Submitted to *Algorithms*. Pages 1 - 25.

OPEN ACCESS

Article

Local Search Approaches in Stable Matching Problems

Mirco Gelain [1], Maria Silvia Pini [1*], Francesca Rossi [1], K. Brent Venable [2], and Toby Walsh [3]

[1] University of Padova, Italy

[2] Tulane University and IHMC, USA

[3] NICTA and UNSW, Australia

* Author to whom correspondence should be addressed; pini@dei.unipd.it

Version September 4, 2013 submitted to Algorithms. *Typeset by LaTeX using class file mdpi.cls*

Abstract: The stable marriage (SM) problem has a wide variety of practical applications, ranging from matching resident doctors to hospitals, to matching students to schools, or more generally to any two-sided market. In the classical formulation, n men and n women express their preferences (via a strict total order) over the members of the other sex. Solving a SM problem means finding a stable marriage where stability is an envy-free notion: no man and woman who are not married to each other would both prefer each other to their partners or to being single. We consider both the classical stable marriage problem and one of its useful variations (denoted SMTI) where the men and women express their preferences in the form of an incomplete preference list with ties over a subset of the members of the other sex. Matchings are permitted only with people who appear in these preference lists, an we try to find a stable matching that marries as many people as possible. Whilst the SM problem is polynomial to solve, the SMTI problem is NP-hard. We propose to tackle both problems via a local search approach, which exploits properties of the problems to reduce the size of the neighborhood and to make local moves efficiently. We evaluate empirically our algorithm for SM problems by measuring its runtime behaviour and its ability to sample the lattice of all possible stable marriages. We evaluate our algorithm for SMTI problems in terms of both its runtime behaviour and its ability to find a maximum cardinality stable marriage. Experimental results suggest that for SM problems, the number of steps of our algorithm grows only as $O(n\log(n))$, and that it samples very well the set of all stable marriages. It is thus a fair and efficient approach to generate stable marriages. Furthermore, our approach for SMTI problems is able to solve large problems, quickly returning stable matchings of large and often optimal size despite the NP-hardness of this problem.

Keywords: local search, stable matching, sampling, ties and incomplete preference lists

1. Introduction

The stable marriage problem (SM) [16] is a well-known problem of matching men to women to achieve a certain type of "stability". Each person expresses a strict preference ordering over the members of the opposite sex. The goal is to match men to women so that there are no two people of opposite sex who would both rather be matched with each other than with their current partners. The stable marriage problem has a wide variety of practical applications, ranging from matching resident doctors to hospitals, sailors to ships, primary school students to secondary schools, as well as in market trading. Surprisingly, such a stable marriage always exists and one can be found in polynomial time. Gale and Shapley give an algorithm, which is linear in the size of the input, to solve this problem based on a series of proposals of the men to the women (or vice versa) [6].

There are many variants of the traditional formulation of the stable marriage problem. Some of the most useful in practice include incomplete preference lists (SMI), that allow one to model unacceptability for certain members of the other sex, and preference lists with ties (SMT), that model indifference in the preference ordering. With a SMI problem, the goal is to find a stable marriage in which the married people accept each other. It is known that all solutions of a SMI problem have the same size (that is, number of married people) [43]. In SMT problems, instead, solutions are stable marriages where everybody is married. Both of these variants are polynomial to solve. In real world situations, both ties and incomplete preference lists may be needed. Unfortunately, when we allow both, the problem becomes NP-hard [31]. In a SMTI (Stable Marriage with Ties and Incomplete lists) problem, there may be several stable marriages of different sizes, and solving the problem means finding a stable marriage of maximum size.

In this paper we investigate the use of a local search approach to tackle both the classical and the NP-hard variants of the problem. In particular, when we consider the classical problem, we investigate the fairness of stable marriage procedures based on local search, i.e., we investigate how well these procedures sample the lattice of stable marriages. On the other hand, for SMTI problems, we focus on efficiency in terms of time and effectiveness at finding large stable marriages. Our algorithms are based on the same schema: they start from a randomly chosen marriage and, at each step, we move to a neighbor marriage by minimizing the distance to stability, which is measured by the number of unstable pairs. To avoid redundant computation due to the possibly large number of unstable pairs, we consider only those that are undominated, since their elimination minimize the distance to stability. Random moves are also used, to avoid stagnation in local minima. The algorithms stop when they find a solution or when a given limit on the number of steps is reached. A solution for an SMTI instance is a perfect stable matching (that is, a stable marriage with no singles), whereas, for an SM instance, a solution is just a stable marriage.

For the SM problem, we performed experiments on randomly generated problems with up to 500 men and women. It is interesting to notice that our algorithm always finds a stable marriage. Also, its runtime behaviour shows that the number of steps grows as little as $O(n \log(n))$ [28]. We also tested the fairness of our algorithm at generating stable marriages, measuring how well the algorithm samples the set of all stable marriages. As it is non-deterministic, it should ideally return any of the possible stable marriages with equal probability. We measure this capability in the form of an entropy that should be as close to that of an uniform sample as possible. The computed entropy is about 70% of that of an uniform sample, and even higher on problems with small size.

For the SMTI problem, we performed experiments on randomly generated problem instances of size 90 and in some cases also of size 100. We observe that our algorithm is able to find stable marriages with at most two singles on average in tens of seconds at worst. The SMTI problem has been tackled also in [12], where the problem is modeled in terms of a constraint optimization problem and solved employing a constraint solver. This systematic approach is guaranteed to find always an optimal solution. However, our experimental results show that our local search algorithm in practice always appears to find optimal solutions. Moreover, it scales well to sizes much larger than those considered in [12]. An alternative approach to local search is to use approximation methods.

The paper is an extended and revised version of [7,8,11].

2. Related work

In this paper we consider the fairness of the methodology to generate stable marriages. Other works have considered the fairness with the meaning of finding a stable marriages where the overall happiness of the persons is maximized. One kind of fairer stable marriage that has been considered, is the *minimum regret* stable marriage [15,28]. The regret for each person is the position in his/her preference list of the persons to whom he/she is married. The regret of a marriage M is the maximum regret of any person. Another way characterize the overall happiness of a marriage is to consider sum the regret of every person. The *egalitarian* stable marriage [21] minimize the total sum of the regrets. Both minimum regret and egalitarian stable marriage can be found in polynomial time [15,21]. In [44] Roth and Vande Vate show that, beginning from an arbitrary marriage, and satisfying a blocking pair at random, we will eventually reach a stable marriage with probability one. Our local search approaches exploit this result by building sequences of blocking pairs removal that rapidly lead to stability thanks to the use of undominated blocking pairs.

In this paper we consider also the solution of stable marriage problems with ties and incomplete lists. It is known that weakly stable matchings may have different cardinality. Furthermore, finding the maximum (or minimum) cardinality weakly stable matching for a given instance of SMTI is NP-hard. This holds even if the ties are at the tails of lists and on one side only, and each tie has length 2 [31], though the largest matching is at most twice the size of the smallest [31]. It has also been established that these problems are not approximable within δ, unless P=NP, for some $\delta > 1$, even if the preference lists are of constant length, there is at most one tie per list, and the ties occur on one side only [18]. Above we noted that a maximum cardinality weakly stable matching is at most twice the size of a minimum cardinality weakly stable matching. Therefore, if we break all ties in an arbitrary way and apply the GS algorithm to the resulting instance of SMI we get what is simultaneously an approximation algorithm for the problem of finding a maximum (resp., minimum) stable matching with a performance ratio of 2. In [18,45] an improved performance bound is shown for instances of SMTI with sparse ties. Three other pieces of work relating to approximating maximum cardinality weakly stable matchings have appeared in the literature. In [19], Halldorsson et al. present a randomised approximation algorithm with expected performance guarantee $\frac{10}{7}$ for instances of SMTI in which ties occur on one side only, there is at most one tie per list, and each tie has length 2. In [17], the same authors present an approximation algorithm with performance guarantee $\frac{2}{(1+\frac{1}{L^2})}$ for instances of SMTI in which ties occur on one side only, and each tie has length at most L. Additionally, they show a ratio of $\frac{13}{7}$ where ties are allowed on both sides, and are of length 2. In [24] Iwama et al. present an approximation algorithm for a general instance of SMTI with guarantee $2 - c\frac{log(n)}{n}$, for an instance of size n, where c is an arbitrary positive constant. Recently, in [26] Iwama et. al improve the approximation ratio to $\frac{25}{17}$ for instances with one-sided ties. This approximation ratio also holds for the hospitals/residents problem (i.e., many-one variant) with one-sided ties (see [42] for the relationship between approximability of the stable marriage problem and the hospitals/residents problem). Other approximation results with a higher ratio have been shown in [25,27,35]. A detailed overview of approximation algorithms is presented on pages 136-137 of [30].

In our paper we consider a local search approach to solve SMTI instances. Other local search methods have been presented for SMTI instances but in terms of parameterized complexity in the framework introduced by [5]. In SMTI instances the parameter can be the number of ties, the maximum or the overall length of ties [33]. In [33] the authors investigate the applicability of a local search algorithm for the problem and they examine the possibilities for giving an FPT algorithm or an FPT approximation algorithm for finding an egalitarian or a minimum regret stable matching. In general, few papers have investigated the connection of parameterized complexity and local search, although attention to this topic has been increasing recently [32]. In [34] the framework of parameterized complexity is used to deal with the Hospitals/Residents with Couples problem, a variant of the classical Stable Marriage problem. This is the extension of the Hospitals/Residents problem where residents are allowed to form pairs and submit joint rankings over hospitals. In this problem the authors consider the number of couples as a parameter, they apply a local search approach, and examine the possibilities for giving FPT algorithms applicable in this context.

In [12], Gent and Prosser give an exhaustive empirical study of the stable marriage problem with ties and incomplete lists, using a constraint programming encoding of the problem. Then, the encoded problem can be solved using off the shelf CP technology. They present results for the decision problem "Is there a stable matching of size n?" and for the optimization problem of finding a maximum or minimum cardinality stable matching. In particular, regarding the optimization problem of finding the largest stable marriage, their complete method (based on the solution of the CP encoding of the problem using the Choco constraint programming toolkit [29]) finds stable marriages of size 9.3 (in average) considering problems of size 10 with no ties. When the amount of ties increases the size increases as well. Our local search approach obtains very similar results using a test set generated in the same way.

Gent and Prosser in [13] give a SAT encoding of the stable marriage problem with ties and incomplete lists. Using such an encoding they obtain very good results in the decision problem of whether there is a perfect matching. Even though in our experiments we often find a perfect matching we consider a different problem from the one solved in [13].

In [3] Brito and Meseguer, propose a distributed approach to the stable marriage problem with ties and incomplete lists with the aim of keeping preference lists private for privacy reasons. They extend some specialized centralized algorithms (such as the Extended Gale Shapley algorithm) to the distributed case. Moreover, they provide a generic distributed constraint programming model. In their experimental evaluation, they consider the communication effort and the computational cost (in terms of constraint checks) which are not applicable to our centralized approach. However, they show also the maximum cardinality of the marriages found by their algorithms considering SMTI instances. Considering problems of the same size, probability of ties and, incompleteness they used, we obtain marriages of very similar cardinality.

In [22] Irving and Manlove present two heuristic approaches to find the largest stable matching in the context of the hospital resident-oriented (HR) problems with incomplete lists and ties only in the hospitals' preference lists. One of the algorithms is based on the hospital-oriented version of Gale-Shapley algorithm and the other one is based on the resident version. Heuristics are used to decide how breaking ties in order to maximize the size of the returned marriage. In fact, the ways in which ties are broken can significantly affect the size of the stable matching found and, in the extreme case, there may be two matchings differing in size by a factor of 2 [31]. When hospitals have capacity equal to 1, the problem becomes an SMTI instance with ties on one side only, thus the algorithms proposed in [22] can also be used to solve such restricted SMTIs.

In [2] the authors give complexity and approximation results regarding the problem of finding a maximum cardinality matching that admits the smallest number of blocking pairs in an SMI instance. They show that such a problem is NP-hard. Our experimental results show that our local search approach is able to find marriages of large size and with a very small number of blocking pairs within a small number of steps.

In our local search approach we exploit the Gale-Shapely stable matching procedure. The GS algorithm is computationally easy to manipulate and favors one gender over the other. In [36,37] it is shown that there exist stable marriage procedures which are NP-hard to manipulate and that voting rules which are NP-hard to manipulate can be used to define stable marriage procedures which are themselves NP-hard to manipulate. Moreover, it is shown how to use voting rules to make any stable marriage procedure gender neutral. Manipulation issues have been also considered in the context of stable matching procedures with weighted preferences where new notions of stability and optimality have been provided [38–40]. Besided manipulation, stability, and optimality, also uniqueness of weakly stable matchings has been studied in the context of stable matching procedures with partially ordered preferences [9,10].

3. Background

In this section we give some basic notions about the stable marriage problem. In addition, we present some basic notions about local search.

3.1. Stable marriage problem

A stable marriage (SM) problem instance [16] consists of matching members of two different sets, usually called men and women. When there are n men and n women, the SM problem is said to have size n. Each person strictly ranks all members of the opposite sex. The goal is to match the men with the women so that there are no two people of opposite sex who would both rather marry each other than their current partners. If there are no such pairs (called blocking pairs) the marriage is *"stable"*.

Definition 1 (Marriage) *Given an SM instance P of size n, a marriage M is a one-to-one matching of the men and the women. If a man m and a woman w are matched in M, we write $M(m) = w$ and $M(w) = m$.*

Definition 2 (Blocking pair) *Given a marriage M, a pair (m, w), where m is a man and w is a woman, is a blocking pair iff m and w are not partners in M, but m prefers w to $M(m)$ and w prefers m to $M(w)$.*

Definition 3 (Stable Marriage) *A marriage M is stable iff it has no blocking pairs.*

A convenient and widely used SM representation is showed in Table 1, where each person is followed by his/her preference list in decreasing order.

Table 1. An example of an SM instance of size 8.

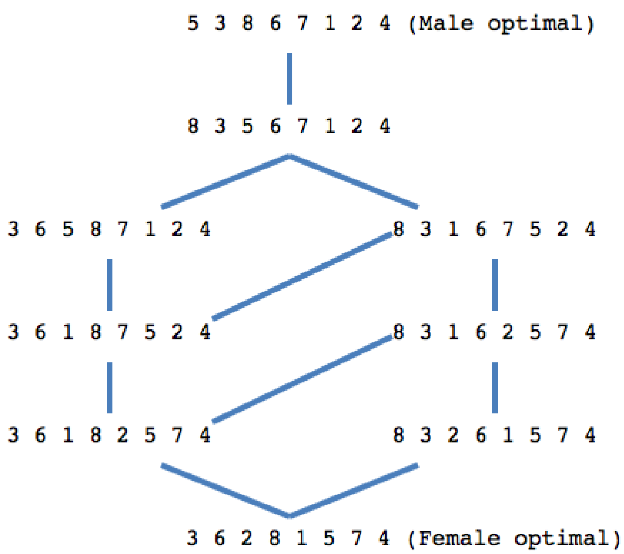
men's preference lists	women's preference lists
1: 5 7 1 2 6 8 4 3	1: 5 3 7 6 1 2 8 4
2: 2 3 7 5 4 1 8 6	2: 8 6 3 5 7 2 1 4
3: 8 5 1 4 6 2 3 7	3: 1 5 6 2 4 8 7 3
4: 3 2 7 4 1 6 8 5	4: 8 7 3 2 4 1 5 6
5: 7 2 5 1 3 6 8 4	5: 6 4 7 3 8 1 2 5
6: 1 6 7 5 8 4 2 3	6: 2 8 5 4 6 3 7 1
7: 2 5 7 6 3 4 8 1	7: 7 5 2 1 8 6 4 3
8: 3 8 4 5 7 2 6 1	8: 7 4 1 5 2 3 6 8

For example, Table 1 shows that man 1 prefers woman 5 to woman 7 to woman 1 and so on. It is known that, at least one stable marriage exists for every SM problem. For a given SM instance, we can define a partial order relation on the set of stable marriages.

Definition 4 (Dominance) *Let M and M' be two stable marriages. M dominates M' iff every man has a partner in M who is at least as good as the one he has in M'.*

Under the partial order given by the dominance relation, the set of stable marriages forms a distributive lattice [28]. Gale and Shapley give a polynomial time algorithm (GS) to find the stable marriage at the top (or bottom) of this lattice [6]. The top of such lattice is the male optimal stable marriage M_m, that is optimal from the men's point of view. This means that there are no other stable marriages in which each man is married with the same woman or with a woman he prefers to the one in M_m. The GS algorithm can also be used to find the female optimal stable marriage M_w (that is the bottom of the stable marriage lattice), which is optimal from the women's perspective, by just replacing men with women (and vice versa) before applying the algorithm. A clear way to represent this lattice is a Hasse diagram representing the transitive reduction of the partial order relation. Figure 1 shows the Hasse diagram of the SM in Table 1.

Figure 1. The Hasse diagram of the set of all stable marriages for the SM in Table 1.



A common concern with the standard Gale-Shapley algorithm is that it unfairly favors one sex at the expense of the other. This gives rise to the problem of finding "fairer" stable marriages. Previous work on finding fair marriages has focused on algorithms for optimizing an objective function that captures the happiness of both genders [15,21]. A different approach is to investigate non-deterministic procedures that can generate a random stable marriage from the lattice with a distribution that is as uniform as possible.

In [1] the authors use a Markov chain approach to sample the stable marriage lattice. More precisely, the edges of the lattice dictate exactly how to formalize the moves to walk from one stable marriage to another one, so that there are at most a linear number of moves at each step, these are easily identifiable, and they form reversible moves that connect the state space and converge to the uniform distribution. Unfortunately, Bhatnagar et al. show that this random walk has an exponential convergence time, which would appear to suggest that the approach may not be feasible in practice.

In this paper we also consider a variant of the SM problem where preference lists may include ties and may be incomplete. This variant is denoted by SMTI [23]. Ties express indifference in the preference ordering, while incompleteness models unacceptability only for certain partners.

Definition 5 (SMTI marriage) *Given a SMTI problem instance with n men and n women, a marriage M is a one-to-one matching between men and women such that partners accept each other. If a man m and a woman w are matched in M, we write $M(m) = w$ and $M(w) = m$. If a person p is not matched in M we say that he/she is single.*

Definition 6 (Marriage size) *Given a SMTI problem instance of size n and a marriage M, its size is the number of men (or women) that are married.*

Definition 7 (Blocking pairs in SMTI problems) *Consider a SMTI problem instance P, a marriage M for P, a man m and a woman w. A pair (m, w) is a blocking pair in M iff m and w accept each other and m is either single in M or he strictly prefers w to $M(m)$, and w is either single in M or she strictly prefers m to $M(w)$.*

Definition 8 (Weakly Stable Marriages) *Given a SMTI problem instance P, a marriage M for P is weakly stable iff it has no blocking pairs.*

As we will consider only weakly stable marriages, we will simply call them stable marriages. Given a SMTI problem instance, there may be several stable marriages of different size. If the size of a marriage coincides with the size of the problem, it is said to be a perfect matching. Solving a SMTI problem instance means finding a stable marriage with maximal size. This problem is NP-hard [31].

3.2. Local search

Local search [20] is one of the fundamental paradigms for solving computationally hard combinatorial problems. Local search methods in many cases represent the only feasible way for solving large and complex instances. Moreover, they can naturally be used to solve optimization problems.

Given a problem instance, the basic idea underlying local search is to start from an initial search position in the space of all solutions (typically a randomly or heuristically generated candidate solution, which may be infeasible, sub-optimal or incomplete), and to improve iteratively this candidate solution by means of typically minor modifications. At each *search step* we move to a position selected from a *local neighborhood*, chosen via a heuristic evaluation function. The evaluation function typically maps the current candidate solution to a number such that the global minima correspond to solutions of the given problem instance. The algorithm moves to the neighbor with the smallest value of the evaluation function. This process is iterated until a *termination criterion* is satisfied. The termination criterion is usually the fact that a solution is found or that a predetermined number of steps is reached, although other variants may stop the search after a predefined amount of time.

Different local search methods vary in the definition of the neighborhood and of the evaluation function, as well as in the way in which situations are handled when no improvement is possible. To ensure that the search process does not stagnate in unsatisfactory candidate solutions, most local search methods use randomization: at every step, with a certain probability a random move is performed rather than the usual move to the best neighbor.

4. Local search on Stable Marriages

We now present an adaptation of the local search schema to deal with the classical stable marriage problem. Then, we will point out the aspects that have to be changed to deal with SMTI problems.

Given an SM instance P, we start from a randomly generated marriage M. Then, at each search step, we compute the set BP of blocking pairs in M and compute the neighborhood, which is the set of all marriages obtained obtained by removing one of the blocking pairs in BP from M. Consider a blocking pair $bp = (m, w)$ in M, $m' = M(w)$, and $w' = M(m)$. Then, removing bp from M means obtaining a marriage M' in which m is married with w and m' is married with w', leaving the other pairs unchanged. To select the neighbor M' of M to move to, we use an evaluation function $f : \mathcal{M}_n \rightarrow Z$, where $\mathcal{M}_n$ is the set of all possible marriages of size n, and $f(M) = nbp(M)$. For each marriage M, $nbp(M)$ is the number of blocking pairs in M, and we move to one with the smallest value of f.

To avoid stagnation in a local minimum of the evaluation function, at each search step we perform a random walk with probability p (where p is a parameter of the algorithm), which removes a randomly chosen blocking pair in BP from the current marriage M. In this way we move to a randomly selected marriage in the neighborhood. The algorithm terminates if a stable marriage is found or when a maximal number of search steps or a timeout is reached.

This basic algorithm, called SML, has been improved in the computation of the neighborhood, obtaining SML1. When SML moves from one marriage to another one, it takes as input the current marriage M and the list $PAIRS$ of its blocking pairs and returns the marriage in the neighborhood of M with the best value of the evaluation function, i.e. the one with fewest blocking pairs. However, the number of such blocking pairs may be very large. Also, some of them may be useless, since their removal would surely lead to new marriages that will not be chosen by the evaluation function. This is the case for the so-called *dominated* blocking pairs. Algorithm SML1 considers only undominated blocking pairs.

Definition 9 (Dominance in blocking pairs) *Let (m, w) and (m, w') be two blocking pairs. Then (m, w) dominates (from the men's point of view) (m, w') iff m prefers w to w'. There is an equivalent concept from the women's point of view.*

Definition 10 (Undominated blocking pair) *A men- (resp., women-) undominated blocking pair is a blocking pair such that there is no other blocking pair that dominates it from the men's (resp., women's) point of view.*

It is easy to see that, if M is an unstable marriage, (m, w) a men- (resp., women-) undominated blocking pair in M, $m' = M(w)$, $w' = M(m)$, and M' is obtained from M by removing (m, w), there are no blocking pairs in M' in which m (resp., w) is involved. This property would not be true if we removed a dominated blocking pair. This is why we focus on the removal of undominated blocking pairs when we pass from one marriage to another in our local search algorithm.

Considering again the SM in Table 1 and the marriage 2 7 4 8 6 3 5 1. The blocking pair (m_8, w_4) dominates (from the men's point of view) (m_8, w_2). If we remove (m_8, w_2) from the marriage, (m_8, w_4) will remain. On the other hand, removing (m_8, w_4) also eliminates (m_8, w_2). Thus, removing (m_8, w_4) is more useful than removing (m_8, w_2).

By using the undominated blocking pairs instead of all the blocking pairs, we also limit the size of the neighborhood, since each man or woman is involved in at most one undominated blocking pair. Hence we have at most $2n$ neighbor marriages to evaluate.

Let us now analyse more carefully the set of blocking pairs considered by SML1. Consider the case in which a man m_i is in two blocking pairs, say (m_i, w_j) and (m_i, w_k), and assume that (m_i, w_j) dominates (m_i, w_k) from the men's point of view. Then, let w_j be in another blocking pair, say (m_z, w_j), that dominates (m_i, w_j) from the women's point of view. In this situation, SML1 returns (m_z, w_j) because it computes the undominated blocking pairs from men's point of view (which are (m_i, w_j) and (m_z, w_j)) and, among those, maintains the undominated ones from the women's point of view ((m_z, w_j) in this case). The removal of (m_z, w_j) automatically eliminates (m_i, w_j) from the set of blocking pairs of the marriage, since it is dominated by (m_z, w_j). However, the blocking pair (m_i, w_k) is still present because the blocking pair that dominated it (i.e. (m_i, w_j)) is not a blocking pair any longer. We also consider a procedure that will return in addition the blocking pair (m_i, w_k), so to avoid having to consider it again in the subsequent step of the local search algorithm. We call SML2 the algorithm obtained from SML1 by using this new way to compute the blocking pairs.

Since dominance between blocking pairs is defined from one gender's point of view, at the beginning of our algorithms we randomly choose a gender and, at each search step we change the role of the two genders. For example, in SML1, if we start by finding the undominated blocking pairs from the men's point of view and, among those, we keep only the undominated blocking pairs from the women's point of view, in the following second step we do the opposite, and so on. In this way we ensure that SML1 and SML2 are gender neutral.

Summarizing, we have defined three algorithms, called SML, SML1, and SML2, to find a stable marriage for a given SM instance. Such algorithms differ only by the set of blocking pairs considered to define the neighborhood.

5. Local search for SMTI problems

To adapt the SML algorithm to solve problems with ties and incomplete lists it is important to recall that an SMTI instance may have several stable marriages of different size. Thus, solving an SMTI problem instance means finding a stable marriage with maximal size. If the size of the marriage coincides with the size of the problem, it is said to be perfect and the algorithm can stop before the step limit. Otherwise the algorithm returns the best marriage found during search, defined as follows: if no stable marriage has been found, then the best marriage is the one with the smallest value of the evaluation function; otherwise, it is the stable marriage with fewest singles.

The SML algorithm is therefore modified in the following ways:

- the evaluation function has to take into account that some person may be not married, so we use: $f(M) = nbp(M) + ns(M)$, where, for each marriage M, $ns(M)$ is the number of singles in M which are not in any blocking pair.
- When we remove a blocking pair (m, w) from a marriage M, their partners $M(m)$ and $M(w)$ become single.
- The algorithm performs a random restart when a stable marriage is reached, since its neighborhood is empty (because it has no blocking pairs).

We call LTIU the modified algorithm for SMTI problems, obtained from SML by the above modifications and by using undominated blocking pairs.

6. Experiments

We tested our algorithms on randomly generated sets of SM and SMTI instances. For SM problems, we generated stable marriage problems of size n using the impartial culture model (IC) [14] which assigns to each man and to each woman a preference list uniformly chosen from the $n!$ possible total orders of n persons. This means that the probability of any particular ordering is $1/n!$.

For SMTI problems, we generated problem instances using the same method as in [12]. More precisely, the generator takes three parameters: the problem's size n, the probability of incompleteness

p_1, and the probability of ties p_2. Given a triple (n, p_1, p_2), a SMTI problem instance with n men and n women is generated, as follows:

1. For each man and woman, we generate a random preference list of size n, i.e., a permutation of n persons;

2. We iterate over each man's preference list: for a man m_i and for each women w_j in his preference list, with probability p_1 we delete w_j from m_i's preference list and m_i from w_j's preference list. In this way we get a possibly incomplete preference list.

3. If any man or woman has an empty preference list, we discard the problem and go to step 1.

4. We iterate over each person's (men and women's) preference list as follows: for a man m_i and for each woman in his preference list, in position $j \geq 2$, with probability p_2 we set the preference for that woman as the preference for the woman in position $j - 1$ (thus putting the two women in a tie).

Note that this method generates SMTI problem instances in which the acceptance is symmetric. If a man m does not accept a woman w, m is removed from w's preference list as well. This does not introduce any loss of generality because m and w cannot be matched together in any stable marriage.

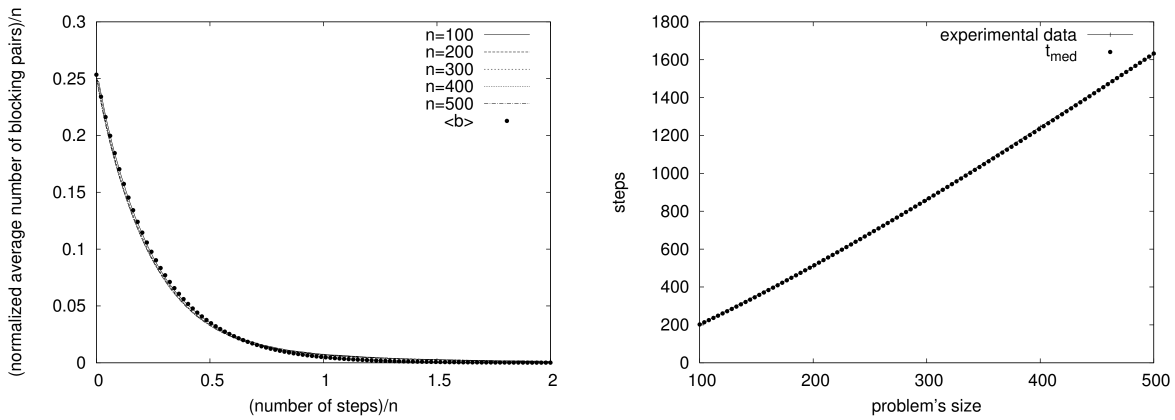
7. Results on SM problems

We measured the performance of our algorithms in terms of number of search steps. For these tests, we generated 100 SM problem instances for each of the following sizes: 100, 200, 300, 400 and 500. In the following we show only the results of our best algorithm, which is SML2. We studied how fast SML2 converges to a stable marriage, by measuring the ratio between the number of blocking pairs and the size of the problem during the execution. Figure 2(a) shows that SML2 has a very simple scaling behavior. Let us denote by $\langle b \rangle$ the average number of blocking pairs of the marriage found by SML2 for SM problem instances of size n after t steps. Then the experimental results shown in Figure 2(a) have a very good fit with the function $\langle b \rangle = an^2 2^{-bt/n}$, where a and b are constants computed empirically ($a \approx 0.25$ and $b \approx 5.7$). Figure 2(a) shows that the analytical function $\langle b \rangle$ has practically the same curve as the experimental data. The figure shows also that the average number of blocking pairs, normalized by dividing it by n, decreases during the search process in a way that is independent of the size of the problem.

We can use function $\langle b \rangle$ to conjecture the runtime behavior of our local search method. Consider the median number of steps, t_{med}, taken by SML2. Assume this occurs when half the problems have one blocking pair left and the other half have zero blocking pairs. Thus, $\langle b \rangle = \frac{1}{2}$. Substituting this value in the equation for $\langle b \rangle$, taking logs, solving for t_{med}, and grouping constant terms, we get $t_{med} = cn(d + 2\log_2(n))$ where c and d are constants. Hence, we can conclude that t_{med} grows as $O(n \log(n))$.

We then fitted this equation for t_{med} to the experimental data (using $c \approx 0.26$ and $d \approx -5.7$). The result is shown in Figure 2(b), where we see that the experimental data have the same curve as function t_{med}. This suggests that we can use such an equation to predict the number of steps our algorithms needs to solve a given SM instance.

Figure 2. Results using SML2.



(a) Blocking pair ratio during the execution. (b) Number of steps necessary to find a stable marriage.

7.1. Sampling the stable marriage lattice

We also evaluated the ability of SML2 to sample the lattice of stable marriages of a given SM problem. To do this, we randomly generated 100 SM problems for each size between 10 and 100, with step 10. Then, we ran the SML2 algorithm 500 times on each instance. To evaluate the sampling capabilities of SML2, we first measured the distance of the found stable marriages (on average) from the male-optimal marriage (the one that would be returned by the GS algorithm).

Given a SM problem instance P, consider a stable marriage M for P. The distance of M from M_m is the number of arcs from M to M_m in the Hasse diagram of the stable marriage lattice for P. This diagram can be computed in $O(n^2+n|S|)$ time [15], where S is the set of all possible stable marriages of a given SM instance. For each SM problem instance, we compute the average normalized distance from the male-optimal marriage considering 500 runs. Notice that normalizations is needed since different SM instances with the same size may have a different number of stable lattices. Then, we compute the average D_m[1] of these distances over all the 100 problems with the same size, which is therefore formally defined as $D_m = \frac{1}{100}\sum_{j=1}^{100}\frac{1}{500}\sum_{i=1}^{500}\frac{d_m(M_i,P_j)}{d_m(M_i,P_j)+d_w(M_i,P_j)}$, where $d_m(M_i,P_j)$ (resp., $d_w(M_i,P_j)$) is the distance of M_i from the male (resp., female)-optimal marriage in the lattice of an SM instance P_j. If $D_m = 0$, it means that all the stable marriages returned coincide with the male-optimal marriage. On the other extreme, if $D_m = 1$, it means that all stable marriages returned coincide with the female-optimal one. Figure 3(a) shows that, for the stable marriages returned by algorithm SML2, the average distance from the male-optimal is around 0.5.

This is encouraging but not completely informative, since an algorithm which returns the same stable marriage all the times, with distance 0.5 from the male-optimal would also have $D_m = 0.5$. To have more informative results, we consider the entropy of the stable marriages returned by SML2. This measures the randomness in the solutions. Let $f(M_i)$ be the frequency that SML2 finds a marriage M_i (for i in $[1,|S|]$) that is: $f(M_i) = \frac{1}{500}\sum_{j=1}^{500}\mathbb{1}_{M_i}(j)$, where $\mathbb{1}_{M_i}(j)$ is the indicator function that returns 1 if in

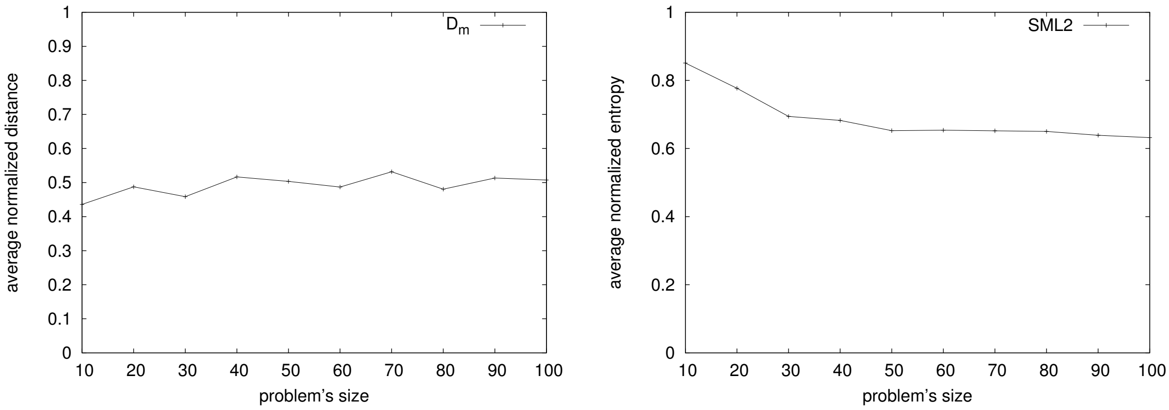
[1] With this measure we want to evaluate how far from the two extremes of the lattice are the marriages we find. However, it possible to give other definitions of stable matchings that belong to the middle of the lattice such as the one presented in [4].

the j-th execution the algorithm finds M_i, and 0 otherwise. The entropy $E(P)$ for each SM instance P (i.e., for each lattice) of size k is then: $E(P) = -\sum_{i=1 \in \{1..|S|\}} f(M_i) \log_2(f(M_i))$. In an ideal case, when each stable marriage in the lattice has a uniform probability of $1/k!$ to be reached, the entropy is $\log_2(|S|)$ bits. On the other hand, the worst case is when the same stable marriage is always returned, and the entropy is thus 0 bits. As we want a measure that is independent of the problem's size, we consider a normalized entropy, that is $E(P)/\log_2(|S|)$, which is in [0,1].

As we have 100 different problems for each size, we compute the average of the normalized entropies for each class of problems with the same size: $E_n = \frac{1}{100}\sum_{i=1}^{100} E(P_i)/\log_2(|S_i|)$, where S_i is the set of stable marriages of P_i.

Figure 3(b) shows that SML2 is not far from the ideal behavior. The normalized entropy starts from a value of 0.85 per bit at size 10, decreasing to just above 0.6 per bit as the problem's size grows.

Figure 3. Sampling with SML2.



(a) Average normalized distance D_m varying n.

(b) Entropy.

Considering both Figures 3(b) and 3(a), it appears that SML2 samples the stable marriage lattice very well. Considering also the distance D_m (Figure 3(a)), the possible outcomes appear to be equally distributed along the paths from the top to the bottom of the lattice.

To better evaluate the sampling capability of our approach, here we compare it to a *Markov chain* approach (MC) [1], defined by using rotations exposed in each stable marriage.

More precisely, suppose that M_i is current marriage. Then the next marriage M_{i+1} is computed follows:

- (i) with probability 1/3: it randomly chooses a man and, if he is part of a woman-improving rotation ρ, it moves to $M_{i+1} = M_i/\rho$;
- (ii) with probability 1/3: it randomly chooses a man and, if he is part of a man-improving rotation ρ, it moves to $M_{i+1} = M_i/\rho$;
- (iii) with probability 1/3, it moves to $M_{i+1} = M_i$.

Since a rotation and its inverse contain the same people, and the probability of picking a particular rotation is proportional to the number of couples it contains, this Markov chain is reversible. This approach converges in exponential time to the uniform distribution over the stable marriages. We consider the entropy and distance from the male-optimal of MC computed on executions where we vary the number of steps from 10 to 200. While the entropy of MC increases quite rapidly, the distance from the top of the lattice (i.e., from the male-optimal) increases more slowly (see Fig. 4(a) and Fig. 4(b)). For each problem instance in the test set, we start MC from the male-optimal marriage and take the stable marriage returned by MC after exactly the same number of steps needed by our algorithm to find a stable marriage for that instance. Then we measure and compare the entropy and the distance from the male-optimal for MC to those of our algorithm (SML2). While the entropy of MC is roughly the same as that of our algorithm, the distance from the male-optimal achieved by our approach (about 0.5) is on average higher that that achieved by MC (about 0.2) (see Fig. 4(c)).

Summarizing, our approach is efficient and it has sampling capabilities comparable with a Markov chain approach considering the same number of steps, and may even perform slightly better considering the distance measured from the top or the bottom of the lattice.

8. Results on SMTI problems

We generated random SMTI problem instances of size 100, by letting p_2 vary in [0, 1.0] with step 0.1, and p_1 vary in [0.1, 0.8] with step 0.1 (above 0.8 the preference lists start to be empty). For each parameter combination, we generated 100 problem instances. Moreover, the probability of the random walk is set to p=20% and the search step limit is s=50000.

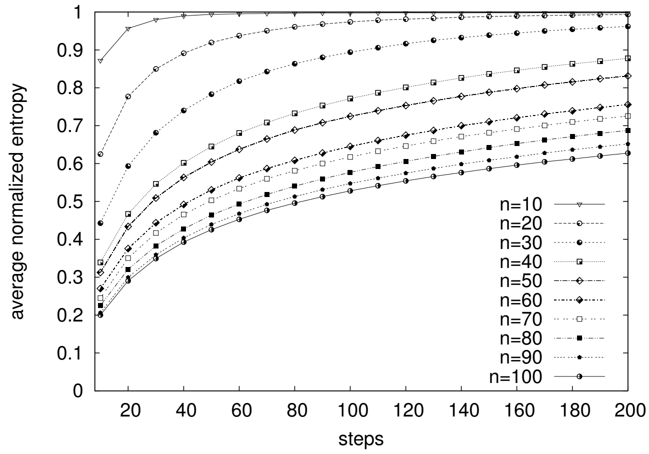
We start by showing the average size of the marriages returned by LTIU. In Figure 5(a) we see that LTIU almost always finds a perfect marriage (that is, a stable marriage with no singles). Even in settings with a large amount of incompleteness (that is, $p_1 = 0.7$ - 0.8) the algorithm finds very large marriages, with only 2 singles on average.

We also consider the number of steps needed by our algorithm. From Figure 5(b), we can see that the number of steps is less than 2000 most of the time, except for problems with a large amount of incompleteness (i.e. $p_1 = 0.8$). As expected, with $p_1 > 0.6$ the algorithm requires more steps. In some cases, it reaches the step limit of 50000. Moreover, as the percentage of ties rises, stability becomes easier to achieve and thus the number of steps tends to decrease slightly. From the results we see that complete indifference (p_2=1) is a special case. In this situation, the number of steps increases for almost every value of p_1. This is because the algorithm makes most of its progress via random restarts. In these problems every person (if accepted) is equally preferred to all others accepted. The only blocking pairs are those involving singles who both accept each other. Hence, after a few steps all singles that can be married are matched, stability is reached, and the neighborhood becomes empty. The algorithm therefore randomly restarts. In this situation it is very difficult to find a perfect matching and the algorithm therefore often reached the step limit.

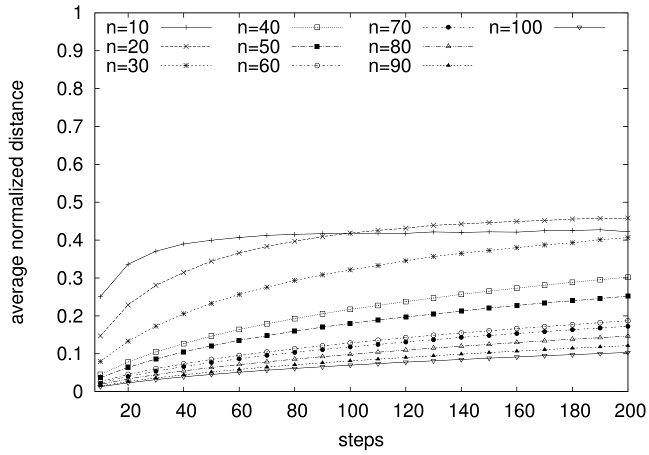
The algorithm is fast. It takes, on average, less than 40 seconds to give a result even for very difficult problems (see Figure 5(c)). As expected, with $p_2 = 1$ the time increases for the same reason discussed above concerning the number of steps.

Re-considering Figure 5(a) and the fact that all the marriages the algorithm finds are stable, we notice that most of the marriages are perfect. From Figure 5(d) we see that the average percentage of matchings

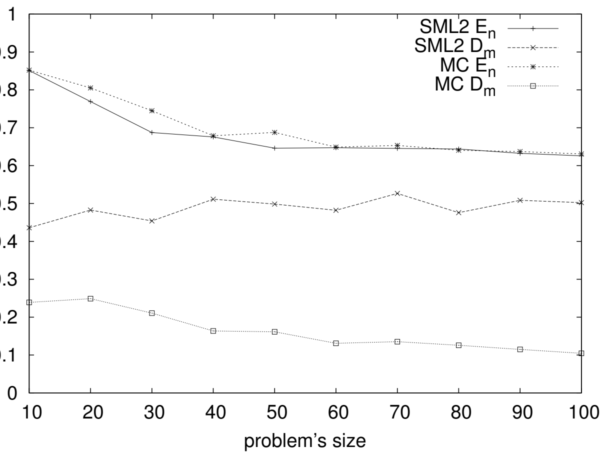
Figure 4. Average runtime entropy of MC (a), average runtime distance from the male-optimal of MC (b), Local Search vs. MC in terms of entropy and distance from the male-optimal (c).



(a)

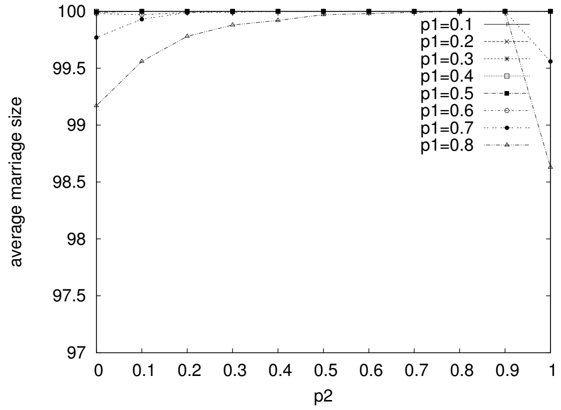


(b)

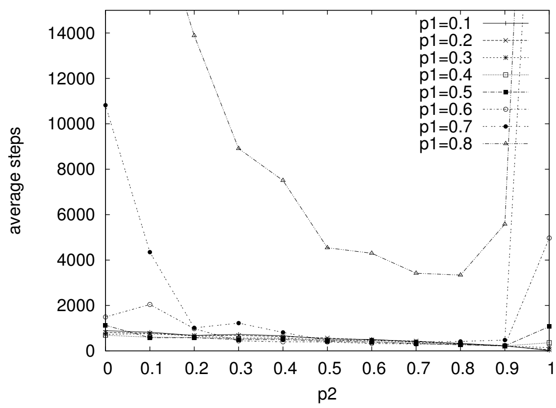


(c)

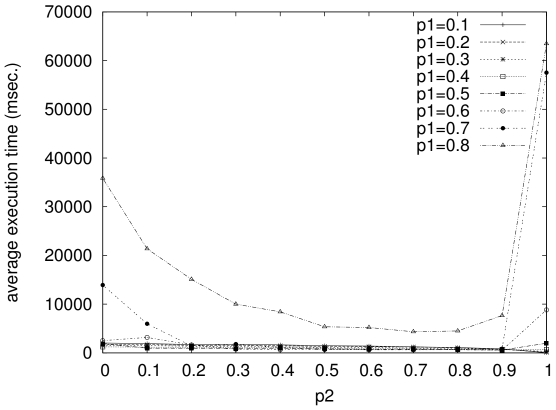
Figure 5. LTIU varying p_2 for different values of p_1.



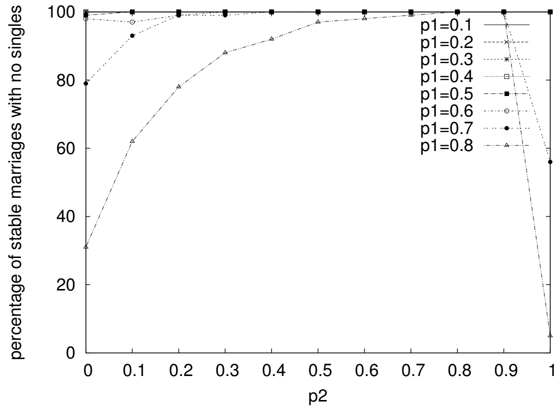
(a) Average size of marriages.



(b) Average number of steps.



(c) Average execution time.

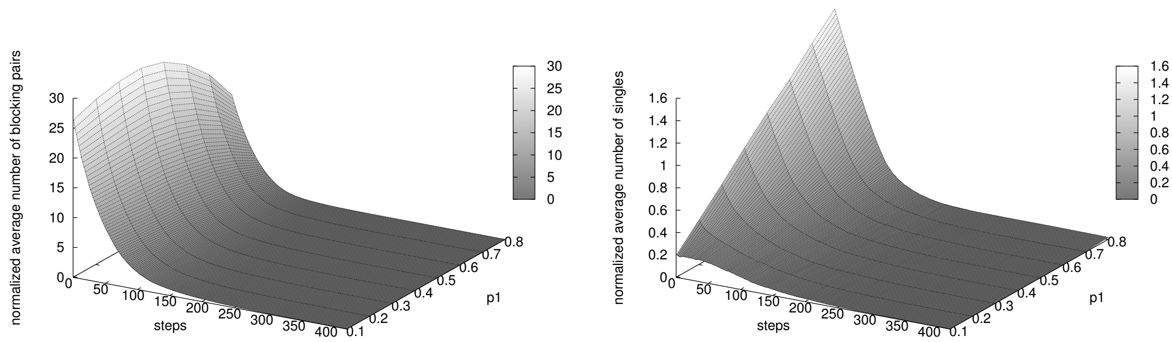


(d) Percentage of perfect matchings.

that are perfect is almost always 100% and this percentage only decreases when the incompleteness is large. We compared our local search approach to the one in [12]. In their experiments, they measured the maximum size of the stable marriages in problems of size 10, fixing p_1 to 0.5 and varying p_2 in [0,1]. We did similar experiments, and obtained stable marriages of a very similar size to those reported in [12]. This means that although our algorithm is incomplete in principle, it always appears to find an optimal solution in practice, and for small sizes it behaves like a complete algorithm in terms of size of the returned marriage. However, it can also tackle problems of much larger sizes, still obtaining optimal solutions most of the times.

We also considered the runtime behavior of our algorithm. In Figure 6(a) we show the average normalized number of blocking pairs and, in Figure 6(b), the average normalized number of singles of the best marriage as the execution proceeds. Although the step limit is 50000, we only plot results for the first steps because the rest is a long plateau that is not very interesting. We show the results only for $p_2 = 0.5$. However, for greater (resp., lower) number of ties the curves are shifted slightly down (resp., up). From Figure 6(a) we see that the average number of blocking pairs decreases very rapidly, reaching 5 blocking pairs after only 100 steps. Then, after 300-400 steps, we almost always reach a stable marriage, irrespective of the value of p_1. Considering Figure 6(b), we see that the algorithm starts with more singles for greater values of p_1. This happens because, with more incompleteness, it is more

Figure 6. LTIU runtime behaviour (p_2=0.5).



(a) Average normalized number of blocking pairs. (b) Average normalized number of singles.

difficult for a person to be accepted. However, after 200 steps, the average number of singles becomes very small no matter the incompleteness in the problem.

Looking at both Figures 6(a) and 6(b), we observe that, although we set a step limit s = 50000, the algorithm reaches a very good solution after just 300-400 steps. After this number of steps, the best marriage found by the algorithm usually has no blocking pairs nor singles. This appears to be largely independent of the amount of incompleteness and the number of ties in the problems. Hence, for SMTI problem instances of size 100 we could set the step limit to just 400 steps and still be reasonably sure that the algorithm will return a stable marriage of a large size, no matter the amount of incompleteness and ties.

9. Local search by swapping ties

In previous sections we have presented two local search algorithms which start from a random marriage and try to converge to a stable marriage with maximum size by removing blocking pairs. In this section we present another local search approach suggested by Prosser [41] to find the largest stable marriage of a given SMTI instance I. This approach is based on the observation that, by breaking all ties, I becomes an SMI instance, say I', and a stable marriage in I' is also stable in I, since we are considering weak stability. Furthermore, we recall that all stable marriages of a given SMI have the same size, and one of them can be found in polynomial time using the Gale Shapley algorithm.

More precisely, we consider SMTIs with ties of length two (the problem of finding a maximum size stable matching still remains NP-hard in this special case of SMTI [31]), and we associate a weight in [0,1] to each way of breaking a tie. Initially, such weights are all set to 0.5.

Our method, which is described in Algorithm LST, works as follows. It takes as input an SMTI instance P, an integer max_steps and a random walk probability p. First, it breaks the ties in P thus obtaining the SMI instance Q, then it repeats a sequence of actions as long as the number of steps is lower than max_steps. The first of these actions is to compute *GS(Q)* that finds the male optimal stable matching M of Q by applying the Gale Shapley algorithm to the SMI instance Q. If the returned marriage M is perfect, then the algorithm returns this marriage. Otherwise, if *rand()* $\leq p$, i.e., if the random number in interval [0,1] generated by the function *rand()* is lower than or equal to the random walk probability p, then it selects a random tie in P, and assigns this tie to *tie_neighbest*, it applies the procedure *swap_tie*(Q, t_i) which returns an SMI instance Q that is obtained by Q where the order of the elements in tie ti in P is swapped. Then, it finds the male optimal stable matching M of Q and it recalls the size of M as *max_neigh_size* and M as *M_neighbest*. Otherwise, if *rand*$(p) > p$, for every allowed tie (see the next paragraph) in P, it applies the procedure *swap_tie*(Q, t_i) which returns an SMI instance R that is obtained by Q where the order of the elements in tie ti in P is swapped. Then, it finds the male optimal stable matching M of R. If M is the best stable matching found in the neighbourhood then it recalls that tie as *tie_neighbest* and that marriage as *M_neighbest*. After having considered all the allowed ties, it moves to a new SMI instance Q obtained from Q and *tie_neighbest* by applying the procedure *swap_tie(Q,tie_neighbest)*. If the size of the obtained stable matching is larger than the overall best one obtained so far it increases the weight of *tie_neighbest* by 0.05, otherwise it decreases the weight of *tie_neighbest* by 0.05.

We can have different versions of the algorithm LST depending on the meaning of the sentence *allowed tie* t *in* P in line 12. We consider as "allowed" the ties which have a weight greater than a fixed threshold or certain percentage of ties with highest weight. In this way we speed up the search by reducing the size of the neighborhood. We call LSTt (where t is the threshold) the algorithm that limits the neighborhood via a threshold and LSTk (where k is the percentage of best ties considered) the other one.

9.1. Experimental evaluation

We generated SMTI problem instances as in previous section except for the probability of ties (p_2). For example, if we generate a problem of size n=100, with probability of incompleteness p_1=0.1 and probability of ties p_2=0.2, then, since p_1=0.1, the average length of preference lists will be 90 and, since p_2=0.2, each preference list will have about 9 ties of length 2.

We generated 100 problems for each combination of n, p_1 and p_2 varying n in {10, 30, 50, 70, 90}, p_1 in [0.1, 0.8] and p_2 in [0.1, 1.0] and fixing a limit of 20000 steps.

We ran our algorithms LSTt and LSTk on this test set and we also compare the results against our LTIU algorithm.

We first measured the average size (normalized w.r.t. the size of the problem) of the stable marriages returned by our algorithms. All three algorithms, find larger marriages when the number of ties increases and when the incompleteness in preference lists decreases. In fact, with more ties and longer preference lists, there is less probability of having a blocking pair and more chances for singles to get married.

For instance, Figure 7(a) shows the results for LSTk when n=10 and k=50%. The results for LSTt and LTIU are very similar. Only for p_1=0.7-0.8 and high values of p_2 LTIU finds slightly smaller marriages. Figure 7(b) shows a comparison of the three algorithms on problems of size 10 and 30.

For n=10, the size of the marriages vary at most of only 0.02 comparing LSTk versus LSTt (LTIU gives practically the same results as LSTt) when we vary the size of the problems. We can also notice that the size of the marriages tends to increase when the size of the problems increases. For instance, Figure 7(c) shows the results for LSTt and it easy to see that, for the same values of the other parameters, it finds larger marriages as n increases. The same results are obtained by the other algorithms. We

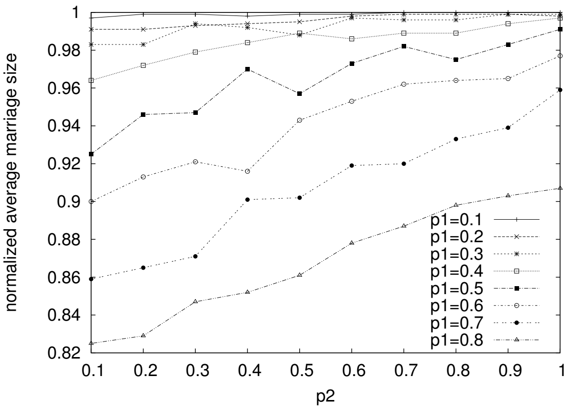
Algorithm 1: LST

```
input  : a SMTI problem instance P, an integer max_steps, a probability p of random walk
output: a marriage
Q ← breakties(P)
steps ← 0
max_size ← −1
repeat
    max_neigh_size ← −1
    M ← GS(Q)
    if M is a perfect matching then
        return M
    if rand() ≤ p then
        tie_neighbest ← a random tie in P
        Q ← swap_tie(Q, ti)
        M ← GS(Q)
        max_neigh_size ← |M|
        M_neighbest ← M
    else
        foreach allowed tie ti in P do
            R ← swap_tie(Q, ti)
            M ← GS(R)
            if |M| > max_neigh_size then
                max_neigh_size ← |M|
                M_neighbest ← M
                tie_neighbest ← ti
        Q ← swap_tie(Q, tie_neighbest)
    if max_neigh_size > max_size then
        max_size ← max_neigh_size
        M_best ← M_neighbest
        increase weight of tie_neighbest
    else
        decrease weight of tie_neighbest
    steps ← steps + 1
until steps ≥ max_steps;
return M_best
```

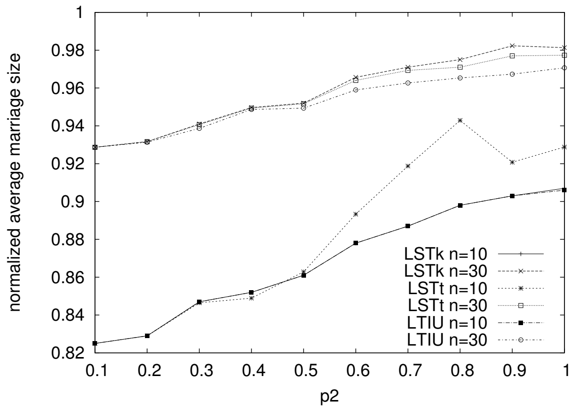
conjecture that the reason for this behavior is that, considering SMI instances, the probability of having a certain person in at least one preference list, say P_l, is very high even with small sizes and a lot of incompleteness. More precisely, the probability of having a person p in at least one preference list in an SMI of size n, denoted by $P_l(n, p_1)$, is $1 - p_1^n$. Moreover, the probability to be in exactly k lists is:

$$\left[(1 - p_1)^k \cdot p_1^{n-k}\right] \binom{n}{k} \tag{1}$$

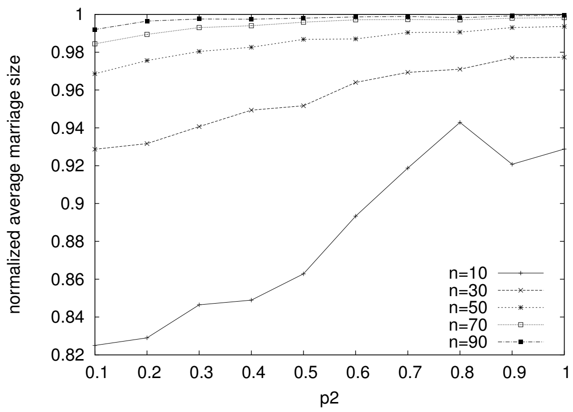
Figure 7. Normalized average size of marriages for LSTk, LSTt and LTIU.



(a) Normalized average size of marriages found by LSTk using $k = 50\%$ on SMTIs of size 10.



(b) Normalized average size for LSTk, LSTt and LTIU on problems of size 10 and 30. Fixing p_1=0.8, $k = 50\%$ and $t = 0.5$.



(c) Normalized average size for LSTt varying n and fixing p_1=0.8 and $t = 0.5$.

Then, the probability to be in at least k lists is:

$$\sum_{i=k}^{n} \left\{ \left[(1 - p_1)^i \cdot p_1^{n-i} \right] \binom{n}{i} \right\} \tag{2}$$

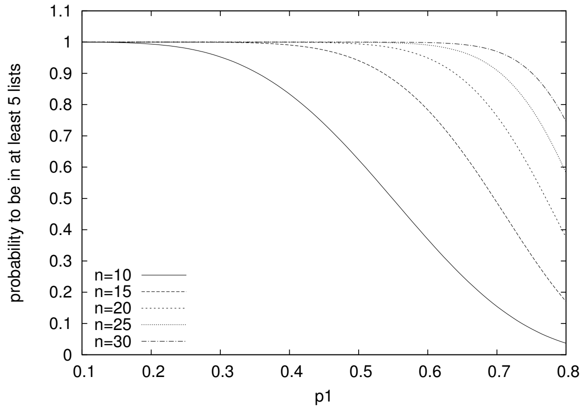
Finally, since our generator rejects problems with empty preference lists, in our test set each person is always in at least one preference list. Thus the probability to be in at least k lists becomes:

$$P(n, p_1, k) = \frac{\sum_{i=k}^{n} \left\{ \left[(1 - p_1)^i \cdot p_1^{n-i} \right] \binom{n}{i} \right\}}{1 - p_1^n} \tag{3}$$

For example, Figure 8 shows how slowly $P(n, p_1, 5)$ decreases when varying p_1 for different values of n. Thus, in general, the probability for a person to be in more than one preference list rises with the size of the problem. Therefore, having a perfect matching or a marriage with very high cardinality is more probable in bigger problems than in smaller ones.

We then considered the average number of steps needed by the algorithms to finish their execution. As can be expected, the number of steps increases as the incompleteness p_1 rises. This happens for all

Figure 8. Probability for a person to be in at least 5 preference lists varying p_1.



algorithms and all problems sizes, and it is more clear as n increases. This can be seen for example in Figure 9(a) that shows the average number of steps for LSTt on problems of size 10 and in Figure 9(b) that shows the results for n=30.

Figure 9(c) shows that the number of steps needed by LSTt for p_1=0.8 decreases as n increases. Moreover, it decreases as the amount of ties (p_2) increases. This behavior is the same for the other algorithms and is due to the increased probability of finding a perfect matching on larger problems. For instance, Figure 9(d) shows how steps vary considering problems of size n=30 and n=90.

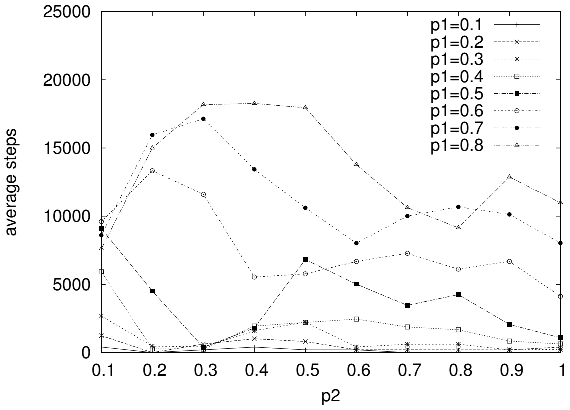
We also measured the execution time of our algorithms. The execution time is mainly influenced by the size and nature of the neighborhood that has to be explored at each search step. The neighborhood used by LTIU depends on blocking pairs and so it is larger in problems with few ties. On the other hand, the neighborhoods defined for LSTt and LSTk are bigger as the number of ties arises. For these reasons, the execution time of LTIU tends to slightly decrease as p_2 increases no matter the size of the problem for fixed values of p_1 (see Figure 10(a)). Figures 10(b) and 10(c) show respectively the execution time of LSTk and LSTt. In both cases the execution is longer as p_2 becomes larger. The difference is that the size of the neighborhood in LSTt varies dynamically according the weights of the ties and the threshold t. This speeds up drastically the algorithm and, as we can see, the execution time of LSTt is about half of the execution time of LSTk.

Summarizing, both LSTk and LSTt are effective in terms of the size of the returned marriages but, when we take into account also the execution time, LSTt has to be preferred.

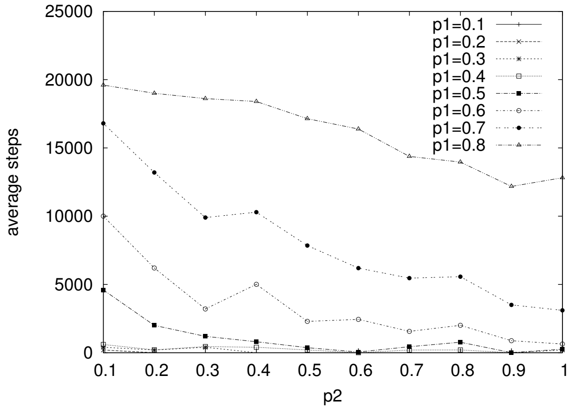
10. Conclusions and future work

We have presented a local search approach for solving the classical stable marriage (SM) problem instances and its variant with ties and incomplete lists (SMTI). Our algorithm for SM problem instances has a simple scaling and size independent behavior and it is able to find a solution in a number of steps which grows as little as $O(n \log(n))$. Moreover it samples the stable marriage lattice reasonably well also when compared with a Markov chain approach. It is thus a fair method to generate random stable marriages We also provided an algorithm for SMTI problems which is both fast and effective at finding large stable marriages for problems of sizes not considered before in the literature. The algorithm was usually able to obtain a very good solution after a small amount of time.

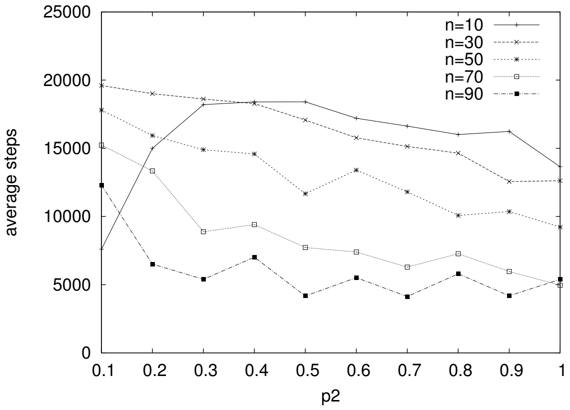
Figure 9. Average number of steps for LSTt, LSTk, and LTIU.



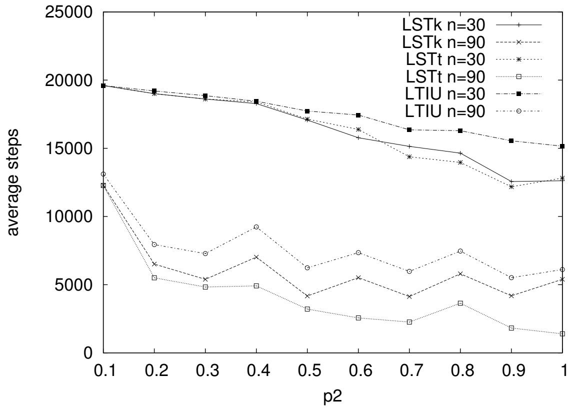
(a) Average number of steps for LSTt for n=10.



(b) Average number of steps for LSTt for n=30.



(c) Average number of steps for LSTt varying n and fixing p_1=0.8.



(d) Average number of steps for LSTk, LSTt, and LTIU on problems of size 30 and 90. Fixing p_1=0.8, $k = 50\%$ and $t = 0.5$.

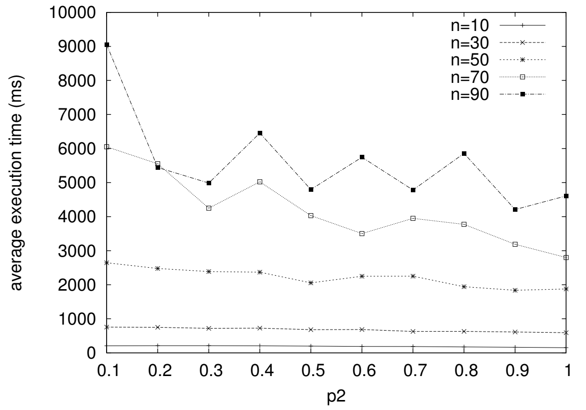
Notice that it is important to validate our local search techniques on larger problem instances. We plan to do that in our future research. Moreover, we intend to compare the algorithms shown in Section 9 with Algorithm ShiftBrk in [17]. We plan also to apply a local search approach also to the hospital-resident problem and to compare our algorithms to the ones in [22], where residents express their preferences in strict order and hospitals allow ties in their preferences and have a finite number of posts each.

Acknowledgements We would like to thank the reviewers for their very useful comments. This work has been partially supported by the MIUR PRIN 20089M932N project “Innovative and multi-disciplinary approaches for constraint and preference reasoning”.

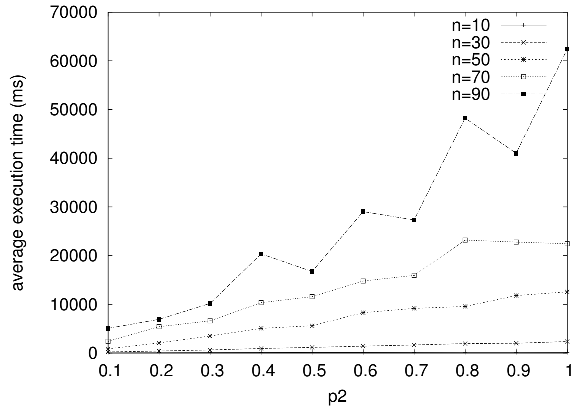
References

1. N. Bhatnagar, S. Greenberg, and D. Randall. Sampling stable marriages: why spouse-swapping won’t work. In *SODA ’08: Proceedings of the nineteenth annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1223–1232. Society for Industrial and Applied Mathematics, 2008.

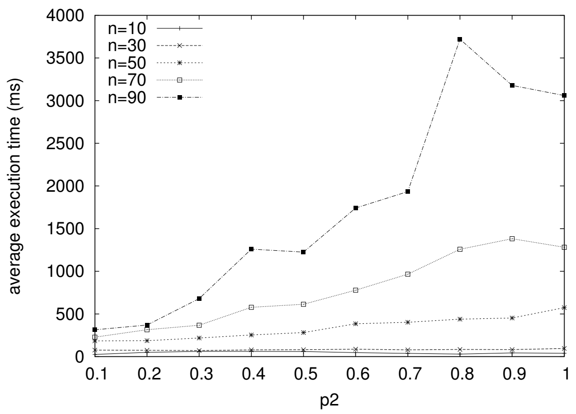
Figure 10. Average execution time for LTIU, LSTk, and LSTt.



(a) Average execution time for LTIU varying n for p_1=0.8.



(b) Average execution time for LSTk varying n for p_1=0.8 and k=50%.



(c) Average execution time for LSTt varying n for p_1=0.8 and t=0.5.

2. P. Biró, D. F. Manlove, and S. Mittal. Size versus stability in the marriage problem. *Theoretical Computer Science*, 411:1828–1841, March 2010.
3. I. Brito and P. Meseguer. Distributed stable matching problems with ties and incomplete lists. In Frédéric Benhamou, editor, *Principles and Practice of Constraint Programming - CP 2006*, volume 4204 of *LNCS*. Springer, 2006.
4. C. T. Cheng. Understanding the generalized median stable matching. *Algorithmica*, 58(1):34–51, 2010.
5. R. G. Downey and M. R. Fellows. *Parametrized complexity*. Springer, 1999.
6. D. Gale and L. S. Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.
7. M. Gelain, M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Local search algorithms on the stable marriage problem: Experimental studies. In *Proceedings of ECAI 2010*, volume 215 of *Frontiers in Artificial Intelligence and Applications*, pages 1085–1086. IOS Press, 2010.

8. M. Gelain, M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Local search for stable marriage problems with ties and incomplete lists. In *Proceedings of PRICAI 2010*, volume 6230 of *LNCS*, pages 64–75. Springer, 2010.
9. M. Gelain, M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Male optimal and unique stable marriages with partially ordered preferences. In *Proc. CARE 2009/2010*. Springer LNAI 6066, 2010.
10. M. Gelain, M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Male optimality and uniqueness in stable marriage problems with partial orders - Extended abstract. In *Proc. AAMAS'10*, 2010.
11. M. Gelain, M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Procedural fairness in stable marriage problems. In *Proceedings of AAMAS 2011*, pages 1209–1210. IFAAMAS, 2011.
12. I. P. Gent and P. Prosser. An empirical study of the stable marriage problem with ties and incomplete lists. In *Proc. ECAI 2002*, pages 141–145, 2002.
13. I. P. Gent, P. Prosser, B. M. Smith, and T. Walsh. Sat encodings of the stable marriage problem with ties and incomplete lists. In *SAT 2002*, pages 133–140, 2002.
14. G.T. Guilbauld. Les théories de l'intérêt général et le problème logique de l'agrégation. *Économie Appliquée*, 5(4):501–584, 1952.
15. D. Gusfield. Three fast algorithms for four problems in stable marriage. *SIAM Journal on Computing*, 16(1):111–128, 1987.
16. D. Gusfield and R. W. Irving. *The Stable Marriage Problem: Structure and Algorithms*. MIT Press, Boston, MA, 1989.
17. M. M. Halldórsso, K. Iwama, S. Miyazaki, and H. Yanagisawa. Improved approximation results for the stable marriage problem. *ACM Transactions on Algorithms*, 3(3), 2007.
18. M. M. Halldórsson, R. W. Irving, K. Iwama, D. Manlove, S. Miyazaki, Y. Morita, and S. Scott. Approximability results for stable marriage problems with ties. *Theor. Comput. Sci.*, 306(1-3):431–447, 2003.
19. M. M. Halldórsson, K. Iwama, S. Miyazaki, and H. Yanagisawa. Randomized Approximation of the Stable Marriage Problem. In *Proceedings of COCOON 2003*, volume 2697 of *LNCS*, pages 339–350. Springer, 2003.
20. H. H. Hoos and E. Tsang. Local search methods. In F. Rossi, P Van Beek, and T. Walsh, editors, *Handbook of Constraint Programming*. Elsevier, 2006.
21. R. W. Irving, P. Leather, and D. Gusfield. An efficient algorithm for the "optimal" stable marriage. *J. ACM*, 34(3):532–543, 1987.
22. R. W. Irving and D. F. Manlove. Finding large stable matchings. *J. Exp. Algorithmics*, 14:2:1.2–2:1.30, January 2010.
23. K. Iwama, D. F. Manlove, S. Miyazaki, and Y. Morita. Stable marriage with incomplete lists and ties. In *Proc. ICALP*, volume 1644 of *LNCS*, pages 443–452, 1999.
24. K. Iwama, S. Miyazaki, and K. Okamoto. A (2-c(log N/N))-Approximation Algorithm for the Stable Marriage Problem. In *Algorithm Theory - SWAT 2004*, volume 3111 of *LNCS*, pages 349–361. Springer, 2004.
25. K. Iwama, S. Miyazaki, and N. Yamauchi. A 1.875: approximation algorithm for the stable marriage problem. In *Proceedings of SODA'07*, pages 288–297. SIAM, 2007.

26. K. Iwama, S. Miyazaki, and H. Yanagisawa. A 25/17-Approximation Algorithm for the Stable Marriage Problem with One-Sided Ties. *Algorithmica*, 2012.
27. Z. Király. Better and Simpler Approximation Algorithms for the Stable Marriage Problem. *Algorithmica*, 60(1):3–20, 2011.
28. D.E. Knuth. *Marriages Stables*. Les Presses du l'Universitè de Montrèal, 1976.
29. F. Laburthe. Choco, a constraint programming kernel for solving combinatorial optimization problems. http://choco-solver.net.
30. D. F. Manlove. *Algorithmics of Matching Under Preferences* . World Scientific Publishing, 2013.
31. D. F. Manlove, R. W. Irving, K. Iwama, S. Miyazaki, and Y. Morita. Hard variants of stable marriage. *Theoretical Computer Science*, 276(1-2):261–279, 2002.
32. D. Marx. Local search. *Parameterized Complexity News*, 3:7–8, 2008.
33. D. Marx and I. Schlotter. Parameterized complexity and local search approaches for the stable marriage problem with ties. *Algorithmica*, 58(1):170–187, 2010.
34. D. Marx and I. Schlotter. Stable assignment with couples: Parameterized complexity and local search. *Discrete Optimization*, 8(1):25–40, 2011.
35. E. McDermid. A 3/2-approximation algorithm for general stable marriage. In *Proceedings of ICALP (1)*, volume 5555 of *LNCS*, pages 689–700. Springer, 2009.
36. M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Manipulation and gender neutrality in stable marriage procedures. In *Proc. AAMAS'09*, volume 1, pages 665–672, 2009.
37. M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Manipulation complexity and gender neutrality in stable marriage procedures. *Jornal of Autonomous Agents and Multi-Agent Systems*, 22(1):183–199, 2011.
38. M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Stability in Matching Problems with Weighted Preferences. In *Proc. ICAART'11*, pages 45–53. SciTePress, 2011.
39. M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Weights in stable marriage problems increase manipulation opportunities. In *Proc. TARK'11* - Best Poster Award, 2011.
40. M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Stability and Optimality in Matching Problems with Weighted Preferences. In *Agents and Artificial Intelligence 2011 - ICAART'11 - Revised Selected Papers*, pages 319–333. CCIS 271 Springer-Verlag, 2012.
41. P. Prosser. Private communication. 2010.
42. D. Manlove R. W. Irving. Approximation algorithms for hard variants of the stable marriage and hospitals/residents problems. *J. Comb. Optim*, 16(3):279–292, 2008.
43. A. E. Roth. On the allocation of residents to rural hospitals: A general property of two-sided matching markets. *Econometrica*, 54(2):425427, 1986.
44. A. E. Roth and J. H. V. Vate. Random paths to stability in two-sided matching. *Econometrica*, 58(6):1475–1480, 1990.
45. S. Scott. *A study of stable marriage problems with ties*. PhD thesis, University of Glasgow, 2005.