The Stable Marriage Problem: An Application of Induction in Understanding Algorithms

A matchmaker must match up n men and n women. Each man has an ordered *preference list* of the n women, and each woman has a similar list of the n men. Is there a good algorithm to pair[1] them up?

Consider for example $n = 3$ men represented by numbers 1, 2, and 3 and three women A, B, and C, with the following preference lists:

Men	Women		
1	A	B	C
2	B	A	C
3	A	B	C

Women	Men		
A	2	1	3
B	1	2	3
C	1	2	3

There are many possible pairings for this example, two of which are {(1,A), (2,B), (3,C)} and {(1,B), (2,C), (3,A)}. How do we decide which pairing to choose? Let us look at an algorithm for this problem that is simple, fast, and widely-used.

The Propose-And-Reject Algorithm[2]

Every Morning: Each man goes to the first woman on his list not yet crossed off and proposes to her.

Every Afternoon: Each woman says "maybe, come back tomorrow" to the man she likes best among the proposals (she now has him *on a string*) and "never" to all the rest.

Every Evening: Each rejected suitor crosses off the woman who rejected him from his list.

The above loop is repeated each successive day until there are no more rejected suitors. On this day, each woman marries the man she has on a string.

How is this algorithm used in the real world?

[1]Notice here that the focus is on actually doing the matching not in helping people discover their preferences. Preference-discovery is a separate problem. We are assuming that all n of these men already know all n of these women quite well and vice-versa. The only question is who will get married to whom.

[2]This algorithm, also known as the Gale-Shapley Algorithm, is based on a stereotypical model of courtship where the men propose to the women, and the women accept or reject these proposals.

The Residency Match

Perhaps the most well-known application of the Propose-And-Reject Algorithm is the residency match program, which pairs medical school graduates and residency slots (internships) at teaching hospitals. Graduates and hospitals submit their ordered preference lists, and the stable pairing produced by a computer matches students with residency programs.

The road to the residency match program was long and twisted[3] Medical residency programs were first introduced about a century ago. Since interns offered a source of cheap labor for hospitals, soon the number of residency slots exceeded the number of medical graduates, resulting in fierce competition. Hospitals tried to outdo each other by making their residency offers earlier and earlier. By the mid-40s, offers for residency were being made by the beginning of junior year of medical school, and some hospitals were contemplating even earlier offers — to sophomores! The American Medical Association finally stepped in and prohibited medical schools from releasing student transcripts and reference letters until their senior year. This sparked a new problem, with hospitals now making "short fuse" offers to make sure that if their offer was rejected they could still find alternate interns to fill the slot. Once again the competition between hospitals led to an unacceptable situation, with students being given only a few hours to decide whether they would accept an offer.

Finally, in the early 50s, this unsustainable situation led to the centralized system called the National Residency Matching Program (N.R.M.P.) in which the hospitals ranked the residents and the residents ranked the hospitals. The N.R.M.P. then produced a pairing between the applicants and the hospitals, though at first this pairing was not stable. It was not until 1952 that the N.R.M.P. switched to the Propose-And-Reject Algorithm, resulting in a stable pairing.

Most recently, Lloyd Shapley and Alvin Roth won the Nobel Prize[4] in Economic Sciences 2012, by extending the Propose-And-Reject Algorithm we study in this lecture!

---

[3] The same can actually be said about actual courtship processes in the United States. A readable reference for this is the book "From front porch to back seat: courtship in twentieth-century America" by Beth Bailey.

In brief, historically courtship processes in the USA were built around an emphasis on the preference-discovery phase. Lots of cultural institutions existed to encourage mixing, and core ideals of hospitality and politeness were invoked to prevent early binding. Socially, people existed in exactly three categories: single, engaged to be married (a brief transitional period), and married. "Dates" complemented and then largely supplanted the older "calling" tradition as technology and living arrangements changed. But there was no exclusivity for single people. Marriage ages were relatively stable and in the mid-twenties.

The war created a huge shock to the system. With many men killed or wounded, a fear of being left alone accelerated the process (arguably helped along by consumerism and the rising cult of female domesticity). Everything seemed to shift to younger and younger ages. "Going steady" (sticking to a single partner exclusively in terms of going on dates) emerged and while it prompted *severe* criticism from the older generation (who, quite reasonably, argued that going steady violated the most basic principle of preference-discovery — to actually interact with *different* people at the same time *and* then also help set them up with others through introductions — and furthermore, only exposed vulnerable young people to "temptation"), it became socially acceptable as people raced to lock-in a partner. The marriage age plummeted until the median hit 18 years old by the early 1960s.

In the subsequent decades, the marriage age crept back up slowly. However, for a long time, the actual ages of "going steady" (early-binding) and having a single partner for dates stayed stable and in most cases, began below even 18 years old. There is some evidence that perhaps now, this trend is reversing and the social hold of "going steady" is being broken among youth.

[4] See `http://www.nobelprize.org/nobel_prizes/economic-sciences/laureates/2012/popular-economicsciences2012.pdf` for more details.

Properties of the Algorithm

We wish to show that the propose-and-reject algortihm is fast and finds a good pairing. But first, we must show that it halts. Here is a simple argument: on each day that the algorithm does not halt, at least one man must eliminate some woman from his list (otherwise the halting condition for the algorithm would be invoked). Since each list has n elements, and there are n lists, this means that the algorithm must terminate in at most n^2 days. Next, we need to show that the propose-and-reject algorithm finds a good pairing. Before we do this, we should discuss what we consider to be a good pairing.

Stability

What properties should a good pairing have? One possible criterion for a "good" pairing is one in which the number of first ranked choices is maximized. Another possibility is to minimize the number of last ranked choices. Or perhaps minimizing the sum of the ranks of the choices, which may be thought of as maximizing the average happiness.

In this lecture we will focus on a very basic criterion: *stability*. A pairing is unstable if there is a man and a woman who prefer each other to their current partners. We will call such a pair a *rogue couple*. So a pairing of n men and n women is stable if it has no rogue couples.

An unstable pairing from the example given in the beginning is: $\{$(1,C), (2,B), (3,A)$\}$. The reason is that 1 and B form a rogue couple, since 1 would rather be with B than C (his current partner), and since B would rather be with 1 than 2 (her current partner).

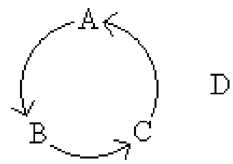
An example of a stable pairing is: $\{$(2,A), (1,B), (3,C)$\}$. Note that $(1,A)$ is not a rogue couple. It is true that man 1 would rather be with woman A than his current partner. Unfortunately for him, she would rather be with her current partner than with him. Note also that both 3 and C are paired with their least favorite choice in this matching. Nonetheless, it is a stable pairing, since none of their preferred choices would rather be with them.

Before we discuss how to find a stable pairing, let us ask a more basic question: do stable pairings always exist? Surely the answer is yes, since we could start with any pairing and make it more and more stable as follows: if there is a rogue couple, modify the current pairing so that they are together. Repeat. Surely this procedure must result in a stable pairing! Unfortunately this reasoning is not sound. To demonstrate this, let us consider a slightly different scenario, the roommates problem. Here we have $2n$ people who must be paired up to be roommates (the difference being that unlike the courtship scenario, a person can be paired with any of the remaining $2n-1$). The point is that nothing about the intuition about the progress made by the propose-and-reject algorithm relied on the fact that men can only be paired with women, so the same intuition should apply to the roommates problem as well. The following counter-example illustrates the fallacy in the reasoning:

Roommates Problem:

Roommates			
A	B	C	D
B	C	A	D
C	A	B	D
D	–	–	–

Visually, we have the following situation:



What is interesting about this problem is that there is no stable pairing (i.e., there is always a rogue couple). For example, the pairing {(A,B), (C,D)} contains the rogue couple B and C. Using the reasoning above, we might decide to pair B and C together, giving us the pairing: {(B,C), (A,D)}. But this pairing is also unstable because now A and C are a rogue couple. [Exercise: Verify that in this example there is *no* stable pairing!] Thus any proof that there must be a stable pairing in the courtship problem must use the fact that there are two genders in an essential way.

Analysis

Let us now return to our analysis of the men-propose-and-women-reject algorithm. To establish that it outputs a stable pairing, we need the following crucial lemma:

Improvement Lemma: If M proposes to W on the kth day, then on every subsequent day she has someone on a string whom she likes at least as much as M.

Proof: Suppose towards a contradiction that the jth day for $j > k$ is the first counterexample where W has either nobody or some M* inferior to M on a string. On day $j-1$, she has M' on a string and likes M' at least much as M. According to the algorithm, M' still proposes to W on the jth day since she said "maybe" the previous day. So W has the choice of at least one man on the jth day; moreover, her best choice is at least as good as M', and according to the algorithm she will choose him over M*. This contradicts our initial assumption. ♠

What proof techniques did we use to prove this lemma? It is really an induction on j, the number of days. Recall that there are many equivalent forms of induction — simple induction, strong induction, and the well-ordering principle. We used the well-ordering principle in the first sentence of the proof when we asserted that if the statement of the lemma is not true, then there must be a *smallest* counterexample: "the first day j ..." [Exercise: How would you restate this proof using simple induction or strong induction?]

Let us now proceed to prove that at the end of the algorithm all $2n$ people are paired up. Before reading the proof, see if you can convince yourself that this is true. The proof is remarkably short and elegant and is based crucially on the Improvement Lemma:

Lemma: The algorithm terminates with a pairing.

Proof: Suppose for contradiction that there is a man M who is left unpaired at the end of the algorithm. He must have proposed to every single woman on his list. By the Improvement Lemma, each of these women thereafter has someone on a string. Thus when the algorithm terminates, n women have n men on a string not including M. So there must be at least $n+1$ men. Contradiction. ♠

Now, before we prove that the output of the algorithm is a stable pairing, let us first do a sample run-through of the propose-and-reject algorithm. We will use the preference lists given earlier, which are duplicated here for convenience:

Men	Women		
1	A	B	C
2	B	A	C
3	A	B	C

Women	Men		
A	2	1	3
B	1	2	3
C	1	2	3

The following table shows which men propose to which women on the given day (the circled men are the ones who were told "maybe"):

Days	Women	Proposals
1	A	①, 3
	B	②
	C	—
2	A	①
	B	②, 3
	C	—
3	A	①
	B	②
	C	③

Thus, the stable pairing which the algorithm outputs is: {(1,A), (2,B), (3,C)}.

Theorem: The pairing produced by the algorithm is always stable.

Proof: We will show that no man M can be involved in a rogue couple. Consider any couple (M,W) in the pairing and suppose that M prefers some woman W* to W. We will argue that W* prefers her partner to M, so that (M,W*) cannot be a rogue couple. Since W* occurs before W in M's list, he must have proposed to her before he proposed to W. Therefore, according to the algorithm, W* must have rejected him for somebody she prefers. By the Improvement Lemma, W* likes her final partner at least as much, and therefore prefers him to M. Thus no man M can be involved in a rogue couple, and the pairing is stable. ♠

Optimality

Consider the situation in which there are 4 men and 4 women with the following preference lists:

Men	Women			
1	A	B	C	D
2	A	D	C	B
3	A	C	B	D
4	A	B	C	D

Women	Men			
A	1	3	2	4
B	4	3	2	1
C	2	3	1	4
D	3	4	2	1

For these preference lists, there are exactly two stable pairings: $S = \{$(1,A), (2,D), (3,C), (4,B)$\}$ and $T = \{$(1,A), (2,C), (3,D), (4,B)$\}$. The fact that there is more than one stable pairing brings up an interesting question. What is the best possible partner for each person, say man 2 for example?

The trivial answer is his first choice (i.e., woman A), but that is just not a realistic possibility for him. Pairing man 2 with woman A would simply not be stable, since he is so low on her preference list. And indeed there is no stable pairing in which 2 is paired with A. Examining the two stable pairings, we can see that the best possible realistic outcome for man 2 is to be matched to woman D.

Let us make some definitions to better express these ideas: we say the *optimal* woman for a man is the highest woman on his list whom he could be paired with in some *stable* pairing. In other words, the optimal woman is the best that a man could do under the condition of social stability. In the above example, woman D is the optimal woman for man 2. So the best that each man can hope for is to be paired with his optimal woman. But can they achieve this optimality *simultaneously*? I.e., is there a stable pairing such that each man is paired with his optimal woman? If such a pairing exists, we will call it a *male optimal* pairing. Turning to the example above, S is a male optimal pairing since A is 1's optimal woman, D is 2's optimal woman, C is 3's optimal woman, and B is 4's optimal woman. Similarly, we can define a female optimal pairing, which is the pairing in which each woman is paired with her optimal man. [Exercise: Check that T is a female optimal pairing.] We can also go in the opposite direction and define the *pessimal* woman for a man to be the lowest ranked woman whom he is ever paired with in some stable pairing. This leads naturally to the notion of a *male pessimal* pairing — can you define it, and also a female pessimal pairing?

Now, a natural question to ask is: Who is better off in the men-propose-and-women-reject algorithm: men or women? Think about this before you read on...

Theorem: The pairing output by the men-propose algorithm is male optimal.

Proof: Suppose for the sake of contradiction that the pairing is *not* male optimal. Assume the first day when a man got rejected by his optimal woman was day k. On this day, M was rejected by W^* (his optimal mate) in favor of M^* who proposed to her. By definition of optimal woman, there must exist a stable pairing T in which M and W^* are paired together. Suppose T looks like this: $\{\ldots,(M,W^*),\ldots,(M^*,W'),\ldots\}$. We will argue that (M^*,W^*) is a rogue couple in T, thus contradicting stability.

First, it is clear that W^* prefers M^* to M, since she rejected M in favor of M^* during the execution of the men-propose algorithm. Moreover, since day k was the first day when some man got rejected by his optimal woman, on day k M^* had not yet been rejected by his optimal woman. Since he proposed to W^* on the k-th day, this implies that M^* likes W^* at least as much as his optimal woman, and therefore at least as much as W'. Therefore, (M^*,W^*) form a rogue couple in T, and so T is not stable. Contradiction. Thus, our assumption was wrong and the pairing is male optimal. ♠

What proof techniques did we use to prove this theorem? Again it is a proof by induction, structured as an application of the well-ordering principle. How do we see it as a regular induction proof? This is a bit subtle to figure out. See if you can do so before reading on. As a hint, the proof is really showing by induction on k the following statement: for every k, no man gets rejected by his optimal woman on the kth day. [Exercise: can you complete the induction?]

So men appear to fare very well by following this algorithm. How about the women? The following theorem confirms the seemingly sad truth:

Theorem: If a pairing is male optimal, then it is also female pessimal.

Proof: Let T = {..., (M, W), ...} be the male optimal pairing (which we know is output by the algorithm). Suppose for the sake of contradiction that there exists a stable pairing S = {..., (M*, W), ..., (M, W′), ...} such that M* is lower on W's list than M (i.e., M is not her pessimal man). We will argue that S cannot possibly be stable by showing that (M,W) is a rogue couple in S. By assumption, W prefers M to M* since M* is lower on her list. And M prefers W to his partner W′ in S because W is his partner in the male optimal pairing T. Contradiction. Therefore, the male optimal pairing is female pessimal. ♠

All this seems a bit unfair to the women! Are there any lessons to be learned from this?

Back to the National Residency Matching Program, until recently the algorithm was run with the hospitals doing the proposing, and so the pairings produced were hospital optimal. In the nineties, the roles were reversed so that the medical students were proposing to the hospitals. More recently, there were other improvements made to the algorithm which the N.R.M.P. used. For example, the pairing takes into account preferences for married students for positions at the same or nearby hospitals.

Further reading (optional!)

Though it was in use 10 years earlier, the propose-and-reject algorithm was first properly analyzed by Gale and Shapley, in a famous paper dating back to 1962 that still stands as one of the great achievements in the analysis of algorithms. The full reference is:

D. Gale and L.S. Shapley, "College Admissions and the Stability of Marriage," *American Mathematical Monthly* **69** (1962), pp. 9–14.

Stable marriage and its numerous variants remains an active topic of research in optimization. Although it is by now twenty years old, the following very readable book covers many of the interesting developments since Gale and Shapley's algorithm:

D. Gusfield and R.W. Irving, *The Stable Marriage Problem: Structure and Algorithms*, MIT Press, 1989.