another if there is a stable matching in which they are matched.

Theorem 1 ([2] and [3]). *The hospital-proposing algorithm matches each hospital to its most-preferred attainable doctor, and each doctor to her least preferred attainable hospital.*

Proof. Let h be the first hospital rejected by an attainable doctor. The doctor, d, rejected h, because she already had an offer from h', a preferred hospital. By assumption, h' has not been rejected by an attainable doctor. Now consider the matching where (h, d) are matched (to verify they are attainable for each other). Since h' has not been rejected by an attainable doctor (and h' just offers down its list), it is matched to a doctor lower on its list than d. But then d and h' each prefer each other to their matches, and d is not actually attainable for h (a contradiction).

For the second part, let M be the matching from the hospital-proposing algorithm. And suppose h, d are matched in M. For contradiction, suppose there is some h' such that in a stable matching M', h' is matched to d and d prefers h to h'. In M', d is matched to some h'' To ensure stability of M', h must prefer d'' to d, but then h did not get its best attainable doctor in M, contradicting our previous claim. □

There are instances with matchings which are neither man- nor woman-optimal. One might hope for an algorithm that could find these matchings as well. In fact, there is such an algorithm![1]

2 An Algorithm in the Simple Case

The following "Instability Chaining Algorithm" was published by Roth and Vande Vate in 1990 [6]. The goal of the algorithm is to build a set (called A) such that there is no blocking pair contained completely inside the set A. Start with a matching that is individually rational, but not necessarily stable.

Theorem 2 (Roth Vande Vate [6]). *The Instability Chaining Algorithm produces a stable matching.*

Proof. We will keep the invariant that at the top of the main while loop, if $x \in A$ and x is matched by μ then $\mu(x)$ is also in A and there is no blocking pair with both members in A.

We proceed by induction. Suppose we take the "if" branch, with x, y the pair picked. Let x be a man, and y a woman (the other case is similar). After we add y to A, there is no

[1] There are actually many, we'll talk about one.

```
function InstabilityChaining(M, W, μ)
    A ← {m_1, μ(m_1)}
    while A ≠ M ∪ W do    ▷ invariant: at top of this loop, no blocking pair is fully in A
        if μ is not stable then
            Let x, y be a blocking pair such that (1) x ∈ A, y ∉ A (2) y does not form a
blocking pair with any z ∈ A such that z >_y x.
            Add (x, y) to μ Leave the former mate of x (call it x′) unmatched.
            add y to A.
            while x′ forms an blocking pair with u ∈ A do
                Let u be the most preferred potential blocker with x′, and u′ be matched to
u.
                Match u, x′ in μ
                x′ ← u′
            end while
        else pick some pair (x, y) ∈ μ but with x, y ∉ A and add to A
        end if
    end while
end function
```

blocking pair involving y and a man in A (because we selected x to be the best blocking partner for y in A). Similarly, x has only improved, so it is not part of any blocking pair fully in A. Only x' had their partner changed, so the only possible blocking pairs involve x'. If such a pair exists, the second loop will similarly eliminate any blocking pairs involving x' and the cost of possibly introducing x'_2. Since the men only improve their match at each of these steps, the second loop must stop, and so no blocking pair will be fully in A. Thus we have that all matches of elements of A stay in A and there is no blocking pair fully in A, as required.

If we take the "else" branch and add x, y then there cannot be a blocking pair inside A involving x or y by the fact we didn't take the "if" branch. This completes the induction.

Note that every iteration of the main loop adds at least one agent to A, thus A will eventually equal all agents. At this point, the matching must be stable by the invariant. □

Example:

$m_1 : w_1, w_2, w_3, w_4$
$m_2 : w_2, w_3, w_4, w_1$
$m_3 : w_3, w_4, w_1, w_2$
$m_4 : w_4, w_1, w_2, w_3$

$w_1 : m_2, m_3, m_4, m_1$
$w_2 : m_3, m_4, m_1, \varnothing, m_2$
$w_3 : m_4, m_1, m_2, m_3$
$w_4 : m_1, m_2, m_3, m_4$

Execution: initialize A to m_2, w_4. m_2, w_3 is blocking A becomes m_2, w_3, w_4. w_4 cannot find a better partner in A. m_3, w_4 is blocking. A adds m_3. No one left to search for a better partner. No blocking in A. Add m_4, w_2. m_4, w_1 is blocking. A adds w_1. w_2 cannot find a better partner in A. m_1, w_2 is blocking. A becomes full set. Matching is stable.

Remark: This matching is neither the man-optimal nor the woman-optimal matching.

Remark: One can recover the men-proposing algorithm by initializing A to be all women.

3 Adapting for the NRMP

The National Resident Matching Program matches new doctors to hospitals at which they will do their residency. Before the 1950s, there was no central authority performing a match – In the '40s it worked somewhat similarly to admissions processes we're familiar with – hospitals sent offers, which could be held by students until a uniform time (10 days after the initial offers were sent), and hospitals could offer down their wait-list when they got rejections. But at the time, there were far more spots available than residents to fill them. Thus students will hold their offers until the last possible second, hoping to get a better one, and no one rejects/accepts an offer until the last possible second. Over the years, hospitals tried moving the offer expiration date forward to "fix" this problem. By the late 40's offers lasted for 12 hours, and then eventually became "exploding" offers, where the offer expired when you hung up the phone. [2]

Everyone found this unacceptable, so the NRMP switched to a deferred acceptance type

[2] www.nobelprize.org/nobel_prizes/economic-sciences/laureates/2012/roth-lecture_slides.pdf

algorithm to replace this system. The original version was primarily program proposing [3]. Changes were made to the algortihm in the '70s, and again in the mid-90s.

3.1 Changes in the '70s – Accounting for Couples

The NRMP doesn't fit into the class of instances we've discussed. First programs can have more than one slot available (there's an obvious reduction but it actually causes subtle changes in considering optimality of matchings and in considering strategic options). There's a bigger problem though – the NRMP allows couples to list pairs of programs to be matched to simultaneously (trying to ensure they end up in the same city). It turns out this substantially changes the theory of stable matchings

Theorem 3 (Roth [4])**.** *A matching instance with couples might not have a stable matching.*

Proof. $h_1 : d_4 > d_2 > d_1 > d_3$
$h_2 : d_4 > d_3 > d_2 > d_1$
$h_3 : d_2 > d_3 > d_1 > d_4$
$h_4 : d_3 > d_4 > d_1 > d_3$

$[d_1, d_2] : [h_1, h_2] > [h_4, h_1] > [h_4, h_3] > [h_4, h_2] > [h_1, h_4] > [h_1, h_3] > [h_3, h_4] > [h_3, h_1] > [h_3, h_2] > [h_2, h_3] > [h_2, h_4] > [h_2, h_1]$
$[d_3, d_4] : [h_4, h_2] > [h_4, h_3] > [h_4, h_1] > [h_3, h_1] > [h_3, h_2] > [h_3, h_4] > [h_2, h_4] > [h_2, h_1] > [h_2, h_3] > [h_1, h_2] > [h_1, h_4] > [h_1, h_3]$

□

The NRMP algorithm has allowed for couples (in some form) since the '70s, but a stable matching has always been found, so this appears to not show up in practice. But at least in theory, it's possible that one day there just won't be a stable matching. .

3.2 Changes in the '90s – Switching who Proposes

The changes in the '90s are perhaps even more interesting. Sometime in the mid-90's a few doctors opened up the theory/economics literature on stable matchings and freaked out (See: [7]). They complained that

[3] `www.nobelprize.org/nobel_prizes/economic-sciences/laureates/2012/roth-lecture_slides.pdf`

1. The NRMP was "misleading" students with technically correct (but perhaps misleading to non-expert medical interns) statements like you get "the most-preferred program on [your] Rank Order List that offered [you] a position." In fact, of course the literature available would suggest they are getting their worst attainable match [4]

2. Perhaps more importantly, the NRMP informs students their best strategy is always to list their true preferences. There is no basis for this, even in theory.

The NRMP didn't ignore these calls, and hired some researchers to change the algorithm (a looming DOJ anti-trust investigation may have helped lead them to this magnanimous decision). The updated algorithm is primarily applicant-proposing. If we don't look any closer, this looks like a great advertisement for theory research – only with the theoretical understanding of markets did people realize that gaming the system was possible and that the students were getting their worst attainable match.

But if we take a closer look, it seems it might not matter all that much. When the algorithm was being redesigned by Roth and Peranson, they ran the old version and the new version on five years worth of data. Each year, only about 0.1% of applicants are matched to different programs between the two algorithms, and even fewer could lie about their preferences to improve their match [5] [5]. Perhaps the craziest observation, though, is that some doctors were matched to *worse* programs under the new algorithm compared to the old one. More doctors (and fewer hospitals) would prefer the new algorithm to the old one, but it wouldn't be unanimous [5]. It turns out those optimality guarantees from the standard matching case simply don't transfer over once you add couples.

Theorem 4 (Aldershof, Carducci [1])**.** *Even if a matching instance with couples has a stable matching, it need not have hospital- or intern-optimal matchings.*

Proof.
$h_1 : d_4 > d_3 > \varnothing$
$h_2 : d_2 > d_3 > d_1 > \varnothing$
$h_3 : d_2 > d_4 > d_1 > \varnothing$
$h_4 : d_2 > d_3$

$[d_1, d_2] : [h_3, h_2] > [h_2, h_3] > [h_2, h_4] > [h_3, h_4] > [\varnothing, h_3] > [\varnothing, h_2] > [\varnothing, h_4] > [h_3, \varnothing] > [h_2, \varnothing]$
$[d_3, d_4] : [h_2, h_1] > [h_2, h_3] > [h_1, h_3] > [h_4, h_1] > [h_4, h_3]$

[4] They weren't – more on this later

[5] Technically, they can't prove this limit, as for all they know everyone was lying about their preferences before they got them, but their interpretation is fairly convincing.

One can verify there are two stable matchings:

$$M_1 : (h_1, d_4), (h_2, d_3), (h_3, d_2), (h_4, \varnothing), (\varnothing, d_1)$$

$$M_2 : (h_1, d_4), (h_2, d_2), (h_3, d_1), (h_4, d_3)$$

Neither is hospital optimal as h_2 prefers M_2 and h_3 prefers M_1. Couple $[d_1, d_2]$ prefers M_2, $[d_3, d_4]$ prefer M_1 $\square$

Perhaps the lesson here is to be careful in adapting our theoretical results to more complicated settings. It seems that almost none of our favorite results about the standard stable matching problem survive when we add in couples. But I suppose if you're an average doctor, the complaints filed on your behalf did improve your match in expectation (just by a very small amount).

The NRMP, for its part, still claims that "no applicant could obtain a better outcome than the one produced by the algorithm." [6]

3.3 The Actual Algorithm

So how does the algorithm actually work in practice.

Start with A equal to all hospitals. Take a new intern (not in A), they propose until they get a tentative match, and the algorithm proceeds as expected until A has all non-couple applicants. Then the algorithm starts handling couples. Each couple proposes to the top pair on its list. If both prefer their new applicant, any displaced applicants are placed on the "applicant stack" otherwise, this is considered a rejection, and the couple checks their next option. When a member of a couple is displaced, their partner is also removed from their match, and the now student-less program is placed on the "program stack." When you've matched someone, match the top member of the applicant stack. Once the applicant stack is empty, grab the top of the program stack, and add to the applicant stack any applicant in A who forms a blocking pair with the program is added to the applicant stack.

References

[1] Brian Aldershof and Olivia M Carducci. Stable matchings with couples. *Discrete Applied Mathematics*, 68(1-2):203–207, 1996.

[6] `http://www.nrmp.org/match-process/match-algorithm/`

[2] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

[3] David G McVitie and Leslie B Wilson. The stable marriage problem. *Communications of the ACM*, 14(7):486–490, 1971.

[4] Alvin E Roth. The evolution of the labor market for medical interns and residents: a case study in game theory. *The Journal of Political Economy*, pages 991–1016, 1984.

[5] Alvin E Roth and Elliott Peranson. The redesign of the matching market for american physicians: Some engineering aspects of economic design. Technical report, National bureau of economic research, 1999.

[6] Alvin E Roth and John H Vande Vate. Random paths to stability in two-sided matching. *Econometrica: Journal of the Econometric Society*, pages 1475–1480, 1990.

[7] Kevin Jon Williams. A reexamination of the nrmp matching algorithm. national resident matching program. *Academic Medicine*, 70(6):470–6, 1995.