The Least-Unpopularity-Factor and Least-Unpopularity-Margin Criteria for Matching Problems with One-Sided Preferences

Richard Matthew McCutchen

Department of Computer Science, University of Maryland, College Park, MD 20742
rmccutch@umd.edu

Abstract. We consider the problem of choosing the best matching of people to positions based on preferences expressed by the people, for which many different optimality criteria have been proposed. A matching is *popular* if no other matching beats it in a majority vote of the people. The popularity criterion has a manipulation-resistance property, but unfortunately, some sets of preferences admit no popular matching. In this paper, we introduce the *least-unpopularity-factor* and *least-unpopularity-margin* criteria, two generalizations of popularity that preserve the manipulation-resistance property but give an optimal matching for every set of preferences. Under each of these generalizations, we show that the "badness" of a given matching can be calculated efficiently but it is NP-hard to find an optimal matching.

Keywords: matching, one-sided preferences, algorithms, NP-hardness, popular matching, voting.

1 Introduction

One of the most common administrative tasks that many organizations perform is assigning people to positions of some kind based on preferences expressed by the people, the positions, or both. For example, the University of Maryland Department of Computer Science assigns teaching assistants to classes according to the teaching assistants' preferences. The National Resident Matching Program (see [10]) assigns residents to hospitals based on the preferences of both residents and hospitals. Gale and Shapley [4] even suggested that students could be assigned to colleges by a central organization that observes the preferences of both sides.

All of these organizations face the problem of choosing a matching of people to positions that gives fair consideration to everyone's preferences. In this paper, we consider the problem in which the people express preferences for the positions, but not vice versa: the preferences are *one-sided*. An *instance* of this problem consists of a set of people, a set of positions, and a *preference list* for each person giving his preference ordering of the subset of the positions that he would be willing to occupy; these orderings may contain ties. The problem is to find the matching that is best overall in light of the preferences. In a valid matching, each

E.S. Laber et al. (Eds.): LATIN 2008, LNCS 4957, pp. 593–604, 2008.

person p is either matched to a position on her preference list or left without a position, which we represent by matching her to a *last resort* position $LR(p)$ available only to her as her last choice.

What makes this problem interesting is that it is not clear what it means for a matching to be "best": since it is almost never possible to give everyone her first choice simultaneously, any matching will inevitably please some people at the expense of others. If we can change a matching to make someone better off without making anyone else worse off (a *Pareto improvement*), of course we should do so, but there are still many *Pareto efficient* matchings that admit no such improvement. To decide among these, we need an *optimality criterion* that, for each instance, designates one or more matchings as optimal. Then, to apply the criterion in practice, we need an efficient algorithm to compute an optimal matching for a given instance. Ideally, our criterion should be "fair" in some sense and should resist attempts by the people to obtain better positions by lying about their preferences.

Many different optimality criteria have been proposed, studied, and used. Some are based on minimizing functions of the rank numbers that the people give to the positions they receive, but such criteria tend to be easy to manipulate. For example, MIT once assigned incoming students to residence halls by minimizing the sum of the cubes of the rank numbers, and a student could improve her chances of being assigned to her true first-choice residence hall by inserting other highly desirable residence halls near the top of her preference list [9].

One criterion that does not use rank numbers and is therefore less susceptible to this kind of manipulation is *popularity*. A matching M is *popular* if no other matching N beats it by majority vote, where each person votes for the matching that gives him the position he likes better or abstains if he likes the two positions equally well. For example, matching M in Figure 1 is not popular because N beats it by majority vote (B and C outvote A in favor of N), while N is popular. (We display an instance as a table that gives the rank number ($-$ if unwilling) for each person (row) and non-last-resort position (column) and a matching by parenthesizing the rank numbers of the matched pairs.)

Abraham et al. [2] gave an efficient algorithm to compute a popular matching for a given instance when one exists. Unfortunately, some instances have no popular matching because of nontransitivity in the voting. For example, no matching for I_2 is popular because we can obtain a matching that beats it by a vote of 2 to 1 by promoting the occupant of y to x and the occupant of x to w and demoting the occupant of w to y.

$$M = \begin{array}{c} \\ A \\ B \\ C \end{array}\!\!\begin{array}{c} \begin{array}{cccc} w & x & y \end{array} \\ \begin{pmatrix} (\mathbf{1}) & - & 2 \\ 1 & (\mathbf{2}) & - \\ - & 1 & (\mathbf{2}) \end{pmatrix} \end{array}, \qquad N = \begin{array}{c} \\ A \\ B \\ C \end{array}\!\!\begin{array}{c} \begin{array}{cccc} w & x & y \end{array} \\ \begin{pmatrix} 1 & - & (\mathbf{2}) \\ (\mathbf{1}) & 2 & - \\ - & (\mathbf{1}) & 2 \end{pmatrix} \end{array}, \qquad I_2 = \begin{array}{c} \\ A \\ B \\ C \end{array}\!\!\begin{array}{c} \begin{array}{cccc} w & x & y \end{array} \\ \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix} \end{array}$$

Fig. 1. Example instances and matchings

In general, when no matching meets the standard of popularity, we would still like to choose a merely "poor" matching over a "terrible" one. To this end, we will propose two numerical measures of a matching's "badness": its *unpopularity factor* and its *unpopularity margin*. The *least-unpopularity-factor* and *least-unpopularity-margin* optimality criteria, respectively, minimize these two measures. We will show that a given matching's unpopularity factor and margin can be calculated efficiently using algorithms based on shortest paths and minimum-cost flow, respectively. However, surprisingly, it is possible to encode the structure of a 3-satisfiability instance in the people's preferences so as to prove that finding an optimal matching under either criterion is NP-hard; we will present this rather unusual reduction in detail.

The inability to find optimal matchings severely limits the immediate practical applicability of these two criteria. Nevertheless, by ruling them out, this paper makes a step toward the goal of finding a criterion that has all the desired properties. Such a criterion would make it possible for organizations to solve their matching problems easily, fairly, and objectively by computer.

2 Related Work

The notion of popularity was first proposed by Gärdenfors [5] in the context of two-sided preferences. More recently, Abraham et al. [2] discussed it for one-sided preferences and gave the first polynomial-time algorithm to find popular matchings. On an instance with n people and a total of m preference-list entries, the algorithm runs in $O(m\sqrt{n})$ if there are ties and $O(n+m)$ if not. Abraham and Kavitha suggested at the end of [3] that, when no popular matching exists, a matching could be chosen based on the graph induced by the "beats by majority vote" relation among matchings; in this paper, we take a different approach by generalizing the "beats" relation itself.

A matching M is *rank-maximal* if it has the lexicographically maximum tuple $(n_1, n_2, \ldots)$, where n_i is the number of people assigned to positions they respectively ranked ith. Irving et al. [7] found an algorithm to compute a rank-maximal matching in $O(\min(n + C, C\sqrt{n})m)$ time, where C is the worst numerical rank to which any person is assigned in the result. Unfortunately, just as MIT's criterion induced people to artificially increase the rank numbers of the positions they desire, the rank-maximality criterion induces people to *decrease* the rank numbers (by omitting positions they are unlikely to receive) because it gives lower rank numbers priority over higher ones. The least-unpopularity criteria, which do not consider rank numbers at all, may be fairer.

If each person specifies utilities for the positions instead of ranking them, the most natural rule, known as *weighted matching* [8], is to maximize the total utility of all pairs. The most commonly used criterion for instances with two-sided preferences (we call the two sides applicants and recruiters) is *stability*. A matching is *stable* unless some currently unpaired applicant and recruiter both prefer each other to their current partners. Gale and Shapley [4] proved that every instance has a stable matching and gave an algorithm to find one.

3 The Unpopularity Factor

Definition 3.1. *If M and N are two matchings for the same instance, N* dominates *M* by a factor of *u/v, where u is the number of people who strictly prefer N to M and v is the number of people who strictly prefer M to N. The* unpopularity factor *of a matching M is the maximum factor by which it is dominated by any other matching (ignoring matchings that give $u = v = 0$).*

Note that a matching N dominates a matching M by a factor of ∞ if and only if N is a Pareto improvement over M; thus M has a finite unpopularity factor if and only if it is Pareto efficient. Furthermore, a matching is popular as defined by Abraham et al. [2] if and only if its unpopularity factor is at most 1.

The *least unpopularity factor* of an instance is the minimum unpopularity factor of all of its matchings, and the matching(s) that achieve this minimum are considered optimal. Thus, different instances have different numbers of optimal matchings, but every instance has at least one.

Matching M from the Introduction has unpopularity factor 2 because N dominates it by a factor of 2. However, N has unpopularity factor 1: the only person who might favor a different matching is A, and to promote A we must demote B, achieving a dominance factor of 1. In I_2, the six matchings that fill all three positions all have unpopularity factor 2 because the people in x and y can improve at the expense of the person in w; these six matchings are optimal.

To determine the unpopularity factor of a matching directly from the definition, we would have to consider all possible alternative matchings and calculate the factor by which each dominates M, which would take exponential time. Fortunately, there is an efficient way to calculate the unpopularity factor using the concept of *pressures*, which we will develop next.

3.1 Differences Between Matchings: Reassignments, Paths and Cycles

We can think of an instance as a bipartite graph whose vertex sets are the people and the positions and whose edges are the preference-list entries. A matching of the instance is then just a matching of the graph that uses all the people (because every person has a position, though it might just be her last resort).

In what ways can two matchings M and N differ? Their symmetric difference, which we will denote $M \oplus N$, consists of vertex-disjoint paths and/or cycles that are alternating in the sense that their edges come alternately from M and N. Furthermore, the alternating paths stop at positions, because if a path stopped at a person, she would lack a position in one matching, which is not allowed. We can think of an alternating path as a sequence of reassignments: one person moves to a different position, ejecting its original occupant; the occupant takes another position, ejecting *its* occupant; and so forth until someone takes a previously unoccupied position. Each reassignment may constitute a *promotion* or *demotion* of the reassigned person according to his preferences. An alternating cycle is similar.

Conversely, we say that a path or cycle X is *applicable* to a matching M if $M \oplus X$ is a valid matching. If so, *applying* X to M gives $M \oplus X$; X represents the change from M to $M \oplus X$.

With this background, we can define pressures.

3.2 Pressures

Definition 3.2. *Let M be a Pareto efficient matching. The* pressure *of a filled position p in M is the largest k for which there exists an alternating path applicable to M that promotes k people without demoting anyone and then ends with the demotion of the occupant of p to her last resort. Note that the demotion by itself constitutes such a path for $k = 0$. (The term "pressure" comes from the idea of k people stacked up behind p, wishing that its occupant will leave so they can all become better off.)*

Theorem 3.3. *The unpopularity factor of a Pareto efficient matching M is the greatest pressure of any of its filled positions.*

Proof. Recall that the unpopularity factor of M is the greatest pressure by which any other matching dominates M. To establish that the two maxima are equal, we'll show that each is at least as great as the other, i.e., (a) if M has a position p of pressure k, then there exists a matching N that dominates M by a factor of at least k and (b) if a matching N dominates M by a factor of f, then M has a position of pressure at least $\lceil f \rceil$.

(a): Simply let N be the result of applying to M the path that determines the pressure of p. The path promotes k people and demotes one person, so N dominates M by a factor of k.

(b): Let u and v be the total number of people better and worse off in N than in M, so that $f = u/v$. First modify N so that everyone who is worse off in N than in M is demoted all the way to his last resort in N; this does not change u or v. Decompose $M \oplus N$ into a collection of paths and cycles $X_1, \ldots, X_c$, discarding those that neither promote nor demote anyone. Each X_i must demote at least one person so that it is not a Pareto improvement over M. That person is demoted to his last resort, and since there is no one else to leave the last resort, the demotion must be the last reassignment in X_i; thus X_i is a path (not a cycle). A path has only one last reassignment, so each X_i demotes *exactly* one person to his last resort, and $c = v$.

For each i, let u_i be the number of people promoted by X_i; of course, $\sum_i u_i = u$. Choose an i such that $u_i \geq \lceil u/v \rceil = \lceil f \rceil$; by the Pigeonhole Principle, one must exist. Let p be the position whose occupant is demoted by X_i. Observe that X_i has exactly the form considered in Definition 3.2 for the pressure of p, and it makes at least $\lceil f \rceil$ people better off; thus the pressure of p is at least $\lceil f \rceil$.

Corollary 3.4. *The unpopularity factor of a matching, if finite, is an integer.*

3.3 Computing the Unpopularity Factor

We can reduce computation of the pressures of a matching to a shortest path problem. Let n' and n be the numbers of people and positions, and let m be the total number of entries in the preference lists.

Algorithm 3.5. *In $O(m\sqrt{n})$ time, determines whether a matching M is Pareto efficient and, if so, finds the pressure of each filled position in M.*

Method. Construct a graph G whose vertices are the positions of M. A pair of vertices (p_1, p_2) is connected in G by an edge of length -1 if p_1 is filled by a person who strictly prefers p_2, an edge of length 0 if p_1 is filled by a person indifferent to p_2, or no edge otherwise. Run Goldberg's shortest-path algorithm [6] on G, using all positions as sources so that the algorithm finds the shortest path ending at each position. If Goldberg's algorithm finds a negative cycle in G or a negative-length path ending at an unfilled position, conclude that M is Pareto inefficient. Otherwise, the pressure of each position is the negative of the length of the shortest path ending at it.

A path or cycle X in G represents a sequence of reassignments that demotes no one and is applicable to M after the ending position (in the case of a path) is vacated; furthermore, the length of X in G is the negative of the number of people it promotes. In light of this, Pareto-improving cycles and paths applicable to M correspond to negative cycles and negative-length paths ending at unfilled positions, respectively, in G. Thus, the algorithm correctly determines whether M is Pareto efficient. If it is, then paths in G ending at a position p represent paths applicable to M of the form considered in Definition 3.2, with the negative of a path's length corresponding to k in that definition. Thus, the negative of the shortest length of a path ending at p gives the pressure of p, as desired.

Each preference-list entry of the instance accounts for at most one edge of G, so G has at most m edges. The running time is dominated by that of Goldberg's algorithm, which is $O(m\sqrt{n})$ since edge lengths are at least -1.

To find the unpopularity factor of a matching, we use this algorithm to find the pressures and then simply take the highest pressure. This algorithm can be seen as a generalization of the algorithm given by Abraham et al. [1] to determine in $O(m)$ time only whether M is Pareto efficient; both algorithms are based on the same graph G.

4 The Unpopularity Margin

The unpopularity margin of a matching is defined the same way as the unpopularity factor, except we subtract the numbers of votes instead of dividing them:

Definition 4.1. *If M and N are two matchings for the same instance, N* dominates *M* by a margin of *$u - v$, where u is the number of people who strictly prefer N to M and v is the number of people who strictly prefer M to N. The* unpopularity margin *of a matching M is the maximum margin by which it is dominated by any other matching.*

The unpopularity margin of a matching M is an integer; it is 0 if M is popular (because M dominates itself by a margin of 0) or otherwise positive. We can reduce calculating the unpopularity margin to a min-cost max-flow problem. Since we use integer edge capacities, we assume that edge flows are also integers.

Algorithm 4.2. *Finds the unpopularity margin u of a matching M in $O((u+1)m\sqrt{n'+n})$ time.*

Method. Construct a flow graph G having as vertices a source, a sink, and the people and positions of M. Add an edge of unit capacity and zero cost from the source to each person and from each position to the sink. For each preference-list entry submitted by a person A for a position p, add a unit-capacity edge from A to p whose cost is -1, 0 or 1 as A likes p better than, the same as, or worse than her position in M.

A max-flow of G must put one unit of flow through each person, and those units must reach the sink via different positions, so the max-flows of G correspond exactly to the possible matchings of the instance. Furthermore, the cost of a max-flow is the negative of the margin by which the corresponding matching dominates M. Thus, we find the min-cost max-flow by starting from the max-flow representing M itself and augmenting negative cycles found using Goldberg's shortest-path algorithm [6]. The negative of the cost of this flow gives the unpopularity margin of M.

To find each negative cycle, we run Goldberg's algorithm on a graph with $n'+n+2$ vertices and m edges, taking $O(m\sqrt{n'+n})$ time since edge lengths are at least -1. Each cycle decreases the cost by at least 1 until we reach cost $-u$, so we find at most u cycles and then perform one more failed search for a cycle. The running time bound follows.

5 NP-Hardness of Finding Least-Unpopularity Matchings

We now use a reduction from 3-satisfiability (3SAT) to prove that it is NP-hard to find the least unpopularity factor or margin of a given set of preferences; it happens that the same reduction works for both problems. Abraham et al. [2] analyze preference sets with no ties separately from the general case of ties. We have had no reason to make this distinction so far, but the reduction will always generate preference lists with no ties in order to prove that even the no-ties versions of the problems are NP-hard.

The reduction converts an instance S of 3SAT to a polynomial-size preference set P and an *ideal* unpopularity factor. We will show that any tuple of truth values that satisfies S can be converted to a matching of P whose unpopularity factor does not exceed the ideal value, and vice versa; thus, the least unpopularity factor of P is at most the ideal value if and only if S is satisfiable. Therefore, computing the least unpopularity factor of P is NP-hard because an algorithm to do that could be used to determine whether S is satisfiable. This paragraph applies equally to unpopularity margins, and henceforth "unpopularity" will refer to either the factor or the margin.

5.1 Overview of the Reduction Design

The reduction, like most, builds P from *gadgets* that represent pieces of S. Each gadget will contain some *internal* people and positions and some *linking people*; there will also be *linking positions* that do not belong to any gadget. An internal person is willing to occupy only internal positions of her own gadget, but a linking person is also open to exactly one linking position, which is always her first choice. Any reassignment of the occupant of a linking position is a demotion, which (from the perspective of voting) could just as well be to his last resort as into a gadget; thus, the dominance that can be achieved by replacing him depends only on the identity and state of the gadget providing the replacement, not on the states of any other gadgets. In other words, gadgets are isolated from one another unpopularity-wise; their only interactions are in *which gadget gets to fill each linking position.* Thus, we can analyze each gadget's contribution to the unpopularity of the matching separately as a function of which linking positions the gadget gets.

Motivated by this idea, we introduce three types of gadget, each of which is designed to enforce a certain constraint on which linking positions it must get by producing a low unpopularity if the constraint is satisfied or a higher one if it is not. To represent S, we start with a set of key linking positions representing its variables; the choice of which gadget gets each of these positions represents a tuple of truth values for the variables. We then add gadgets so that satisfaction of all of the gadget constraints is equivalent to satisfaction of S, and we let the ideal unpopularity of P be the low unpopularity that would result if every gadget's constraint were satisfied. Note that the unpopularity factor of the matching is the highest pressure produced by any gadget, while the unpopularity margins of separate gadgets roughly add.

5.2 The Gadgets

A *box* consists of four internal positions, three internal people (i_1, i_2, and i_3), and three linking people (w, n_1, and n_2). Figure 2(a) shows its structure, including the linking positions. w is known as the *wide* person, and n_1 and n_2 are the *narrow* people. A box is satisfied, and produces a pressure of 2 and a margin of 1, if either the wide person or both narrow people get their linking positions; however, if both the wide person and at least one narrow person are denied their linking positions, a pressure of 3 and a margin of 2 result.

A *peg* (Figure 2(b)) consists of one internal position available to three linking people, all of whom prefer the same linking position. Its purpose is very simple: to always produce a pressure of 2 on l and provide a way to replace its occupant at margin 1.

A *pool* (Figure 2(c)) consists of two internal positions and three linking people. If k of the people are denied their linking positions, the pool has one linking position with a pressure of k and can replace its occupant at a margin of $\max(k-1, 0)$. We want to use the pool to distinguish between two and three people being denied linking positions. To this end, we attach a peg to each linking position;

then, all positions have pressure 2 and replacement margin 1, except when all three positions are taken by people from other gadgets, one of them develops a pressure of 3 and a replacement margin of 2.

$$\begin{array}{c} \\ w \\ i_1 \\ i_2 \\ i_3 \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{ccccccc} x & y & z & u & l_w & l_{n1} & l_{n2} \end{array} \\ \begin{pmatrix} 2 & 3 & 5 & 4 & 1 & - & - \\ 1 & 2 & 3 & 4 & - & - & - \\ 1 & 2 & 3 & 4 & - & - & - \\ 1 & 2 & 3 & 4 & - & - & - \\ - & - & - & 2 & - & 1 & - \\ - & - & - & 2 & - & - & 1 \end{pmatrix} \end{array}$$

(a)

$$\begin{array}{c} \\ f_1 \\ f_2 \\ f_3 \end{array} \begin{array}{c} \begin{array}{cc} x & l \end{array} \\ \begin{pmatrix} 2 & 1 \\ 2 & 1 \\ 2 & 1 \end{pmatrix} \end{array}$$

(b)

$$\begin{array}{c} \\ f_1 \\ f_2 \\ f_3 \end{array} \begin{array}{c} \begin{array}{ccccc} x & y & l_{f1} & l_{f2} & l_{f3} \end{array} \\ \begin{pmatrix} 2 & 3 & 1 & - & - \\ 2 & 3 & - & 1 & - \\ 2 & 3 & - & - & 1 \end{pmatrix} \end{array}$$

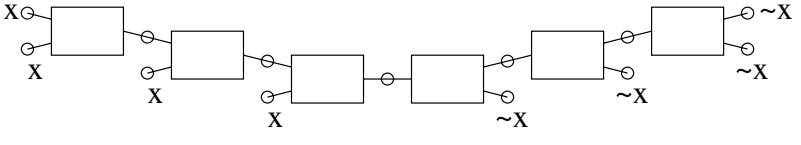
(c)

Fig. 2. The three gadgets: (a) box, (b) peg, and (c) pool

5.3 Constructing the Preference Set

A box is a "two-for-one" device: if another gadget takes its wide linking position, it demands both narrow linking positions. For any k, we can construct a k-for-one device from $k-1$ boxes by identifying the wide position of each box after the first with a narrow position of the previous box. If we identify the ultimate wide positions of two such devices, we can get a u-for-v device for any desired u and v.

For each variable x_i of the 3SAT instance S, we generate a many-for-many device whose two sets of narrow positions represent the references to x_i and the references to $\neg x_i$ in the clauses of S, respectively. The device for a variable x with four ordinary references and four negated ones could be drawn like this:



(Boxes represent boxes, circles represent linking positions, and lines represent linking people. Internal people and positions are not shown.) In a matching that obeys all the gadget constraints, we may assign all the linking people either to the right, filling the $\neg x$ positions and leaving the x positions open, or to the left, filling the x positions and leaving the $\neg x$ positions open. These two possibilities correspond to making x true or false, respectively. Either way, a linking position is left open if and only if the reference it represents evaluates to true.

Now, we add a pool for each clause of S and identify its three linking positions with linking positions of the variable devices according to the clause's three references. The pool demands that at least one of its linking people receive a linking position. This is possible if and only if the clause is satisfied, so we can see that a matching that obeys all the constraints represents a solution to S.

5.4 Correctness of the Reduction

We will now give the details of the argument that S is satisfiable if and only if P has a matching of the ideal unpopularity factor (margin) and, in doing so, specify the ideal unpopularities.

Suppose the tuple of truth values $(t_1, \ldots, t_v)$ satisfies S; we construct a matching M as follows. We match the device for each variable x_i in a manner that depends on t_i. If t_i is true, we match each box on the x_i side according to the first table below, filling its wide linking position, and each box on the $\neg x_i$ side according to the second, filling its two narrow linking positions. In the first table, we let n_1 be a/the person whose linking position is shared with a pool (rather than a box) so that the pressure of 2 on l_{n1} from the box is subsumed by that from the attached peg; we then assign n_2 to his last resort. Each table's superscripts give the pressures generated by the box shown; other gadgets may account for higher pressures on some linking positions.

	x^2	y^1	z^0	u^1	l_w^0	l_{n1}^2	l_{n2}^1
w	2	3	5	4	**(1)**	–	–
i_1	**(1)**	2	3	4	–	–	–
i_2	1	**(2)**	3	4	–	–	–
i_3	1	2	**(3)**	4	–	–	–
n_1	–	–	–	**(2)**	–	1	–
n_2	–	–	–	2	–	–	1

	x^2	y^1	z^0	u^0	l_w^1	l_{n1}^0	l_{n2}^0
w	2	3	5	**(4)**	1	–	–
i_1	**(1)**	2	3	4	–	–	–
i_2	1	**(2)**	3	4	–	–	–
i_3	1	2	**(3)**	4	–	–	–
n_1	–	–	–	2	–	**(1)**	–
n_2	–	–	–	2	–	–	**(1)**

Each of the device's linking positions is filled exactly once except for those representing references to x_i. If t_i is instead false, we use the same construction but with the two sides of the device switched. Either way, exactly those linking positions that represent references evaluating to true are left open, and no pressure exceeds 2.

Now we assign each pool linking person to her linking position if it is available or otherwise to the best available position in her pool. Since the t_i satisfy S, at least one linking person from each pool will get a linking position, so each pool only needs to accommodate at most two people. A pool can hold two people as in the table below (the pressure superscripts consider the attached pegs as well as the pool itself), and additional people can be moved to linking positions without increasing the pressures.

	x^1	y^0	l_{f1}^2	l_{f2}^2	l_{f3}^2
f_1	**(2)**	3	1	–	–
f_2	2	**(3)**	–	1	–
f_3	2	3	–	–	**(1)**

Nowhere did M incur a pressure exceeding 2, so it has unpopularity factor 2, which we designate as ideal. To bound its unpopularity margin, we use the following lemma, whose proof we omit due to space constraints:

Lemma 5.1. *The unpopularity margin of a Pareto efficient matching M does not exceed the number of filled positions in M that have pressure 2 or greater.*

M has exactly $6c - 2v$ positions of pressure 2, namely the $3c$ positions bearing pegs and the internal position x of each of the $3c - 2v$ boxes (one per variable reference minus $2v$ at the ends of the variable devices), so M has unpopularity margin at most $6c - 2v$, which we designate as ideal.

For the other direction of the proof, suppose that S is unsatisfiable and let M be an arbitrary Pareto efficient matching of P; we will show that M achieves neither the ideal unpopularity factor nor the ideal unpopularity margin. It should be clear that M cannot satisfy all the gadget constraints, but we must show that non-ideal unpopularities result.

A peg may or may not get its linking position in M, but either way, x must be filled for Pareto efficiency, and at least one person is left at her last resort. Starting from M, we "cycle" each peg by promoting a last-resort person to x, promoting the occupant of x to the linking position, and demoting the occupant of the linking position to his last resort. In each box, the three people i_1, i_2, and i_3 are all eager to fill the three positions x, y, z, so M must fill those positions for Pareto efficiency. We cycle the box by promoting z's occupant (who could be w rather than an i_j) to y and y's occupant to x, demoting the occupant of x. Let N be the resulting matching. We have performed two demotions and one promotion for each of the $3c$ pegs and $3c - 2v$ boxes, so N dominates M by the ideal margin of $6c - 2v$.

Now we modify N to exploit the gadget dissatisfaction in M. Suppose M dissatisfies a box, i.e., both the wide person w and at least one narrow person (say n_1) are denied linking positions. The four people w, i_1, i_2, and i_3 are eager to fill the four positions x, y, z, and u, so M must fill those positions. Instead of cycling this box, we do the following. If either w or an i_j is at her last resort, we promote her to z and z's occupant to y. Otherwise, n_1 must be at his last resort; we promote him to u and u's occupant (who must be w or an i_j) to y. Either way, we then promote y's occupant to x and demote the occupant of x. We now have 3 promotions in the box instead of 2, so N dominates M by a margin of $6c - 2v + 1$. Furthermore, the chain of promotions exerts a pressure of 3 on x.

On the other hand, if M dissatisfies a pool (by denying all three of its people their linking positions), then one of the people must be in x, one must be in y, and the third (call him p) must be at his last resort. Let l_{fi} be the linking position of the occupant of x. Instead of cycling the peg attached to l_{fi}, we promote p to y, y's occupant to x, and x's occupant to l_{fi}, demoting the occupant of l_{fi}. This strategy similarly raises the dominance margin to $6c - 2v + 1$ and reveals a pressure of 3 on l_{fi}.

In either case, the pressure of 3 shows that M fails to achieve the ideal unpopularity factor of 2 and N's dominance margin of $6c - 2v + 1$ shows that M fails to achieve the ideal unpopularity margin of $6c - 2v$, so the proof is complete.

By means of the reduction, we have proved the following result:

Theorem 5.2. *It is NP-hard to calculate the least unpopularity factor or margin of a given preference set. Thus, it is also NP-hard to compute an optimal matching.*

How well the least unpopularity factor and margin of an instance can be approximated is an open problem. The above proof shows that it is NP-hard to approximate the least unpopularity factor within a factor better than 3/2. The author examined three heuristic algorithms (see the supplementary material) that often find good matchings but could not prove an approximation bound for any of them; a search for a simple construction to increase the additive gap in the least unpopularity factor was also unsuccessful. Of course, a more important goal for future work is to find a good manipulation-resistant criterion under which *optimal* matchings always exist and can be found efficiently.

Acknowledgments. I would like to thank Samir Khuller, my advisor, for introducing me to previous work on matching with one-sided preferences; Brian Dean for suggesting the use of Goldberg's algorithm in Algorithm 3.5; and Dr. Khuller, Bobby Bhattacharjee, Glenda Torrence, Nancy Zheng, my parents, and others for suggesting improvements to this paper and its precursors.

Supplementary materials for this work, including Java implementations of some of the algorithms, are available at `http://mattmccutchen.net/lumc/`.

References

1. Abraham, D., Cechlárová, K., Manlove, D., Mehlhorn, K.: Pareto Optimality in House Allocation Problems. In: Fleischer, R., Trippen, G. (eds.) ISAAC 2004. LNCS, vol. 3341, pp. 3–15. Springer, Heidelberg (2004)
2. Abraham, D., Irving, R., Kavitha, T., Mehlhorn, K.: Popular matchings. In: Proceedings of SODA 2005: The 16th ACM-SIAM Symposium on Discrete Algorithms, pp. 424–432 (2005)
3. Abraham, D., Kavitha, T.: Dynamic matching markets and voting paths. In: Arge, L., Freivalds, R. (eds.) SWAT 2006. LNCS, vol. 4059, pp. 65–76. Springer, Heidelberg (2006)
4. Gale, D., Shapley, L.: College admissions and the stability of marriage. American Mathematical Monthly 16, 9–15 (1962)
5. Gärdenfors, P.: Match Making: assignments based on bilateral preferences. Behavioural Sciences 20, 166–173 (1975)
6. Goldberg, A.: Scaling algorithms for the shortest paths problem. In: Proceedings of SODA 1993: The 4th ACM-SIAM Symposium on Discrete Algorithms, pp. 222–231 (1993)
7. Irving, R., Kavitha, T., Mehlhorn, K., Michail, D., Paluch, K.: Rank-maximal matchings. In: Proceedings of SODA 2004: the 15th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 68–75 (2004)
8. Papadimitriou, C., Steiglitz, K.: Combinatorial Optimization: Algorithms and Complexity, ch. 11. In: Weighted Matching. Prentice-Hall, Englewood Cliffs (1982)
9. Price, E.: Personal communication (August 2005)
10. Roth, A.: The evolution of the labor market for medical interns and residents: A case study in game theory. Journal of Political Economy 92, 991–1016 (1984)