Two-sided profile-based optimality in the stable marriage problem

Frances Cooper$^1$ and David Manlove$^{1\dagger}$

$^1$School of Computing Science, University of Glasgow

Abstract

We study the problem of finding “fair” stable matchings in the Stable Marriage problem with Incomplete lists ($\text{SMI}$). In particular, we seek stable matchings that are optimal with respect to profile, which is a vector that indicates the number of agents who have their first-, second-, third-choice partner, etc. In a rank maximal stable matching, the maximum number of agents have their first-choice partner, and subject to this, the maximum number of agents have their second-choice partner, etc., whilst in a generous stable matching $M$, the minimum number of agents have their $d$th-choice partner, and subject to this, the minimum number of agents have their $(d-1)$th-choice partner, etc., where $d$ is the maximum rank of an agent’s partner in $M$. Irving et al. $^{[13]}$ presented an $O(n^5 \log n)$ algorithm for finding a rank-maximal stable matching, which can be adapted easily to the generous stable matching case, where $n$ is the number of men / women. An $O(n^{1.5})$ algorithm for the rank-maximal stable problem was later given by Feder $^{[6]}$. However these approaches involve the use of weights that are in general exponential in $n$, potentially leading to inaccuracies or memory issues upon implementation. In this paper we present an $O(n^5 \log n)$ algorithm for finding a rank-maximal stable matching using an approach that involves weights that are polynomially-bounded in $n$. We show how to adapt our algorithm for the generous case to run in $O(n^2 d^3 \log n)$ time. Additionally we conduct an empirical evaluation to compare various measures over many different types of “fair” stable matchings, including rank-maximal, generous, egalitarian, sex-equal and median stable matchings. In particular, we observe that a generous stable matching is typically considerably closer than a rank-maximal stable matching in terms of the egalitarian and sex-equality optimality criteria.

1 Introduction

Background. The Stable Marriage problem ($\text{SM}$) was first introduced in Gale and Shapley’s seminal paper “College Admission and the Stability of Marriage”. In an instance of $\text{SM}$ we have two sets of agents, men and women (of equal number, henceforth $n$), such that each man ranks every woman in strict preference order, and vice versa. An extension to $\text{SM}$ known as the Stable Marriage problem with Incomplete lists ($\text{SMI}$) allows each man (woman) to rank a subset of women (men).

Generalisations of $\text{SMI}$ in which one or both sets of agents may be multiply assigned have been extensively applied in the real-world. The National Resident Matching Program (NRMP) is a long standing matching scheme in the US (beginning in 1952) which assigns graduating medical students to hospitals $^{[20]}$. Other examples include the assignment of children to schools in Boston $^{[1]}$ and the allocation of high-school students to university places in China $^{[27]}$.

Gale and Shapley $^{[5]}$ described linear time algorithms to find a stable matching in an instance of $\text{SM}$. These classical algorithms find either a man-optimal (or woman-optimal) stable matching in which every man (woman) is assigned to their best partner in any stable matching and every woman (man) is assigned to their worst partner in any stable matching. Favouring one set of agents over the
other is often undesirable and so we look at the notion of a “fair” matching in which the happiness of both sets of agents is taken into account.

There may be many stable matchings in any given instance of SM and there are several different criteria that may be used to describe an optimal or “fair” stable matching. The rank of an agent $a$ in a stable matching $M$ is the position $a$’s partner on $a$’s preference list, while the degree $d$ of $M$ is the highest rank of any agent in $M$. We might wish to limit the number of agents with large rank. A minimum regret stable matching is a stable matching that minimises $d$ and can be found in $O(n^2)$ time [10]. Another type of optimality criteria, uses an arbitrary weight function to find a minimum (maximum) weight stable matching, which is a stable matching that has minimum (maximum) weight among the set of all stable matchings. A special case of this is known as the egalitarian stable matching which minimises the sum of ranks of all agents. Irving et al. [13], gave an algorithm to find an egalitarian stable matching in $O(n^3)$ time and discussed how to generalise their method to the minimum (and maximum) weight stable marriage problem. Feder [6] later improved on this showing that a minimum weight stable matching may be found in $O(n^{3.5})$ time using weighted SAT. A sex-equal stable matching seeks to minimise the difference in the sum of ranks between men and women. Finding a sex-equal stable matching was shown to be NP-hard [14].

Other notions of fairness involve the profile of a matching which is a vector representing the number of agents assigned to their first, second, third choices etc., in the matching. A rank-maximal stable matching $M$ is a stable matching whose profile is lexicographically maximum, i.e. $M$ maximises the number of agents assigned to their first choice and, subject to that their second choice, and so on. Meanwhile, a generous stable matching $M$ is a stable matching whose reverse profile is lexicographically minimum, i.e. $M$ minimises the number of agents with rank $d$, and subject to that, rank $d-1$, and so on. Profile-based optimality such as rank-maximality or the generous criteria provide guarantees that do not exist with other optimality criteria giving a distinct advantage to these approaches in certain scenarios.

Irving et al. [13] describe the use of weights that are exponential in $n$ in order to find a rank-maximal stable matching using a maximum weight approach. This requires an additional factor of $O(n)$ time complexity to take into account calculations over exponential weights, giving an overall time complexity of $O(n^5 \log n)$ [1]. The choice of max flow algorithm in Irving et al.’s approach is important. Irving et al. [13] stated that the strongly polynomial $O(n^4 \log n)$ Sleator-Tarjan algorithm [23] was the best option (at the time of writing). The Sleator-Tarjan algorithm [23] is an adapted version of Dinic’s algorithm [5] and finds a maximum flow in a network in $O(|V||E|\log |V|)$ time. Since $|V| \leq n^2$ and $|E| \leq n^2$ [11 pg. 112], this translates to $O(n^4 \log n)$ for the maximum weight stable matching problem and an overall time complexity of $O(n^5 \log n)$ for the rank-maximal stable matching problem. However in 2013 Orlin [19] described an improved strongly polynomial max flow algorithm with an $O(|V||E|)$ (translating to $O(n^4)$) time complexity, giving a total overall time complexity for finding a rank-maximal stable matching of $O(n^5)$. Feder’s weighted SAT approach [6] has an overall $O(n^{5.5})$ time complexity for finding a rank-maximal stable matching. Neither Irving et al. [13] nor Feder [6] considered generous stable matchings, however, a generous stable matching may be found in a similar way to a rank-maximal stable matching with the use of weights that are exponential in $n$.

**Motivation.** For the rank-maximal stable matching problem, Irving et al. [13] suggest a weight of $n^{n-1}$ for each agent assigned to their $i$th choice and a similar approach can be taken to find a generous stable matching as we demonstrate later in this paper. In both the rank-maximal and generous cases, the use of exponential weights introduces the possibility of overflow and accuracy errors upon implementation. This may occur as a consequence of limitations of data types: for example, the int and long primitive types restrict the number of integers that can be represented, and the double primitive type may introduce inaccuracies when the number of significant figures is greater than 15. Using a weight of $n^{n-1}$ for each agent assigned to their $i$th choice as above, it may be that we need to distribute $n$ capacities of size $n^{n-1}$ across the network [13]. As a theoretical example the long data type has a maximum possible value of $2^{63} - 1 < 10^{19}$ [17]. Since $16^{15} < 10^{19} < 17^{16}$,

---

1 Irving et al. [13] actually state a time complexity of $O(n^5 \log n \log n)$, however, we believe that this time complexity bound is somewhat pessimistic and that a bound of $O(n^5 \log n)$ applies to this approach.
when we are dealing with flows or capacities of order \( n^{a-1} \), the largest \( n \) possible without risking errors is 16. Alternative data structures such as `BigInteger` do allow an arbitrary limit (currently the implementation limit of Java’s `BigInteger` is \( 2^{31}-1 \) [18]), meaning we are more likely to be dependent on the size of computer memory than this bound.

When looking for a rank-maximal or generous stable matching, we describe an alternative approach to finding a maximum flow that does not require exponential weights. This approach is based on using polynomially-bounded weight vectors that involve profiles of matchings rather than exponentially-large scalars used to represent profiles. On the surface, performing operations over polynomially-bounded weight vectors rather than over equivalent exponential weights, would appear not to improve the time or space complexity of the algorithm, since an exponential number would naturally be stored as an equivalent list of integers in memory. However, even for instances of \( SMR \) with uniformly distributed preference lists, weight vectors of the flow network are not uniformly distributed, allowing us to explore vector compression that is unavailable in the exponential case. Lossless vector compression was performed by saving the index and value of each non-zero vector element. We then calculated the minimum space requirement\(^2\) to store an array of indices and an array of values for this compressed vector. The degree of a vector \( v \) is the position of the final non-zero element in \( v \). We compressed the exponential weight as far as reasonably possible by finding the maximum degree \( d_t \) over all vector-based weights in the instance, before calculating the minimum space requirements to store a number of size \( 2^{d_t} \). The average number of bits required to store the flow network when using the exponential weight representation of weights was compared to the polynomially-bounded vector-based representation of weights.

Figure [14] shows a plot comparing the average number of bits required to store capacities of the flow network using these two approaches for instance size up to \( n = 100,000 \). In this plot, circles represent the average number of bits required for different values of \( n \). The exact space requirements for vector-based weights for the flow network were found experimentally for 1000 randomly generated instances each of size \( n \in \{10, 20, ..., 100, 200, ..., 1000\} \). These instances were also used for experimental work for this paper and more information on their generation can be seen in Section [6]. Additionally, 5 instances were tested each for instances of size \( n \in \{2000, 3000, 4000, 5000\} \). Solid circles represent data points \( n \in \{100, 200, ..., 1000\} \) and these were used to calculate the best fit curves shown when assuming a second order polynomial model. 90% confidence intervals for each representation are also displayed.

We can see clearly that the exponential representation requires more space on average than the compressed vector representation and that this difference increases as \( n \) grows large. Above \( n = 1000 \) we extrapolate up until \( n = 100,000 \), showing the expected trend with an increasing \( n \). Note that the additional small number of data points at \( n \in \{2000, 3000, 4000, 5000\} \) fit this model well. We can see that at \( n = 100,000 \), we expect the exponential approach to be around 100 times more costly in terms of space than the compressed vector-based approach. These differences naturally will have an effect on the time taken to perform operations over these weights in a Max-Flow algorithm. Combining this with the fact that the time complexity of Irving et al.’s [13] \( O(n^5 \log n) \) algorithm to find a rank-maximal matching is dominated substantially by the maximum flow algorithm (no other part taking more than \( O(n^2) \)), it is arguably important to ensure that the flow network fit comfortably in RAM.

**Our contribution.** In this paper we present an \( O(n^5 \log n) \) algorithm to find a rank-maximal stable matching in an instance of \( SMR \) using a vector-based weight approach rather than using exponential weights. We also show that a similar process can be used to find a generous stable matching in \( O(n^2d^3 \log n) \) time, where \( d \) is the degree of the matching. In addition to theoretical contributions we also ran experiments using randomly generated \( SMR \) instances. In these experiments we evaluate differences between egalitarian, sex-equal, median, rank-maximal and generous stable matchings.

\(^2\)Space requirement calculations for both vector-based weights and exponential weights did not assume any particular implementation, but more generally indicated the minimum number of bits required theoretically.

\(^3\)Unlike the instances described in Section [6], calculation of the space requirements for instances of size \( n \in \{2000, 3000, 4000, 5000\} \) were carried out on a machine running Ubuntu version 14.04 with 4 cores, 16GB RAM and Intel\(^\text{®}\) Core\(^\text{TM}\) i7-4790 processors, and compiled using Java version 1.7.0. A far larger timeout was given for these instances at 24 hours for each run of the extended Gale-Shapley Algorithm and Minimal Differences Algorithm – see Section [8] for more information on these algorithms. Each instance was run on a single thread, and one instance of size \( n = 5000 \) timed out over these instance.
Figure 1-1: A log-log plot of the average number of bits required to store a flow network for varying instance sizes $n$ up to $n = 100,000$, comparing exponential and compressed vector-based representation.
based on profile, cost and degree measures. In particular, we find that a generous stable matching typically outperforms a rank-maximal stable matching when considering egalitarian and sex-equal cost measures.

Related work. Work undertaken by Cheng et al. [2] describes the happiness of an agent \(a\) in a stable matching \(M\), defined \(s(a, M)\), as a map from all agents over a given matching to \(\mathbb{R}\). The map \(s(a, M)\) is said to have the independence property if it is only reliant upon information contained in \(M(a)\). The Hospitals/Residents problem (HR) is a more general case of \(\text{SMI}\) in which women may be assigned more than one man. Cheng et al. [2] provide an algorithm for the family of variants of \(\text{HR}\) incorporating happiness functions that exhibit the independence property, to calculate egalitarian and minimum regret stable matchings. For the case that we are given an instance of \(\text{SMI}\), this algorithm has a time complexity of \(O(n^2 f(c) + n^4)\) where \(f(c)\) is the time it takes to calculate the weight of a matching. It is worth noting that the \(n^4\) term of this time complexity is due to Irving et al.’s [13] method of finding a minimum weight stable matching. This method also requires the use of exponential weights which would be problematic for the reasons outlined above.

The House Allocation problem (HA) is an extension of \(\text{SMI}\) in which women do not have preferences over men. The Capacitated House Allocation problem with Ties (CHAT) is an extension of \(\text{HA}\) in which women may be assigned more than one man and men may be indifferent between one or more women on their preference list. With one-sided preferences the notion of stability does not exist. Rank-maximality however, may be described in an analogous way to \(\text{SMI}\) and there is an \(O(\min(n + d, d\sqrt{m}))\) algorithm to find the rank-maximal matching in an instance of \(\text{CHAT}\) [24].

Structure of the paper. Section 2 gives a formal definition of \(\text{SMI}\) and various types of optimal stable matchings. Sections 3 and 4 describe the new approach to find a rank-maximal stable matching and a generous stable matching respectively, without the use of exponential weights. Our experimental evaluation is presented in Section 5 whilst future work is discussed in Section 6.

2 Preliminary results and definitions

2.1 Formal definition of \(\text{SMI}\)

The Stable Marriage problem with Incomplete lists (SMI) comprises a set of men \(U\) and a set of women \(W\). Each man ranks a subset of women in preference order and vice versa. A man \(m_i\), finds a woman \(w_j\) acceptable if \(w_j\) appears on \(m_i\)’s preference list and vice versa. A matching \(M\) in this context is an assignment of men to women such that no man or woman is assigned to more than one person, and if \((m_i, w_j) \in M\), then \(m_i\) finds \(w_j\) acceptable and \(w_j\) finds \(m_i\) acceptable. An example \(\text{SMI}\) instance \(I_0\) with 8 men and women is taken from Gusfield and Irving’s book [11] p. 69 and is given as Figure 2-2. A matching \(M\) is stable if there is no man-woman pair \((m_i, w_j)\) who would rather be assigned to each other than to their assigned partners in \(M\) (if any). By the “Rural Hospitals” Theorem [23][21][4], the same set of men and women are assigned in all stable matchings. We assume that the number of men and women is equal and is denoted \(n\).

It is well known that a stable matching in \(\text{SMI}\) can be found in \(O(m)\) time via the Gale-Shapley algorithm [8], where \(m\) is the total length of all agents preference lists. This algorithm requires either men or women to be the proposers and those of the opposite gender are receivers. However, this procedure naturally produces a proposer-optimal stable matching where members of the proposer group will be assigned to their best possible partner in any stable matching. Unfortunately, this also ensures a receiver-pessimal stable matching in which members of the receiver group will be assigned their worst assigns in any stable matching.

It is natural therefore to want to find some notion of optimality which provides some sense of equality between men and women in a stable matching. This problem has been researched widely and and a summary of the literature is now given.
2.2 Optimality in SMI

Let \( r(m_i, w_j) \) be the rank of woman \( w_j \) on man \( m_i \)’s list with an analogous definition for the rank of man on a woman’s list. Then the egalitarian weight function according to men \( e_m(M) \) is defined as,

\[
e_m(M) = \sum_{(m_i, w_j) \in M} \text{rank}(m_i, w_j).
\]

Similarly, the egalitarian weight function according to women \( e_w(M) \) is defined as,

\[
e_w(M) = \sum_{(m_i, w_j) \in M} \text{rank}(w_j, m_i).
\]

Our combined egalitarian weight function is then,

\[
e(M) = \sum_{(m_i, w_j) \in M} (\text{rank}(m_i, w_j) + \text{rank}(w_j, m_i)).
\]

Let \( I \) be an instance of SMI. One measure of optimality is known as the egalitarian stable matching which optimises the total happiness of all men and women over all stable matchings. An egalitarian stable matching is a stable matching \( M \) such that \( e(M) \) is minimised taken over the set of stable matchings in \( I \). Let \( w(M) \) define some arbitrary weight function of stable matching \( M \). A matching \( M \) is minimum (maximum) weight if \( w(M) \) is minimum (maximum) taken over all stable matchings in \( I \). Thus the minimum weight function \( w(M) \) is a generalisation of the egalitarian weight function \( e(M) \). Irving et al. [19] showed that an egalitarian stable matching can be found in \( O(n^4) \) time and a minimum weight stable matching in \( O(n^4 \log n) \) time. Additionally, Irving et al. [13] described a simple transformation that allows the minimum weight stable matching algorithm to be used to find a maximum weight stable matching in the same time complexity. Feder [6] improved on their method, giving an \( O(n^{3.5}) \) algorithm for finding a minimum weight stable matching.

A sex-equal stable matching in \( I \) is a stable matching \( M \) such that the difference \( e_s(M) = |e_m(M) - e_w(M)| \) is minimum. Kato [14] showed that the problem of finding a sex-equal stable matching is NP-hard.

The degree \( d(M) \) of a matching \( M \) is the highest rank of any assigned pair in \( M \). Formally, \( d(M) \) may be defined as follows:

\[
d(M) = \max_{(m_i, w_j) \in M} \max\{\text{rank}(m_i, w_j), \text{rank}(w_j, m_i)\}
\]

A minimum regret stable matching \( M \) is then a stable matching in \( I \) such that \( d(M) \) is minimised and can be found in \( O(n^2) \) time [10].

A median stable matching may be described in the following way. Let \( M \) denote the set of all stable matchings and \( l_i \) denote the multiset of all women who are assigned to man \( m_i \) in the matchings in \( M \) (in general \( l_i \) is a multiset as \( m_i \) may have the same partner in more than one stable matching). Assume that \( l_i \) is sorted according to \( m_i \)’s preference order (there may be repeated values) and let \( l_{i,j} \) represent the \( j \)th element of this list. Let \( M_f \) denote the set of pairs obtained by assigning \( m_i \) to \( l_{i,j} \) for every \( i (1 \leq i \leq n) \). Teo and Sethuraman [20] showed the surprising result that \( M_f \) is a stable...
Figure 2-3: Rotations for instance $I_0$.

matching for every $j$ such that $1 \leq j \leq |M|$. If $|M|$ is odd then the unique median stable matching is found when $j = \left\lceil \frac{|M|}{2} \right\rceil$. However, if $|M|$ is even, then the set of median stable matchings are the stable matchings such that each man (woman) does no better (worse) than their partner when $j = \left\lfloor \frac{|M|}{2} \right\rfloor$ and no worse (better) than their partner found when $j = \left\lfloor \frac{|M|}{2} \right\rfloor + 1$. For the purposes of this paper, in particular the experimentation section, we define the median stable matching as the stable matching found when $j = \left\lceil \frac{|M|}{2} \right\rceil$.

Define a rank-maximal matching $M$ in $\text{SMI}$ to be a matching in which the largest number of agents gain their first choice, then subject to that, their second choice and so on. More formally we define a profile as a finite vector of integers (positive or negative) and the profile of a matching as follows. Given a matching $M$, let the profile of $M$ be given by the vector $p(M) = (p_1, p_2, ..., p_n)$ where $p_k = |\{(m_i, w_i) \in M : \text{rank}(m_i, w_i) = k\}| + |\{(m_i, w_i) \in M : \text{rank}(w_i, m_i) = k\}|$ for some $k : (1 \leq k \leq n)$. Thus we define a stable matching $M$ in an instance $I$ of $\text{SMI}$ to be rank-maximal if $p(M)$ is lexicographically maximum, taken over all stable matchings in $I$. We define the reverse profile $p_r(M)$ to be the vector $p_r(M) = (p_k, p_{k-1}, ..., p_1)$. A stable matching $M$ in an instance $I$ of $\text{SMI}$ is generous if $p_r(M)$ is lexicographically minimum, taken over all stable matchings in $I$.

2.3 Finding a rank-maximal stable matching using exponential weights

In this section we will describe how Irving et al.’s [13] maximum weight stable matching algorithm works and how it can be used to find a rank-maximal stable matching using exponential weights.

Graphical structures. Irving et al. [13] define a rotation $\rho = (m_1, w_1), (m_2, w_2), ..., (m_n, w_n)$ as a list of man-woman pairs in a stable matching $M$, such that when their assignments are permuted (each man $m_i$ moving from $w_i$ to $w_{i+1}$, where $i$ is incremented modulo $k$), we obtain another stable matching. A rotation $\rho$ is exposed in $M$ if all of the pairs in $\rho$ are in $M$. Permuting the assignments of an exposed rotation $\rho$ is known as eliminating $\rho$. A list of rotations of instance $I_0$ is given in Figure 2-3.

In order to describe profiles of rotations we must first describe arithmetic over profiles. Addition over profiles may be defined in the following way. Let $p = (p_1, p_2, ..., p_n)$ and $p' = (p'_1, p'_2, ..., p'_n)$ be profiles of length $n$. Then the addition of $p$ to $p'$ is taken pointwise over elements from $1...n$. That is, $p + p' = (p_1 + p'_1, p_2 + p'_2, ..., p_n + p'_n)$. We define $p = p'$ if $p_i = p'_i$ for $1 \leq i \leq n$. Now suppose $p \neq p'$. Let $k$ be the first point at which these profiles differ, that is, suppose $p_k \neq p'_k$ and $p_i = p'_i$ for $1 \leq i < k$. Then we define $p < p'$ if $p_k < p'_k$. Finally, we say $p \leq p'$ if either $p_k < p'_k$ or $p = p'$. It is trivial to show that an addition or comparison of two profiles would take $O(n)$ time in the worst case (since the length of any profile is bounded by $n$). Let $p'' = (p_1, p_2, ..., p_i, 0, ..., 0)$ be a profile, where $i \leq n$. Then for ease of description we may shorten this profile to $p'' = (p_1, p_2, ..., p_i)$.

Suppose we have a rotation $\rho$ that, when eliminated, takes us from stable matching $M$ to stable matching $M'$, where $M$ and $M'$ have profiles $p(M) = (p_1, p_2, ..., p_n)$ and $p(M') = (p'_1, p'_2, ..., p'_n)$ respectively. Then the profile of $\rho$ is defined as the net change in profile between $M$ and $M'$, that is, $p(\rho) = (p'_1 - p_1, p'_2 - p_2, ..., p'_n - p_n)$. Hence, $p(M') = p(M) + p(\rho)$. It is easy to see that a particular rotation will give the same net change in profile regardless of which stable matching it is eliminated from. For a set of rotations $R = \{p_1, p_2, ..., p_k\}$, we define the profile over $R$ as $p(R) = p(p_1) + p(p_2) + ... + p(p_k)$.

A rotation poset may be constructed as a directed graph which indicates the order in which rotations may be eliminated. Informally, if one rotation $\rho$ precedes another, $\tau$, in the rotation poset then $\tau$ is not exposed until $\rho$ has been eliminated. A closed subset of the rotation poset may be defined as a set of rotations $P = \{p_1, p_2, ..., p_k\}$ such that for every $p_i$ in $P$, all of $p_i$’s predecessors
are also in \( P \). It has been shown that there is a 1-1 correspondence between the closed subsets of the rotation poset and the set of all stable matchings [13, Theorem 3.1]. The rotation poset for \( I_0 \), denoted \( R_\rho(I_0) \), is shown in Figure 2-4a.

[Irving et al.’s][13] method for finding a maximum weight stable matching involves finding a maximum weight closed subset of the rotation poset. In order to find this maximum weight closed subset, other graphical structures need to be defined. A description of the creation of a rotation digraph now follows. First, retain each rotation from the rotation poset as a node. There are two types of predecessor relationships to consider.

1. Suppose pair \((m_i, w_j) \in \rho\). We have a directed edge in our digraph from \( \rho' \) to \( \rho \) if \( \rho' \) is the unique rotation that moves \( m_i \) to \( w_j \). In this case we say that \( \rho' \) is a type 1 predecessor of \( \rho \).
2. Let \( \rho \) be the rotation that moves \( m_i \) below \( w_j \) and \( \rho' \neq \rho \) be the rotation that moves \( w_j \) above \( m_i \). Then we add a directed edge from \( \rho' \) to \( \rho \) and say \( \rho' \) is a type 2 predecessor of \( \rho \).

The rotation digraph for instance \( I_0 \), denoted \( R_d(I_0) \), is shown in Figure 2-4b.

Using the rotation digraph structure, Gusfield and Irving [11] were able to enumerate all stable matchings in \( O(n^2 + n |\mathcal{M}|) \), where \( \mathcal{M} \) is the set of all stable matchings. All stable matchings of instance \( I_0 \) are listed in Figure 2-5.

We must now convert the rotation digraph to a flow network \( R_a(I) \). First we add two extra nodes; a source node \( s \) and a sink node \( t \). An edge of capacity \( \infty \) replaces each original edge in the digraph. Since we are finding a rank-maximal stable matching, capacities on other edges of \( R_a(I) \) are calculated by converting each profile of a rotation to a single exponential weight. We decide on a weight function of \((2n + 1)^{n-i}\) for each person assigned to their ith choice. From this point onwards we refer to the use of this weight function as the high-weight scenario, and denote it as \( w \).

**Definition 2.1.** Given a profile \( p = (p_1, p_2, ..., p_n) \) such that \( |p_1| + |p_2| + ... + |p_n| \leq 2n \) and \( 1 \leq a \leq n \), define the high-weight function \( w \) as,

\[
w(p) = p_1(2n + 1)^{n-1} + p_2(2n + 1)^{n-2} + ... + p_a(2n + 1)^{n-a}.
\]
Lemma 2.3 shows that when the above function $w$ is used, a matching of maximum weight will be a rank-maximal matching.

**Proposition 2.2.** Let $p = (p_1, p_2, ..., p_n)$ and $p' = (p'_1, p'_2, ..., p'_n)$ be profiles such that $|p_1| + |p_2| + ... + |p_n| \leq 2n$ and $|p_1| + |p_2| + ... + |p_n| \leq 2n$. Let $w_i(p) = p_i(2n+1)^{n-1}$ denote the $i$th term of $w(p)$ and let $w_i^+(p) = \sum_{j=i}^{n-1} p_j(2n+1)^{n-j}$ denote the sum of $w(p')$ terms for all $j$ such that $i \leq j \leq n$. If $p_i > p'_i$, then $w_i(p) > w_i^+(p')$. Additionally, if $i$ is the first point at which $p$ and $p'$ differ, then $w(p) > w(p')$.

**Proof.** Assume $p_i > p'_i$. Then $p_i$ must be at least 1 larger than $p'_i$ since each profile element is an integer by definition. A value of 1 for $p_i$ will contribute $(2n+1)^{n-1}$ to $w_i(p)$ and so it follows that $w_i(p) \geq w_i(p') + (2n+1)^{n-i}$.

Since $(2n+1)^{n-k}$ decreases as $k$ increases and $|p'_1| + |p'_2| + ... + |p'_n| \leq 2n$, the maximum weight contribution that $p_{i+1}', p_{i+2}', ..., p'_n$ can make to $w_i^+(p')$ is when $p'_{i+1} = 2n$.

Through the following series of inequalities,

$$w_i^+(p') \leq w_i(p') + 2n(2n+1)^{n-(i+1)}$$

$$\leq w_i(p) - (2n+1)^{n-i} + \frac{2n}{2n+1} (2n+1)^{n-i}$$

$$\leq w_i(p) + \left(\frac{2n}{2n+1} - 1\right) (2n+1)^{n-i}$$

$$< w_i(p)$$

it follows that $w_i(p) > w_i^+(p')$ as required. If $i$ is the first point at which $p$ and $p'$ differ then it follows that $w(p) > w(p')$.

**Lemma 2.3.** Let $I$ be an instance of [SM] and let $M$ be a stable matching in $I$. If $w(p(M))$ is maximum amongst all stable matchings of $I$, where $p(M)$ is the profile of $M$, then $M$ is a rank-maximal stable matching.

**Proof.** Suppose $w(p(M))$ is maximum amongst all stable matchings of $I$. Now, assume for contradiction that $M$ is not rank-maximal. Then, there exists some stable matching $M'$ in $I$ such that $M'$ lexicographically larger than $M$. Let $i$ be the first point at which $p(M) = (p_1, p_2, ..., p_n)$ and $p(M') = (p'_1, p'_2, ..., p'_n)$ differ. Since $M'$ is lexicographically larger than $M$ we know that $p'_i > p_i$ and by Proposition 2.2 it follows that $w(p(M')) > w(p(M))$.

But this contradicts the fact that $w(p(M))$ is maximum over all stable matchings of $I$. Therefore our assumption that $M$ is not rank-maximal is false, as required.

We now continue describing [Irving et al.]'s technique for finding a maximum closed subset of the rotation poset. The rotations are divided into positive and negative nodes as follows. A rotation $\rho$ is positive if $w(\rho) > 0$ and negative if $w(\rho) < 0$. A directed edge is added from the source to each negative node and is given a capacity equal to $|w(\rho)|$. A directed edge is also added between each positive node and $t$ with capacity $w(\rho)$. The high-weight flow network of instance $I_0$ is denoted $R_n(I_0)$ and is shown in Figure 2.4.

**Minimum cut of $R_n(I_0)$.** In the flow network, we denote the flow over a node or edge as $f$ and an $s$-$t$ cut as $c_{st}$ with capacity given by $c(c_{st})$.

By the Max Flow-Min Cut Theorem [17] we need only find a maximum flow through $R_n(I_0)$ in order to find a minimum cut in $R_n(I)$. Irving et al. [13] used the Sleator-Tarjan algorithm [23] to find a maximum flow. Several analogous definitions used in the Sleator-Tarjan algorithm are required when we move to our new approach and so are described below.

In order to search for augmenting paths we construct a new network known as the residual graph. Given a flow network $R_n(I)$ and a flow $f$ in $R_n(I)$, the residual graph relative to $R_n(I)$ and $f$, denoted $R_{res}(I, f)$, is defined as follows. The vertex set of $R_{res}(I, f)$ is equal to the vertex set of $R_n(I)$. An edge $(u, v)$, known as a forward edge, is added to $R_{res}(I, f)$ with capacity $c(u, v) - f(u, v)$ if $(u, v) \in E$ and $f(u, v) < c(u, v)$. Similarly an edge $(u, v)$, known as a backwards edge, is added to $R_{res}(I, f)$ with capacity $f(v, u)$ if $(v, u) \in E$ and $f(v, u) > 0$. Using a breadth-first search in $R_{res}(I, f)$ we may find an augmenting path or determine that none exists in $O(|E|)$ time. Once an augmenting path $P$ is found we augment $R_n(I)$ in the following way:
Figure 2-6: The high-weight flow network $R_n(I_0)$.

- The residual capacity $c_a$ is the minimum of the capacities of the edges in $P$ in $R_{res}(I, f)$;
- For each edge $(u, v) \in P$, if $(u, v)$ is a forwards edge, the flow through $(u, v)$ is increased by $c_a$, whilst if $(u, v)$ is a backwards edge, the flow through $(v, u)$ is decreased by $c_a$.

Ford and Fulkerson [7] showed that if no augmenting path in $R_n(I_0)$ can be found then the flow $f$ in $R_n(I_0)$ is maximum. In Figure 2-6 we show the high-weight flow network $R_n(I_0)$ with a maximum flow highlighted. There is one minimum cut, $c_T = \{ (s, \rho_0), (\rho_4, \rho_t) \}$. Note that this must be a minimum cut since the flow over edge $(s, \rho_0)$ is equal to its capacity, and the flow over $(s, \rho_3)$ is limited entirely by the capacity of $(\rho_4, t)$. For this cut $c_T$ we list every rotation $\rho$ such that $(\rho_t, \rho) \in c_T$.

Then a maximum weight closed subset of the rotation poset is given by this set of rotations and their predecessors. $c_T$ has associated closed subset of $\{ \rho_0, \rho_1, \rho_2 \}$ which is precisely the maximum weight closed subset of $R_p(I_0)$. The man-optimal stable matching of $I_0$ is

$$M = \{ (m_1, w_5), (m_2, w_3), (m_3, w_8), (m_4, w_6), (m_5, w_7), (m_6, w_1), (m_7, w_2), (m_8, w_4) \}.$$  

By eliminating rotations $\{ \rho_0, \rho_1, \rho_2 \}$ from the man-optimal stable matching, we find the rank-maximal stable matching

$$M' = \{ (m_1, w_3), (m_2, w_9), (m_3, w_1), (m_4, w_8), (m_5, w_7), (m_6, w_5), (m_7, w_2), (m_8, w_4) \}.$$  

The following Theorem summarises the work in this section.

**Theorem 2.4.** Let $I$ be an instance of $\text{SMM}$. A rank-maximal stable matching $M$ of $I$ can be found in $O(n^5 \log n)$ using weights that are exponential in $n$ [13].

An alternative to high-weight values when looking for a rank-maximal stable matching, is to use a new approach, involving polynomially-bounded weight vectors, to find a maximum weight closed subset of rotations. This is the focus of the rest of this paper.

3 Finding a rank-maximal stable matching using polynomially-bounded weight vectors

3.1 Strategy

Following a similar strategy to Irving et al. [13], we aim to show that we can return a rank-maximal stable matching in $O(n^5 \log n)$ time without the use of exponential weights. The process we to follow is described below.

1. Calculate man-optimal and woman-optimal stable matchings using the Extended Gale-Shapley Algorithm – $O(n^3)$ time;
2. Find all rotations using the minimal differences algorithm – $O(n^2)$ time;
3. Build the rotation digraph and flow network – $O(n^2)$ time;
4. Find a minimum cut of the flow network in $O(n^5 \log n)$ time without reverting to high weights;
5. Use this cut to find a maximum profile closed subset $S$ of the rotations \(-O(n^2)\) time;

6. Eliminate the rotations of $S$ from the man-optimal matching to find the rank-maximal stable matching.

In the next section we discuss required adaptions to the high-weight procedure.

### 3.2 Vb-networks and vb-flows

In this section we look at steps in the strategy to find a rank-maximal stable matching without the use of exponential weights (Section 3.1) which either require adaptions or further explanation.

In Step 5 of our strategy we eliminate the rotations of a maximum profile closed subset of the rotation poset from the man-optimal stable matching. We now present Lemma 3.1, an analogue of Corollary 3.6.1 of [11], which shows that eliminating a maximum profile closed subset of the rotation poset from the man-optimal stable matching results in a rank-maximal stable matching.

**Lemma 3.1.** Let $I$ be an instance of $\text{SMI}$ and let $M_0$ be the man-optimal stable matching in $I$. A rank-maximal stable matching $M$ may be obtained by eliminating a maximum profile closed subset $S$ of the rotation poset from $M_0$.

**Proof.** Let $R_p(I)$ be the rotation poset of $I$. By Gusfield and Irving [11, Theorem 2.5.7], there is a 1-1 correspondence between closed subsets of $R_p(I)$ and the stable matchings of $I$. Let $S$ be a maximum profile closed subset of the rotation poset $R_p(I)$ and let $M$ be the unique corresponding stable matching. Then, $p(M) = p(M_0) + \sum_{\rho \in S} p(\rho)$. Suppose $M$ is not rank-maximal. Then there is a stable matching $M'$ such that $p(M') > p(M)$. As above, $M'$ corresponds to a unique closed subset $S'$ of the rotation poset. Also $p(M') = p(M_0) + \sum_{\rho \in S'} p(\rho)$. But $p(M') > p(M)$ and so $S$ cannot be a maximum profile closed subset of $R_p(I)$, a contradiction. $\blacksquare$

Steps 3 and 4 of our strategy are the only places where we are required to check that it is possible to directly substitute an operation involving large weights taking $O(n)$ time with a comparable profile operation taking $O(n)$ time.

The first deviation from Gusfield and Irving's method (described in Section 1) is in the creation of a vector-based flow network (abbreviated to vb-flow network). For ease of description we denote this new vb-flow network as $R'_p(I)$ to distinguish it from the high-weight version $R_p(I)$. We now define a vb-capacity in $R'_p(I)$ which is of similar notation to that of a profile.

**Definition 3.2.** In a vb-flow network $R'_p(I)$, the vector-based capacity (vb-capacity) of an edge $e$ is a vector $c(e) = (c_1, c_2, ..., c_n)$, where $n$ is the number of men or women in $I$ and $c_i \geq 0$ for $1 \leq i \leq n$.

As before we add a source $s$ and sink $t$ node to the rotation digraph. We replace each original digraph edge with an edge with vb-capacity $\langle \infty, \infty, ..., \infty \rangle$ ($\infty$ repeated $n$ times). For convenience these edges are marked with ‘\^\infty’ in the network flow diagram. The definition of a positive and negative rotation is also amended. Let $\rho$ have profile $p(\rho) = (p_1, p_2, ..., p_n)$. Let $p_k$ be the first non-zero profile element where $1 \leq k \leq n$. We now define a positive rotation $\rho$ as a rotation where $p_k > 0$, and a negative rotation is one where $p_k < 0$. Define the absolute value operation, denoted $|p(\rho)|$, as follows. If $p_k > 0$, then leave all elements unchanged. If $p_k < 0$, then reverse the sign of all non-zero profile elements. Figure 3-7 shows the profile and absolute profile for each rotation of $I_0$. Then we add a directed edge to the vb-flow network from $s$ to each negative rotation node $\rho$ to $t$ with a vb-capacity of $|p(\rho)|$ and a directed edge from each positive rotation node $\rho$ to $t$ with a vb-capacity of $p(\rho)$.

**Definition 3.3.** In a vb-flow network $R'_p(I) = (V, E)$, a vector-based flow (vb-flow) is a function $f : E \rightarrow \mathbb{R}^n$ such that

i) (vb-capacity) $f(e) \geq 0$ and $f(e) \leq c(e)$ for all $e \in E$;

ii) (vb-conservation) $\sum_{(u,v) \in E} f(u,v) = \sum_{(v,w) \in E} f(v,w)$ for all $v \in V \setminus \{s,t\}$.

Sleator and Tarjan’s algorithm [23] (used later in this section) may be used in a way that assumes integer value flows, and hence for the rest of this paper we assume vb-flow value elements will only ever be integers. That is, a vb-flow is a function $f : E \rightarrow \mathbb{N}^n$. 

---

\[\text{Sleator and Tarjan’s algorithm} \quad [23]\]

---

11
positive vb-flow. For example, $n$ in $\pm e$ element of $\{1\}$. As an example, the vb-value $\langle e \rangle$ vector-based residual network. First, we show that the Max Flow-Min Cut Theorem can be extended to a vb-flow network. We now show how we are able to use our vb-flow network to find a rank-maximal stable matching.

### 3.3 Rank-maximal stable matchings

Vb-flows are non-negative by the vb-capacity constraint. Let $f(e) = \langle f_1, f_2, ..., f_n \rangle$ be a vb-flow over edge $e$ in $R_v(I)$. Note that it is possible for $f_i < 0$ for some $i : 2 \leq i \leq n$ and for $f$ to be a positive vb-flow. For example $(0,0,0) < (0,1,-10)$, and hence we are not bounding each individual element of a vb-flow through an edge by a minimum of zero. However, we give bounds for each element of $\pm 2n$ (the total number of agents in an instance) and the modulus of flow value elements must sum to a maximum of $2n$. This is done in order for proofs in the next section describing the equivalence of vector-based and high-weight approaches to work.

In addition we define the following notation and terminology for vb-flows. Let $I$ be a vb-flow in a vb-flow network, where $c$ capacities is created in the same way as the residual network of $I$. In Lemma 3.5 we show that vb-flows in $R_v(I)$ hold for vb-flow networks.

Identical results hold for vb-capacities.

In addition we define the following notation and terminology for vb-flows. Let $I$ be a vb-flow in a vb-flow network, where $c$ capacities is created in the same way as the residual network of $I$. In Lemma 3.5 we show that vb-flows in $R_v(I)$ hold for vb-flow networks.

Identical results hold for vb-capacities.

### 3.3 Rank-maximal stable matchings

We now show how we are able to use our vb-flow network to find a rank-maximal stable matching. First, we show that the Max Flow-Min Cut Theorem can be extended to a vb-flow network.

In Lemma 3.5 we show that vb-flows in $R_v(I)$ correspond to high-weight flows in $R_v(I)$. Let $f$ be a vb-flow in a vb-flow network, where $val(f) = \{f_1, f_2, ..., f_n\}$ and let $c'_f$ be a cut where $c(\rho) = \{c'_f, c'_1, c'_2, ..., c'_n\}$.

**Proposition 3.4.** Let $f$ and $f'$ be vb-flows. Let $w_i(val(f))$ denote the $i$th term of $w(p)$ and let $w_i'(\langle val(f') \rangle)$ denote the sum of $w(val(f'))$ terms for all $j$ such that $i \leq j \leq n$. If $f_i > f'_i$, then $w_i(val(f)) > w_i'(val(f'))$. Additionally, if $i$ is the first point at which $f$ and $f'$ differ, then $w_i(val(f)) > w_i'(val(f'))$.

Identical results hold for vb-capacities.
Lemma 3.5. Let $f$ and $f'$ be vb-flows in $R_n'(I_0)$. Let the total high-weight values of $f$ and $f'$, according to Definition 2.1, be denoted $w(val(f))$ and $w(val(f'))$ respectively. Then $val(f) < val(f')$ if and only if $w(val(f)) < w(val(f'))$.

Similar to above, let $c_T$ and $c'_T$ be cuts in $R_n'(I_0)$ and let $w(e(c_T))$ and $w(e(c'_T))$ denote the high-weight capacities of $c_T$ and $c'_T$ respectively. Then $c(c_T) < c(c'_T)$ if and only if $w(e(c_T)) < w(e(c'_T))$.

Proof. Suppose that $val(f) < val(f')$. We know $val(f) \neq val(f')$, and at the first point $i$ at which $val(f)$ and $val(f')$ differ $f_i < f'_i$. By Proposition 3.4, $w(val(f)) < w(val(f'))$ as required.

Now assume $w(val(f)) < w(val(f'))$ and suppose for contradiction that $val(f) \geq val(f')$. If $val(f) = val(f')$ then clearly $w(val(f)) = w(val(f'))$ a contradiction. Therefore suppose $val(f) > val(f')$. Then, we can use identical arguments to the preceding paragraph to prove that $w(val(f)) > w(val(f'))$. But this contradicts our original assumption that $w(val(f)) < w(val(f'))$. Therefore, $val(f) < val(f')$. 

Proof. Recall from the footnote of Definition 3.3 that vb-flow values must contain only integer elements.

The only difference between the structure of a profile $p$ and a vb-flow $f$ is that each profile element must take a value between 0 and $2n$ inclusive, whereas the lower bound of vb-flow elements is relaxed to $-2n$. This difference does not affect the validity of Proposition 2.2 and so we may use identical reasoning to show that if $f_i > f'_i$, then $w_i(val(f)) > w_i(val(f'))$, and if $i$ is the first point at which $f$ and $f'$ differ, then $w(val(f)) > w(val(f'))$.

Since vb-capacities have an identical structure to vb-flows the all results also hold for the vb-capacity case.

Figure 3-8: Vector-based flow network $R_n'(I_0)$ and flow network $R_n(I_0)$ with both vector-based and high-weight capacities respectively.
Using identical reasoning to the vb-flow case for the vb-capacity case, we can show that $e(c_T') < e(c_T')$ if and only if $w(e(c_T')) < w(e(c_T'))$.

 Lemma 3.6 shows that if there is no augmenting path in a vb-flow network, then the vb-flow existing in this network is maximum.

**Lemma 3.6.** Let $R_n(I)$ define the vb-flow networks of $I$, and let $R_n(I)$ define the vb-flow and flow networks of $I$ respectively. For all vb-flows and vb-capacities we define a corresponding flow or capacity for $R_n(I)$ using the high-weight function $w$. Suppose $f$ is a vb-flow in $R_n(I)$ that admits no augmenting path. Then $f$ is a maximum vb-flow in $R_n(I)$.

**Proof.** Let $f$ be the flow corresponding to $f$ in $R_n(I)$. First, we show that $f$ is a maximum flow in $R_n(I)$. Suppose for contradiction that $f$ is not a maximum flow. Then there must exist an augmenting path relative to $f$ in $R_n(I)$. Let $E$ denote the edges involved in this augmenting path. Then, for each edge $(u, v) \in E$, we have

- $f(u, v) > 0$, in which case the vb-flow $f$ through edge $(u, v) \in R_n(I)$ may increase by $(0, 0, ..., 1)$, or;
- $f(v, u) > 0$, and so the vb-flow $f$ through edge $(v, u) \in R_n(I)$ may decrease by $(0, 0, ..., 1)$.

Therefore, there exists an augmenting path relative to $f$ in $R_n(I)$. But this contradicts the fact that $f$ is a vb-flow in $R_n(I)$ that admits no augmenting path. Hence our assumption that $f$ is not a maximum flow in $R_n(I)$ is false.

We now show that $f$ is a maximum vb-flow in $R_n(I)$. Suppose for contradiction that this is not the case. Then there must exist a vb-flow $f'$ such that $val(f') > val(f)$. By Lemma 3.6, $w(val(f')) > w(val(f))$. Let $f'$ be the flow corresponding to $f'$ in $R_n(I)$. Then we have the following inequality:

$$val(f') = w(val(f')) > w(val(f)) = val(f)$$

contradicting the fact that $f$ is a maximum flow in $R_n(I)$. Therefore $f$ is a maximum vb-flow in $R_n(I)$.

This means if we use any max-flow algorithm that terminates with no augmenting paths (such as the Ford-Fulkerson Algorithm adapted to work with vb-flows and vb-capacities) we have found a maximum flow in a vb-flow network.

We now show that the Max Flow-Min Cut Theorem can be extended to a vb-flow network.

**Theorem 3.7.** Let $I$ be an instance of $\text{SM}$ and let $R_n(I)$ and $R_n(I)$ define the vb-flow and flow networks of $I$ respectively. For all vb-flows and vb-capacities we define a corresponding flow or capacity for $R_n(I)$ using the high-weight function $w$. Suppose $f$ is a vb-flow in $R_n(I)$ such that $e(c_T') = val(f)$.

**Proof.** Given $f$ is a maximum vb-flow in $R_n(I)$, we define a cut $c_T'$ in $R_n(I)$ in the following way. A partial augmenting path is an augmenting path from the source vertex $s$ to vertex $u \neq t$ in $V$ with respect to $f$ in $R_n(I)$. Note that, by Lemma 3.6, no augmenting path from $s$ to $t$ may exist at this point since $f$ is a maximum vb-flow. Let $A$ be the set of reachable vertices along partial augmenting paths, and let $B = V \setminus A$. Then $s \in A$ and $t \in B$. Define $c_T' = \{(u, v) : u \in A, v \in B\}$. Then $c_T'$ is a cut in $R_n(I)$. Since there is no partial augmenting path extending between vertices in $A$ and vertices in $B$, we know that for vertices $u \in A$ and $v \in B$,

- if $(u, v) \in E'$ then $f(u, v) = c(u, v)$, and;
- if $(v, u) \in E'$ then $f(v, u) = 0$.

Therefore $val(f) = e(c_T')$ (see, for example, Cormen et al. [4] pg. 721, Lemma 26.4] for a proof of this statement).

Let $f$ be the flow corresponding to $f$ in $R_n(I)$. We now show that $c_T'$ is a minimum cut in $R_n(I)$. Suppose for contradiction that this is not the case. Then there must exist a cut $c_T''$ in $R_n(I)$ such that $e(c_T'') < e(c_T')$. By Lemma 3.5, $w(e(c_T'')) < w(e(c_T'))$ and so we have the following inequality.

$$e(c_T') = w(e(c_T')) < w(e(c_T')) = val(f)$$

But then $c_T'$ is a cut with smaller capacity than $val(f)$ in $R_n(I)$ contradicting the Max Flow-Min Cut Theorem in $R_n(I)$. Hence $c_T'$ is a minimum cut in $R_n(I)$.  \qed
Figure 3-9: Maximum vb-flow and flow in the flow networks $R^*_n(I_0)$ and $R_n(I_0)$ with both vector-based and high-weight capacities respectively. (Vb-)flows through edges are highlighted in grey.

As an example Figure 3-9a shows the maximum flow over the vb-flow network $R^*_n(I_0)$, and Figure 3-9b shows these vb-flows translated into the high-weight flow network $R_n(I_0)$. In Figure 3-9a each edge flow is positive, that is, the first non-zero element of each vb-flow is positive as required. The maximum vb-flow shown in this figure has saturated both edge $(s, \rho_0)$ leaving the source $s$ and edge $(\rho_4, t)$ entering the sink $t$. It is easy to see that there are no vertices reachable from $s$ in the residual network $R^*_{r_{ve}}(I_0, f)$. This means that edges $\{(s, \rho_0), (\rho_4, t)\}$ comprise the minimum cut of $R^*_n(I_0)$ with a summed vb-capacity of $(-3, -3, -3, 1, 0, 2)$. The equivalent situation is shown in Figure 3-9b.

In order to determine which rotations must be eliminated from the man-optimal stable matching, we must first determine a maximum profile closed subset of the rotation poset. As with the example in Section 2.3, we must find the positive nodes which have edges into $t$ that are not in the minimum cut. These are $\rho_1$ and $\rho_2$. In Theorem 3.8 we will prove that a maximum profile closed subset of the vb-flow network comprises these nodes and their predecessors: $\{\rho_0, \rho_1, \rho_2\}$. It was shown in Section 2.3 that this was indeed the maximum closed subset of $R^*_n(I_0)$.

The following theorem is a restatement of Gusfield and Irving’s theorem [11, pg. 130] proving that a maximum profile closed subset of the rotation poset $R_p(I)$ can be found by finding a minimum $s$-$t$ cut of the vb-flow network $R^*_n(I)$, when using a vector-based weight function.

**Theorem 3.8.** Let $I$ be an instance of $\text{BMI}$ and let $R_p(I)$, $R_n(I)$ and $R^*_n(I)$ denote the rotation poset, rotation digraph and vb-flow network of $I$ respectively. Let $c_T$ be the minimum $s$-$t$ cut in $R^*_n(I)$, and let $P_{c_T}$ be the positive nodes of the network whose edges into $t$ are not in $c_T$. Then the nodes $P_{c_T}$...
and their predecessors define a maximum profile closed subset of $R_p(I)$. Further $\mathcal{P}_c^I$ is exactly the set of positive nodes of this closed subset of the rotation poset.

**Proof.** Let $S$ be an arbitrary set of rotations in $I$ and define $w(S) = \sum_{s_i \in S} p(s_i)$, that is, $w(S)$ is the total vector-based weight of these rotations. Let $\mathcal{P}$ be the set of all positive rotations. For any set of rotations $S \subseteq \mathcal{P}$, let $N(S)$ be the set of all negative rotation predecessors of $S$ in the rotation digraph $R_d(I)$. Let $Q$ denote a maximum profile closed subset of $R_d(I)$. In order for a negative rotation node to exist in $Q$ it must precede at least one positive rotation node, otherwise $Q$ could not be of maximum weight. Hence $Q$ can be found by maximising $w(S) + w(N(S))$ over all subsets of $S \subseteq \mathcal{P}$.

We now show that $w(S) + w(N(S))$ is equivalent to $w(S) - |w(N(S))|$ according to our vector-based weight function. We know that $w(N(S))$ is negative (i.e. the first non-zero element of $w(N(S))$ is negative) and therefore taking the absolute value of $w(N(S))$ will reverse the signs of all non-zero elements. Taking the negative of $|w(N(S))|$ reverses the element signs once more and so we have $w(S) + |w(N(S))| = w(S) - |w(N(S))| = w(S) - w(N(S))$.

Therefore we may say that $Q$ can be found by maximising $w(S) - |w(N(S))|$ over all subsets of $S \subseteq \mathcal{P}$. But by maximising this function, we also minimise $w(P) - (w(S) - |w(N(S))|) = w(P\backslash S) + |w(N(S))|$. That is, we are minimising the total weight of the positive rotations not in $S$ added to the absolute value of the negative rotations that are $S$’s predecessors. This becomes clearer when looking at the vb-flow network $R_v^I$.

Let $c(e^*_P)$ denote the capacity of the minimum cut $e^*_P$. We want to show that $c(e^*_P)$ is at least as small as $w(P\backslash S) + |w(N(S))|$ for any $S \subseteq \mathcal{P}$. We can find an upper bound for $c(e^*_P)$ by doing the following. If we have a set of edges $e^*_P$ that comprises (1) all edges from $s$ to nodes in $N(S)$, and (2) all edges from nodes in $P\backslash S$ to $t$, then $e^*_P$ is certainly a cut since there can be no flow through $R_v^I$. Moreover $c(e^*_P) = w(P\backslash S) + |w(N(S))|$ and therefore $c(e^*_P) \leq w(P\backslash S) + |w(N(S))|$ for any $S \subseteq \mathcal{P}$. Now let $S^* \subseteq P$ be the set of positive rotation nodes that have edges into $t$ that are not in $e^*_P$. Then $e^*_P$ must contain all edges from $P\backslash S^*$ to $t$. Since $e^*_P$ has finite capacity all the edges within it must also have finite capacity and consequently $e^*_P$ must also contain all edges in $N(S^*)$. Therefore,

$$c(e^*_P) = w(P\backslash S^*) + |w(N(S^*))| \leq w(P\backslash S) + |w(N(S))|$$

for all $S \subseteq \mathcal{P}$. Hence, $\mathcal{P}_c^I = S^*$ and, $\mathcal{P}_c^I$ and their predecessors define a maximum profile closed subset of the rotation poset $R_p(I)$.

It remains to show that we can adapt Sleator and Tarjan’s O(n^4 log n) Max Flow algorithm to work with vb-flow networks. This is shown in Lemma 3.9.

**Lemma 3.9.** Let $I$ be an instance of [SBM] and let $R_v^I$ be a vb-flow network. We can use a version of Sleator and Tarjan’s Max Flow algorithm [23] adapted to work with vb-flow networks in order to find a maximum flow $f$ of $R_v^I$ in $O(n^3 \log n)$.

**Proof.** A blocking flow in the high-weight setting is a flow such that each path through the flow network from $s$ to $t$ has a saturated edge. Note that this is different from a maximum flow, since a blocking flow may still allow extra flow to be pushed from $s$ to $t$ using backwards edges in the residual graph. The Sleator-Tarjan algorithm [23] is an adapted version of Dinic’s algorithm [5] which improves the time complexity of finding a blocking flow. This is achieved by the introduction of a new dynamic tree structure.

The following operations are required from Sleator and Tarjan’s dynamic tree structure in the max flow setting [23]: link, capacity, cut, mincost, parent, update and cost. Each of these processes (not described here) consists of straightforward graph operations (such as adding a parent node, deleting an edge etc.) and comparisons, additions, subtractions and updating of edge capacities and flows.

Since we have a vector-based interpretation of comparison, addition and subtraction operations, it is possible to adapt Sleator and Tarjan’s Max Flow algorithm to work in the vector-based setting. Sleator and Tarjan’s algorithm [23] terminates with a flow that admits no augmenting path. Let $f$ be a vb-flow given at the termination of Sleator and Tarjan’s algorithm, as applied to $R_v^I$. Since $f$ admits no augmenting path, it follows, by Lemma 3.6 that $f$ is a maximum vb-flow in $R_v^I$. Sleator and Tarjan’s algorithm runs in $O(n^3 \log n)$ time assuming constant time operations for comparison, addition and subtraction. However, in the vb-flow setting, each of these operations takes $O(n)$ time in the worst case. Therefore, using the Sleator and Tarjan algorithm, we have a total time complexity of $O(n^3 \log n)$ to find a maximum flow of a vb-flow network. 

\[\Box\]
Finally, we now show that there is an $O(n^5 \log n)$ algorithm for finding a rank-maximal stable matching in an instance of $\text{SM}$ based on polynomially-bounded weight vectors.

**Theorem 3.10.** Given an instance $I$ of $\text{SM}$, there is an $O(n^5 \log n)$ algorithm to find a rank-maximal stable matching in $I$ that is based on polynomially-bounded weight vectors.

**Proof.** We use the process described in Section 3.1. All operations from this are well under the required time complexity except number 4. Let $R_n(I) = (V, E')$ be a vb-flow network of $I$. Here, bounds on the number of edges and number of vertices are identical to the maximum weight case, that is $|E'| \leq n^2$ and $|V| \leq n^2$. This is because, despite having alternative versions of flows and capacities, we have an identical graph structure to the high-weight case.

By using the adaptation of Seator and Tarjan’s Max Flow algorithm from Lemma 3.9, we achieve an overall time complexity of $O(n^5 \log n)$ to find a maximum vb-flow $f$ in $R_n(I)$. Let $c_f$ denote a minimum cut in $R_n(I)$. By Theorem 3.7, $c(c_f) = \text{val}(f)$. Therefore using the process described in Section 3.1, with vector-based adaptations, we can find a rank-maximal stable matching in $O(n^5 \log n)$ without reverting to high weights.

Hence we have an $O(n^5 \log n)$ algorithm for finding a rank-maximal stable matching, without reverting to high-weight operations.

### 4 Generous stable matchings

We now show how to adapt the techniques in Section 3 to the generous setting. Let $I$ be an instance of $\text{SM}$ and let $M$ be a matching in $I$ with profile $p(M) = \langle p_1, p_2, \ldots, p_k \rangle$. Recall the reverse profile $p_r(M)$ is the vector $p_r(M) = \langle p_k, p_{k-1}, \ldots, p_1 \rangle$.

As with the rank-maximal case, we wish to use an approach to finding a generous stable matching that does not require exponential weights. Recall $n$ is the number of men in $I$. A simple $O(n)$ operation on a matching profile allows the rank-maximal approach described in the previous section to be used.

Let $M$ be a stable matching in $I$ with degree $k$ and profile $p(M) = \langle p_1, p_2, \ldots, p_k \rangle$. Since we wish to minimise the reverse profile $p_r(M) = \langle p_k, p_{k-1}, \ldots, p_1 \rangle$, we can simply maximise the reverse profile where the value of each element is negated. A short proof of this is given in Proposition 4.1.

We denote this profile by $p'_r(M)$, where

$$p'_r(M) = \langle -p_k, -p_{k-1}, \ldots, -p_2, -p_1 \rangle. \hspace{1cm} (2)$$

Thus in general, profile elements corresponding to $p'_r(M)$ can take negative values. All profile operations described in Section 2,3 still apply to profiles of this type.

**Proposition 4.1.** Let $M$ be a matching in an instance $I$ of $\text{SM}$. Then, $M \in \arg\min\{p_r(M') : M' \text{ is a matching in } I\}$ if and only if $M \in \arg\max\{p'_r(M') : M' \text{ is a matching in } I\}$.

**Proof.** Suppose $M$ is a matching in $I$ such that $p_r(M)$ is minimum taken over all matchings in $I$ and $p'_r(M)$ is not maximum taken over all matchings in $I$. Then, there is a matching $M'$ in $I$ such that $p'_r(M') > p'_r(M)$.

Let $p_r(M) = \langle p_1, p_2, \ldots, p_k \rangle$ (note that we use indices from 1 to $k$, despite $p_r(M)$ being a reverse profile) and therefore $p'_r(M) = \langle -p_1, -p_2, \ldots, -p_k \rangle$. Also let $p'_r(M') = \langle p'_1, p'_2, \ldots, p'_l \rangle$. Since $p'_r(M') > p'_r(M)$, there must exist some $i$ ($1 \leq i \leq l \leq \min(k, l)$) such that $p'_i > -p_i$ and $p'_j = -p_j$ for $1 \leq j < i$.

Then,

$$p_r(M') = \langle -p'_1, -p'_2, \ldots, -p'_l \rangle$$

$$= \langle p_1, p_2, \ldots, p_{i-1}, -p'_{i-1}, \ldots, -p'_l \rangle \hspace{1cm} (3)$$

$$< \langle p_1, p_2, \ldots, p_i, -p'_{i+1}, \ldots, -p'_l \rangle.$$}

Hence, $p_r(M)$ cannot be minimum taken over all matchings in $I$, a contradiction. Therefore, $M$ is a matching such that $p'_r(M)$ is maximum taken over all matchings in $I$. 

Now conversely, suppose that $M$ is a matching such that $p'_r(M)$ is maximum taken over all matchings in $I$, but $p_r(M)$ is not minimum taken over all matchings in $I$. Then there is a matching $M'$ in $I$ such that $p_r(M') < p_r(M)$.

Let $p_r(M) = (p_1, p_2, ..., p_k)$ and therefore $p'_r(M) = (-p_1, -p_2, ..., -p_k)$. Also let $p'_r(M') = (p'_1, p'_2, ..., p'_l)$ and so $p'_r(M') = (-p'_1, -p'_2, ..., -p'_l)$. Since $p_r(M') < p_r(M)$, there must exist some $i$ ($1 \leq i \leq \min(k, l)$) such that $-p'_i < p_i$ and $-p'_j = p_j$ for $1 \leq j < i$.

Then,

\[ p'_r(M') = (p'_1, p'_2, ..., p'_l) = (-p_1, -p_2, ..., -p_{i-1}, p'_i, ..., p'_l) \]

\[ > (-p_1, -p_2, ..., -p_i, p'_{i+1}, ..., p'_l) \] (4)

Hence, $p'_r(M)$ cannot be maximum taken over all matchings in $I$, a contradiction, meaning that $M$ is a matching such that $p_r(M)$ is minimum taken over all matchings in $I$. \qed

We now show that a generous stable matching may be found by eliminating a maximum profile closed subset of the rotation poset as in the rank-maximal case. Let $P$ be the rotation that takes us from stable matching $M$ to stable matching $M'$, where $M$ and $M'$ have profiles $p(M) = (p_1, p_2, ..., p_k)$ and $p(M') = (p'_1, p'_2, ..., p'_l)$ with each profile having length $k$ without loss of generality. Then $p(p) = (p_1 - p_1, p_2 - p_2, ..., p_k - p_k)$ and so $p(M') = p(M) + p(p)$. We know that $p'_r(M) = (-p_k, -p_{k-1}, ..., -p_1)$ and $p'_r(M') = (-p'_k, -p'_{k-1}, ..., -p'_1)$.

Now, since $p'_r(p) = (p_k - p'_k, p_{k-1} - p'_{k-1}, ..., p_1 - p'_1)$, in the generous case we have $p'_r(M') = p'_r(M) + p'_r(p)$. We next present Lemma 4.2 which is an analogue of Lemma 3.1 and shows that a generous stable matching may be found by eliminating a maximum profile closed subset of the rotation poset.

**Lemma 4.2.** Let $I$ be an instance of $\text{SM}$ and let $M_0$ be the man-optimal stable matching in $I$. A generous stable matching $M$ may be obtained by eliminating a maximum profile closed subset of the rotation poset $S$ from $M_0$. 

**Proof.** Let $R_p(I)$ be the rotation poset of $I$. Note that by Gusfield and Irving [11] Theorem 2.5.7, there is a 1-1 correspondence between closed subsets of $R_p(I)$ and the stable matchings of $I$. Let $S$ be a maximum profile closed subset of the rotation poset $R_p(I)$, whose rotation profiles are built according to Equation 2 and let $M$ be the unique corresponding stable matching. Then $p'_r(M) = p'_r(M_0) + \sum_{\rho_1 \in S} p'_r(\rho_1)$. Suppose $M$ is not generous. Then there is a stable matching $M'$ such that $p'_r(M') > p'_r(M)$. Since $p'_r(M') = p'_r(M) + p'_r(p)$, $M'$ corresponds to a unique closed subset $S'$ of the rotation poset, such that $p'_r(M') = p'_r(M_0) + \sum_{\rho_1 \in S'} p'_r(\rho_1)$. But $p'_r(M') > p'_r(M)$ and so $S$ cannot be a maximum profile closed subset of $R_p(I)$, a contradiction.

By Proposition 4.1 since $M$ is a matching such that $p'_r(M)$ is maximum among all stable matchings, $M$ is also a matching such that $p_r(M)$ is minimum among all stable matchings. Therefore $M$ is a generous stable matching in $I$. \qed

Recall that in Definition 2.1 we defined the high-weight function $w$ in order to show that a stable matching of maximum weight is a rank-maximal stable matching and then showed that vb-flows and vb-capacities correspond directly with this high-weight setting. In the generous case, since we seek a matching $M$ that maximises $p'_r(M)$, the negation of the reverse profile, the constraints of Definition 2.1 still apply. By Proposition 4.1 and Lemma 4.2 all processes from the previous section to find a rank-maximal stable matching may now be used to find a generous stable matching in $O(n^3 \log n)$ time.

However, it is also possible to exploit the structure of a generous stable matching to bound some part of the overall time complexity by the generous stable matching degree rather than by the number of men or women $n$.

Let $I$ be an instance of $\text{SM}$. First we find a minimum regret stable matching $M'$ of $I$ as described in Section 1 in $O(n^2)$ time. It must be the case that the degree $d(M)$ of a generous stable matching $M$ is the same as the degree of $M'$. Therefore, since no man or women can be assigned to a partner of rank higher than $d(M)$, it is possible to simply truncate all preference lists beyond rank $d(M)$, which has a positive effect on the overall time complexity of finding a generous stable matching.

Finally, Theorem 4.3 shows that a generous stable matching may be found in $O(n^2 d^3 \log n)$ time.
Theorem 4.3. Given an instance $I$ of $\text{SMI}$ there is an $O(n^2 d^3 \log n)$ algorithm to find a generous stable matching in $I$ using polynomially-bounded weight vectors, where $d$ is the degree of a minimum regret stable matching.

Proof. Each step required to find a generous stable matching in $I$ is outlined below along with its time complexity.

1. Calculate the degree $d$ of a minimum regret stable matching and truncate preference lists accordingly. A minimum regret stable matching may be found in $O(n^2)$ time \[10\]. We now assume all preference lists are truncated below rank $d$.

2. Calculate the man-optimal and woman-optimal stable matchings. The Extended Gale-Shapley Algorithm takes $O(nd)$ time since the number of acceptable pairs is now $nd$.

3. Find all rotations using the Minimal Differences Algorithm \[11\] in $O(nd)$ time, since the number of acceptable pairs is now $nd$.

4. Build the rotation digraph and vb-flow network, where rotation profiles are built according to Equation 2, using the process described in Section 3. We know that no man-woman pair can appear in more than one rotation. This means that the number of vertices in each of the associated rotation poset, rotation digraph and vb-flow network is also $O(nd)$. Identical reasoning (with adapted time complexities to suit the generous case) to that of Gusfield and Irving \[11\] pg. 112 may be used to obtain a bound of $O(nd)$ on the number of edges. This is because we have a bound of $O(nd)$ for the creation of both type 1 and type 2 edges of the rotation digraph. Therefore we may build the rotation digraph and vb-flow network in $O(nd)$ time.

5. Find a minimum cut of the vb-flow network using the process described in Section 3. With $O(nd)$ vertices and edges and a maximum length of $d$ for any preference list, the Sleator and Tarjan algorithm \[23\] has a time complexity of $O(n^2 d^2 \log n)$, since $d \leq n$, with an additional factor of $O(d)$ to perform operations over vectors. Hence this step takes a total of $O(n^2 d^3 \log n)$ time.

6. Use this cut to find a maximum profile closed subset $S$ of the rotations in $O(nd)$ time, since the numbers of vertices and edges are bounded by $O(nd)$.

7. Eliminate the rotations of $S$ from the man-optimal matching to find the corresponding rank-maximal stable matching.

Therefore the operation that dominates the time complexity is still Step 5 and the overall time complexity to find a generous stable matching for an instance $I$ of $\text{SMI}$ is $O(n^2 d^3 \log n)$.

5 Experiments and evaluations

5.1 Methodology

For our experiments we used randomly-generated data to compare properties such as egalitarian cost, sex-equal cost and degree over several types of optimal stable matchings (rank-maximal, generous, median, egalitarian and sex-equal). We also investigated the effect of varying instance size (in terms of the number of men or women) on these properties.

Our experiments explored 19 separate instance sizes with the number of men (and women) taking the values of $\{10, 20, ..., 100, 200, ..., 1000\}$ with 1000 instances tested in each case. Preliminary experimentation showed that complete preference lists generated according to a uniform distribution, in general, produced a larger number of stable matchings than using incomplete lists or linear distributions in which the most popular man was $p$ times more popular than the least popular man (similar for women). Therefore since we wished to compare properties of different stable matchings, all experimental cases below have complete, uniformly distributed preference lists.

For each generated instance we ran the Extended Gale-Shapley Algorithm \[8\] twice, finding both the man-optimal and woman-optimal stable matchings. Then, the Minimal Differences Algorithm \[11\] was used to find all rotations of an instance, and the rotation digraph was created in order to enumerate all stable matchings \[10\]. From this we were then able to compute all the types of optimal stable matchings described above. A total timeout of 1 hour was used for these three stages. Experiments were carried out on a machine running Ubuntu version 17.10 with 32 cores, $8 \times 64$GB.
RAM and Dual Intel® Xeon® CPU E5-2697A v4 processors. Instance generation and statistics programs were written in Python and run on Python version 2.7.14. All other code was written in Java and compiled using Java version 1.8.0. All Python and Java code was run on a single thread, with GNU Parallel [25] used to run multiple instances in parallel. Java garbage collection was run in serial and a maximum heap size of 1GB was distributed to each thread. Code and data repositories for these experiments can be found at https://doi.org/10.5281/zenodo.2545798 and https://doi.org/10.5281/zenodo.2542703 respectively.

Correctness testing was conducted in the following way. All stable matchings produced by all instances were checked for (1) capacity: each man or woman may only be assigned to one partner; and (2) stability: no man-woman pair who would rather be assigned to each other than their allocated partners exists with respect to the computed matching. Additional correctness testing was also conducted for all instances of size $n = 10, \ldots, 60$. For these instances, in addition to the above testing, a process took place to determine whether the number of stable matchings found matched the number found by an Integer Programming (IP). This was developed in Python version 2.7.14 with the IP modelling framework PuLP (version 1.6.9) [16] using the CPLEX solver [12], version 12.8.0. Each instance was run on a single thread with a time limit of 10 hours (all runs completed within this time), using the same machine described above. All correctness tests passed successfully.

### 5.2 Experimental results summary

Results of all experiments are shown in Tables 1, 2, 3, 4 and 5. Table 1 shows the 19 experiments with varied instance size $n$. Note the correlation between the naming of each experiment case and the instance size $n$. In this table we also show the number of instances per experiment $N_i$ in column 2, the number of instances that did not complete within the 1 hour per instance (timeout) is shown in column 3, and the average time taken is shown in column 4. Table 2 shows the average number of rotations $|R|$ and the average number of stable matchings $|M|$ per experiment. It also shows the minimum, maximum and average, egalitarian cost $e$ and sex-equal cost $e_s$ over all instances of each experiment. Tables 3, 4 and 5 display statistics for rank-maximal, generous and median stable matchings. Here, the minimum, maximum and average degree, egalitarian cost and sex-equal cost for those optimal matchings are given, in addition to the minimum, maximum and average number of first choices $f$ and the number of assignments in the last $a\%$ of the preference list $l_a$.

The main findings of these experiments are:

- **Number of first choices**: As expected, rank-maximal stable matchings obtain the largest number of first choices by some margin, when compared to generous and median stable matchings. When looking at the average number of first choices, this margin appears to increase from almost 1:1 in experiment S10 (6.9 for rank-maximal compared to 6.0 and 6.1 for generous and median respectively) to approximately 3:1 in experiment S1000 (158.4 for rank-maximal compared to 63.5 and 71.5 for generous and median respectively). Generous and median stable matchings are far more aligned, however generous is increasingly outperformed by median on the average number of first choices with ratios starting at around 1:1 for S10, gradually increasing to 1.1:1 for S1000. This is summarised in the plot shown in Figure 5-10.

- **Number of last $a\%$ choices**: For rank-maximal stable matchings, the average number of assignments in the final 10\% of preference lists was low, increasing from 0.4 for experiment S10 to 1.4 for experiment S1000. Note that this increase is far lower than the rate of instance size increase. The average number of generous stable matching choices in the final 50\% of preference lists decreased from 2.4 to 0.0 over all experiments. As the generous criteria minimises final choices, this is likely due to the number of stable matchings increasing with larger instance size. Finally, it is interesting to note that the average number of median stable matching choices in the final 20\% of preference lists decreases from 0.5 to 0.0 despite the number of lower ranked choices not being directly minimised.

Figure 5-11 shows how the average matching degree changes with respect to $n$ for rank-maximal, median and generous stable matchings. We can see that on average the rank-maximal criteria performs badly, putting men or women very close to the end of their preference list. As above the generous criteria outperforms either of the other optimisations, with median somewhere in between.
| Case | $N_f$ | Timeout | $n$  | time (ms) |
|------|------|---------|------|-----------|
| S10  | 1000 | 0       | 10   | 50.3      |
| S20  | 1000 | 0       | 20   | 60.2      |
| S30  | 1000 | 0       | 30   | 75.2      |
| S40  | 1000 | 0       | 40   | 93.2      |
| S50  | 1000 | 0       | 50   | 113.8     |
| S60  | 1000 | 0       | 60   | 133.9     |
| S70  | 1000 | 0       | 70   | 163.7     |
| S80  | 1000 | 0       | 80   | 205.5     |
| S90  | 1000 | 0       | 90   | 235.1     |
| S100 | 1000 | 0       | 100  | 278.0     |
| S200 | 1000 | 0       | 200  | 1084.5    |
| S300 | 1000 | 0       | 300  | 2886.1    |
| S400 | 1000 | 0       | 400  | 7972.7    |
| S500 | 1000 | 0       | 500  | 15934.8   |
| S600 | 1000 | 0       | 600  | 32925.3   |
| S700 | 1000 | 0       | 700  | 50802.4   |
| S800 | 1000 | 1       | 800  | 87169.2   |
| S900 | 1000 | 0       | 900  | 128878.0  |
| S1000| 1000 | 1       | 1000 | 196029.9  |

Table 1: General instance information.

- **Number of stable matchings**: From Table 2, we can see that the number of stable matchings increases with instance size. In [2009], Lennon and Pittel [15] showed that the number of expected stable matchings in an instance of size $n$ tends to the order of $n \log n$. Our experiments confirm this result and show a reasonably linear correlation between $n \ln n$ and the average number of stable matchings for instances with $n \geq 100$ (Figure 5-12).

- **Egalitarian cost and sex-equal cost**: The range of egalitarian costs and sex-equal costs over all experiments (Table 2) is small when compared with results found for these measures in the rank-maximal, generous and median stable matching experiments. Comparing the average sex-equal cost of the rank maximal (Table 3), generous (Table 4), median (Table 5) and optimal sex-equal case (Table 2), we can see that a generous stable matching is a far closer approximation of a sex-equal matching in practice (Figure 5-13). This is followed by the median and then the rank-maximal solution concepts. A similar, though less pronounced, result holds for the egalitarian cost (Figure 5-14).
| Case | | \( |R| \) | | \( |M| \) | \( \min(e) \) | \( \max(e) \) | \( \text{av}(e) \) | \( \min(e_d) \) | \( \max(e_d) \) | \( \text{av}(e_d) \) |
|------|------|------|------|------|------|------|------|------|------|------|
| S10  | 1.8  | 3.0  | 40   | 78   | 58.0 | 0    | 41   | 5.2  |
| S20  | 4.2  | 6.5  | 113  | 215  | 167.2| 0    | 71   | 9.9  |
| S30  | 6.5  | 10.9 | 243  | 371  | 311.2| 0    | 137  | 13.6 |
| S40  | 8.9  | 15.7 | 396  | 572  | 484.3| 0    | 130  | 16.1 |
| S50  | 11.2 | 20.9 | 567  | 821  | 679.5| 0    | 218  | 19.8 |
| S60  | 13.4 | 27.2 | 730  | 1057 | 896.2| 0    | 310  | 23.3 |
| S70  | 15.9 | 34.0 | 955  | 1299 | 1133.2| 0   | 255  | 24.2 |
| S80  | 18.2 | 40.6 | 1164 | 1609 | 1390.0| 0  | 217  | 27.2 |
| S90  | 20.0 | 46.4 | 1447 | 1909 | 1660.9| 0  | 178  | 31.6 |
| S100 | 22.4 | 54.2 | 1663 | 2179 | 1947.0| 0  | 314  | 33.2 |
| S200 | 41.9 | 138.8| 4865 | 6257 | 5554.9| 0  | 411  | 62.0 |
| S300 | 59.2 | 231.0| 9444 | 11301| 10250.0| 0  | 528  | 93.4 |
| S400 | 76.2 | 337.6| 14755| 16999| 15847.1| 0  | 885  | 116.3|
| S500 | 90.7 | 442.0| 20610| 23767| 22137.6| 0  | 1158 | 136.3|
| S600 | 105.8| 566.1| 27502| 31147| 29147.8| 0  | 1311 | 170.8|
| S700 | 119.1| 675.5| 35117| 38975| 36772.6| 0  | 1352 | 206.2|
| S800 | 131.4| 804.0| 42372| 47962| 44930.2| 0  | 1809 | 230.0|
| S900 | 144.9| 937.6| 50650| 56289| 53658.2| 0  | 1812 | 251.3|
| S1000| 157.6|1115.2| 59776| 65571| 62875.9| 0  | 2295 | 269.4|

Table 2: General statistical results.
| Case | min(f) | max(f) | av(f) | min(l_{10}) | max(l_{10}) | av(l_{10}) | min(d) | max(d) | av(d) | min(e) | max(e) | av(e) | min(e_d) | max(e_d) | av(e_d) |
|------|--------|--------|-------|-------------|-------------|------------|--------|--------|-------|--------|--------|------|----------|----------|-------|
| S10  | 0      | 13     | 6.9   | 0.0         | 3.0         | 0.4        | 4      | 10     | 8.6   | 40     | 87     | 60.4 | 0        | 63       | 16.5  |
| S20  | 2      | 19     | 10.6  | 0.0         | 4.0         | 0.4        | 9      | 20     | 17.0  | 124    | 275    | 182.3 | 0        | 205      | 64.8  |
| S30  | 6      | 22     | 13.5  | 0.0         | 4.0         | 0.5        | 12     | 30     | 25.3  | 247    | 496    | 349.3 | 0        | 374      | 142.4 |
| S40  | 6      | 26     | 16.0  | 0.0         | 4.0         | 0.6        | 18     | 40     | 34.0  | 407    | 810    | 564.2 | 1        | 650      | 265.6 |
| S50  | 7      | 32     | 18.2  | 0.0         | 6.0         | 0.6        | 22     | 50     | 42.4  | 604    | 1150   | 809.7 | 1        | 902      | 405.0 |
| S60  | 11     | 35     | 20.4  | 0.0         | 4.0         | 0.6        | 25     | 60     | 51.4  | 742    | 1617   | 1095.5 | 13       | 1332     | 596.8 |
| S70  | 11     | 36     | 22.6  | 0.0         | 7.0         | 0.6        | 29     | 70     | 60.2  | 1040   | 2054   | 1421.3 | 2        | 1730     | 818.4 |
| S80  | 11     | 42     | 24.6  | 0.0         | 4.0         | 0.7        | 34     | 80     | 68.9  | 1328   | 2586   | 1780.0 | 3        | 2106     | 1067.6 |
| S90  | 14     | 43     | 26.6  | 0.0         | 5.0         | 0.7        | 41     | 90     | 78.4  | 1501   | 3156   | 2186.0 | 7        | 2644     | 1363.4 |
| S100 | 13     | 45     | 28.7  | 0.0         | 5.0         | 0.7        | 40     | 100    | 87.2  | 1807   | 3609   | 2617.4 | 32       | 2983     | 1693.5 |
| S200 | 24     | 73     | 45.8  | 0.0         | 7.0         | 0.9        | 98     | 200    | 177.6 | 5617   | 12532  | 8616.7 | 1277     | 11096    | 6494.2 |
| S300 | 33     | 96     | 61.6  | 0.0         | 6.0         | 1.0        | 110    | 300    | 269.4 | 10333  | 24028  | 17608.0 | 1677     | 21562    | 14213.9 |
| S400 | 46     | 110    | 76.5  | 0.0         | 8.0         | 1.1        | 230    | 400    | 361.3 | 19149  | 39409  | 29521.9 | 10331    | 35863    | 24778.8 |
| S500 | 55     | 132    | 90.4  | 0.0         | 7.0         | 1.1        | 271    | 500    | 454.4 | 26729  | 60640  | 43725.3 | 15185    | 56490    | 37529.6 |
| S600 | 67     | 155    | 105.3 | 0.0         | 9.0         | 1.2        | 365    | 600    | 546.5 | 41102  | 81496  | 61248.3 | 28874    | 76436    | 53676.6 |
| S700 | 65     | 162    | 118.5 | 0.0         | 9.0         | 1.3        | 396    | 700    | 641.9 | 52098  | 108000 | 80778.1 | 37230    | 101646   | 71714.5 |
| S800 | 77     | 178    | 131.4 | 0.0         | 8.0         | 1.3        | 402    | 800    | 732.9 | 60915  | 134559 | 102579.6 | 40623    | 126963   | 91993.6 |
| S900 | 94     | 198    | 144.5 | 0.0         | 8.0         | 1.3        | 491    | 900    | 824.0 | 86785  | 183870 | 127944.3 | 69419    | 175936   | 115909.3 |
| S1000| 104    | 208    | 158.4 | 0.0         | 8.0         | 1.4        | 652    | 1000   | 921.2 | 104836 | 205341 | 154730.8 | 83732    | 195439   | 141113.6 |

Table 3: Rank-maximal stable matching statistical results.
| Case | min(f) | max(f) | av(f) | min(l50) | max(l50) | av(l50) | min(d) | max(d) | av(d) | min(e) | max(e) | av(e) | min(eₜ) | max(eₜ) | av(eₜ) |
|------|--------|--------|-------|----------|----------|---------|--------|--------|-------|--------|--------|-------|----------|--------|-------|
| S10  | 0      | 12     | 6.0   | 0.0      | 6.0      | 2.4     | 4      | 10     | 7.6   | 40     | 81     | 58.8  | 0        | 41     | 8.3   |
| S20  | 2      | 17     | 8.8   | 0.0      | 7.0      | 2.4     | 8      | 20     | 13.8  | 113    | 225    | 170.0 | 0        | 93     | 22.2  |
| S30  | 3      | 20     | 10.9  | 0.0      | 7.0      | 2.2     | 12     | 30     | 19.3  | 243    | 396    | 317.2 | 0        | 161    | 40.7  |
| S40  | 3      | 21     | 12.4  | 0.0      | 8.0      | 1.8     | 14     | 40     | 24.2  | 397    | 626    | 492.8 | 0        | 294    | 60.5  |
| S50  | 5      | 28     | 14.0  | 0.0      | 7.0      | 1.4     | 18     | 49     | 28.7  | 567    | 875    | 691.0 | 0        | 398    | 85.2  |
| S60  | 6      | 28     | 15.2  | 0.0      | 7.0      | 1.2     | 21     | 56     | 33.0  | 730    | 1105   | 910.5 | 0        | 470    | 166.9 |
| S70  | 6      | 29     | 16.8  | 0.0      | 6.0      | 0.9     | 24     | 68     | 37.0  | 955    | 1330   | 1151.6 | 0       | 602    | 140.0 |
| S80  | 7      | 28     | 17.8  | 0.0      | 5.0      | 0.7     | 24     | 76     | 40.6  | 1164   | 1683   | 1411.4 | 0       | 670    | 163.2 |
| S90  | 9      | 32     | 18.9  | 0.0      | 7.0      | 0.5     | 26     | 75     | 44.4  | 1455   | 1981   | 1683.9 | 0       | 825    | 186.4 |
| S100 | 8      | 34     | 20.0  | 0.0      | 5.0      | 0.4     | 30     | 74     | 47.8  | 1704   | 2276   | 1974.6 | 1       | 951    | 219.4 |
| S200 | 13     | 55     | 28.2  | 0.0      | 2.0      | 0.0     | 51     | 119    | 78.5  | 4865   | 6375   | 5622.4 | 1       | 2428   | 566.6 |
| S300 | 19     | 55     | 34.7  | 0.0      | 1.0      | 0.0     | 70     | 165    | 104.3 | 9460   | 12079  | 10366.3 | 2       | 6095   | 1015.7 |
| S400 | 22     | 62     | 40.0  | 0.0      | 1.0      | 0.0     | 86     | 203    | 126.3 | 14802  | 17981  | 16050.4 | 0       | 7130   | 1503.9 |
| S500 | 24     | 76     | 44.8  | 0.0      | 0.0      | 0.0     | 107    | 223    | 147.7 | 20625  | 24592  | 22358.6 | 2       | 9214   | 2056.5 |
| S600 | 30     | 71     | 49.4  | 0.0      | 0.0      | 0.0     | 124    | 292    | 165.9 | 27514  | 32564  | 29406.9 | 2       | 11850  | 2603.3 |
| S700 | 33     | 80     | 52.7  | 0.0      | 0.0      | 0.0     | 135    | 264    | 183.7 | 35117  | 39710  | 37086.4 | 2       | 14023  | 3201.3 |
| S800 | 35     | 83     | 57.3  | 0.0      | 0.0      | 0.0     | 145    | 291    | 200.3 | 42579  | 49508  | 45308.3 | 0       | 19508  | 3883.3 |
| S900 | 30     | 81     | 59.5  | 0.0      | 0.0      | 0.0     | 160    | 305    | 216.4 | 50957  | 59447  | 54104.8 | 2       | 22087  | 4939.8 |
| S1000| 40     | 89     | 63.5  | 0.0      | 0.0      | 0.0     | 176    | 350    | 230.6 | 60181  | 69426  | 63364.8 | 2       | 26076  | 5163.1 |

Table 4: Generous stable matching statistical results.
Table 5: Median stable matching statistical results.

| Case | min($f$) | max($f$) | av($f$) | min($l_{20}$) | max($l_{20}$) | av($l_{20}$) | min($d$) | max($d$) | av($d$) | min($e$) | max($e$) | av($e$) | min($e_d$) | max($e_d$) | av($e_d$) |
|------|----------|----------|---------|--------------|--------------|-------------|----------|----------|--------|----------|----------|--------|------------|------------|----------|
| S10  | 0        | 12       | 6.1     | 0.0          | 3.0          | 0.5         | 4        | 10       | 8.2    | 40       | 79       | 59.9   | 0          | 41         | 9.5      |
| S20  | 2        | 18       | 8.9     | 0.0          | 4.0          | 0.4         | 8        | 20       | 15.3   | 113      | 224      | 173.7  | 0          | 121        | 27.3     |
| S30  | 4        | 20       | 11.0    | 0.0          | 5.0          | 0.3         | 12       | 30       | 22.0   | 243      | 416      | 323.9  | 0          | 228        | 50.8     |
| S40  | 3        | 22       | 12.6    | 0.0          | 3.0          | 0.3         | 17       | 40       | 28.2   | 402      | 693      | 504.4  | 0          | 403        | 83.8     |
| S50  | 5        | 25       | 14.2    | 0.0          | 5.0          | 0.2         | 19       | 50       | 33.9   | 570      | 924      | 709.6  | 0          | 537        | 123.9    |
| S60  | 6        | 30       | 15.6    | 0.0          | 3.0          | 0.2         | 23       | 60       | 39.8   | 756      | 1232     | 938.0  | 0          | 756        | 173.1    |
| S70  | 8        | 30       | 17.0    | 0.0          | 2.0          | 0.2         | 25       | 70       | 45.0   | 970      | 1487     | 1186.7 | 0          | 973        | 222.8    |
| S80  | 7        | 30       | 18.2    | 0.0          | 2.0          | 0.1         | 28       | 80       | 50.0   | 1164     | 1858     | 1457.6 | 0          | 1171       | 280.2    |
| S90  | 9        | 32       | 19.5    | 0.0          | 4.0          | 0.1         | 31       | 90       | 55.4   | 1447     | 2484     | 1744.8 | 1          | 1850       | 355.7    |
| S100 | 7        | 34       | 20.5    | 0.0          | 2.0          | 0.1         | 34       | 100      | 60.5   | 1663     | 2604     | 2045.8 | 1          | 1553       | 415.6    |
| S200 | 14       | 52       | 29.5    | 0.0          | 2.0          | 0.1         | 57       | 199      | 105.2  | 5006     | 8226     | 5917.3 | 1          | 6108       | 1477.0   |
| S300 | 21       | 69       | 36.8    | 0.0          | 3.0          | 0.0         | 79       | 294      | 144.4  | 9541     | 17224    | 11046.9| 4          | 13374      | 3115.2   |
| S400 | 21       | 72       | 42.8    | 0.0          | 3.0          | 0.0         | 99       | 393      | 178.1  | 14934    | 26115    | 17149.7| 17         | 20951      | 5052.7   |
| S500 | 25       | 99       | 48.3    | 0.0          | 9.0          | 0.0         | 110      | 496      | 212.0  | 20725    | 48487    | 24257.5| 15         | 43025      | 7752.5   |
| S600 | 31       | 102      | 53.5    | 0.0          | 4.0          | 0.0         | 127      | 595      | 244.3  | 28028    | 58399    | 32052.6| 14         | 50581      | 10459.2  |
| S700 | 32       | 113      | 58.0    | 0.0          | 2.0          | 0.0         | 141      | 665      | 276.7  | 35634    | 71039    | 40774.9| 34         | 60999      | 13863.8  |
| S800 | 37       | 129      | 63.0    | 0.0          | 3.0          | 0.0         | 162      | 797      | 304.0  | 42713    | 105477   | 50215.3| 10         | 95243      | 18038.7  |
| S900 | 36       | 136      | 66.0    | 0.0          | 2.0          | 0.0         | 174      | 833      | 331.2  | 51568    | 117599   | 60037.6| 46         | 104663     | 21334.7  |
| S1000| 44       | 163      | 71.5    | 0.0          | 3.0          | 0.0         | 191      | 975      | 362.5  | 60270    | 155476   | 71456.3| 11         | 142570     | 27270.2  |
Figure 5-10: Plot of the average number of first choices vs $n$ for rank-maximal, median and generous stable matchings. A second order polynomial model has been assumed for all best-fit lines.

Figure 5-11: Plot of the average matching degree vs $n$ for rank-maximal, median and generous stable matchings. A second order polynomial model has been assumed for all best-fit lines.
Figure 5-12: Plot of the average number of stable matchings $|M|$ vs $n \log n$. A first order polynomial model has been assumed for the best-fit line.

Figure 5-13: A log-log plot of the average sex-equal cost vs $n$ for rank-maximal, median, generous and sex-equal stable matchings. A first order polynomial model has been assumed for all best-fit lines.
Figure 5-14: Plot of the average egalitarian cost vs $n$ for rank-maximal, median, generous and egalitarian stable matchings. A second order polynomial model has been assumed for all best-fit lines.
6 Future work

In this paper we have described a new method for computing rank-maximal and generous stable matchings for an instance of $\text{smi}$ using polynomially-bounded weight vectors that avoids the use of weights that can be exponential in the number of men. By using this approach we are able to avoid high-weight calculation problems such as overflow, inaccuracies and limitations in memory.

In Section 1, two potential improvements that could be made to the process of finding a rank-maximal stable matching in an instance of $\text{smi}$ were highlighted. First was the adaptation of Orlin's \cite{19} max flow algorithm to work in the vector-based setting. This adaptation would result in a time complexity of $O(n^5)$ to find a rank-maximal stable matching, improving on the method outlined in this paper by a factor of $\log n$, however it is not clear that Orlin's algorithm can be adapted to the vb-flow setting. Additionally, Feder \cite{6} used an entirely different technique based on weighted $\text{SAT}$ for finding a rank-maximal stable matching in $O(n^{5/2})$ time. It remains to be seen if this could be adapted to work in the vector-based setting.

References

[1] Atila Abdulkadiroğlu, Parag A. Pathak, Alvin E. Roth, and T. Sönmez. The Boston public school match. American Economic Review, Papers and Proceedings, 95(2):368–371, 2005.

[2] Christine Cheng, Eric McDermid, and Ichiro Suzuki. A unified approach to finding good stable matchings in the hospitals/residents setting. Theoretical Computer Science, 400:84–99, 2008.

[3] Christine T. Cheng. The generalized median stable matchings: Finding them is not that easy. In Proceedings. LATIN ’08: The 8th Latin-American Theoretical INformatics symposium, volume 4957, pages 568 – 579. Springer, 2008.

[4] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms. MIT Press, 3rd edition, 2009.

[5] Yefim A. Dinic. Algorithm for solution of a problem of maximum flow in networks with power estimation. Soviet Math. Dokl., 11(5):12771280, 1970.

[6] Tomás Feder. A new fixed point approach for stable networks and stable marriages. Journal of Computer and System Sciences, 45:233–284, 1992.

[7] Lester R. Ford and Delbert R. Fulkerson. Maximal flow through a network. Canadian Journal of Mathematics, 8:399–404, 1956.

[8] David Gale and Lloyd S. Shapley. College admissions and the stability of marriage. The American Mathematical Monthly, 69(1):9–15, 1962.

[9] David Gale and Marilda Sotomayor. Some remarks on the stable matching problem. Discrete Applied Mathematics, 11:223–232, 1985.

[10] Dan Gusfield. Three fast algorithms for four problems in stable marriage. Siam Journal of Computing, 16(1):111–128, 1987.

[11] Dan Gusfield and Robert W. Irving. The Stable Marriage Problem. The MIT Press, 1989.

[12] IBM. CPLEX optimizer. \url{https://www.ibm.com/analytics/cplex-optimizer} 2019.

[13] Robert W. Irving, Paul Leather, and Dan Gusfield. An efficient algorithm for the "optimal" stable marriage. Journal of the Association for Computational Machinery, 34(3):532–543, 1987.

[14] Akiko Kato. Complexity of the sex-equal stable marriage problem. Japan Journal of Industrial and Applied Mathematics, 10:1–19, 1993.

[15] Craig Lennon and Boris Pittel. On the likely number of solutions for the stable marriage problem. Combinatorics, Probability and Computing, 18:371–421, 2009.
[16] Stuart Mitchell, Michael O’Sullivan, and Iain Dunning. PuLP: A linear programming toolkit for Python. Optimization Online, 2011.

[17] Oracle. Primitive data types. https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html, 2017. [Java version 8. Online; accessed 30 January 2019].

[18] Oracle. Class BigInteger. https://docs.oracle.com/javase/8/docs/api/java/math/BigInteger.html, 2018. [Java version 8. Online; accessed 07 August 2018].

[19] James B. Orlin. Max flows in O(nm) time, or better. In Proceedings. STOC ’13: forty-fifth annual ACM symposium on Theory of computing, pages 765–774. Association for Computing Machinery, 2013.

[20] Elliott Peranson and Richard R. Randlett. The NRMP matching algorithm revisited: Theory versus practice. Academic Medicine, 70(6):477–484, 1995.

[21] Alvin E. Roth. The evolution of the labor market for medical interns and residents: a case study in game theory. Journal of Political Economy, 92(6):991–1016, 1984.

[22] Alvin E. Roth. On the allocation of residents to rural hospitals: A general property of two-sided matching markets. The Econometric Society, 54(2):425–427, 1986.

[23] Daniel D. Sleator and Robert Endre Tarjan. A data structure for dynamic trees. Journal of Computer and System Sciences, 26(3):362–391, 1983.

[24] Colin T.S. Sng. Efficient Algorithms for Bipartite Matching Problems with Preferences. PhD thesis, 2008.

[25] Ole Tange. GNU parallel - the command-line power tool. The USENIX Magazine, pages 42–47, 2011.

[26] Chuny-Piaw Teo and Jay Sethuraman. The geometry of fractional stable matchings and its applications. Mathematics of Operations Research, 23(4):874–891, 1998.

[27] Yu Zhang. The determinants of national college entrance exam performance in China - with an analysis of private tutoring. Master’s thesis, Columbia University, 2011.