Regular Paper

# Hardness of Instance Generation with Optimal Solutions for the Stable Marriage Problem

Yuki Matsuyama[1,a)] Shuichi Miyazaki[2,b)]

Received: March 10, 2020, Accepted: November 5, 2020

**Abstract:** In a variant of the stable marriage problem where ties and incomplete lists are allowed, finding a stable matching of maximum cardinality is known to be NP-hard. There are a lot of experimental studies for evaluating the performance of approximation algorithms or heuristics, using randomly generated or artificial instances. One of standard evaluation methods is to compare an algorithm's solution with an optimal solution, but finding an optimal solution itself is already hard. In this paper, we investigate the possibility of generating instances with known optimal solutions. We propose three instance generators based on a known random generation algorithm, but unfortunately show that none of them meet our requirements, implying a difficulty of instance generation in this approach.

**Keywords:** stable marriage problem, NP-hardness, instance generation

## 1. Introduction

Consider a situation where there are $n$ men and $n$ women, where each person has a preference list that ranks the members of the opposite gender, and we are seeking for a matching, i.e., $n$ pairs of a man and a woman, based on the preference lists. A matching that does not have an unmatched pair each of whom prefers the other to the matched partner is called a *stable matching*. The problem of finding a stable matching is called the *stable marriage problem* (or *SM* for short) [2] and has been extensively studied. The stable marriage problem is widely used in matching or assignment systems, such as assigning residents to hospitals and assigning students to labs in universities. It is known that for any instance, at least one stable matching exists, and one can be found in $O(n^2)$ time by using the so-called Gale-Shapley algorithm [2].

Note that, in the above setting, each person must rank *all* the members of the opposite gender in a *strict* order. These restrictions are clearly strict for applications so there are two major extensions considered. One is to allow ties in preference lists, where two or more persons with the same preference are tied in a preference list. This variant is denoted as *SMT*. When ties are introduced, there exist three stability notions, super, strong, and weak stabilities. Throughout this paper, we consider only the weak stability, which is the most natural notion (formal definitions will be given later). In SMT, a stable matching always exists and one can be found in $O(n^2)$ time. The other is to allow incomplete lists, where each person's preference list can contain only acceptable persons. This variant is denoted as *SMI*. Similarly, in SMI, a stable matching exists in any instance and one can be found in $O(n^2)$ time. In this case, a stable matching may no longer be a perfect matching, but all the stable matchings have the same size [3], [28], [29]. Hence in these three problems, it is easy to find a maximum size stable matching.

However, when both ties and incomplete lists are allowed (denoted *SMTI*), stable matchings may have different sizes. Therefore, although a stable matching can still be obtained in polynomial time, there is no guarantee that its size is maximum. In fact, the problem of finding a largest stable matching is NP-hard [12], [21], and hence the existence of a polynomial time algorithm cannot be expected. For this reason, heuristics such as local search methods [4], [5], [24], [25], mathematical programming approaches [6], [17], [22], and approximation algorithms with theoretical guarantee [1], [9], [10], [13], [14], [15], [16], [18], [23], [26] have been devised.

In experimental studies, approximation algorithms and/or heuristics are implemented and run on computers to evaluate their performance. In some experiments, the sizes of an optimal solution and the algorithm's solution are compared to evaluate the goodness of algorithms [7], [11], [27], but since the problem is NP-hard, it takes a fair amount of time to find an optimal solution for large instances. Therefore, there is a problem that experiments can be performed only on small size instances. If an instance whose optimal solution is known in advance can be used, this problem can be solved.

In this study, we investigate the possibility of constructing an instance generation algorithm that produces instances whose optimal solution is known. To evaluate the performance of algorithms fairly and accurately, it is desirable to use a wide variety of instances. However, by using the same argument as the literature [19], [30], it can be shown that NP=coNP holds if there is a polynomial-time algorithm that can generate any instance with an optimal solution. Therefore, we aim to devise algorithms that

[1] Graduate School of Informatics, Kyoto University, Kyoto 606–8501, Japan
[2] Academic Center for Computing and Media Studies, Kyoto University, Kyoto 606–8501, Japan
[a)] matsuyama@net.ist.i.kyoto-u.ac.jp
[b)] shuichi@media.kyoto-u.ac.jp

can generate a wide range, if not all, of the instances. In this paper, we focus on an instance generation algorithm based on the method of Gent and Prosser [7], which is widely used as input for an experimental performance evaluation. Specifically, we first create an SMI instance $I$ and find its stable matching $M$. In the SMI instance, $M$ is the maximum size stable matching of $I$ because, as mentioned above, all the stable matchings are of the same size. Then, by adding ties to preference lists one after another, we eventually obtain an SMTI instance $I'$. Since the stable matching of the instance before adding a tie is also stable in the instance after adding the tie, $M$ is a stable matching of $I'$. Our goal is to make $M$ be a maximum stable matching of $I'$. To do so, it is necessary not to create a stable matching larger than $M$ when adding ties.

We first considered an algorithm that determines whether a stable matching larger than $M$ will be created when adding a tie, and adds this tie if not. If we can make this decision correctly, it is possible to generate various types of instances. However, we show that P=NP holds if there exists a polynomial time algorithm to make this decision.

Next, when adding a tie on the list of a person $p$, we put some restrictions on putting $p$'s partner in $M$ into the tie. This is a sufficient condition that a stable matching larger than $M$ is not created. However, we found that this condition is too strong to the extent that the algorithm does not create stable matchings smaller than $M$ either, so the size of stable matchings is always constant.

We then relaxed the above constraint, while maintaining the condition that a stable matching larger than $M$ is not created. Although we performed a more detailed analysis and devised a new generator that actually relaxes the constraints, we found that the size of the stable matching is still constant.

To summarize, the first method is flexible but cannot be expected to run in polynomial time. The second and the third methods run in polynomial time but can produce only instances having stable matchings of the same size, which are useless for the purpose of experiments since for such instances, it suffices to find one stable matching, e.g., by the Gale-Shapley algorithm. From our study in this paper, we think that if we insist on using Gent and Prosser's algorithm [7], more consideration is needed, or otherwise we may have to seek for completely different methods.

## 2. Preliminaries

### 2.1 Stable Marriage Problem

An SMTI instance of size $n$ consists of $n$ men, $n$ women, and each person's incomplete preference list that may contain ties. If the preference list of a person $p$ includes a person $q$, we say that $q$ is *acceptable* to $p$. Here, we assume without loss of generality that woman $w$ is acceptable to man $m$ if and only if $m$ is acceptable to $w$.

Consider three persons $p$, $q$, and $q'$, where $q$ and $q'$ are of the same gender and $p$ is of a different gender. In the preference list of $p$, if $q$ is written higher than $q'$, we write $q \succ_p q'$. If both are written in the same order (i.e., $q$ and $q'$ are tied in $p$'s list), we write $q =_p q'$. The notation $q \succeq_p q'$ means that either $q \succ_p q'$ or $q =_p q'$ holds.

A set of pairs $(m, w)$ in which each person appears at most once is called a *matching*. The *size* of a matching $M$, written as $|M|$, is the number of pairs included in $M$. If $(m, w) \in M$ for a matching $M$, then we write $M(m) = w$ and $M(w) = m$. In this case, $m$ is called the *partner* of $w$, and similarly, $w$ is called the *partner* of $m$. If the person $p$ does not appear in $M$, then $p$ is said to be *single* in $M$.

```
The Gale-Shapley algorithm [2]

Initially all the persons are single.
The following steps are repeated until every man either is
engaged or has an empty list.
Step 1
    A single man $m$ whose preference list is nonempty proposes
    to the highest woman $w$ on his preference list.
Step 2
    (1) If $w$ is single, $m$ becomes engaged with $w$.
    (2) If $w$ is already engaged with $m'$,
        (a) If $m' \succ_w m$, $m$ and $w$ delete each other from
            the list.
        (b) If $m \succ_w m'$, $m'$ becomes single and $w$ becomes
            engaged with $m$. $m'$ and $w$ delete each
            other from the list.
Output the set of engaged pairs.
```

**Fig. 1** The Gale-Shapley algorithm [2].

We then define the stability of matchings. When ties are present, there are three types of stability notions, weak stability, strong stability, and super-stability. In this paper, we deal with only weak stability. A *blocking pair* is a pair $(m, w)$ that satisfies all the following three conditions:

(1) $M(m) \neq w$ but $m$ and $w$ are mutually acceptable.
(2) $w \succ_m M(m)$ or $m$ is single in $M$.
(3) $m \succ_w M(w)$ or $w$ is single in $M$.

A matching that has no blocking pair is a *weakly stable matching*. In this paper, we call a weakly stable matching simply a *stable matching*. Different sizes of stable matchings may exist in an SMTI instance. We write the size of a maximum stable matching of an instance $I$ as $opt(I)$. The problem of finding a maximum size stable matching is denoted *MAX SMTI*.

### 2.2 The Gale-Shapley Algorithm

Gale and Shapley proposed a polynomial-time algorithm (the Gale-Shapley algorithm) [2] for finding a stable matching in SM and SMI. Instance generators proposed in this paper use this algorithm as a subroutine. The Gale-Shapley algorithm is described in **Fig. 1**.

### 2.3 Instance Generators and Hardness of Constructing a Complete Generator

A MAX SMTI instance generation algorithm considered in this paper is a nondeterministic algorithm that, given an input $1^n$, outputs a pair $(I, opt(I))$ of an SMTI instance of size $n$ and its optimal solution. When an instance generator $G$ can generate any SMTI instance, we say that $G$ is *complete*.

An ultimate goal of this research is to find a complete instance generator that runs in polynomial time in $n$. However, this is un-

Gent and Prosser's instance generator [7]

**Input**
- Instance size $n$
- Probability $p_1$ of removing a person from the list
- Probability $p_2$ of tying a person with a higher person

**Step 1**
A random preference list of size $n$ is produced for each man and each woman.

**Step 2**
We iterate over each man's preference list as follows. For a man $m_i$ and for all women $w_j$ in his preference list, we generate a random number $0 \leq p < 1$. If $p \leq p_1$ we delete $w_j$ from $m_i$'s preference list and delete $m_i$ from $w_j$'s preference list.

**Step 3**
If any man or woman has an empty preference list, discard the instance and go to Step 1.

**Step 4**
We iterate over each person's (men and women's) preference list as follows. For a man $m_i$ and for his choices $c_i$ ranging from his second to his last, we generate a random number $0 \leq p < 1$. If $p \leq p_2$ then the preference for his $c_i$th choice is the same as his $c_{i-1}$th choice, otherwise his $c_i$th choice is one greater than (i.e., one lower than) his $c_{i-1}$th choice.

**Fig. 2** Gent and Prosser's instance generator [7].

likely due to the following proposition, whose proof is almost the same as that of [19], [30].

**Proposition 1.** If there is a complete polynomial time instance generator for MAX SMTI, then NP=coNP.

**Proof.** Consider the following language $L = \{(I, k) \mid I$ is an SMTI instance, $opt(I) \geq k\}$ corresponding to the MAX SMTI decision problem. Since $L$ is NP-complete [12], [21] and the language $\{(I, k) \mid I$ is an SMTI instance, $k$ is an integer$\}$ belongs to the class P, the complement of $L$, $\overline{L} = \{(I, k) \mid I$ is an SMTI instance, $opt(I) < k\}$, is coNP-complete.

Given a complete polynomial-time instance generator $H$, consider the following algorithm $A$ that uses it to nondeterministically determine whether an input $(I, k)$ belongs to $\overline{L}$ or not. $A$ runs $H$ on input $n$, and obtains its output $(I', k')$. If $I' = I$ and $k' < k$, accept $(I, k)$; otherwise, reject. This works in polynomial time and determines membership in $\overline{L}$ correctly, so $\overline{L} \in NP$. From this, NP = coNP is derived. □

### 2.4 Gent and Prosser's Instance Generation Algorithm

The instance generator used in Ref. [7], which is the basis of our proposed method, is described in **Fig. 2**.

In Step 1, a complete list without ties is created. Next, in Step 2, an incomplete list having an expected length of $p_1 n$ is obtained. After Step 2, we obtain an SMI instance (a set of incomplete preference lists without ties). Then, in Step 4, each person on a preference list is tied with a person just above him/her with probability $p_2$, and as a result, we obtain an SMTI instance.

General Framework

**Step 1**
Generate a list of length $n$ for all persons, i.e., $n$ permutations of the members of the opposite gender.

**Step 2**
For each pair $(m, w)$, we decide whether to remove this pair from the instance. If so, remove $w$ from the list of $m$ and remove $m$ from the list of $w$.

**Step 3**
Let $I$ be the created SMI instance.

**Step 4**
Solve $I$ using the Gale-Shapley algorithm, and let $M$ be the obtained stable matching.

**Step 5**
The following is repeated a certain number of times. Arbitrarily choose a person $p$ and two adjacent ties $t_1$ and $t_2$ of $p$. According to the condition (specified by each concrete algorithm), merge or do not merge $t_1$ and $t_2$ to a single tie.

**Step 6**
Let $I'$ be the created SMTI instance. Output $(I', |M|)$.

**Fig. 3** General framework.

## 3. Proposed Instance Generators

All three instance generators proposed in this section are based on Gent and Prosser's one [7] described in Section 2.4. Here we explain how we modify their generator.

First, we are not interested in probability distribution of instances. Hence we do not use probabilities $p_1$ and $p_2$ used in Ref. [7]. Instead, we perform deletion of persons or tying persons nondeterministically.

Second, Step 3 of Ref. [7] is not essential so we omit this operation (i.e., we allow an instance with empty lists).

Third, which is essential, we compute a stable matching $M$ after we obtain an SMI instance at Step 2. As all the stable matchings have the same size, $M$ is clearly an optimal solution. After this, we add ties in preference lists. For convenience, we consider a person who is not included in a tie as in a tie of length one. Initially, all the ties are of length one, and we repeatedly and nondeterministically merge two consecutive ties to one. Note that a stable matching before merging ties is also stable after merging, because by merging ties no new blocking pair can arise (this is formally proved in the proof of Lemma 1). Hence $M$ is a stable matching throughout this process. However, by merging ties, an unstable matching may turn stable. To guarantee that $M$ is of maximum cardinality in an output SMTI instance, we have to prevent a matching that is larger than $M$ (which is of course initially unstable) from becoming stable.

A framework of our method is given in **Fig. 3**. Step 5 executes the operation of merging ties. Our three generators only differ in a condition for allowing two ties to be merged.

### 3.1 Algorithm Using a Decision Problem

In our first method, we check if two ties can be merged with-

Algorithm 1

**Step 5**

The following is repeated a certain number of times. Arbitrarily choose a person $p$ and two adjacent ties $t_1$ and $t_2$ of $p$. Using these as inputs, solve IS_SIZE_SAME. If the answer is YES, ties are merged, and if NO, ties are not merged.

**Fig. 4** Step 5 of Algorithm 1.

out changing the optimal solution (the size of a maximum stable matching). If so, then we merge them, otherwise, we do not merge. Consider the following decision problem.

**Problem:** IS_SIZE_SAME
**Input:** SMTI instances $I$ and $\tilde{I}$, where $\tilde{I}$ is the result of merging two adjacent ties on the preference list of a person in $I$.
**Output:** YES if $opt(\tilde{I}) = opt(I)$. Otherwise NO.

Algorithm 1 solves IS_SIZE_SAME before merging adjacent ties, and merge them if and only if the answer is YES. In **Fig. 4**, we describe only Step 5 of Algorithm 1 (as other parts are unchanged from Fig. 3).

If IS_SIZE_SAME can be solved in polynomial time, Algorithm 1 terminates in polynomial time. However, as we will show in Theorem 1, IS_SIZE_SAME is NP-hard. First, we prove the following Lemma 1.

**Lemma 1.** Let $I$ and $\tilde{I}$ be input SMTI instances for IS_SIZE_SAME. Then, $opt(\tilde{I}) \geq opt(I)$ holds.

**Proof.** Let $M^*$ be a maximum stable matching of $I$. It suffices to show that $M^*$ is stable in $\tilde{I}$. Suppose not. Then, there is a blocking pair $(m, w)$ of $M^*$ in $\tilde{I}$. By definition, "$w \succ_m M^*(m)$ or $m$ is single in $M^*$" and "$m \succ_w M^*(w)$ or $w$ is single in $M^*$" hold in $\tilde{I}$. However, by construction of $\tilde{I}$, $I$ is a result of breaking a tie of $\tilde{I}$ in some way, so both relations $w \succ_m M^*(m)$ and $m \succ_w M^*(w)$ also hold in $I$. This implies that $(m, w)$ is a blocking pair of $M^*$ in $I$, contradicting the stability of $M^*$ in $I$. □

By Lemma 1, the answer of NO means that $opt(\tilde{I}) > opt(I)$. Lemma 2 guarantees that $opt(\tilde{I})$ is at most one more than $opt(I)$.

**Lemma 2.** Let $I$ and $\tilde{I}$ be input SMTI instances for IS_SIZE_SAME. Then, $opt(\tilde{I}) \leq opt(I) + 1$ holds.

**Proof.** Without loss of generality, let the person whose ties are merged be a man $m$. Let $M$ be one of the maximum stable matchings of $I$, and $\tilde{M}$ be one of the maximum stable matchings of $\tilde{I}$. That is, $opt(I) = |M|$ and $opt(\tilde{I}) = |\tilde{M}|$.

It is known that there is a way of breaking ties of $\tilde{I}$ so that $\tilde{M}$ is stable in the resulting instance (Lemma 3.1 of Ref. [20]). Let $\tilde{I}_1$ be the SMI instance obtained in this way. Next, let $I_1$ be the SMTI instance obtained from $I$ by breaking ties of all the people, except for the man $m$, in the same way as in $\tilde{I}_1$, and $M_1$ be one of the maximum stable matchings of $I_1$. Since $M_1$ is also a stable matching of $I$, $|M_1| \leq |M|$ holds.

Consider the bipartite graph $G = (V, E)$ with vertex set $V = \{m_1, m_2, \ldots, m_n, w_1, w_2, \ldots, w_n\}$ and edge set $E = \{(m, w) \mid (m, w) \in (M_1 \cup \tilde{M})\}$. This graph represents $M_1$ and $\tilde{M}$ being superimposed. Since the degree of each vertex is at most two, each connected component is an isolated vertex, a path, or a cycle (possibly of length two). In the following, we show that if a path exists, it must contain the man $m$. Then since there can be at most one path, the sizes of $M_1$ and $\tilde{M}$ differ by at most 1, and we have that $|\tilde{M}| \leq |M_1| + 1 \leq |M| + 1$, as desired.

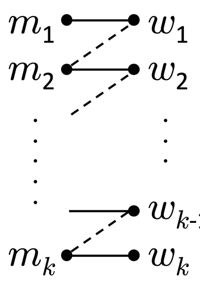

**Fig. 5** A path on the bipartite graph $G = (V, E)$.

Suppose that there exists a path that does not contain $m$. By renaming persons, let $m_1, w_1, m_2, \ldots, m_k, w_k$ be the men and women along this path, where $(m_i, w_i) \in M_1$ for $1 \leq i \leq k$ and $(m_{i+1}, w_i) \in \tilde{M}$ for $1 \leq i \leq k-1$. In **Fig. 5**, solid edges represent the pairs of $M_1$ and dotted edges represent the pairs of $\tilde{M}$. This path starts from a man and ends with a woman, but the following argument holds for any other case. Note that every person in this path has the same strict preference list in $I_1$ and $\tilde{I}_1$.

Since $(m_1, w_1) \in M_1$, $w_1$ is acceptable to $m_1$. If $m_1 \succ_{w_1} m_2$ then $(m_1, w_1)$ is a blocking pair for $\tilde{M}$ in $\tilde{I}_1$, contradicting the stability of $\tilde{M}$ in $\tilde{I}_1$. Hence we have that $m_2 \succ_{w_1} m_1$. If $w_1 \succ_{m_2} w_2$ then $(m_2, w_1)$ is a blocking pair for $M_1$ in $I_1$, contradicting the stability of $M_1$ in $I_1$. Hence we have that $w_2 \succ_{m_2} w_1$. Continuing in this way, we eventually have that $w_k \succ_{m_k} w_{k-1}$. Then $(m_k, w_k)$ is a blocking pair for $\tilde{M}$ in $\tilde{I}_1$ since $m_k$ is acceptable to $w_k$. This is a contradiction and the proof is completed. □

**Theorem 1.** If there exists a deterministic polynomial time algorithm that solves IS_SIZE_SAME, then P=NP.

**Proof.** Suppose there is a deterministic polynomial time algorithm $A$ that solves IS_SIZE_SAME. Given a MAX SMTI instance $I$, construct an algorithm $B$ to solve it as follows. Prepare a variable $x$ and set its initial value to 0. Also, let $I_1 = I$.

If there is a tie of length 2 or more in $I_i$, construct an SMTI instance $I_{i+1}$ by dividing it arbitrarily into two, and give $I_i$ and $I_{i+1}$ to $A$. If $opt(I_{i+1}) \neq opt(I_i)$, increment $x$ by 1. If $opt(I_{i+1}) = opt(I_i)$, do not change the value of $x$. This is repeated as long as the instance includes a tie, and let $I_k$ be the finally obtained SMI instance.

By Lemmas 1 and 2, $opt(I_{i+1}) \neq opt(I_i)$ implies $opt(I_{i+1}) = opt(I_i) - 1$, so $opt(I_k) = opt(I) - x$. Since $I_k$ is an SMI instance, $opt(I_k)$ can be computed in polynomial time. Therefore, $opt(I)$ can be computed in polynomial time, and $B$ is a polynomial time algorithm for finding the optimal solution of MAX SMTI. Since MAX SMTI is NP-hard, P=NP is implied. □

## 3.2 Algorithm Using a Sufficient Condition

Theorem 1 implies that it is hard to precisely determine whether we can merge two ties without changing the optimal solution. One way of avoiding this problem is to rely on a sufficient condition for preserving the optimal solution when merging two ties. In this section, we propose Algorithm 2, which is described in **Fig. 6**. As before, we give only Step 5.

```
Algorithm 2
Step 5
    The following is repeated a certain number of times.
    Arbitrarily choose a person $p$ and two adjacent ties of
    $p$. Let the upper tie be $t_1$, and the lower tie be $t_2$. Do not
    merge these ties if $t_1$ contains $M(p)$. Otherwise merge
    them.
```

**Fig. 6** Step 5 of Algorithm 2.

```
1 : Ⓒ A B        A : ② 1 3 4
2 : Ⓐ B C        B :  2 1 4
3 :  A           C :  4 ① 2
4 :  A Ⓓ C B     D : ④
```

**Fig. 7** An SMI instance and a matching $M$ obtained after Step 4.

```
1 : Ⓒ A B        A : ② 1 3 4
2 : Ⓐ(B C)       B :  2 (1 4)
3 :  A           C :  4 ① 2
4 : (A Ⓓ)C B     D : ④
```

**Fig. 8** A generated SMTI instance.

### 3.2.1 Example

We show an execution of Algorithm 2 using an example of $n = 4$. Let 1, 2, 3, and 4 be men and $A$, $B$, $C$, and $D$ be women. Suppose that after Step 3, we obtained an SMI instance given in **Fig. 7**. Here, each row represents a preference list of each person. For example, in the man 1's preference list, $C$, $A$ and $B$ are the first, the second, and the third choices, respectively.

In Step 4, a stable matching $M$ is obtained. In Fig. 7, the person written inside the circle is the partner in $M$. A person who does not have a circle on the list is single in $M$.

In Step 5, we repeatedly merge two ties (recall that initially all the ties are of length one). Ties that do not include a circle can be merged freely, such as $B$ and $C$ in 2's list or 1 and 4 in $B$'s list. A tie with a circle can be merged with an upper tie, such as $D$ and $A$ in 4's list. However, a tie with a circle cannot be merged with a lower tie, such as $C$ and $A$ in 1's list or 2 and 1 in $A$'s list.

**Figure 8** shows an example instance after executing Step 5.

### 3.2.2 Proof of Correctness

Here we show that Algorithm 2 works correctly.

**Theorem 2.** $opt(I') = |M|$ holds.

**Proof.** To show this, it suffices to show that (1) the matching $M$ obtained in Step 4 is a stable matching of $I'$ and (2) there is no stable matching of $I'$ larger than $M$. (1) can be proved by repeatedly applying the argument in the proof of Lemma 1, so in the rest we prove (2).

Assume that there exists a stable matching $M'$ of $I'$ that is larger than $M$. As in the proof of Lemma 2, we consider the bipartite graph $G = (V, E)$ with vertex set $V = \{m_1, m_2, \ldots, m_n, w_1, w_2, \ldots, w_n\}$ and edge set $E = \{(m, w) \mid (m, w) \in (M \cup M')\}$. Since $|M'| > |M|$, $G$ contains an augmenting path, i.e., a path starting from and ending with edges of $M'$. Let the persons along this path be $m_1, w_1, m_2, \ldots, m_k, w_k$, where

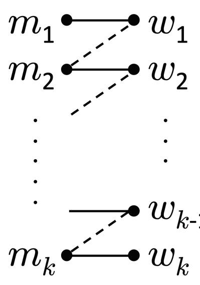

**Fig. 9** A path on the bipartite graph $G = (V, E)$.

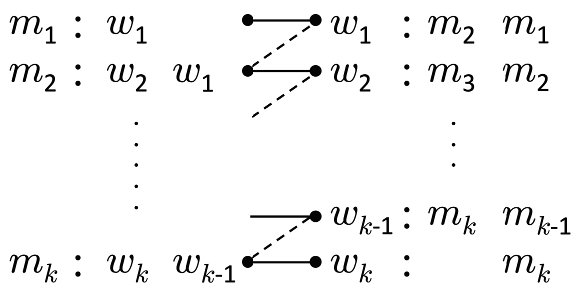

**Fig. 10** Preference lists in $I$ added to Fig. 9.

$M'(m_1) = w_1, M(w_1) = m_2, M'(m_2) = w_2, \ldots, M(w_{k-1}) = m_k$, and $M'(m_k) = w_k$. In **Fig. 9**, the solid edges represent the pairs of $M'$, and the dotted edges represent the pairs of $M$.

If $m_1 \succ_{w_1} m_2$ holds in $I$, then $(m_1, w_1)$ becomes a blocking pair of $M$ in $I$, which contradicts the stability of $M$ in $I$. Since preference lists of $I$ contain no ties, it must be the case that $m_2 \succ_{w_1} m_1$ in $I$. By the rule of Step 5, $M(w_1)$ $(= m_2)$ is not tied with the lower man in $w_1$'s preference list. Hence $m_2$ and $m_1$ are not tied in $I'$ and $m_2 \succ_{w_1} m_1$ also holds in $I'$.

Next, if $w_1 \succ_{m_2} w_2$ holds in $I$, $w_1$ and $w_2$ do not belong to the same tie in $I'$ due to the condition of Step 5, because $M(m_2) = w_1$ cannot belong to the same tie with a lower-ranked woman. Hence $w_1 \succ_{m_2} w_2$ also holds in $I'$. Then, $(m_2, w_1)$ becomes a blocking pair of $M'$ in $I'$, contradicting the stability of $M'$ in $I'$. Thus $w_2 \succ_{m_2} w_1$ holds in $I$.

By a similar argument, we can deduce that $w_i \succ_{m_i} w_{i-1}$ holds in $I$ for $2 \le i \le k$, and $m_{i+1} \succ_{w_i} m_i$ holds in $I$ for $1 \le i \le k-1$. **Figure 10** shows the preference lists in $I$ added to Fig. 9.

Then, $w_k \succ_{m_k} w_{k-1}$ holds in $I$, and $(m_k, w_k)$ becomes a blocking pair for $M$ in $I$, which contradicts the stability of $M$ in $I$. Therefore, we can conclude that there is no stable matching of $I'$ larger than $|M|$. □

### 3.2.3 Proof that the Sizes of Stable Matchings are Equal

We show that all the stable matchings of an output of Algorithm 2 are equal.

**Theorem 3.** All the stable matchings of $I'$ are of the same size.

**Proof.** Let $S = \{I_1, I_2, \ldots, I_k\}$ be the set consisting of all the SMI instances obtained by breaking all the ties of $I'$. When merging ties to form $I'$ from $I$ in Step 5, in each person $p$'s preference list, $M(p)$ is not tied with a lower ranked person. Hence if $M(p) \succ_p q$ holds in $I$, it also holds in $I'$, so $M(p) \succ_p q$ holds in any $I'' \in S$. Thus $M$ is stable in all the SMI instances of $S$.

Let $M'$ be an arbitrary stable matching of $I'$. Then, by Lemma 3.1 of Ref. [20], $M'$ is stable in at least one SMI instance

Algorithm 3

**Step 5**

The following is repeated a certain number of times. Arbitrarily choose a person $p$ and two adjacent ties of $p$. Let the upper tie be $t_1$, and the lower tie be $t_2$. Do not merge these ties if $t_1$ contains $M(p)$ and $t_2$ contains $r$ who is single in $M$ or $p \succeq_r M(r)$ in the current list. Otherwise merge them.

**Fig. 11** Step 5 of Algorithm 3.

| | |
|---|---|
| 1 : Ⓒ A B | A : ② 1 3 4 |
| 2 : Ⓐ B C | B : 2 1 4 |
| 3 : A | C : 4 ① 2 |
| 4 : A Ⓓ C B | D : ④ |

**Fig. 12** An SMI instance and a matching $M$ obtained after Step 4.

| | |
|---|---|
| 1 : (Ⓒ A) B | A : ② 1 3 4 |
| 2 : Ⓐ(B C) | B : 2 (1 4) |
| 3 : A | C : 4 (① 2) |
| 4 : (A Ⓓ) C B | D : ④ |

**Fig. 13** A generated SMTI instance.

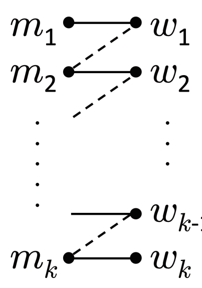

**Fig. 14** A path on the bipartite graph $G = (V, E)$.

of $S$, say $I_i$. As discussed above, $M$ is also stable in $I_i$. Since in SMI the sizes of all the stable matchings are equal, $|M| = |M'|$ holds. Therefore, the size of any stable matching of $I'$ is $|M|$. □

### 3.3 Algorithm Using a Relaxed Sufficient Condition

Theorem 3 implies that the condition of Algorithm 2 at Step 5 is too strong. In this section, we propose Algorithm 3 in **Fig. 11**, which relaxes the condition in Step 5 of Algorithm 2.

Recall that, in Algorithm 2, ties $t_1$ and $t_2$ cannot be merged if $t_1$ contains $M(p)$. However, in Algorithm 3, there is additional condition. Hence, ties allowed to be merged in Algorithm 2 are also allowed in Algorithm 3. As will be shown in the next section, it can happen that ties not allowed to be merged in Algorithm 2 can be merged in Algorithm 3. Thus Algorithm 3 is actually a relaxation of Algorithm 2.

#### 3.3.1 Example

We show an execution of Algorithm 3 using the same example as in Section 3.2.1, which is given in **Fig. 12**.

Let the person $p$ in Step 5 be the man 1 and his upper tie $t_1$ consists of the woman $C$ and a lower tie consists of the woman $A$. Recall that in Algorithm 2, $t_1$ and $t_2$ are not allowed to be merged. However, this time we can merge them because the only woman $A$ in $t_2$ neither is single nor satisfies $1 \succeq_A 2$ in the current list. For the same reason, 2 and 1 in $A$'s list can be merged.

However, $D$ and $C$ in 4's list cannot be merged because $4 \succeq_C 1$ holds. ($p$, $r$, and $M(r)$ in Step 5 correspond to the man 4, the woman $C$, and the man 1, respectively). **Figure 13** shows a possible output of Algorithm 3, which Algorithm 2 cannot produce.

It should be noted that men 2 and 1 in $A$'s list can be merged in Fig. 12, but it is no longer possible in Fig. 13 because $C$ and $A$ are now tied in 1's list.

#### 3.3.2 Proof of Correctness

In this section, we show that the output of Algorithm 3 is correct.

**Theorem 4.** $opt(I') = |M|$ holds.

**Proof.** As in the proof of Theorem 2, the matching $M$ obtained in Step 4 is a stable matching of $I'$, so we show that there is no stable matching of $I'$ larger than $M$. This part goes like that of Theorem 2, with a bit more sophisticated analysis. Assume that there exists a stable matching $M'$ of $I'$ that is larger than $M$. Then, on a bipartite graph with vertex set $V = \{m_1, m_2, \ldots, m_n, w_1, w_2, \ldots, w_n\}$ and edge set $E = \{(m, w) \mid (m, w) \in (M \cup M')\}$, there exists an augmenting path starting from a man $m_1$ who is single in $M$ but matched in $M'$, and ending with a woman $w_k$ who is single in $M$ but matched in $M'$. Hence $(m_i, w_i) \in M'$ for $1 \le i \le k$ and $(m_{i+1}, w_i) \in M'$ for $1 \le i \le k-1$. In **Fig. 14**, the solid edges represent the pairs of $M'$, and the dotted edges represent the pairs of $M$.

If $m_1 \succ_{w_1} m_2$ holds in $I$, then $(m_1, w_1)$ becomes a blocking pair of $M$ in $I$, which contradicts the stability of $M$. Hence $m_2 \succ_{w_1} m_1$ holds in $I$, and by construction of $I'$, $m_2 \succeq_{w_1} m_1$ holds in $I'$. Since $m_1$ is single in $M$, $m_2 =_{w_1} m_1$ does not hold in $I'$ by the condition of Step 5. Hence $m_2 \succ_{w_1} m_1$ holds in $I'$.

Next, if $w_1 \succ_{m_2} w_2$ holds in $I'$, then $(m_2, w_1)$ becomes a blocking pair of $M'$ in $I'$, which contradicts the stability of $M'$ in $I'$. Therefore, $w_2 \succeq_{m_2} w_1$ holds in $I'$.

If $w_2 \succ_{m_2} w_1$ holds in $I'$, $w_2 \succ_{m_2} w_1$ also holds in $I$. If $w_1 =_{m_2} w_2$ holds in $I'$, then either $w_1 \succ_{m_2} w_2$ or $w_2 \succ_{m_2} w_1$ may hold in $I$. Hence it suffices to consider the following two cases:

**(1) $w_2 \succ_{m_2} w_1$ holds in $I$**

If $m_2 \succ_{w_2} m_3$ holds in $I$, then $(m_2, w_2)$ is a blocking pair for $M$ in $I$. So $m_3 \succ_{w_2} m_2$ holds in $I$. Since $w_2 \succ_{m_2} w_1$ holds in $I$, the tie containing $m_3$ and the tie containing $m_2$ are not merged due to the condition of Step 5 (by identifying $p = w_2$, $M(p) = m_3$, and $r = m_2$ in Step 5). So $m_3 \succ_{w_2} m_2$ holds in $I'$.

**(2) $w_1 \succ_{m_2} w_2$ holds in $I$ and $w_1 =_{m_2} w_2$ holds in $I'$**

Suppose that $m_2 \succ_{w_2} m_3$ holds in $I$. Since $M(m_2) = w_1$ and $w_1 \succ_{m_2} w_2$ holds in $I$, by the condition of Step 5, $w_1$ and $w_2$ are not in the same tie. This contradicts $w_1 =_{m_2} w_2$, so $m_3 \succ_{w_2} m_2$ in $I$. Note that $w_1 =_{m_2} w_2$ in $I'$ by the condition of this case. When $w_1$ and $w_2$ are put into the same tie, it must be the case that $m_3 \succ_{w_2} m_2$ as otherwise, the condition of Step 5 prohibits merging the tie containing $w_1$ and the tie containing $w_2$. After this, the relation $m_3 \succ_{w_2} m_2$ is maintained by the condition of Step 5; since

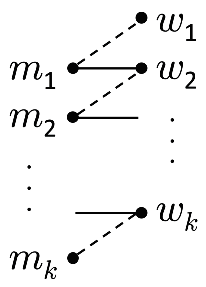

**Fig. 15** A path on the bipartite graph $G = (V, E)$.

$w_1 =_{m_2} w_2$, the tie containing $m_3$ and the tie containing $m_2$ will not be merged. Therefore $m_3 \succ_{w_2} m_2$ holds in $I'$.

Hence, in either of the two cases (1) and (2), $m_3 \succ_{w_2} m_2$ holds in $I'$. By repeating the same argument along this path, we have that $w_{i+1} \succeq_{m_{i+1}} w_i$ in $I'$ and $m_{i+1} \succ_{w_i} m_i$ in $I'$, and eventually we have that $w_k \succeq_{m_k} w_{k-1}$ in $I'$. This means that $w_k \succ_{m_k} w_{k-1}$ in $I$. Then, $(m_k, w_k)$ is a blocking pair for $M$ in $I$, which contradicts the stability of $M$. Therefore, there is no stable matching of $I'$ larger than $|M|$. □

#### 3.3.3 Proof that the Sizes of Stable Matchings are Equal

**Theorem 5.** All the stable matchings of $I'$ are of the same size.

**Proof.** By Theorem 4, there is no stable matching $M'$ of $I'$ such that $|M'| > |M|$. Assume that there exists a stable matching $M'$ of $I'$ such that $|M'| < |M|$. Then, on the bipartite graph with vertex set $V = \{m_1, m_2, \ldots, m_n, w_1, w_2, \ldots, w_n\}$ and edge set $E = \{(m, w) \mid (m, w) \in (M \cup M')\}$, there exists a path starting from a woman $w_1$ who is matched in $M$ but is single in $M'$, and ending with a man $m_k$ who is matched in $M$ but is single in $M'$. Hence $M(w_1) = m_1, M'(m_1) = w_2, M(w_2) = m_2, \ldots, M'(m_{k-1}) = w_k$, and $M(w_k) = m_k$. In **Fig. 15**, the solid edges represent the pairs of $M'$, and the dotted edges represent the pairs of $M$.

As a starting point, $w_2 \succeq_{m_1} w_1$ holds in $I'$, as otherwise, $(m_1, w_1)$ is a blocking pair for $M'$ in $I'$. By applying the same argument as the proof of Theorem 4, we have that $m_k \succ_{w_k} m_{k-1}$ in $I'$. Then, $(m_k, w_k)$ becomes a blocking pair of $M'$, which contradicts the stability of $M'$. Therefore, all the stable matchings of $I'$ have the same size $|M|$. □

## 4. Conclusion

In this paper, we have investigated possibilities of constructing a polynomial time instance generator for an NP-hard problem MAX SMTI. We require a generator to output the optimal value, as well as an instance itself.

By following an argument of Refs. [19], [30], we first observed that if there exists an instance generator that can produce any input, then NP=coNP holds. Hence we focused on constructing generators that can produce a subset of instances.

We have proposed three generators based on Gent and Prosser's [7]. In all of them, we first construct an SMI instance (that has no tie) and find a candidate optimal solution $M$. After that, we introduce ties while keeping $M$ to be an optimal solution. The three generators differ in the conditions to guarantee this property when merging ties. In the first generator, the condition is too general to the extent that the generator cannot be expected to run in polynomial time, while in the second and the third ones, the conditions are too strong to the extent that the sizes of stable matchings in an output instance are constant and hence such instances are far from useful for empirically evaluating algorithms.

One possible future direction is to further relax the condition of Step 5 of Algorithm 3. Another line of research is to consider a new generation method whose base is different than Gent and Prosser's [7].

**Acknowledgments** This work was supported by JSPS KAKENHI Grant Numbers JP16K00017 and JP20K11677. The authors would like to thank the anonymous reviewers for their comments for pointing errors and typos in the submitted version, which improved the rigorousness of the paper.

## References

[1] Huang, C.-C. and Kavitha, T.: Improved approximation algorithms for two variants of the stable marriage problem with ties, *Mathematical Programming*, Vol.154, No.1-2, pp.353–380 (2015).
[2] Gale, D. and Shapley, L.S.: College admissions and the stability of marriage, *The American Mathematical Monthly*, Vol.120, No.5, pp.386–391 (2013).
[3] Gale, D. and Sotomayor, M.: Some remarks on the stable matching problem, *Discrete Applied Mathematics*, Vol.11, pp.223–232 (1985).
[4] Gelain, M., Pini, M.S., Rossi, F., Venable, K.B. and Walsh, T.: Local search algorithms on the stable marriage problem: Experimental studies, *Proc. ECAI'10*, pp.1085–1086 (2010).
[5] Gelain, M., Pini, M.S., Rossi, F., Venable, K.B. and Walsh, T.: Local search approaches in stable matching problems, Vol.6, pp.591–617 (2013).
[6] Gent, I.P., Prosser, P., Smith, B. and Walsh, T.: SAT encodings of the stable marriage problem with ties and incomplete lists, *SAT 2002*, pp.133–140 (2002).
[7] Gent, I.P. and Prosser, P.: An empirical study of the stable marriage problem with ties and incomplete lists, *Proc. ECAI'02, IOS Press*, pp.141–145 (2002).
[8] Gusfield, D. and Irving, R.W.: *The Stable Marriage Problem: Structure and Algorithms*, MIT Press, Cambridge, MA, USA (1989).
[9] Halldórsson, M.M., Iwama, K., Miyazaki, S. and Yanagisawa, H.: Improved approximation results for the stable marriage problem, *ACM Transactions on Algorithms*, Vol.3, No.3, p.30 (2007).
[10] Irving, R.W. and Manlove, D.F.: Approximation algorithms for hard variants of the stable marriage and hospitals/residents problems, *Journal of Combinatorial Optimization*, Vol.16, No.3, pp.279–292 (2008).
[11] Irving, R.W. and Manlove, D.F.: Finding large stable matchings, *J. Exp. Algorithmics*, Vol.14, pp.2:1.2–2:1.30 (2010).
[12] Iwama, K., Manlove, D., Miyazaki, S. and Morita, Y.: Stable marriage with incomplete lists and ties, *Proc. ICALP'99*, pp.443–452 (1999).
[13] Iwama, K., Miyazaki, S. and Yamauchi, N.: A 1.875-approximation algorithm for the stable marriage problem, *Proc. 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, *SODA '07*, pp.288–297 (2007).
[14] Iwama, K., Miyazaki, S. and Yanagisawa, H.: A 25/17-approximation algorithm for the stable marriage problem with one-sided ties, *Algorithmica*, Vol.68, No.3 (2014).
[15] Király, Z.: Better and simpler approximation algorithms for the stable marriage problem, *Algorithmica*, Vol.60, No.1, pp.3–20 (2011).
[16] Király, Z.: Linear time local approximation algorithm for maximum stable marriage, *Algorithms*, Vol.6, No.3, pp.471–484 (2013).
[17] Kwanashie, A. and Manlove, D.F.: An integer programming approach to the hospitals/residents problem with ties, *Operations Research Proc. 2013*, Huisman, D., Louwerse, I. and Wagelmans, A.P. (Eds.), pp.263–269, Springer International Publishing (2014).
[18] Lam, C.-K. and Plaxton C.G.: A $(1 + 1/e)$-approximation algorithm for maximum stable matching with one-sided ties and incomplete lists, *Proc. 30th Annual ACM-SIAM Symposium on Discrete Algorithms*, *SODA 2019*, pp.2823-2840 (2019).
[19] Leggett, E.W. Jr. and Moore, D.J.: Optimization problems and the polynomial hierarchy, *Theoretical Computer Science*, Vol.15, pp.279–289 (1981).
[20] Manlove, D.F.: Algorithmics of Matching under Preferences, World Scientific, Singapore (2013).
[21] Manlove, D.F., Irving, R.W., Iwama, K., Miyazaki, S. and Morita, Y.: Hard variants of stable marriage, *Theor. Comput. Sci.*, Vol.276, No.1-2, pp.261–279 (2002).

[22] Manlove, D.F., O'Malley, G., Prosser, P. and Unsworth, C.: A constraint programming approach to the hospitals/residents problem, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, Van Hentenryck, P. and Wolsey, L. (Eds.), pp.155–170, Springer Berlin Heidelberg (2007).
[23] McDermid, E.: A 3/2-approximation algorithm for general stable marriage, *Automata, Languages and Programming*, Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S. and Thomas, W. (Eds.), *ICALP 2009*, pp.689–700, Springer Berlin Heidelberg (2009).
[24] Munera, D., Diaz, D., Abreu, S., Rossi, F., Saraswat, V. and Codognet, P.: A local search algorithm for SMTI and its extension to HRT problems, *3rd International Workshop on Matching Under Preferences*, Glasgow, United Kingdom, University of Glasgow (2015).
[25] Munera, D., Diaz, D., Abreu, S., Rossi, F., Saraswat, V. and Codognet, P.: Solving hard stable matching problems via local search and cooperative parallelization, *Proc. 29th AAAI Conference on Artificial Intelligence*, *AAAI'15*, pp.1212–1218, AAAI Press (2015).
[26] Paluch, K.: Faster and simpler approximation of stable matchings, *Algorithms*, Vol.7, No.2, pp.189–202 (2014).
[27] Podhradský, A.: Stable marriage problem algorithms, Faculty of informatics, Masaryk University (2010), available from ⟨https://is.muni.cz/th/172646/fi_m⟩.
[28] Roth, A.E.: The evolution of the labor market for medical interns and residents: A case study in game theory, *Journal of Political Economy*, Vol.92, No.6, pp.991–1016 (1984).
[29] Roth, A.E.: On the allocation of residents to rural hospitals: A general property of two-sided matching markets, *Econometrica*, Vol.54, No.2, pp.425–27 (1986).
[30] Sanchis, L.A.: On the complexity of test case generation for NP-hard problems, *Inf. Process. Lett.*, Vol.36, pp.135–140 (1990).

**Yuki Matsuyama** was born in 1996. He received Bachelor of Engineering and Master of Informatics degrees from Kyoto University in 2018 and 2020, respectively.

**Shuichi Miyazaki** is an associate professor at Academic Center for Computing and Media Studies, Kyoto University. He received B.E., M.E., and Ph.D. degrees from Kyushu University in 1993, 1995 and 1998, respectively. He is a member of IPSJ, IEICE, ACM, and EATCS. His research interests include discrete algorithms and computational complexity theory.