This work was partially funded by the National Science Foundation under grant number MCS-81/05894.

THREE FAST ALGORITHMS FOR FOUR PROBLEMS IN STABLE MARRIAGE

Dan Gusfield

YALEU/DCS/TR-407

July, 1985

THREE FAST ALGORITHMS FOR FOUR PROBLEMS IN STABLE MARRIAGE

Dan Gusfield
Department of Computer Science, Yale University

Table of Contents

1 Introduction 1
2 Definitions and Background Results. 1
3 Finding all stable pairs in time $O(n^2)$ 4
3.1 Rotations and Algorithm A 9
4 Enumerating all stable marriages in optimal time and space 10
4.1 The enumeration algorithm 14
5 The minimum regret stable marriage problem 17
6 Open question 20
7 Appendix. 22
8 References 25
9 Acknowledgements 25

This work was partially funded by the National Science Foundation under grant number MCS-81/05894.

1. Introduction

The stable marriage problem is a well-known problem of matching n men to n women to achieve a certain type of "stability"; the Gale-Shapley [GS] algorithm for finding two particular, but extreme, stable marriages (out of a possibly exponential number of stable marriages) is also well known, and is the basis of a national system for matching hospitals to resident doctors [R]. In this paper we consider four problems concerned with finding information about the set of all stable marriages, and with finding stable marriages other than those obtained by the Gale-Shapley algorithm. In particular, we give an $O(n^2)$ time algorithm which, for any problem instance of n men and n women, finds every man-woman pair that is contained in at least one stable marriage; we show that same algorithm finds all the "rotations" for the problem instance in $O(n^2)$ time (rotations are central in efficiently finding the optimal or most "egalitarian" stable marriage [ILG], and in the efficient enumeration of all stable marriages); we give an $O(n^2+n|S|)$ time and $O(n^2)$ space bounded algorithm (which is time and space optimal) to enumerate all stable marriages, where S is the set of them; and we give an $O(n^2)$ time algorithm to find the minimum regret stable marriage (the best marriage, as measured by the person who is worst off in it). We believe the previous best time bounds for these problems are respectively $O(n^4)$ (from a related problem in [K]), $O(n^3)$ [ILG], $O(n^3|S|)$ [K], and $O(n^4)$ [K]. The basic idea leading to the improved running times is to exploit theorems (from [K], [MW], [IL], and this paper) about the structure of stable marriages in order to avoid back-up and duplicated work inherent in earlier algorithms.

2. Definitions and Background Results

An instance of the stable marriage problem consists of n men and n women, each of whom has a rank-ordered preference list of the n people of the opposite sex. A marriage M is a one-one matching of the men and the women. Marriage M is said to be *unstable* if there is a man m and a woman w who are not matched to each other in M, but who both prefer each other to their respective mates given in M. Such a pair is said to *block* M. A marriage that is not unstable is called *stable*. The fundamental theorem [GS] is that there is a stable marriage for any problem instance. It is known [K] that there can be an exponential number of stable marriages, and the problem of counting them is #P- complete [IL].

The lattice of stable marriages

Let M and M′ be two stable marriages, and let max_i (M, M′) be the women man i most prefers between his two assigned mates in M and M′. Let min_i(M, M′) denote the other woman. Then max(M, M′) is the mapping of each man i to max_i(M, M′), and min(M, M′) is the opposite mapping. We say that marriage M *dominates* marriage M′ (from the perspective of the men) if and only if M = max(M, M′), and a marriage X is *between* M and M′ if and only if M dominates X and X dominates M′, and X differs from both M and M′. It is surprising, but easy to show

([K], [GS84]) that max(M, M′) and min(M, M′) are both stable marriages. Hence, under the relation of dominance, the set of all stable marriages forms a lattice L where the join and union operations are the max and min operations above. The unique maximum (most dominant) element of L is called the *man- optimal* marriage, and the unique minimum (most dominated) element is called the *woman optimal* marriage. The man optimal marriage has the very strong property that for *every* man m, there is no stable marriage in which m is married to a woman he prefers to his mate in the man optimal marriage.

If dominance is defined from the womens' point of view, and wmax and wmin are the max and min operations with respect to the women, then max(M, M′) = wmin(M,M′) and min(M, M′) = wmax(M,M′), so the lattice obtained using wmax is an inverted copy of the lattice obtained using max. Hence the man optimal marriage is woman *pessimal* i.e. for every woman w, there is no stable marriage in which w is married to a man she prefers less than her mate in the man optimal marriage. Similarly, the woman optimal marriage is man pessimal. In this paper, dominance will always be from the men's point of view unless explicitly stated otherwise. The following is an immediate consequence of the above facts.

Lemma 0 [K], [GS84]: For any two stable marriages M and M′, M dominates M′ from the men's point of view, if and only if M′ dominates M from the women's point of view.

Gale-Shapley Algorithm

The algorithm of [GS] finds the man optimal marriage, although it can, by relabeling, also find the woman optimal marriage. This algorithm will be the basis of two of the algorithms given in this paper, so we briefly review it here.

Algorithm GS

At the start of the algorithm, each person is *free* and becomes *engaged* during the execution of the algorithm. Once a women is engaged she never becomes free again (although who she is engaged to may change), but men can alternate between being free and being engaged. The following step is iterated until all men are engaged:

Choose a free man m, and have m *propose* to the highest (most preferred) woman w on his list, such that w has not already rejected m. If w is free, then w and m become engaged. If w is engaged to man m′, then she rejects the man (m or m′) that she least prefers, and becomes, or remains, engaged to the other man. The rejected man becomes, or remains, free.

When all men are engaged, the engaged pairs are said to be *mated* or *paired* and form the man optimal stable marriage.

Proof of the correctness of this method appear in many places [K], [GS], [L]. There are specializations of the above algorithm derived by imposing rules specifying which free man makes the next proposal. In the remainder of the paper, we will use the rule that if a man m has just been rejected, then man m makes the next proposal. With this rule, it is clear that the Gale-Shapley algorithm can be implemented to run in time $O(n^2)$.

Breakmarriage

Definition: Let M be a stable marriage. Let man m be married (paired) in M to woman w. The operation *breakmarriage*(M,m), developed in [MW], is defined as follows:

With the men and woman paired as in M, restart the Gale-Shapley algorithm by breaking the pairing of m and w, making [redacted]. Hence man m now proposes to the next (most preferred women who he has not proposed to) woman on his list, and this initiates a sequence of proposals, rejections and acceptances as given by the Gale-Shapley algorithm. Operation breakmarriage(M,m) terminates either when some man has been rejected by all women, or when all men are engaged (this will happen the first time w receives a proposal). Note that during the entire running of breakmarriage(M,m) there is exactly one free man at any time, hence (unlike the Gale-Shapley algorithm above) the sequence of proposals is completely determined: the next proposal is always made by the unique free man.

The following is simple to prove:

Lemma 1 [MW]: If breakmarriage(M,m) terminates with all men engaged, then the engaged pairs form a stable marriage.

The following is the central theorem in [MW].

Theorem 1 [MW]: Every stable marriage M′ can be obtained by a series of breakmarriage operations starting from the man optimal marriage M_0.

The proof is simple, and we will not repeat it. However, the key point in the proof is that if M is any stable marriage which dominates M′, and man m's mate in M′ is different than his mate in M, then breakmarriage(M,m) either results in M′ or in a stable marriage between M and M′ (i.e. breakmarriage(M,m) does not move any man to a woman below his proper mate in M′). Hence M′ can be derived from M_0 by successively (and arbitrarily) choosing a man who isn't yet married to his mate in M′, and executing a breakmarriage operation. Each such operation either results in M′ or in a stable marriage that dominates M′, but which is closer to it.

In order to make the transformation from M_0 to M′ completely deterministic, we can impose an ordering on the men so that when a new breakmarriage operation must be started, we select the first man (in the ordering) who is not yet married to his intended mate in M′. Hence the set of proposals and their order is determined by M_0, M′ and the order of the men. This will be useful in Algorithm A below.

We note two useful corollaries to theorem 1.

Corollary 1: If breakmarriage(M,m) results in marriage M′, then M′ dominates all marriages which are dominated by M and in which m is not married to his mate in M.

Corollary 2: If m is married in M to a woman other than his mate in the woman optimal marriage, then breakmarriage(M,m) terminates with a new stable marriage, i.e. no man is rejected by all the women.

Corollary 1 is proven simply by letting M″ be any stable marriage dominated by M where m is not married to his mate in M. The first step in transforming M to M″ is to execute breakmarriage(M,m), hence breakmarriage(M,m) results in a marriage which dominates M″ and in which m has a different mate than in M. This corollary reverses the orientation of theorem 1: in the theorem, there is a known target marriage which is obtained from M_0 by successive breakmarriage operations, while with corollary 1 we will do breakmarriage operations without having any target marriage in mind, and we will extract information from the results. Corollary 1 will be central in section 5, and Corollary 2 will permit a simpler exposition.

3. Finding all stable pairs in time $O(n^2)$

Definition: Given an instance of the stable marriage problem, a man-woman pair (m,w) is said to be a *stable pair* if and only if m is married to w in some stable marriage.

There is a fairly direct method [K] to test whether any given pair (m,w) is a stable pair. The pair is stable if and only if there is a stable marriage omitting m and w, where each man who w prefers to m is mated to a woman he prefers to w, and where each woman who m prefers to w is mated to a man she prefers to m. This can be tested by a direct modification of the preference lists and of the Gale-Shapley algorithm. This approach would then need $\Theta(n^2)$ executions of the modified algorithm, so the best resulting bound would be $O(n^4)$, although it would not only identify the stable pairs, but marriages which contain them.

Knuth [K] mentions the usefulness of knowing the non-stable pairs, and shows how the man optimal and woman optimal marriages can be used to identify some, but not necessarily all, non-stable pairs. In this section we give a method to exactly identify all stable pairs (hence all non-stable pairs) in $O(n^2)$ time (the same time needed to find the man and woman optimal marriages); this is also the best time bound that we know of to determine if even a single pair is stable.

Theorem 2: Let M_0 and M_t be the man optimal and woman optimal marriages respectively. Let M_0, M_1, ... ,M_t be a sequence of stable marriages such that for each i from 0 to t-1, M_i dominates M_{i+1} and there is no stable marriage between M_i and M_{i+1}. Then every stable pair appears in at least one of the marriages in the sequence.

Proof: Let M_i and M_{i+1} be two consecutive stable marriages on the sequence, and let m be a man who is married to w_i in M_i and to $w_{i+1} \neq w_i$ in M_{i+1}. Of course, man i prefers w_i to w_{i+1}. Now let w be a woman who man m prefers to w_{i+1} but not to w_i. If there exists a stable marriage M in which m and w are married, then $M' = \min[M_i, \max(M, M_{i+1})]$ is also a stable marriage in which m and w are married, hence M′ is different from both M_i and M_{i+1}. But then, since M_i dominates M′ and M′ dominates M_{i+1}, M′ is between M_i and M_{i+1}, a contradiction. □

Corollary 3: Let H be the Hasse diagram (the graph representing the transitive reduction) of the lattice of all stable marriages. Then the marriages along any path (directed by the dominance relation) in H between the man optimal and woman optimal marriages contain all the stable pairs.

Figure 1 shows the problem instance given in [MW] and displays the Hasse diagram of the set of all stable marriages. Corollary 3 and the facts about operations max, min, wmax, and wmin are easy to verify in this example.

We now show how to efficiently find all stable pairs by enumerating a short sequence of stable marriages which satisfy the conditions of theorem 2. The particular marriages are obtained as a by-product of successive breakmarriage operations that transform the man optimal marriage into the woman optimal marriage. This method will later be used in the algorithm to find all stable marriages efficiently.

Algorithm A: pausing breakmarriage

The following algorithm finds a sequence of stable marriages that satisfy the conditions of theorem 2. The key algorithmic idea is to modify the breakmarriage operation so that it *pauses* at certain points where the next marriage in the sequence is output. In particular, the algorithm will pause when the proposal sequence generated by going from M_0 to M_t discovers a certain type of cycle called a *p-cycle* (we will see later that these are the rotations in [IL]). At each pause, the p-cycle is output, and the next marriage in the sequence is generated from the previous marriage by making changes dictated by the p-cycle. To more quickly understand the algorithm and its running time, it is helpful to keep in mind that the sequence of proposals, acceptances and rejections, is exactly the same sequence as used in transforming M_0 to M_t by successive breakmarriage operations, as discussed above, without pauses. The additional detail in the algorithm, which is interwoven into the proposal sequence, is used to extract and output p- cycles and the sequence of desired marriages.

Algorithm A

0. Set $i = 0$; find and output the man optimal marriage M_0; find the woman optimal marriage M_t. All women are *unmarked* at this point.

1. If $M_i = M_t$ then stop. Else unmark any marked women and let m be the first man (in the fixed ordering of men) whose mate in M_i is different from his mate in M_t.

2. Set M to M_i. Let w be m's mate in M. Mark w and initiate breakmarriage(M,m); carry out (or continue, if returning from 3d.) the sequence of proposals, acceptances and rejections as determined by breakmarriage(M,m), with the following modifications:

a) Whenever any unmarked woman *accepts* a proposal, mark that woman.

b) Whenever a *marked* woman, say w′ (which could be w), *receives* a proposal from a man, say m′, who she prefers to her mate in M_i, go to PAUSE breakmarriage(M,m).

Note that the comparison here is to the mate of w′ in M_i, not to who she is presently engaged to. Note also that the pause comes before w′ decides to accept or reject the proposal from m′.

3. PAUSE: When breakmarriage(M,m) pauses do:

3a. Let R(W) be the set containing w′ and all the women who were marked since the most recent time that w′ became marked; let R(M) be the mates of R(W) in M_i; and let *p-cycle* R_i be the ordered set of pairs consisting of R(W) and their respective mates in R(M), where the pairs are ordered in the order that the women in R(W) were most recently marked.

3b. Let M_{i+1} be the marriage where w′ is paired with m′, where every other woman in R(W) is paired to the man she is currently engaged to (as given in the current status of breakmarriage(M,m)), and all other women are paired to their mates in M_i. Output the p-cycle R_i and marriage M_{i+1}; set $i = i+1$ and unmark all women in R(W).

3c. If w = w′, then let w′ accept the proposal from m′ (which completes breakmarriage(M,m)), and go to step 1.

3d. If w ≠ w′, then let w′ accept or reject the proposal from m′ (by comparing m′ to the man she is currently engaged to in the current status of breakmarriage(M,m)).

If w′ rejects the proposal from m′, then mark w′; return to step 2 and continue with breakmarriage(M,m) (i.e. m′ next proposes to the woman on his list below w′).

If w′ accepts the proposal from m′, then leave w′ unmarked; go to step 2 and continue with breakmarriage(M,m) (i.e. the man who was engaged to w′ when m′ made his proposal is now free and makes the next proposal).

Note that by Corollary 2, each breakmarriage(M,m) operation, initiated in step 2, will ultimately finish with a new stable marriage (i.e. no man is rejected by all the women). It should also be clear that each M_i in the sequence is a marriage, although we must still demonstrate stability.

The five p-cycles output by algorithm A on the problem instance given in figure 1 are $\pi_1 = \{(1,5), (3,8)\}$, $\pi_2 = \{(1,8), (2,3), (4,6)\}$, $\pi_3 = \{(3,5), (6,1)\}$, $\pi_4 = \{(5,7), (7,2)\}$, $\pi_5 = \{(3,1), (5,2)\}$, where the first number in each pair in a p-cycle is a man and the second a woman. The p-cyles are listed in the order that the algorithm finds them, where the algorithm has used the given numerical order of the men. Notice that π_4 and π_5 were found in the running of a single breakmarriage operation, which started with man 3. All the other breakmarriage operations discovered exactly one p-cycle.

Before proving the correctness of algorithm A, the following interpretation of the algorithm may be helpful, especially in explaining step 3d. Suppose m is mated to w in M_i and breakmarriage(M_i,m) pauses when a marked woman w′ $\neq$ w is proposed to. Consider all of the proposals made from the start of breakmarriage(M_i,m) up to and including the first proposal that w′ accepts; let the sequence of these proposals be called P(w′). Now consider breakmarriage(M_{i+1},m). The key point to note is that breakmarriage(M_{i+1},m) initially executes exactly the same sequence of proposals P(w′) in exactly the same order. The proposal immediately following P(w′) in breakmarriage(M_{i+1},m) differs from the one in breakmarriage(M_i,m), and is, in fact, the next proposal made by algorithm A in step 2, after returning from the pause in breakmarriage(M_i,m) caused by the proposal to w′. Algorithm A can be thought of as an optimized algorithm that successively runs breakmarriages on M_0, M_1, ..., M_{t-1}, each until a p-cycle is encountered. The optimization makes sure that in each successive pair of breakmarriage operations no proposal in P(w′) is repeated. Step 3d of algorithm A adjusts the mark of woman w′ appropriately so that the last proposal in P(w′) is not repeated.

The following facts are easy to establish by examining the actions of algorithm A. They will be needed in the proof of correctness and time.

Fact 1. The men and women who have different mates in M_k than in M_{k+1} are exactly the men and women in the pairs of R_k. The men in R_k (strictly) prefer their mates in M_k to their mates in M_{k+1}, and the women in R_k (strictly) prefer their mates in M_{k+1} to their mates in M_k.

Fact 2. If m_1 and m_2 are the men in any two consecutive pairs (in the circular order) of R_k, and they are married to w_1 and w_2 respectively in M_k, then m_1 is married to w_2 in M_{k+1}.

Fact 3. If m_1 and m_2 are as above, then w_2 is the first woman below w_1 on m_1's list such that w_2 prefers m_1 to m_2, her mate in M_k.

Time Analysis and Correctness of Algorithm A

With the exception of the time needed to output the marriages, algorithm A runs in time $O(n^2)$, since step 1 is within this time bound, and since no man proposes to the same woman twice, and all other work is proportional to the number of proposals. In more detail: at any point in the algorithm the unique free man makes the next proposal which is to the next woman on his list; a linked list connecting the women in the order that they are marked allows R_i to be found in constant time per pair; and the total number of pairs in all the p-cycles is $O(n^2)$ since no pair is in more than one p-cycle (this follows from fact 1). For the purpose of efficiently outputting the pairs which appear in the sequence of marriages, we can simply output the p-cycles and marriage M_t; each pair that appears in any of the marriages in the sequence is then output exactly once, and hence $O(n^2)$ time suffices to output these pairs.

In order to show that the output pairs are in fact stable, we need the following two lemmas:

Lemma 2: Each M_i found by the algorithm is a stable marriage.

Proof: We prove this by induction. M_0 is stable and we assume that all marriages up through M_k are stable. Suppose M_{k+1} is not stable, then there is a man m and a woman w who block M_{k+1}. Since, by fact 1, each woman either improves in M_{k+1} (over M_k) or keeps her same mate, m must be in a pair in R_k, otherwise m and w would block M_k. For the same reason, m cannot prefer w to his mate in M_k, so w must be strictly between (in order of preference) m's mate in M_k and his mate in M_{k+1}; Let B denote these women. But by fact 3, none of the women in B prefer m to their mates in M_k, and so by fact 1, none of these women prefer m to their mates in M_{k+1}. Hence M_{k+1} is stable. □

Lemma 3: There is no stable marriage between M_k and M_{k+1}.

Proof: Suppose, to the contrary, that M is a stable marriage between M_k and M_{k+1}. We claim first that no man m can be married in M to a woman between his mate in M_k and his mate in M_{k+1}. Let w be such a woman between m's respective mates, and let m_w be the mate of w in M_k. By fact 3, w prefers m_w to m, and since M_k dominates M and m_w is not married to w in M, m_w prefers w to his mate in M. Hence m_w and w block M. So if a stable M exists between M_k and M_{k+1}, then every man is either mated to his mate in M_k or to his mate in M_{k+1}, and there must be at least one man of each type (else M is either M_k or M_{k+1}). Now in the circular order of pairs given in R_k, let m and m′ be any two consecutive men in R_k, and let w and w′ be their respective mates in M_k. Recall that w′ becomes the mate of m in M_{k+1}. Hence it is not possible that in M, m marries his mate in M_{k+1} and m′ marries his mate in M_k, since they both would then marry w′. Similarly, it is not possible that in M, m marries his mate in M_k and m′ marries his mate in M_{k+1}, for then w′ would be unmarried in M. But then either $M = M_k$ or $M = M_{k+1}$, and hence there is no marriage M between M_k and M_{k+1}. □

Hence by lemmas 2 and 3, and theorem 2:

Theorem 3: The marriages found by algorithm A contain all the stable pairs, and all stable pairs can be found and output in time $O(n^2)$.

3.1. Rotations and Algorithm A

Theorems 2 and 3 along with algorithm A above showed that a small set of stable marriages contain all stable pairs. This fact, given by direct proofs above, also follows from a much deeper analysis of the structure of the set of all stable marriages, given in a powerful paper by Irving and Leather [IL]. Here we define rotations, the basic object in [IL], and show that algorithm A finds each of them exactly once. Finding all the rotations is the first step in the algorithm of the next section which enumerates all stable marriages in time $O(n^2+n|S|)$; the $O(n^2)$ term contains the time used in Algorithm A to find the rotations. Finding all the rotations is also the first step in the algorithm in [ILG] to find the optimal or most egalitarian stable marriage.

Rotations

Definition: Let M be a stable marriage. For any man m let S(m) be the first woman w′ on m's list such that i) m prefers his mate in M to w′, and ii) w′ prefers m to her mate in M. Let S′(m) be the man who S(m) is married to in M.

Definition [IL]: Let $R = \{(m_1, w_1), (m_2, w_2), \ldots, (m_z, w_z)\}$ be an ordered list of pairs from M such that for each i from 1 to z, $S'(m_i)$ is $m_{i+1 \pmod z}$. Then R is called a *rotation* (exposed in M).

Note that for a given marriage there may be many or there may be no exposed rotations.

Given an instance of the stable marriage problem, consider the set of all stable marriages for that instance, and consider the set of all rotations exposed by those marriages (any given rotation may be exposed by many marriages).

It is shown in [IL] that

Theorem 4 [IL]: Except for the stable pairs that are in the woman optimal marriage (which are in no rotations), each stable pair is in exactly one rotation, and, of course, each pair in a rotation is stable.

Finding all rotations in $O(n^2)$ time

Clearly, each p-cylce R_k is a rotation (exposed in M_k) and hence algorithm A finds a set of distinct rotations[1] that contain all stable pairs other than those in both the man and woman optimal marriages. Therefore, by theorem 4, algorithm A finds all rotations, and outputs each one exactly once, and so

Theorem 5: Given a n by n instance of the stable marriage problem, all the rotations can be found and output in $O(n^2)$ time.

Combined with Corollary 3, we have

Theorem 6: Let P be any path in H from the man optimal to the woman optimal marriage. Then any two consecutive marriages on P differ by a single rotation, and the set of the rotations between marriages along P contain all rotations exactly once.

It is also now easy to see that any sequence of stable marriages which satisfy the conditions of Theorem 2 must lie on such a path P in H, and so, algorithm A enumerates the marriages along some path in H from M_0 to M_t.

4. Enumerating all stable marriages in optimal time and space

McVitie and Wilson [MW] give an algorithm for enumerating all stable marriages in a problem instance with n men and n women. Their algorithm can be shown to take at least $\Omega(n^3|S|/(\log|S|^2))$ time, and no more than $O(n^3|S|)$ time, where S is the set of stable marriages. Knuth [K] describes a similar algorithm which has the same complexity[2] . Wirth [W] gives a different enumeration algorithm even less efficient than these two.

In this section we give an algorithm for enumerating all stable marriages in $O(n^2 + n|S|)$ time and $O(n^2)$ space. Considering the time needed just to output the marriages, and the space needed just to store the input preference lists, this time and space use is necessary; it is surprising that is is also sufficient. The algorithm depends critically on results in [IL], so we will first briefly review some of the results in [IL]. We will next modify the central construction given in [IL], and then combine these results with algorithm A and the modified construction to obtain the enumeration algorithm. We also note that in the same time bound, the Hasse diagram of the lattice of all stable marriages can be explicitly constructed.

[1]We used the terminology "p-cycle" to avoid any confusion between what is derived directly from algorithm A, and what is known about rotations from [IL].

[2]It is reported in [K] that the time is $O(n^2|S|)$, but this is incorrect. We give constructions in the appendix showing that the algorithm can take $\Omega(n^3|S|/[\log|S|^2])$ time.

Partial orders and precedence graphs

Definition: For a given instance of the stable marriage problem, let π be a rotation exposed in stable marriage M, and let $M(\pi)$ be the marriage obtained by mating each man m in π with S(m), and mating all men not in π with their mates in M. We say that π *moves* each man and women in π from their mates in M to their mates in $M(\pi)$. Note that a rotation always moves a man "down" his preference list, and always moves a woman "up" her preference list, and that the moves made by a rotation are independent of the marriage it is exposed in.

Definition: A pair (m,w), not necessarily a stable pair, is said to be *eliminated* by rotation π if π moves w from m or below in her preference list to strictly above m.

Note that if (m,w) is a pair in π then π eliminates (m,w), and that if (m,w′) is any other pair eliminated by π, then m prefers w to w′, for otherwise no marriage in which π is exposed could be stable. Note also that if w is the woman most preferred by m such that (m,w) is eliminated by a rotation, then (m,w) is a stable pair, and is in the eliminating rotation.

Lemma 4 [IL]: No pair is eliminated by more than one rotation, and for any pair (m,w), at most one rotation moves m to w.

This follows directly from algorithm A and theorem 4.

Now we define the following relation between rotations:

Definition [IL]: Let π and ρ be two distinct rotations. Rotation π is said to *explicitly precede* ρ if and only if π eliminates a pair (m,w), and ρ moves m to a woman w′ such that m (strictly) prefers w to w′. The relation *precedes* is defined as the transitive closure of the relation "explicitly precedes".

It is easy to verify that the relation "precedes" defines a partial order among the rotations. In order to get some intuition for the importance of this relation, we claim (proofs follow from details in [IL]) that if π precedes ρ, then no matter how the men are ordered, algorithm A finds π before it finds ρ. Hence in any transformation of M_0 to a marriage M by breakmarriage operations, the moves specified by rotation ρ will be made only if the moves specified by rotation π are made first. These claims are strengthened in the following definitions and theorem.

Definition: Given an instance of the stable marriage problem, let D be a directed acyclic graph, where the nodes of D are in one-one correspondence with the set of rotations (we give each node the name of its corresponding rotation) and for any two nodes π and ρ, there is a directed edge from π to ρ if and only if rotation π precedes rotation ρ. Note that D may have $\Theta(n^2)$ nodes and $\Theta(n^4)$ edges.

We will often refer informally to a rotation π in D, instead of the node in D corresponding to rotation π; this should cause no confusion.

Definition: A subset of rotations SN of D is *closed* if and only if SN contains all rotations which precede the rotations in SN.

The following is the central theorem of [IL]:

Theorem 7 [IL]: Let S be the set of all stable marriages for a given problem instance, and let D be the corresponding directed graph formed from the set of all rotations. Then there exists a one-one correspondence between S and the family of closed subsets in D, i.e. each closed subset in D specifies a distinct stable marriage, and all stable marriages are specified in this way.

For a closed set SN the corresponding stable marriage is obtained by starting with M_0 and making the moves specified by the rotations in SN, in any order consistent with the precedence relations, i.e. the moves of rotation π can be made only after the moves of all rotations which precede π.

Refining D

The algorithm to enumerate all stable marriages will enumerate each closed subset of D exactly once. The style of the enumeration (without concern for time and space complexity) is not completely new, and a similar method is implicit in [IL], although the question of generating all stable marriages is not explicitly discussed there. The new contributions here are several observations which allow the approach to run fast and in small space. In particular, explicit use of D does not lead to the $O(n^2+n|S|)$ time bound, since construction of D would take more than $O(n^2)$ time ($O(n^5)$ using fast transitive closure is possible), and the enumeration itself would take more than $O(n)$ time per marriage ($O(n^2)$ is possible). Further, D needs $\Theta(n^4)$ space just to store it, hence the $O(n^2)$ space bound could not be achieved using D. The main idea in this part of the paper is to use a sparse subgraph of D which preserves all the closed subsets, and which can be built quickly. It is not difficult to see that any subgraph of D whose transitive closure is D, preserves the closed subsets. We will construct such a subgraph G, in $O(n^2)$ time, with the property that G has $O(n^2)$ edges, and that no node in G has outdegree more than n. The bounded outdegree is the one of the keys to the n|S| term in the time bound, and the sparsity is of course central to the space bound.

Definition: G is a directed acyclic subgraph of D containing all the nodes of D but only edges defined by the following two rules, which are applied for each man m whose mate in M_0 is different than his mate in M_t:

Rule 1. Let $W(m) = \{w_0, w_1, \dots, w_r\}$ be the set of women, in decreasing order of preference by m, such that for each i from 0 to r, (m, w_i) is a stable pair. For i from 0 to r-1, let π_i be the rotation containing pair (m, w_i), and let $\Pi(m)$ be the set of these rotations. Then for i from 0 to r-2, G contains an edge from π_i to π_{i+1}.

Rule 2. Suppose (m,w) is a non-stable pair eliminated by a rotation π, such that m prefers w to any other woman w′ in any other pair (m,w′) eliminated by π. If there are women w_i and w_{i+1} in W(m) such that m prefers w_i to w and m prefers w to w_{i+1}, then G contains an edge from π to π_i.

Note that G is defined to contain only one copy of any edge, even though the same edge may be specified more than once by the above rules.

Figure 2 shows graph G constructed from the problem instance and rotations of figure 1; edge $<\pi_3, \pi_4>$ is defined by an application of rule 2, while all the other edges are defined by rule 1. The graphs are not always so tree-like as in this example.

Lemma 5: G has only $O(n^2)$ edges; it can be constructed in time $O(n^2)$; and no node in G has outdegree more than n.

Proof: Given the rotations, which can be found in $O(n^2)$ time, we label each pair that is eliminated by some rotation with the name of the (unique) eliminating rotation. To do this we examine each rotation π, and for each woman w in a pair in π we note the men that π moves w over; each of these pairs is labeled with π. Since π eliminates a set of pairs corresponding to a contiguous sequence of men in w's preference list, finding these pairs takes constant time per pair. Then, since no pair is eliminated by more than one rotation, these labelings can be done in $O(n^2)$ total time.

Now G can be constructed by processing each man m's list top down, keeping a mark on the most recently encountered stable pair in m's list. When a new stable pair is encountered, we create an edge in G from the rotation labeling the marked pair (if there is one) to the rotation labeling the new pair, and we update the mark. When a non-stable pair is encountered, we check (in unit time using a random access list of the rotations) if its label has already been encountered in m's list. If not, then we create an edge in G from the rotation labeling the marked pair (there will be one) to the rotation labeling the current non-stable pair. Each scan down a man's list takes O(n) time, hence $O(n^2)$ time in total.

Since the total time to build G is $O(n^2)$, it can only have $O(n^2)$ edges. It is also clear from the details above that for any rotation π, the scan down a given man m's list adds at most one edge out of π, hence the outdegree of any node in G is bounded by n, the number of men. □.

To complete our claims about G, we need the following

Lemma 6: For any two rotations π and ρ, π precedes ρ if and only if π reaches ρ by a directed path in G, hence the transitive closure of G is D, and so the closed sets of G and D are identical.

Proof: Clearly, G is a subgraph of D since each edge in G specifies a precedence relation between the rotations at the endpoints of the edge. To prove the other direction, it suffices to show that if π *explicitly* precedes ρ then π reaches ρ in G. By definition of "explicitly precedes", there must be two women w and w^* such that (m,w) is eliminated by π, and ρ moves m to w^*, and m prefers w to w^*. Then w^* is in W(m), and $\rho \in \Pi(m)$; say $\rho = \pi_{i^*}$, where, by definition, ρ moves m from w_{i^*} to w^*. So in G there is a directed path from π_i to ρ for every $\pi_i \in \Pi(m)$ such that $i < i^*$.

Now let w' be the woman most preferred by m such that (m,w') is eliminated by π. By construction of G, there is an edge (associated with the pair (m,w')) from π to $\pi_{i'}$ for some $\pi_{i'} \in \Pi(m)$; let $w_{i'}$ be the woman that $\pi_{i'}$ moves m from. So if $i' \leq i^*$ (i.e. $w_{i'}$ is equal to or is preferred to w_{i^*}), then there is a directed path in G from π to ρ. But m prefers $w_{i'}$ to w, and w^* to w, and since, by the actions of algorithm A, man m is moved over any particular woman by at most one rotation, w_{i^*} cannot be preferred to w'; hence $w_{i'}$ must either be w_{i^*} or be preferred to w_{i^*}, and the lemma follows. □

Note that G is not necessarily the transitive reduction of D. As stated above, any subgraph of D whose transitive closure is D, preserves the closed sets, and since we want a sparse subgraph, the transitive reduction of D would be the best. However, general algorithms to produce the transitive reduction of D (even assuming D is given) would take much more than the $O(n^2)$ time to construct G. Perhaps the transitive reduction of D can be computed from the rotations in $O(n^2)$ time for this special problem, but G is sufficient for the needs of this paper.

4.1. The enumeration algorithm

We will first describe how to use G to build a tree T with root r, where every edge in T is labelled with a rotation, such that the path from the root to any node in T enumerates a distinct closed set SN of rotations in G (and D), and such that each closed set in G is enumerated in this way. Hence by theorem 7, there is a one-one correspondence between the nodes of T and the set of all stable marriages. Further, the order of the rotations along any path will be such that if π is a rotation on a given edge $e = \langle x,y \rangle$, then all rotations that precede (in the partial order of rotations) π will be on the path from the root to x. It follows inductively that the stable marriage corresponding to any node x can be explicitly constructed by starting at the root and successively executing the moves dictated by each rotation on the path to x. Since each such change takes O(n) time, and each node in T corresponds to a distinct stable marriage, it follows that all the stable marriages can be output in O(n) time per marriage, once T has been constructed. In obtaining the output, if T is traversed depth first, then only one complete marriage must be known at any time (the previous, as well as the next, marriage can be obtained from the marriage and the relevant rotation), hence only O(n) additional space is needed for the traversal of T.

Building T

First, we label the rotations numerically according to a topological ordering of G, i.e. every node has a larger label than any of its predecessors. It is well known that these labels can be found in linear time in the number of edges of G, hence in $O(n^2)$ time. There are ways to avoid topological labelling, but the exposition becomes more complex.

To build T, we start at the root r and successively expand from any unexpanded node y in T as follows: Let R(y) be the rotations along the path from r to v in T, and let e = <x,y> be the last edge on this path. Let MR(y) be the set of maximal rotations (nodes in G with indegree zero) when all the rotations in R(y) are removed from G, and let LR(y) be those rotations in MR(y) whose label is larger than the label on edge e. Then y is expanded by adding |LR(y)| edges out of node y, each labelled with a distinct rotation in LR(y).

Lemma 7: Given G, T can be constructed in O(n) time per node.

Proof: We give here more implementation detail on expanding a node. Let e = <x,y> be the last edge on the path to y, and let the rotation on e be π. We will assume, for now, that at node x in T, there is a graph G(x), obtained from G by deleting all nodes in R(x), and all incident edges. We also assume that the indegree of each node in G(x) is known. Then LR(y) is the set of all neighbors of π in G(x) which have indegree 1 (note that these all have larger label than π due to the topological labeling of G), together with the set of rotations in LR(x) whose label is larger than π. The first set can clearly be found in O(n) time since no node in G (hence in G(x)) has outdegree more than n, i.e. there are at most n neighbors of π in G(x). For the second set, we claim that $|LR(x)| \leq n$, hence we can simply scan LR(x) to find those rotations with label larger than π. To see that $|LR(x)| \leq n$, note first that for any fixed m, if (m,w) and (m,w′) are two pairs in (necessarily) distinct rotations, then one of these two rotations must precede (in the partial order) the other. But, by construction or induction, each pair of rotations in LR(x) must be incomparable, and so for any man m, m is in a pair in at most one rotation in LR(x).

So far, we have seen that if G(x) is given at node x, then the edges out of x can be determined and labeled in time O(n). However, constructing the graphs at each of the endpoints of these edges must be done with some care. For example, if T is built in a breath first manner, then |LR(x)| graphs have to be constructed and stored. In addition to the enormous space this would require, it also would need more than O(n) time per node, since the graphs can have $\Theta(n^2)$ nodes and edges. The solution is to expand T depth first: to expand a given node x in T, we find all the maximal elements in G(x) and store them (essentially, constructing all the edges out of x), but we construct a new graph G(y) for only one edge <x,y> out of x; node y is the next node in T to be expanded. Graph G(x) can be transformed into G(y) in O(n) time, by deleting node y and all incident edges from G(x); the indegree in G(y) of each neighbor of x is one less than its indegree in G(x), and all other indegrees remain as in G(x), so the indegrees are also maintained in O(n) time. Backing up from y to x, we use G(y) and the rotation on edge <x,y> to reconstruct G(x) in time O(n). Knowing G(x) and the untraversed edges out of node x, we choose an unexpanded child y′ of x, transform G(x) into G(y′), and then expand y′. So T can be built in O(n) time per node. □

Corollary: Given G, the set of all stable marriage can be enumerated in O(n) time per marriage, and $O(n^2)$ total space.

Proof: The approach above was to first build T, and then to traverse it depth first to explicitly construct the stable marriages. However, T was built depth first in order to obtain the O(n) time bound, so we can construct the stable marriages as we build T. But then, we never need to know the complete T at any given time. What is sufficient at any one time, staying close to the above details, is the path from r to the current node being expanded, and the edges (with their rotations) which directly hang off of that path. The depth of T is at most $O(n^2)$ since each edge on a path corresponds to a distinct rotation, and the outdegree of each node in T is O(n), hence if we construct and output the stable marriages as T is (implicitly) being built, depth first, then we need only $O(n^3)$ space for T. However, all other space use is $O(n^2)$, and the total space bound could be reduced to that if we didn't store the maximal elements of G(x) at each node x. These had been stored to facilitate the backup to x, and the next traversal out of x, and to simplify the exposition. But, when backing up from node y to x, where edge <x,y> is labeled with rotation π, the maximal elements of G(x) can be found from G(y) in O(n) time, since they are the maximal elements of G(y), plus π, minus the neighbors of π in G(x). So both G(x) and its maximal elements can be recomputed in O(n) time on backup. However, we must be careful that no edge out of x is traversed more than once. There are several ways to do this. One simple way is to traverse the edges out of x in increasing order of their labels; each time we enter x we scan the maximal elements of G(x) and choose the one with the smallest label larger than the label on the edge just used to enter x (either backup or first entry). In this way, only a single path of T needs to be kept at any one time, hence the total space is $O(n^2)$, and the time remains O(n) per node. □

Figure 3 shows the tree T built from graph G of figure 2. Each node in T is labelled with the corresponding stable marriage from figure 1.

We still need to show that the nodes in T correspond one-one to the closed sets of G, and that the order of the rotations along a path in G has the desired properties claimed above. This is done in the following lemmas.

Lemma 8: Let x be an arbitrary node in T. Then R(x) is a closed set of rotations in D (hence G).

Proof: By induction on the length of the path to x. The lemma is clearly true for the root, which corresponds to the empty set, and for nodes at distance one from the root, for each of these correspond to a maximal rotation in D. Now let x be a node at distance k from r, and let <x,y> be an edge out of x with label π. By inductive hypothesis, R(x) is a closed set, and, by construction, π is maximal in G(x), so all the predecessors of π are in R(x), and hence R(x) + $\{\pi\}$ is a closed set in D, and this set corresponds to node y. □

Corollary: The stable marriage corresponding to node y can be constructed by traversing the path from r to y, successively making the moves dictated by the rotations on the path.

Lemma 9: Every closed set in D is R(x) for some node x in T.

Proof: By induction on the size of the set. The lemma is clearly true for size zero and one, since these sets are the empty set and the maximal elements in D. Now suppose the lemma holds for sets of size k, and let SN be a closed set of size k+1. SN must have a minimal element with respect to the partial order D; let π be the minimal element of SN with the largest label. By the induction hypothesis, SN - $\{\pi\}$ is R(x) for some node x in T. But then, π is a maximal node in G(x), and since it has the largest label of the rotations in SN, it will label an edge <x,y> out of x. Hence SN is R(y). □

Lemma 10: Let x and x′ be two distinct nodes in T, then R(x) $\neq$ R(x′), hence no closed set is enumerated twice in T.

Proof: Consider a node x and two edges <x,y> and <x,y′> out of x labelled π and ρ, where π has a smaller label than ρ. Note that the labels along any path from r are in increasing order, hence π cannot appear in the subtree of T rooted at y′. The lemma follows by applying this observation inductively on the length of the paths. □

Constructing the Hasse diagram

We will not give any details here, but we claim that the Hasse diagram of the set of all stable marriages can also be explicitly constructed in $O(n^2 + n|S|)$ time, and (excluding the space for the lattice itself) with modest space. To see that this is plausible, note that each node in the Haase diagram can have outdegree of at most n, since each node is associated with a stable marriage, and each edge out of the node is associated with a rotation exposed in that marriage, and there clearly can be no more than n/2 rotations exposed in any stable marriage. Hence the size of the lattice itself is at most $O(n|S|)$.

5. The minimum regret stable marriage problem

Definition: For stable marriage M containing the pair (m,w), the *regret* of m is the position of woman w in m's list, and the regret of w is the position of man m in w's list. The regret of a marriage M, denoted r(M), is defined to be the maximum regret of any person, given the pairing in M, i.e. M is measured by the person who is worst off in it.

Knuth [K] discusses the problem of finding a stable marriage which minimizes r(M) over all stable marriages. The solution given in [K] is attributed to Alan Selkow and naive analysis of it gives a running time of $O(n^4)$. Here we use the breakmarriage operation and corollary 1 to obtain a method that runs in time $O(n^2)$. It is corollary 1 that allows this speed up by avoiding duplicated proposals and rejections.

For ease of exposition, we will break the problem into two problems: find, if one exists, a marriage minimizing r(M) over all stable marriages in which at least one woman is a person of maximum regret in the marriage, and find, if one exists, a marriage minimizing r(M) over all stable marriages in which at least one man is a person of maximum regret in the marriage. Let the first type of marriage be called *woman regret* minimum, and the second type be called *man regret* minimum. Note that for a given problem instance, it is possible that in every stable marriage, all people with maximum regret are women (men) and hence there is no man (woman) regret minimum. This happens if and only if all people of maximum regret in the woman (man) optimal marriage are women (men). These cases are easy to check and adjust for in the algorithm so, for ease of the exposition, we will assume that neither of these two cases occur, and hence both a woman regret and a man regret minimum marriage exist. The minimum regret stable marriage is obtained from these two marriages. The following algorithm finds a woman regret minimum, assuming both a woman and a man regret minimum exist.

Algorithm B

0. Find the man optimal stable marriage M_0, and find the woman optimal stable marriage. Set i = 0.

1. Let w be a woman with regret $r(M_i)$ in M_i, and let m be her mate in M_i. If m and w are a pair in the woman optimal marriage, then stop and output M_i; M_i is a woman regret minimum. Else perform operation breakmarriage(M_i,m) and let M_{i+1} be the resulting stable marriage.

2. If there are no women with regret $r(M_{i+1})$ in M_{i+1}, then stop and output M_i; marriage M_i is a woman regret minimum. Else set i = i+1, and go to step 1.

Correctness of Algorithm B

First, the algorithm must terminate since, after each breakmarriage operation, any woman with a new mate prefers him to her previous mate, and any man with a new mate prefers his previous mate. So unless the conditions in step 1 apply, ultimately there will be no women with the maximum regret, and then the conditions of step 2 will apply. Note that by corollary 2, each breakmarriage operation in the algorithm results in a stable marriage. Now if the algorithm terminates in step 1, then M_i is woman regret minimum, for there is no stable marriage in which w has a better mate than m. Hence we will assume that the algorithm terminates in step 2.

Let M_0, M_1,...,M_z be the sequence of marriages produced by the algorithm (hence the algorithm outputs M_{z-1}). To prove that M_{z-1} is woman regret minimum we need the following two lemmas.

Lemma 11: For i from 0 to z-2, if M_i is woman regret minimum then so is M_{i+1}.

Proof: All woman either have the same mate in both marriages or prefer their mate in M_{i+1}, and, by the algorithm, for i from 0 to z-2, there is a woman in M_{i+1} with regret $r(M_{i+1})$. □

Lemma 12: For every i from 0 to z-1, either M_i is woman regret minimum or it dominates all marriages which are.

Proof: This is clearly true for M_0 since it dominates all marriages. Suppose the claim holds through M_k; let w be a woman with regret $r(M_k)$ in M_k, and let m be her mate in M_k. Now if M_k is not woman regret minimum, then w cannot be married to m in any marriage which is. Hence we know that M_k dominates all woman regret minimum marriages, and that m and w are not paired to each other in any of these marriages. But by corollary 1, breakmarriage(M_k,m) results in a marriage which dominates all marriages which are dominated by M_k and in which m is not married to his mate in M_k. Hence M_{k+1} dominates all woman regret minimum marriages. Hence the lemma follows by inductively applying this argument until either i = z-1, or until M_i is woman regret minimum, where lemma 11 applies. □

Now we can prove correctness of the algorithm.

Theorem 8: The marriage M_{z-1} is woman regret minimum.

Proof: Let MW be a woman regret minimum marriage. Suppose M_{z-1} is not woman regret minimum and let (m,w) be a pair in M_{z-1} where w is a woman with regret $r(M_{z-1})$ in M_{z-1}. Then we know that M_{z-1} dominates MW (from lemma 12), and that m is not married to w in MW (by the assumption that M_{z-1} is not woman regret minimum), so, by corollary 1, M_z dominates MW, and, by corollary 0, MW dominates M_z from the women's point of view. But, by the algorithm, there are no women among the people of maximum regret in M_z, and so (again by corollary 0) there can be no women among the people of maximum regret in any marriage dominated by M_z, in particular, in MW. But this is a contradiction, since at least one woman in MW must have regret r(MW), or else MW is not even in the set of stable marriages over which the woman regret minimum is defined. Hence either M_{z-1} is woman regret minimum, or none exist, so, given our assumptions, the algorithm is correct. □

Implementation and Time Analysis of Algorithm B

The algorithm moves M_0 towards M_t using breakmarriage operations, hence the total number of proposals in $O(n^2)$. Step 0 of the algorithm clearly requires only time $O(n^2)$, but steps 1 and 2 must be implemented with some care in order to obtain an overall $O(n^2)$ time bound. In each iteration of steps 1 and 2 the maximum regret of the men and of the women must be determined and compared, and a woman with overall maximum regret, if one exists, must be found. Simple scanning of the men and women at each iteration would lead to a bound of $O(n^3)$ ($O(n)$ time per iteration, and $O(n^2)$ iterations). Below we sketch the details that give a time bound of $O(n^2)$ and a space bound of $O(n)$ (not counting the space for the preference lists).

At the start of each step 1, the status of the women in the current marriage will be represented by n linked lists, one for each level of regret, where each list i links together (in no particular order) all of the women with regret i in the current marriage. For each i, we let c(i) be the number of women in list i; variable wr keeps the largest i such that $c(i) \neq 0$. We also need two n length vectors of pointers, one to point to the current location of each woman in the list that she is presently in, and one to point to the head of each list. Clearly all the lists and pointers take $O(n)$ space and can be initiated in $O(n)$ time. An identical data structure is kept for the men; K(i) is the number of men in list i, and mr is the largest i such that $K(i) \neq 0$. Of course we also need to record who the paris are in the current marriage, and other information needed to efficiently execute breakmarriage operations, but these details are assumed, since they are trivial and were needed in algorithm A.

Given the above data structures, a woman of regret r(M) in the current marriage M is found at the head of the women's list wr. After a breakmarriage operation, the women with new mates are removed from their current lists (in constant time per woman using the vector of pointers), and inserted at the heads of the appropriate new lists, and the variables c(i) are adjusted. If c(wr) is now zero, then i is decremented from wr until $c(i) \neq 0$ is found, and wr is updated. Since wr only decreases during algorithm B, the overall time for this search is $O(n)$. The men's lists are similarly updated after the breakmarriage, but for any k(i) which changes from zero to a positive count, mr is set to max(mr,i). Step 2 is implemented by comparing wr to mr. When computing the man regret minimum, the roles of men and women and their respective data structures are interchanged.

6. Open question

Consider the following problem: Find a fast algorithm to determine if an input marriage is stable. One obvious way to test for stability is to examine each man m to see if there is a woman w who m prefers to his mate in M, such that w also prefers m to her mate in M. With the obvious storage of the preference lists, each check takes unit time. One can also check each woman's preferences in the above way, but note that we need only check from the perspective either of the men or from the perspective of the women, but not both. If we define r(M,i) to be the number of people who person i prefers to their mate in M, then stability can be checked from the perspective of the men in time $\sum_{\text{all men } m} r(M,m)$. Naively, this could be as many as $n(n-1) = O(n^2)$ checks (table look-ups). Is there a way to beat this bound? Particularly, if preprocessing is allowed, say to build D, or some other "reasonable" work, can the $O(n^2)$ bound per marriage be reduced?

The worst case bound of n(n-1) is, in fact, not optimal: no more than n(n-1)/2 checks are needed, and these can be found and done in that time.

Lemma 13: If M is a stable marriage, then $\sum_{\text{men } m} [r(M,m)] + \sum_{\text{women } w} [r(M,w)] \leq n(n-1)$. Hence one of $\sum_{m} [r(M,m)]$ or $\sum_{w} [r(M,w)]$ is less than or equal to n(n-1)/2.

Proof: If M is a stable marriage and man m prefers woman w to his current mate, then woman w must not prefer man m to her current mate, and similarly, if woman w prefers m to her mate, them m must not prefer w to his mate. Hence the pair (m,w) can contribute at most one to r(M,m) + r(M,w) and the lemma follows. □

Hence we can test for stability by first computing $\sum_{m} r(M,m)$ and $\sum_{w} r(M,w)$ (preprocessing permits us to construct the correct data structure so that this sum can be done in O(n) time). If they sum to more than n(n-1), then M isn't stable. If the sum is less than or equal to n(n-1), then we check for stability from the perspective of the sex with smallest sum.

7. Appendix

The $\Omega(n^3|S|/[\log |S|]^2)$
time bound for the enumeration algorithms in [K] and [MW]

Definition: Assume the men and women are numbered from 1 to n. Let M be a stable marriage, and let S(M,i) be the set of all stable marriages which are dominated by M, and in which all men from 1 to i-1 have the same mates as in M.

Procedure E(M,i) will enumerate each element in S(M,i) exactly once. Clearly to enumerate all the stable marriages we call E(M_0,1), where M_0 is the man optimal marriage. The procedure is defined recursively, and is essentially the algorithm given in [K], but with terminology consistent with the present paper.

Procedure E(M,i):

```
If i = n then output M and terminate E(M,i);

Else, call E(M,i+1).
Upon return from E(M,i+1) do:

  If i is married to his mate in M_t
  (the woman optimal marriage), then terminate E(M,i);

  Else execute operation breakmarriage(M,i)
  with the following modification:  if ever a man numbered
  less than i becomes free, then stop the breakmarriage
  operation and terminate E(M,i).

  If Breakmarriage(M,i) terminates naturally, i.e.
  with new marriage M', then call E(M',i).
  Upon return from E(M',i), terminate E(M,i).
```

The algorithm may be a little clearer if the first line were "If i > n ..."; the present version is correct, and somewhat faster.

It is stated in [K] that this method has worst case running time of $O(n^2|S|)$. We show here that this is incorrect, and that the running time is $\Omega(n^3|S|/[\log |S|]^2)$. The analysis (and the exposition) here is very sloppy, and it is likely that the bound could be improved by better analysis of the constructions, or by another construction; our purpose in discussing it is only to compare to and motivate the $O(n^2 + n|S|)$ time bound in the present paper.

First, we have to be clear on the semantics of the bound $O(n^2|S|)$. In detail, it asserts that there exists positive integers q, n_0 and s_0, such that for any problem instance P with $n > n_0$ men, where there are $|S| > s_0$ stable marriages for P, the running time of the above algorithm is less than or equal to $qn^2|S|$. Hence to show that $O(n^2|S|)$ is not a correct upper bound, it suffices to show that for any s_0, there exists a family of problem instances P, such that each has $|S| > s_0$ stable marriages, and, as a function of n, the time needed for the above algorithm grows faster than any constant times n^2. We start with $|S| = 2$, and for illustration, $n = 5$.

The mens list is:	The womens list is:
1: 1 x x x x 2	1: 5 1 x x x
2: 2 x x x x 3	2: 1 2 x x x
3: 3 x x x x 4	3: 2 3 x x x
4: 4 x x x x 5	4: 3 4 x x x
5: 5 x x x x 1	5: 4 5 x x x

where the x's are arbitrary as long as each row is a permutation of 1 through 5.

In this construction, each man gets his first choice in the man optimal marriage, and each woman gets her second choice; in the woman optimal marriage each woman gets her first choice, and each man his last. There is only one rotation exposed in M_0 (it is {(5,5) (1,1) (2,2) (3,3) (4,4)}), and it moves the woman to the woman optimal marriage, hence there are only two stable marriages in the construction. The crucial property of this construction is that for any man m, during breakmarriage(M_0,m) man m will be rejected by all but his last choice woman.

What happens when E(M_0,1) is executed on this constructions is that E(M_0,1) calls E(M_0,2) which calls E(M_0,3),..., which calls E(M_0,5) which outputs M_0 and terminates. Then E(M_0,4) is reentered and man 4 is rejected by all the women below woman 4 until woman 5, his last choice, is encountered; she then accepts 4 and rejects man 5; man 5 is then rejected to by all the women until his last choice, woman 1, who accepts 5 and rejects 1. However, man 1 is now free, and $1 < 4$, hence E(M_0,4) terminates and E(M_0,3) is reentered. Man 3 is rejected by all the women in his list until his last choice, woman 4, and this frees man 4. But since we are in E(M_0,3), man 4 is again at the top of his list, so exactly the same series of proposals and rejections are made that were made in E(M_0,4). After these are completed, when man 1 become free, E(M_0,3) terminates and E(M_0,2) is reentered. Man 2 makes four proposals freeing man 3, and then the same sequence of proposals are made as were made in E(M_0,3). Continuing in this way we see that man 5 becomes free four times, man 4 four times also, man 3 three times, man 2 twice, and man 1 once. Each time a man becomes free, exactly four proposals are made before the next man becomes free. Hence the number of proposals is 4C(5,2) - 4, where C(5,2) is the binomial coefficient "5 choose 2".

For $|S| = 2$, and n arbitrary, the first choice of each man i is woman i, the second choice of each woman i is man i, the last choices of the men are a clockwise rotation of their first choices, and the first choices of the women are a counter-clockwise rotation of their second choices. In this construction, each man i becomes free i times (except for man n who is free n-1 times), and each time a man is free he makes n-1 proposals before being accepted. Hence the total number of proposals is roughly $(n-1)n^2$. We will call the set of these proposals a "phase", which will be useful below.

Now for larger values of |S| we do the following. Assume $|S| = 2^k$ for some k. To generate an n person instance, for n divisible by k, divide the men and women into k groups of n/k men and women each, where the first group has men and women 1 through n/k, the second has men and women n/k + 1 through 2n/k etc. The preference lists of each group are each a copy of the construction above for n/k men and |S| = 2, with the appropriate renumbering of the names in each copy. Each copy contains exactly one rotation exposed in M_0, and each of these rotations moves the men and the women in the group to the women optimal marriage, hence there are exactly 2^k stable marriages in this construction, as claimed.

For simplicity, we only examine the number of proposals in the last group. What happens over execution of the algorithm is that the marriage status of these men will be reset to M_0 |S|/2 times (plus or minus one), and after each time, the men in this group will execute a sequence of proposals which is identical to a phase discussed above, for a problem instance with n/k men. Hence the number of proposals is (very sloppy) $(n-1)(n/k)^2|S|/2$, and this can be taken to be about $n^3|S|/[\log|S|]^2$. So the work need in this algorithm is $\Omega(n^3|S|/[\log|S|]^2)$.

Very naive analysis of the algorithm gives $O(n^3|S|)$ as an upper bound on the time, but I make no claims for it being tight.

The enumeration algorithm of [MW] is essentially the same as the one above, except that operation breakmarriage(M,i) is modified so that it terminates if a man with number *greater* (rather than less) than i is made free. The same construction above forces this algorithm to take $\Omega(n^3|S|/[\log |S|]^2)$ time, and it is easy to show an upper bound of $O(n^3|S|)$ time.

8. References

[GS] D. Gale, and L. Shapley. College Admissions and the Stability of Marriage. Am. Math. Monthly 69 (1962).

[GS84] D. Gale, and M. Sotomayor. Some Remarks on the Stable Mathching Problem. Discrete Applied Math (to appear).

[IL] R. Irving and P. Leather. The complexity of counting stable marriages. SIAM J. on Computing, to appear.

[ILG] R. Irving, P. Leather and D. Gusfield. An algorithm for the optimal stable marriage problem. July 1985.

[K] D. N. Knuth, *Mariages Stable* (in French). Les Presses de L'Universite de Montreal, 1976.

[L] E. Lawler, *Combinatorial Optimization: networks and matroids*. Holt, Rhinehard and Winston, 1976.

[MW] D. McVitie and L. Wilson. The stable marriage problem. CACM, July 1971.

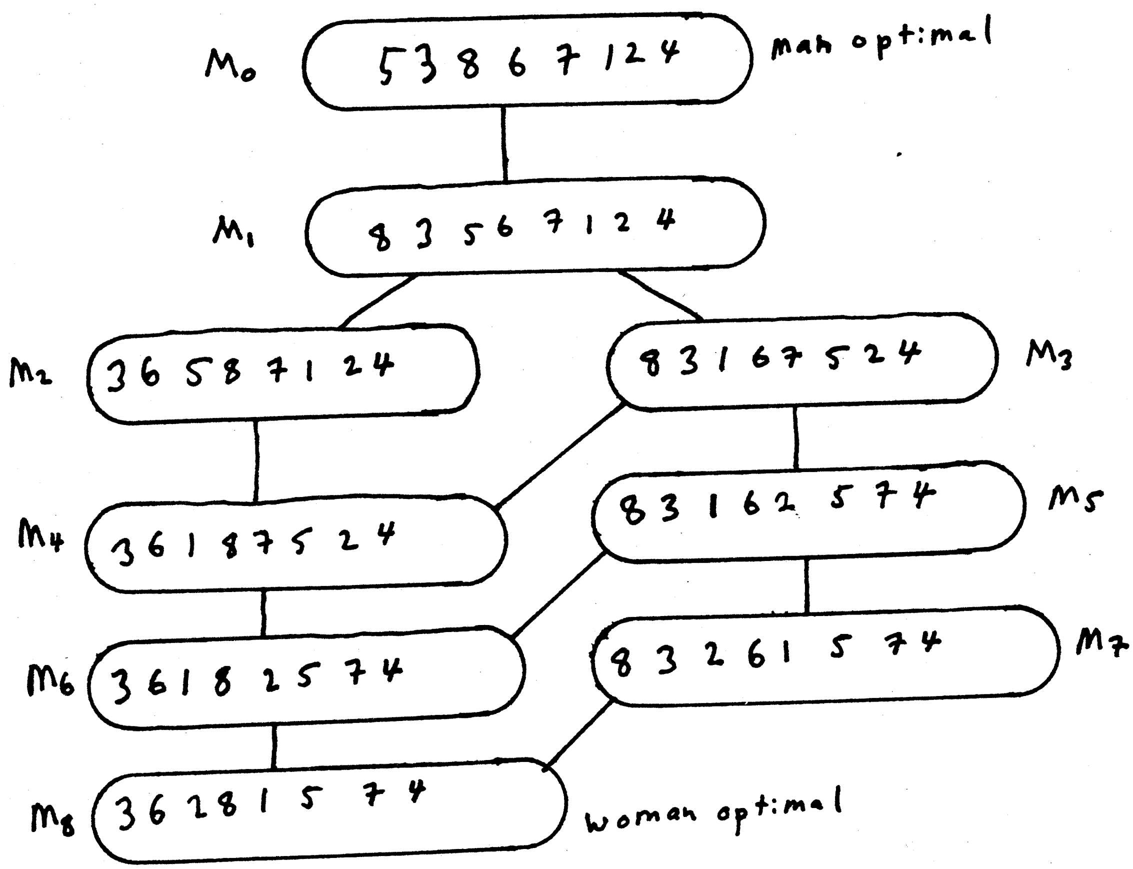
[R] A. Roth. The Evolution of the Labor Market for Medical Interns and Residents: A Case Study in Game Theory. Working paper #165, Dept. of Economics, University of Pittsburgh.

[W] N. Wirth. *Algorithms + Data Structures = Programs*. Prentice-Hall, 1976.

9. Acknowledgements

I would like to thank Dana Angluin, Gregory Sullivan and Robert Irving for listening to parts of this paper as it developed. I would like to especially thank Carrie Shepard for helping me to read [K] and for interesting me in the practical and personal problem of marriage.

Men's preference lists	Women's preference lists
1: 5 7 1 2 6 8 4 3	1: 5 3 7 6 1 2 8 4
2: 2 3 7 5 4 1 8 6	2: 8 6 3 5 7 2 1 4
3: 8 5 1 4 6 2 3 7	3: 1 5 6 2 4 8 7 3
4: 3 2 7 4 1 6 8 5	4: 8 7 3 2 4 1 5 6
5: 7 2 5 1 3 6 8 4	5: 6 4 7 3 8 1 2 5
6: 1 6 7 5 8 4 2 3	6: 2 8 5 4 6 3 7 1
7: 2 5 7 6 3 4 8 1	7: 7 5 2 1 8 6 4 3
8: 3 8 4 5 7 2 6 1	8: 7 4 1 5 2 3 6 8



The Hasse Diagram of the set of all stable marriages. The number in position i of any list indicates the woman married to man i in that marriage.

Figure 1.

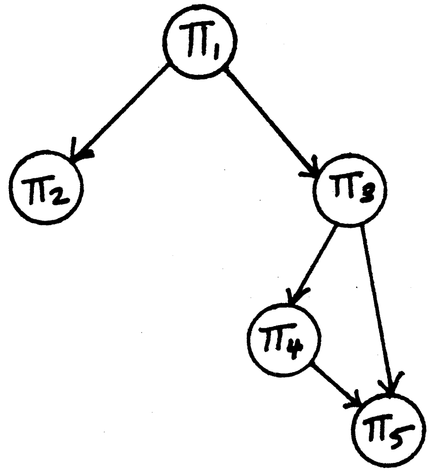


Figure 2: Graph G

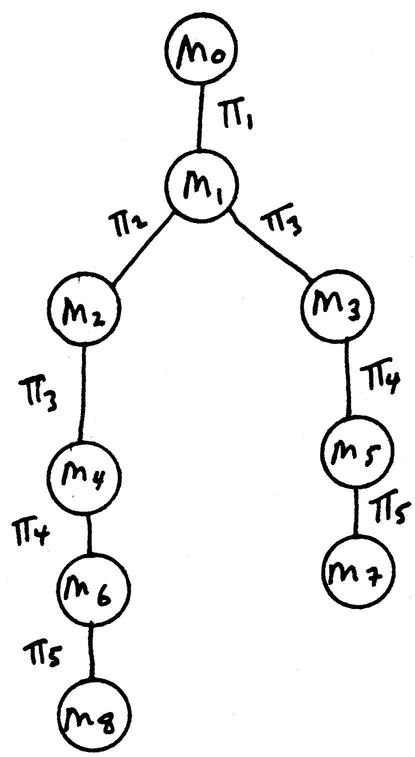


Figure 3 : Tree T with rotations on edges and stable marriages at nodes.