Morehead Electronic Journal of Applicable Mathematics
Issue 2 — MATH-2001-06

Stable Matchings in the Couples Problem[1]

Danielle Bianco — Gwynedd-Mercy College, Gwynedd Valley, PA
Stephen Hartke — University of Dayton, Dayton, Ohio
Anne Larimer — Arizona State University, Tempe, AZ

Abstract: The National Resident Matching Program (NRMP) is a centralized service which matches graduating medical students to hospital residency positions. The NRMP has been successful in producing stable matchings between individuals and hospitals, meaning that no student and hospital both prefer each other to their assigned matching. Recently student couples have been able to submit joint preference lists to obtain positions in close proximity. With the addition of student couples, stable matchings are not guaranteed. In this paper acceptability graphs will be used to characterize the existence of stable matchings in the couples problem. These characterizations can be used to generate instances of the couples matching problem for which it is known whether or not a stable matching exists.

1 Introduction

Several stable matching problems have been studied; the most well-known is the stable marriage problem. The marriage problem was introduced in 1962 by Gale and Shapley (1962); it describes a situation involving a community of m men and n women. Each person creates a preference list by ranking the members of the opposite sex according to his or her preferences for a marriage partner. A matching is then created based on these preferences. A stable matching occurs when the men and women are paired in a way such that no man and woman both prefer each other to their actual mates. Gale and Shapley proved that a stable matching exists for any set of preference lists by devising an algorithm which always produces a stable matching.

The National Resident Matching Program (NRMP) is a real-life situation involving the stable matching problem. The NRMP was developed in 1951 after years of turmoil involving the matching of medical students to residency positions at hospitals. The NRMP is still used today to fill over 20,000 positions a year (Roth, 1990).

[1]Research supported in part by an NSF REU grant no. DMS-9424098 at Lafayette College, 1997.

Originally, the NRMP used an algorithm that was equivalent to Gale and Shapley's algorithm for the stable marriage problem (Roth, 1984). Using Gale and Shapley's work, it can be shown that there exists a stable matching between hospitals and students for every preference list. In recent years, however, the NRMP has modified its algorithm to accommodate changes in the residency labor market. One of these changes has allowed couples to submit joint preference lists so that the members of the couple will be matched to residency positions in close proximity. With the addition of couples, several results about stable marriages no longer apply to the NRMP algorithm (Roth, 1984; Ronn, 1990; Aldershof and Carducci, 1996). In particular, Roth has shown that a stable matching may not exist, and Ronn has shown that determining whether a stable matching exists is NP-complete.

Formally, the objective of a stable matching problem is to create a stable set of pairings of participants. In the marriage problem, pairings are created when members from two disjoint sets, often referred to as men and women, are matched together. The two sets matched by the NRMP are students and hospital residency positions. Each participant creates a strictly-ordered preference list of a subset of the opposite set. If a student s does not list a position offered by hospital h or h does not list s, then s and h are not *mutually acceptable* and can never be assigned to each other in a matching. The couples problem differs in that a couple c_i, consisting of students s_{2i-1} and s_{2i}, submits a joint preference list of ordered pairs of hospitals. For example, couple c_1's preference list might be (h_1, h_2), (h_1, h_3), (h_4, h_5), $\ldots$. Thus, as a couple, c_1 is indicating that their first choice would be for s_1 to be matched with h_1 and s_2 to be matched with h_2. If one or both of those assignments are not possible, then their joint second choice is for s_1 to be matched with h_1 and s_2 to be matched with h_3. We will say $c_i = (s_{2i-1}, s_{2i})$ is matched with (h_j, h_k) when s_{2i-1} is assigned to h_j and s_{2i} is assigned to h_k. When a couple can become matched to a higher preference (h_a, h_b) than the preference they are assigned to, an instability exists. Couple c_i can become matched to a higher preference (h_a, h_b), if either (i) h_a is matched to student s_{2i-1} and h_b prefers s_{2i} to its current assignment; or (ii) h_a prefers s_{2i-1} to its current assignment and h_b is matched to s_{2i}; or (iii) h_a prefers s_{2i-1} to its current assignment and h_b prefers s_{2i} to its current assignment. Thus couple c_i and hospitals h_a and h_b are all at least as well off as they were immediately preceding the change.

The roommates problem is another variation of the stable matching problem where only one set of n individuals rank each other. Gale and Shapley (1962) first gave an example of the roommates problem where a stable matching does not exist. To investigate the existence of a stable matching, Abeledo and Isaak (1991) defined an acceptability graph where a node represents an individual and an edge signifies that two individuals are mutually acceptable. Using this definition, they proved that a stable matching is guaranteed for all possible preference lists in the roommates problem if and only if the acceptability graph is bipartite. In this paper a similar result for the couples problem will be shown characterizing the existence of a stable matching for all possible preference lists based on an acceptability graph.

2 Preferences and Acceptability Graphs

When couples and hospitals submit their preference lists, the lists are purged of all preferences containing students and hospitals who are not mutually acceptable. Hospitals are assumed to have the unmatched preference u at the end of their lists, and couples are assumed to have the unmatched preference (u, u) at the end of their lists. Couples can also include preferences where one student uses the unmatched preference u. This preference u can be considered to be a hospital that will accept any student and that has an unlimited quota. In this paper, a hospital refers to a particular residency position and so has a quota of one student.

An acceptability graph can be formed using the purged preference lists, where each node of the graph represents either a hospital or a couple. An edge between a hospital node and a couple node signifies that the hospital and at least one student in the couple are mutually acceptable.

3 Acceptability Graphs with Cycles

Acceptability graphs are useful in analyzing the relationships between couples and hospitals. The existence of stable matchings is related to characteristics of acceptability graphs, namely the presence of a cycle in the graph. First, the implications of a cycle in the graph will be examined.

Initially we will consider an instance of the couples matching problem consisting of n couples and $2n$ hospitals. We will provide a set of preference lists for which the associated acceptability graph contains a cycle and show that these preference lists do not admit a stable matching. Then we will use this result to construct preference lists that do not admit a stable matching for any acceptability graph that contains a cycle.

Suppose we are to match n couples and $2n$ hospitals and the associated acceptability graph contains the cycle c_1, h_2, c_2, h_4, c_3, h_6, $\ldots$, c_n, h_{2n}. We want to find a set of preference lists for the couples and the hospitals so there is no stable matching. Each couple is adjacent to two hospitals in the cycle. This may be because the same member of the couple has ranked h_{2i-2} and h_{2i} in which case the couple is a *partial couple*; or it may be because one member of the couple ranked h_{2i-2} and the other ranked h_{2i} in which case the couples is a *full couple*. In constructing our preference lists, we will treat full couples and partial couples differently. Knowing whether couple c_i is a full couple or a partial couple also determines how to construct the preference lists of their associated hospitals h_{2i-1} and h_{2i}. Couples' preference lists often include options where one student in the couple is assigned a desirable position, but the other student is left unassigned. It is sometimes convenient to refer to a student being left unassigned as the student being assigned to position "unmatched." Hospital h_{2i} is in the cycle and so must be an actual hospital (may not be unmatched), but hospital h_{2i-1} is not in the cycle and may be unmatched. We will describe h_{2i-1}'s preference list as if it exists; if h_{2i-1} is unmatched, ignore its preference list. Subscript arithmetic is modular when appropriate.

If couple c_i is a partial couple, their preference list is:

$$\{(h_{2i}, h_{2i-1}), (h_{2i-2}, h_{2i-1})\}$$

and the associated hospitals' preferences are:

$$\begin{aligned} h_{2i-1}: &\quad \{s_{2i}\} \\ h_{2i}: &\quad \{s_{2i+1}, s_{2i-1}\} \end{aligned}$$

If couple c_i is a full couple, their preference list is:

$$\{(h_{2i-2}, h_{2i})\}$$

and the associated hospitals' preferences are:

$$\begin{aligned} h_{2i-1}: &\quad \{\} \\ h_{2i}: &\quad \{s_{2i+1}, s_{2i}\} \end{aligned}$$

where c_i consists of students s_{2i-1} and s_{2i}, $s_{2n+1} \equiv s_1$, and $h_0 \equiv h_{2n}$.

Lemma 1 *Given a situation of n couples and $2n$ hospitals ($n \geq 2$) that have preferences as described above, no stable matching exists if the number of full couples is odd.*

Proof. Let $|C_p|$ denote the number of partial couples and $|C_f|$ denote the number of full couples. Since there are only $2|C_p| + |C_f|$ distinct hospitals ranked in the couples' preference lists, there are only $2|C_p| + |C_f|$ acceptable hospitals for $2n$ students. The expression $2n - (2|C_p| + |C_f|)$, which equals $|C_f|$, is the minimum number of students who will be unmatched. Therefore, at least $\left\lceil \frac{|C_f|}{2} \right\rceil$ couples must be unmatched in a possible matching.

With these preference lists it can be shown that in every possible match, one of the following instabilities must occur:

1. If any partial couple c_i is unmatched, then an instability exists because couple c_i can be assigned to their second preference (h_{2i-2}, h_{2i-1}).

2. If an unmatched full couple c_i immediately precedes another unmatched full couple c_{i+1} in the cycle, then an instability exists because the first couple c_i can become matched to (h_{2i-2}, h_{2i}). This can occur since h_{2i} is unmatched and h_{2i-2} prefers s_{2i-1} most.

3. If an unmatched full couple c_i immediately precedes a partial couple c_{i+1} that is matched to their first preference, again couple c_i can become matched to (h_{2i-2}, h_{2i}). This can occur since h_{2i} is unmatched and h_{2i-2} prefers s_{2i-1} most.

4. If a partial couple c_i matched to their second preference immediately precedes an unmatched full couple or a partial couple matched to their first preference, then an instability exists because c_i can become matched to their first preference (h_{2i}, h_{2i-1}). This can occur since both h_{2i-1} and h_{2i} are unmatched.

To prove that a matching is unstable, assume that instabilities 1, 2, and 3 do not occur. It must then be shown that instability 4 occurs in that matching.

Because there are no unmatched partial couples (instability 1), then there are $\left\lceil \frac{|C_f|}{2} \right\rceil$ unmatched full couples. Since $|C_f|$ is odd, the number of unmatched full couples is greater than the number of matched full couples by one. Thus, there must be two unmatched full couples that do not have any matched full couples between them in the cycle. However, since no unmatched full couple immediately precedes another unmatched full couple (instability 2), there exit partial couples separating the two unmatched full couples. Label the ℓ partial couples $c_1, \ldots, c_\ell$, and label the unmatched full couples c_0 and $c_{\ell+1}$. Consider the k^{th} couple, where k is the largest integer less than or equal to ℓ such that c_k is not assigned to their first preference. The k^{th} couple exists because c_1 has their second preference (instability 3, and c_0 is unmatched). If $k = \ell$, then $h_{2\ell} = h_{2k}$ is unmatched, and c_k can become matched to their first preference (h_{2k}, h_{2k-1}). If $k < \ell$, then c_{k+1} has their first preference, and h_{2k} is unmatched. Thus, c_k can be matched with their first preference (h_{2k}, h_{2k-1}).

Note that if there is only one full couple in the cycle, then $c_0 = c_{\ell+1}$ and all partial couples are included in $c_1, \ldots, c_\ell$. □

Example 1 *Consider a problem with three couples, where there are two partial couples and one full couple.*

The preference lists are:

h_1	h_2	h_3	h_4	h_5	h_6
s_2	s_3	s_4	s_5		s_1
	s_1		s_3		s_6

$c_1 = (s_1, s_2)(partial)$	$c_2 = (s_3, s_4)(partial)$	$c_3 = (s_5, s_6)(full)$
(h_2, h_1)	(h_4, h_3)	(h_4, h_6)
(h_6, h_1)	(h_2, h_3)	

At least one couple must be unmatched in each possible matching. All matchings with only one couple unmatched, along with their instabilities are:

h_1	h_2	h_3	h_4	h_5	h_6	instability
s_2	s_1	u	s_5	u	s_6	$c_2 \to (h_2, h_3)$
u	s_3	s_4	s_5	u	s_6	$c_1 \to (h_6, h_1)$
s_2	s_3	s_4	u	u	s_1	$c_2 \to (h_4, h_3)$
s_2	u	s_4	s_3	u	s_1	$c_1 \to (h_2, h_1)$
s_2	s_1	s_4	s_3	u	u	$c_3 \to (h_4, h_6)$

Theorem 1 *For any acceptability graph that contains a cycle, there are preference lists such that no stable matchings exist.*

Proof. Consider a minimal cycle in the acceptability graph. Let the length of the cycle be $2n$ where n represents the number of couples.

If n is even, then find a couple in the cycle that is mutually acceptable to a hospital outside of the cycle. Label that couple c_1 and the hospital outside of the cycle h_1. If no such couple exists, then $h_1 \equiv u$. Label the rest of the couples in the cycle $c_2, \ldots, c_n$. Label the hospitals in the cycle $h_2, h_4, \ldots, h_{2n-2}, h_{2n} \equiv h_0$, where couple c_i is adjacent to h_{2i} and h_{2i-2}.

If n is odd, label the couples in the cycle $c_1, \ldots, c_n$. Label the hospitals in the cycle $h_2, h_4, \ldots, h_{2n-2}, h_{2n} \equiv h_0$, where couple c_i is adjacent to h_{2i} and h_{2i-2}.

The preference lists are based on the lists constructed in Lemma 1. To add extra edges to the acceptability graph, construct the preference lists as follows:

The i^{th} couple has the following preference list:

$$c_i = \begin{cases} \{(h_2, h_1), (h_{2n}, h_1), (h_x, h_y)\} & \text{if i = 1 and n is even} \\ \{(h_{2n}, h_2), (h_w, h_z)\} & \text{if i = 1 and n is odd} \\ \{(h_{2i-2}, h_{2i}), (h_w, h_z)\} & \text{if } 2 \leq \text{i} \leq \text{n} \end{cases}$$

where c_i consists of students s_{2i-1} and s_{2i}. Hospitals h_x and h_y both denote any hospital. There can be as many extra preferences added to the bottom of a couple's preference list as necessary to create the acceptability graph. Hospitals h_w and h_z both denote any hospital, where $w \neq 2i - 2$.

All other couples not in the cycle can have any preference list that creates the given acceptability graph.

The j^{th} hospital has the following preference list:

$$h_j = \begin{cases} \{s_2, s_x\} & \text{if j = 1} \\ \{s_3, s_1, s_x\} & \text{if j = 2 and n is even} \\ \{s_3, s_2, s_x\} & \text{if j = 2 and n is odd} \\ \{s_{j+1}, s_j, s_x\} & \text{if } 2 < \text{j} \leq 2\text{n} \end{cases}$$

where $s_{2n+1} \equiv s_1$ and s_x denotes any other student not already mentioned in that preference list. There can be as many students as necessary.

All hospitals not in the cycle can have any preference list that creates the given acceptability graph.

It remains to be shown that additions made to the preference lists in Lemma 1 to construct these preference lists do not create a stable matching.

Consider couple c_i who is matched to one of the added preferences.

If $i = 1$ and n is even, then an instability occurs because the couple can become matched to their second preference (h_{2n}, h_1). This can happen since h_{2n} ranks s_1 first and h_1 ranks s_2 first.

If $1 < i \leq n$ and n is even, the instabilities as described in the proof of Lemma 1 will occur, since no full couple has a preference of the form (h_w, h_z) where $w = 2i - 2$.

If n is odd, then there must be a full couple who is not matched to their first preference immediately preceding another full couple who is not matched to their first preference. This situation creates instability 2 from Lemma 1, and the first of the these two couples can become matched to their first preference, since the second full couple has no preference of the form (h_w, h_z) where $w = 2i - 2$. □

We can add individuals to this type of problem by viewing an individual as partnered with a dummy student who only ranks unmatched on its preference list. Thus an individuals student's preference list is of the form (h_1, u), (h_2, u), ... This "couple" can be involved in a cycle as a partial couple only, but it behaves like any other partial couple with u as an associated hospital.

4 Tree Acceptability Graphs

Having shown that any acceptability graph that contains a cycle is not guaranteed to have a stable matching, the next logical question is whether or not stable matchings always exist on tree graphs. To explore this question, an algorithm similar to the current NRMP algorithm (Roth 1998) will be constructed to find a stable matching for any set of preference lists. The general method of the algorithm is to add couples one at a time, making the matching stable before a new couple is added. The algorithm changes the matching only when an instability exists. To describe these changes, the following definitions are needed:

In the *couple proposing phase*, a couple c_m begins at the top of their preference list and proposes to hospitals until both hospitals in a preference accept the students' offers. A hospital accepts an offer only if that hospital prefers the proposing student to its current assignment. If no such preference exists, then the couple becomes unmatched. If such a preference (h_a, h_b) does exist, then two possibilities could occur. In most cases, c_m becomes matched to their preference and the hospitals h_a and h_b become matched to the proposing students. In these cases if a student in couple c_h (without loss of generality, assume student s_{2h-1}) was previously matched to either h_a or h_b, then both students in c_h are *bumped* from their assignments, and c_h is added to the queue C_b of bumped couples. The hospital that student s_{2h} was matched to also becomes bumped and is added to the queue H_b of bumped hospitals. It is possible that two couples are bumped from their matches when c_m becomes matched. While in the queues, bumped couples and bumped hospitals are currently unmatched. If a hospital is currently in the bumped queue H_b and accepts an offer from a student, then the hospital is removed from H_b.

The other possibility is that one of h_a or h_b is a bumped hospital and was last matched to a student in c_m. Without loss of generality, assume that h_a is that hospital. Hospital h_a would prefer being matched to being unmatched, but must check to see if a more preferred student than the proposing student s_{2m-1} in c_m is available. This requires h_a to enter hospital proposing phase, with the restriction that h_a cannot propose to s_{2m}, the student in c_m not proposing to h_a.

If h_a proposes to s_{2m-1}, then c_m accepts, becoming matched to the preference (h_a, h_b). If h_a proposes to any other bumped couple, that couple rejects the offer. If h_a is accepted at a more preferred match, then c_m remains in C_b as a bumped couple. A situation in which a hospital enters hospital proposing phase in this way is seen in iteration 5 of Example 2.

In the *hospital proposing phase*, a hospital h_a begins at the top of its preference list and proposes to individual students. A student (without loss of generality, assume s_{2i-1}) in couple c_i will consider the offer if there exists a preference (h_a, h_b) higher on the couple's preference list than the c_i's current assignment. Couple c_i creates a new preference list by purging all preferences from their list except those of the form (h_a, h_b) which are higher than c_i's current assignment. Student s_{2i} proposes down the new preference list until his offer is accepted. If a hospital h_b accepts s_{2i}'s offer, then s_{2i-1} becomes matched to the proposing hospital h_a, and s_{2i} becomes matched to h_b. If h_b had been matched to some student in c_h at a preference (h_b, h_c), then c_h and h_c are bumped and added to their respective queues. If s_{2i} or s_{2i-1} become matched to different hospitals than they were previously matched to when accepting the preference (h_a, h_b), then the hospitals they were previously matched to are bumped and are added to H_b. If no h_b accepts, then c_i remains at their current match, rejecting h_a's offer and h_a continues asking down its list. If no student accepts h_a's offer, then h_a becomes unmatched and is removed from H_b.

Algorithm:

step 1. Create a matching M that contains all of the hospitals and no couples. Set the queues C_b and H_b to empty.

step 2. Add a *new couple* c_n to M. Enter couple proposing phase with c_n to resolve any instabilities that exist with c_n.

step 3. If the queue C_b is not empty, then remove a couple c_i from C_b using last-in-first-out (LIFO) ordering. Enter couple proposing phase with c_i to resolve any instabilities created by the bumping of the couple. Go to step 3.

step 4. If the queue H_b is not empty, then remove a hospital h_j from H_b using LIFO ordering. Enter hospital proposing phase with h_j to resolve any instabilities created by the bumping of the hospital. Go to step 3.

step 5. If not all couples have been added to M, then go to step 2.

An iteration begins at step 2 and ends when step 5 is reached.

Example 2 *Consider the following preference lists and the operation of the algorithm upon the preference lists.*

The hospitals' preference lists are:

h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9
s_2	s_3	s_3	s_5	s_6	s_8	s_7	s_8	s_{10}
	s_1	s_7					s_9	
	s_4	s_4					s_7	
		s_5						

The couples' preference lists are:

$c_1 = (s_1, s_2)$	$c_2 = (s_3, s_4)$	$c_3 = (s_5, s_6)$	$c_4 = (s_7, s_8)$	$c_5 = (s_9, s_{10})$
(h_2, h_1)	(h_2, h_3)	(h_3, h_5)	(h_8, h_6)	(h_8, h_9)
	(h_3, h_2)	(h_4, h_5)	(h_3, h_8)	
	(h_3, u)		(h_7, h_6)	

The operation of the algorithm follows. Insignificant steps have been omitted.

iteration	step	outcome	C_b	H_b
	1	matching has no couples	$\{\}$	$\{\}$
1	2	$c_1 \to (h_2, h_1)$	$\{\}$	$\{\}$
2	2	$c_2 \to (h_2, h_3)$	$\{c_1\}$	$\{h_1\}$
	3	$c_1 \to (u, u)$	$\{\}$	$\{h_1\}$
	4	$h_1 \to u$	$\{\}$	$\{\}$
3	2	$c_3 \to (h_4, h_5)$	$\{\}$	$\{\}$
4	2	$c_4 \to (h_8, h_6)$	$\{\}$	$\{\}$
5	2	$c_5 \to (h_8, h_9)$	$\{c_4\}$	$\{h_6\}$
	3	$c_4 \to (h_3, h_8)$	$\{c_5, c_2\}$	$\{h_6, h_9, h_2\}$
	3	$c_1 \to (h_2, h_1)$ †	$\{c_5, c_2\}$	$\{h_6, h_9\}$
	3	$c_2 \to (h_3, u)$	$\{c_5, c_4\}$	$\{h_6, h_9, h_8\}$
	3	$c_4 \to (h_8, h_6)$	$\{c_5\}$	$\{h_9\}$
	3	$c_5 \to (h_8, h_9)$	$\{c_4\}$	$\{h_6\}$
	3	$c_4 \to (h_7, h_6)$	$\{\}$	$\{\}$
	5	algorithm terminates		

† This occurs since h_2 enters hospital proposing phase when c_2 asks their preference (h_3, h_2). Student s_1 then accepts h_2's offer, and c_2 remains in C_b. The resulting stable matching is:

c_1	c_2	c_3	c_4	c_5
(h_2, h_1)	(h_3, u)	(h_4, h_5)	(h_7, h_6)	(h_8, h_9)

In order to show that this algorithm will always produce a stable matching when operating on tree acceptability graphs, two aspects must be considered: stability and finiteness. To examine these aspects, the following definition and lemma are needed: A *changed item* is a hospital or couple that at some point in the iteration is assigned a different mate than the one they were assigned to at the beginning of the iteration.

Lemma 2 *If the matching is stable at the beginning of the current iteration, then there exists a path of changed items on the graph from any changed couple or hospital back to the new couple* c_n.

Proof. Assume that there is a changed couple or hospital x on the graph that has no path in the induced subgraph of changed items back to the new couple c_n. Determine the first changed item y in the component of the subgraph that contains x. Since y was changed it must have been involved in an instability. When y became changed, it was not adjacent to any changed item. Therefore, the instability must have been present in the original matching. This contradicts the assumption that the matching at the beginning of the iteration was stable. □

Lemma 3 *Upon running the algorithm on a tree acceptability graph, the matching at the end of each iteration of the algorithm is stable.*

Proof. Let the n^{th} iteration be the first iteration whose outcome is unstable. Assume the instability involves couple c_m who can move to a higher preference (h_a, h_b) than their current assignment. Define a state U of the matching where c_m is not in C_b and is matched to a preference lower than (h_a, h_b), and where h_a and h_b are not in H_b, h_a is matched to s_{2m-1} or a lower preference, and h_b is matched to s_{2m} or lower. By assumption, the n^{th} iteration begins out of state U and ends in state U. Consider the last time the matching transitions into state U and the last changed item x (couple c_m, or hospital h_a or h_b) that moves the matching into state U. If $x = c_m$, then c_m must have been bumped from a higher preference, entered couple proposing phase, and became matched to a lower preference. If this was the case, c_m must have proposed to (h_a, h_b), but because h_a and h_b already fulfill the requirements of state U, they would have accepted the offer. State U is not reached; therefore, $x \neq c_m$.

If x is a hospital, assume without loss of generality that $x = h_a$, then h_a must have been bumped from a higher preference involving a student in some couple c_h. If h_a enters hospital proposing phase, then h_a must have proposed to s_{2m-1} and c_m will accept since h_b will accept s_{2m}. State U is not reached since c_m becomes matched to (h_a, h_b). Hospital h_a cannot enter hospital proposing phase. To prevent h_a from proposing, a student s_{2i-1} in c_i, who is less preferred by h_a than s_{2m-1}, must propose to h_a and be accepted. If $c_i = c_h$, then h_a still enters hospital proposing phase and becomes matched to c_m. If $c_i \neq c_h$, then, by Lemma 2, there exists a path P_i of changed items from c_i to c_n, as well as a path of changed items from c_h to c_n. Since $c_i \neq c_h$, P_i does not contain h_a, and the trail of changed items in the acceptability graph from c_h to c_n to c_i to h_a to c_h always contains a cycle. This contradicts the assumption that the acceptability graph is a tree. The transition to state U can never occur; therefore, the outcome of every iteration is stable. □

Lemma 4 *Upon running the algorithm on a tree acceptability graph, the algorithm terminates in a finite number of steps if each hospital is mutually acceptable with at most one student in a couple.*

Proof. Let c_m be a couple who becomes matched to (h_a, h_b). We will show that c_m's assignment will not change in the remainder of the current iteration. Assume without loss of generality that student s_{2h} in a couple c_h is bumped from h_a. If h_a has accepted s_{2m-1}, then s_{2m-1} is more preferred by h_a than s_{2h} and h_a will not accept an offer from s_{2h} while it is matched to s_{2m-1}. From this observation and the hypothesis that s_{2h-1} and h_a are not mutually acceptable, no student in c_h can bump s_{2m-1} from h_a.

It remains to be shown that no student in any other couple can bump s_{2m-1} from being matched with h_a. For c_m to be bumped from h_a by a student in another couple c_i (without loss of generality, assume student s_{2i-1}), s_{2i-1} must propose and be accepted by h_a. Since c_i is proposing, they must have been bumped within this iteration. By Lemma 2, there exists a path P_i of changed items from c_i to c_n. Since h_a has accepted an offer from c_m in this iteration, there exists a path P_a of changed items from h_a through c_m to c_n as well. Because h_a is in c_i's preference list, a trail T of changed items is formed in the acceptability graph from c_i to c_n to h_a to c_i. It contains a cycle except in the case where $P_a \cup \{\text{edge } (h_a, c_i)\} = P_i$. This implies, however, that c_i was bumped from h_a by c_m, which cannot happen since $c_i \neq c_h$. Therefore, trail T always contains a cycle, contradicting the assumption that the acceptability graph is a tree. Hence, c_i can never propose to h_a, and no student can bump s_{2m-1} from being matched with h_a in this iteration.

The same argument can be used to show that s_{2m-1} cannot be bumped from h_b. It follows, then, that c_m can never be bumped from their preference (h_a, h_b) and therefore will never be matched to a lower preference. Since the preference lists are finite, each iteration must end in a finite number of steps. Because there are a finite number of couples, there are a finite number of iterations, and thus the entire algorithm will terminate in a finite number of steps. □

Note that although the problem instance in example 2 does not satisfy the conditions of Lemma 4, the algorithm terminates in a finite number of steps.

Theorem 2 *If the acceptability graph is a tree , then a stable matching exists for any set of preference lists in which each hospital is mutually acceptable with at most one student in a couple.*

Proof. The algorithm is used to generate a matching. Lemma 4 ensures that the algorithm will terminate in a finite number of steps, and Lemma 3 ensures that the matching will be stable. □

The restriction that each hospital be mutually acceptable with at most one student in a couple is automatically satisfied if the students in a couple have different specialties. This algorithm can be modified to accommodate hospitals with q identical positions by allowing the hospital to continue to accept students' proposals until all q positions are filled. In the hospital proposing phase, the hospital only attempts to fill one position for each time it was bumped.

Conjecture 1 *In the couples problem, if the acceptability graph is a tree, then a stable matching exists for any set of preference lists.*

Lemma 3 applies to all tree acceptability graphs, so the only issue is finiteness. There are examples of couples problems in which a single couple c_i is assigned to the same pair of hospitals more than once within an iteration. The authors have not been able to find an upper bound for the number of times c_i can be assigned to the same pair of hospitals, leading to the possibility of an infinite loop in the algorithm presented here. These problems occur when both students in c_i are mutually acceptable to a hospital h_a. If c_i is bumped by another couple c_j that is in turned bumped by a couple different from c_i, then c_i can become matched to a preference they were matched to previously. Couple c_i can revisit the same assignment several times in the same iteration.

5 Conclusion

Acceptability graphs can be used to characterize the existence of stable matchings in the couples problem. If an acceptability graph contains a cycle, then preference lists can be formed such that no stable matching exists. When the acceptability graph is a tree, we conjecture that a stable matching always exists for any set of preference lists. This is the case for problems in which each hospital is mutually acceptable with at most one student in a couple. (See Theorem 2.) These results can be used to develop instances of the couples problem known to have (or not to have) a stable matching. Such instances can be used to test heuristics for solving the couples problem.

6 References

H. Abeledo and G. Isaak A characterization of graphs which assure the existence of stable matchings, Mathematical Social Sciences 22 (1991) 93–96.

B. Aldershof and O. M. Carducci Stable matchings with couples, Discrete Applied Mathematics 68 (1996) 203–207.

D. Gale and L. S. Shapley College admissions and the stability of marriage, American Mathematical Monthly 69 (1962) 9–15.

E. Ronn NP-complete stable matching problems, Journal of Algorithms 11 (1990) 285–304.

A. E. Roth The evolution of the labor market for medical interns and residents: A case study in game theory, Journal of Political Economy 92 (1984) 991–1016.

A. E. Roth New physicians: A natural experiment in market organization. Science 250 (1990) 1524–1528.

A. E. Roth Report on the design and testing of an applicant proposing matching algorithm, and comparison with the existing NRMP algorithm (1996) http://www.economics.harvard.edu/~aroth/phase1.html (Oct. 23, 2001).