Achieving Private Verification in Multi-stakeholder Environment and Application to Stable Matching

Toru Nakamura[a], Hiroki Okada[b], Kazuhide Fukushima and Takamasa Isohara
KDDI Research, Inc., 2-1-15, Ohara, Fujimino-shi, Saitama-ken, Japan

Keywords: Verification, Privacy, Multi-Stakeholder Environment.

Abstract: Services to support decision-making, such as resource allocation and recommendations, are becoming popular and essential. This paper focuses on two-sided matching as a form of decision-making support. The stable marriage problem has been thoroughly studied as an exciting research topic related to two-sided matching. Stability is a property in which there is no man and woman who would agree to leave their assigned partner, and this property is recognized as an ideal condition for participants. This paper assumes a system where participants provide their preference orders to an assignee, and the assignee provides them with a stable matching. When considering a multi-stakeholder environment, not only the participants' requirements but also the assignee's intention should be respected. That is, the assignee should be given the discretion to select the matching which is the best for the assignee among all the stable matchings. However, there is a possibility that if the assignee is malicious, he/she falsifies and provides an unstable matching in order to maximize his/her benefit with ignoring the participants' requirements. It is difficult for the participants to detect it if they want to keep their preference orders secret from others. This paper proposes a solution of protocol including a private verification algorithm to judge whether the received matching is stable while keeping their preference orders private. The proposed protocol is based on fully homomorphic encryption (FHE) and assumes the use of a semi-honest third-party server. This paper also proposes a general solution that does not limit to specific requirements from participants.

1 INTRODUCTION

Services to support decision-making, such as resource allocation and recommendation, are becoming popular and essential. The stable marriage problem (Gale and Shapley, 1962) has been throughly studied as a research topic related to decision-making. Informally, a stable matching is a one-to-one pairing of a set of men to a set of women, containing no man and woman who would agree to leave their assigned partners. This paper assumes a system where participants provide preference orders to an assignee, and the assignee provides a stable matching. When discussing stable matching, we only discuss whether the participants' requirements are satisfied while never considering the intention of the assignee. On the other hand, in the context of the recommendation system, discussion has begun on the multi-stakeholder environment, where not only end-users' requirements but also those of other stakeholders such as item providers and system operators are considered (Abdollahpouri et al., 2020). In this paper, we discuss the issues involved in providing stable matchings in the multi-stakeholder environment.

For example, efficient, stable matching algorithms are used to assign graduating medical students to residency programs at hospitals in the real-world[1]. In this case, participants, that is, students and hospitals, provide their preference orders to an assignee and expect to receive a stable matching which is decided based on their preference orders. Here, the assignee may have another motivation beyond just providing a stable matching, for example, solving the issue of how to reduce labor shortages in rural areas. When respecting the assignee's motivation, an assignee should be given the discretion to select the matching which is the best for the assignee among all the stable matchings. However, there is a possibility that if the assignee is malicious, he/she falsifies and provides an unstable matching in order to maximize his/her own

[a] https://orcid.org/0000-0003-2530-648X
[b] https://orcid.org/0000-0002-5687-620X

[1] National Resident Matching Program (NRMP). http://www.nrmp.org/

Nakamura, T., Okada, H., Fukushima, K. and Isohara, T.
Achieving Private Verification in Multi-stakeholder Environment and Application to Stable Matching.
DOI: 10.5220/0011995800003467
In *Proceedings of the 25th International Conference on Enterprise Information Systems (ICEIS 2023) - Volume 1*, pages 768-775
ISBN: 978-989-758-648-4; ISSN: 2184-4992

benefit. Though participants can easily detect such falsification if they can share their preference order, it may be difficult because of privacy concerns.

There is existing work on private stable matching algorithms, first introduced by Golle (Golle, 2006), based on the Gale-Shapley algorithm with secure multiparty computation (MPC). These proposals can achieve participants' privacy from other participants and matching authorities. There is no assignee in this setting; participants and matching authorities collaborate to execute pre-defined matching algorithms such as the Gale-Shapley algorithm. However, this setting may be unrealistic in a business context; naturally, an assignee would seek benefit commensurate with the effort of attracting participants to the matching service.

1.1 Contributions

The contribution of this paper is to propose a protocol that satisfies the following requirements. More generally, we will use "user," "private input," and "server" instead of "participant," "preference order," and "assignee," respectively, in the rest of the paper.

1. Verifiability: Users can accept a result obtained from the server if and only if the result satisfies the users' requirements.
2. Privacy: The adversary can not obtain any information about the private inputs of the users.
3. Selectability: The server can freely choose a result from among all the results candidates.

We propose a protocol that satisfies the three requirements, verifiability, privacy, and selectability, with a semi-honest third-party server, called a verifier and fully homomorphic encryption (FHE). FHE is cryptography that allows the operation of any functions for encrypted data. We first show a general construction of this protocol which does not limit specific requirements from users but assumes that the verification algorithm of the requirement is available on FHE. Furthermore, we show a specific implementation of an FHE-based verification algorithm for two-sided matching that checks whether matching is stable while users' private inputs kept secret.

Note that we do not consider the server to be an adversary to privacy requirements in this paper; therefore, we do not regard the server knowing the private inputs of the users as a privacy issue. Even if we can exclude the server as a potential adversary of the privacy requirement, it is still worth discussing even in this problem setting because no simple solution satisfies all of the above requirements.

HEREHEREHERE

1.2 Organization

The remainder of the paper is organized as follows. Section 2 describes the existing work related to this paper. Section 3 provides the mathematical model for our protocol and defines the requirements of the protocol. Section 4 describes the proposed general construction of the protocol. Section 5 provides the specific implementation of our solution for the stable matching problem. Section 6 presents the discussion, and Section 7 concludes this paper.

2 RELATED WORK

Homomorphic encryption (HE) is a type of cryptography that allows a third party (e.g., cloud service provider) to perform some mathematical operations on encrypted data without compromising the encryption (Acar et al., 2018). In particular, fully homomorphic encryption (FHE), which can perform arbitrary operations an arbitrary number of times, has been actively studied. Its practicality has been improved significantly since Gentry (Gentry, 2009) first proposed FHE. In this paper, we propose methods based on FHE.

Verifiable computation is a method that enables a user to offload the computation of a function π to another untrusted party. The user can verify whether the received result is truly $\pi(x)$ at a lower cost than executing $\pi(x)$ by the user himself (Parno et al., 2013). However, this technique cannot solve the issue addressed in this paper because users cannot know π in advance to realize selectability.

The stable marriage problem is a well-known problem; informally, a stable matching is a one-to-one pairing of a set of men to a set of women, containing no man and woman who would agree to leave their assigned partners. Gale and Shapley proposed an efficient algorithm to find male-dominated (or female-dominated) stable matching (Gale and Shapley, 1962). Golle proposed the first private stable matching algorithm based on the Gale-Shapley algorithm, where if a majority of matching authorities are honest, the protocol correctly outputs a stable matching and reveals no other information than what can be learned from that match and the preferences of participants controlled by the adversary (Golle, 2006). Franklin et al. pointed out that Golle's algorithm includes a failure in the communication cost and proposed a modified version of the algorithm based on threshold additive homomorphic encryption (TAHE) and secret sharing (SS) (Franklin et al., 2007). Franklin et al. also proposed two more ef-

fective algorithms; one is based on private information retrieval (PIR) in addition to TAHE and SS, and the other is based on secure multiparty computation (MPC) and the Naor-Nissim protocol. Teruya and Sakuma proposed a more effective algorithm based solely on TAHE (Teruya and Sakuma, 2015). Doerner et al. (Doerner et al., 2016) and Riazi et al. (Riazi et al., 2017) proposed practical private stable matching with Oblivious RAM (ORAM). This technique is effective if the algorithm to find a specific matching is determined in advance without considering the existence of a server that manages participants. However, this technique is not suitable for the problem setting of this paper because the requirement in this paper is selectivity, which guarantees the freedom to choose among the matches that satisfy stability for the server.

3 MODEL AND REQUIREMENTS

In this section, we define the mathematical model and the requirements for the discussion in this paper. In addition, the issues of existing methods are summarized from the above viewpoints. The mathematical model assumes applications that can calculate results from only user inputs.

3.1 Model

Let n be the number of users. Each user has his/her private inputs $x_1, \ldots x_n$, respectively, where user i's input is $x_i \in X$. We assume that the users require a server S to provide a result $y \in Y$ for inputs $x_1, \ldots, x_n$ that satisfies a requirement L, where Y is the set of possible ranges of y. L can be represented by $L \subseteq X \times \cdots \times X \times Y$, that is, the requirement of users is to get a result y such that $(x_1, \ldots x_n, y) \in L$. We assume that a server S chooses and provides y to the users. The server is expected to return y such that $(x_1, \ldots, x_n, y) \in L$ from the users; however, if the server is malicious, it may choose y' that is $(x_1, \ldots, x_n, y') \notin L$.

Users communicate and execute a protocol with the server, and in some cases a third-party server, and obtain a result from the server. We assume that there exists an efficient verification algorithm V, that outputs 1 if $(x_1, \ldots, x_n, y) \in L$, otherwise, outputs 0.

3.2 Requirements

We show the requirements of our proposed protocol as follows.

1. Verifiability: the users can accept the result y for private inputs $x_1, \ldots x_n$ if and only if $(x_1, \ldots x_n, y) \in L$.
2. Privacy: the protocol reveals no information about users' private inputs to the adversary.
3. Selectability: The server S can choose any $y \in Y$. In other words, server S does not need to fix the algorithm to be executed in advance.

In this paper, we do not consider the server as an adversary for privacy requirements; therefore, the server is allowed to know the private inputs of users, i.e., we do not regard that as a privacy issue. In the next section, we will propose a protocol with an additional third-party verification server, called verifier. Therefore, it is necessary to regard the other users and the verifier as potential adversaries for the privacy requirement in this case. We also consider the assumed adversaries as semi-honest (passive) adversaries. This setting is not designed to achieve the most desirable level of privacy, that is, this is a weaker assumption than that of a malicious (active) adversary. Nevertheless, this study is still worthwhile because the semi-honest adversary model is generally considered a reasonable assumption. The most important point of this paper is to discuss protocols that simultaneously realize verifiability, privacy, and selectability. Achieving stricter privacy requirements will be subject of our future work.

3.3 Issues of Naive Methods

In this section, we introduce two naive protocols that do not require any novelty without the assumption of a secure communication channel, and describe their issues.

3.3.1 Naive 1: Not Shared among Users

STEP 1. Each user i sends $x_i \in X$ to the server S via a secure channel.

STEP 2. S selects any $y \in Y$ and sends it to each user.

STEP 3. Each user accepts y.

The naive 1 method satisfies privacy and selectability, however, it does not satisfy verifiability because the users accept any y even if $(x_1, \ldots x_n, y) \notin L$.

3.3.2 Naive 2: Shared among Users

STEP 1. Each user i sends $x_i \in X$ to the server S via a secure channel.

STEP 2. S selects any y satisfying $(x_1, \ldots x_n, y) \in L$ and sends it to each user.

STEP 3. Each user i shares x_i with other users and executes $r \leftarrow V(x_1, \ldots x_n, y)$. i accepts y if $r = 1$, otherwise, rejects y.

The naive 2 method satisfies verifiability and selectability, however, it does not satisfy privacy because the users share their private inputs.

4 GENERAL CONSTRUCTION

In this section, we show the general construction of a protocol that simultaneously satisfies verifiability, privacy, and selectivity with a semi-honest verifier and FHE. The general construction does not limit any specific requirement from users but assumes that the verification algorithm of the requirement is available on FHE.

4.1 Fully Homomorphic Encryption (FHE)

FHE can perform arbitrary operations for an arbitrary number of times, We show the definition of FHE (Brakerski and Vaikuntanathan, 2011)(Acar et al., 2018) as follows.

Definition 1. *Fully Homomorphic Encryption* FHE *is a quintet of algorithms* $(\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *which satisfies authenticity, compactness, and security.*

- ***Setup*** $\mathsf{params} \leftarrow \mathsf{Setup}(1^\lambda)$*: outputs a public parameter* param.
- ***Key Generation*** $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Keygen}(\mathsf{param})$*: outputs public encryption key* pk *and secret decryption key* sk.
- ***Encryption*** $c \leftarrow \mathsf{Enc}(\mathsf{pk}, \mu)$*: encrypts a message* $\mu \in \{0,1\}$ *with* pk *and outputs a cyphertext c.*
- ***Decryption*** $\mu \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$*: decrypts cyphertext c with* sk *and outputs* μ.
- ***Homomorphic Evaluation*** $\hat{c} \leftarrow \mathsf{Eval}(\mathcal{C}, (c_1, \ldots c_\ell), \mathsf{pk})$*: computes any circuit* $\mathcal{C} : \{0,1\}^\ell \rightarrow \{0,1\}$ *to* $c_1, \ldots c_\ell$ *with* pk *and outputs* $\hat{c}$.

Authenticity is the property that any ciphertext obtained by the encryption algorithm and the homomorphic evaluation algorithm can be decrypted correctly. Compactness is the property that the size of a ciphertext after homomorphic evaluation does not depend on the size of the circuit in which an operation is performed. Security is the property that the attack success rate is negligibly small for any given attacker. See (Brakerski and Vaikuntanathan, 2011)(Acar et al., 2018) for the details of definitions.

Obviously, it is possible to construct a fully homomorphic encryption for any algorithm, if it is possible to homomorphically compute any circuit.

Therefore, we redefine the FHE as follows. Hereinafter, when we refer to the FHE, the following definition will be applied.

Definition 2. *Fully Homomorphic Encryption* FHE *is a quintet of algorithms* $(\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *which satisfies authenticity, compactness, and security.*

- ***Setup*** $\mathsf{params} \leftarrow \mathsf{Setup}(1^\lambda)$*: outputs a public parameter* param.
- ***Key Generation*** $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Keygen}(\mathsf{param})$*: outputs public encryption key* pk *and secret decryption key* sk.
- ***Encryption*** $c \leftarrow \mathsf{Enc}(\mathsf{pk}, \mu)$*: encrypts a message* $\mu \in X$ *with* pk *and outputs a cyphertext c.*
- ***Decryption*** $\mu \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$*: decrypts cyphertext c with* sk *and outputs* μ.
- ***Homomorphic Evaluation*** $\hat{c} \leftarrow \mathsf{Eval}(\mathcal{A}, (c_1, \ldots c_n), d, \mathsf{pk})$*: computes any algorithm* $\mathcal{A} : X \times \cdots \times X \times Y \rightarrow \{0,1\}$ *with ciphertexts* $c_1, \ldots c_n$*, and d, and public encryption key* pk *and outputs* $\hat{c}$.

If for $1 \leq i \leq n$, $c_i^1 \leftarrow \mathsf{Enc}(\mathsf{pk}, x_i)$, $c^2 \leftarrow \mathsf{Enc}(\mathsf{pk}, y)$, $\hat{c} \leftarrow \mathsf{Eval}(\mathcal{A}, (c_1^1, \ldots c_n^1), c^2, \mathsf{pk})$, then $\mathsf{Dec}(\mathsf{sk}, \hat{c}) = \mathcal{A}((x_1, \ldots x_n), y)$ from the property of authenticity.

4.2 Proposed FHE-based Protocol

Here, we show a protocol which satisfies all the requirements in the general setting. We assume that there exists a verification algorithm V that outputs 1 if $(x_1, \ldots, x_n, y) \in L$, otherwise, outputs 0, and the homomorphic evaluation algorithm on V is available.

STEP 1. Let $\bar{i} \in \{1, \ldots n\}$ be the representative of users. $\bar{i}$ executes $\mathsf{Setup}(1^\lambda)$, obtains param, and publishes it.

STEP 2. $\bar{i}$ executes $\mathsf{Keygen}(\mathsf{param})$ and obtains $(\mathsf{pk}, \mathsf{sk})$. $\bar{i}$ publishes pk and distributes sk to all users.

STEP 3. Each user i sends $x_i \in X$ to the server S via a secure channel.

STEP 4. S selects any $y \in Y$ and sends it to each user.

STEP 5. Each user i sends $c_i^1 \leftarrow \mathsf{Enc}(\mathsf{pk}, x_i)$ and y (practically all it takes is one user (e.g. $\bar{i}$) sends $c^2 \leftarrow \mathsf{Enc}(\mathsf{pk}, y)$) to the verifier D via a secure communication channel.

STEP 6. D executes $r' \leftarrow \mathsf{Eval}(V, (c_1, \ldots, c_n), c^2, \mathsf{pk})$ and sends r' to each user.

STEP 7. Each user i executes $r \leftarrow \mathsf{Dec}(r', \mathsf{sk})$ and i accepts y if $r = 1$, otherwise, i rejects y.

Verifiability is achieved because each user can obtain $r = V(x_1, \ldots, x_n, y)$ by the property of authenticity in FHE. Selectivity is achieved because S can select any $y \in Y$. Regarding privacy, as mentioned in the previous section, we assume that the adversary is semi-honest, that is, passive, hence what the adversary can obtain is at most $\mathsf{pk}, \{c_1^1, \ldots, c_n^1\}, c^2$. The adversary can not obtain any information from this because of the property of security in FHE.

5 APPLICATION TO STABLE MATCHING

In this section, we show a specific implementation of a protocol including an FHE-based verification algorithm for stable matching. Although FHE can theoretically be used to compute any algorithms with encrypted data, the implementation methods are non-trivial. We focus on the case of the stable matching and discuss the method for implementing the verification algorithm on FHE and its efficiency.

5.1 Definition of Stable Matching

The stable marriage problem is well known as the problem of finding stable matching, as shown in (Gale and Shapley, 1962). Informally, a stable matching is a one-to-one pairing of a set of men to a set of women, containing no man and woman who would agree to leave their assigned partners.

An instance of a matching consists of two disjoint sets where the number of elements is n, and set M is men and the set W is women. Each person has a strict total order, called the *preference order*, on members on the other side. A *matching* $\mathcal{M}$ is a pair of M and W, that is, $\mathcal{M} \subseteq M \times W$. If $(m,w) \in \mathcal{M}$, we denote $\mathcal{M}(m) = w, \mathcal{M}(w) = m$. The rank of w in m's preference order is denoted by $P_m(w)$, and it is called w' *preference rank* in m. In the same way, m' preference rank in w is denoted by $P_w(m)$. If a person p prefers a to b in an instance I, we denote $a \succ_p b$ in I.

Definition 3. *Given a matching $\mathcal{M}$, a pair of a man and a woman (m,w) is a blocking pair for $\mathcal{M}$ if all the following conditions are met;*

1. $(m', w') \in \mathcal{M}$ *and* $m' \in M, w' \in W$,
2. $w' \succ_{m'} \mathcal{M}(m')$,
3. $m' \succ_{w'} \mathcal{M}(w')$.

Definition 4. *A matching $\mathcal{M}$ is stable if $\mathcal{M}$ has no blocking pair.*

5.2 Stability Checking Algorithm

Various studies have been conducted on algorithms to find stable matching. Gale and Shapley proved that at least one stable matching exists for all instances and gave an efficient algorithm to find a stable matching (Gale and Shapley, 1962). Gale-Shapley's algorithm is called the man-optimal algorithm because every man has the best partner he can have in any stable matching, but every woman has the worst partner. If applied with the roles of men and women reversed, the algorithm serves as the woman-optimal algorithm. Algorithms for finding stable matching based on optimizations different from that of Gale-Shapley's one have also been studied, such as egalitarian stable matching, minimum regret stable matching (Irving et al., 1987), and sex-equal stable matching (Kato, 1993).

On the other hand, to the best of our knowldege, few studies have been conducted on stability checking algorithms. A simple and efficient $(O(n^2))$ stability checking algorithm (shown in Algorithm 1) is described in (Gusfield and Irving, 1989); hence we construct the implementation of our protocol based on this algorithm.

Algorithm 1: Standard stability checking algorithm.

```
1: for m ← 1 to n do
2:     for each w such that m prefers w to M(m) do
3:         if w prefers to m to M(w) then
4:             return 0
5:             break
6:         end if
7:     end for
8: end for
9: return 1
```

A critical issue with implementing FHE is that the results of conditional expressions in conditional branchings cannot be known. Therefore, the stability checking algorithm is modified not to use any conditional branchings in our implementation.

5.3 Specific Implementation

Some open-source software libraries implement homomorphic encryption such as HElib[2]. Such libraries provide essential functions for homomorphic evaluations. We provide a specific implementation of the stability checking algorithm with the essential functions from libraries such as HElib. The essential

[2]HElib Documentation, https://homenc.github.io/HElib/

functions which are used for our implementation are shown as follows.

1. Addition of two integers: $\mathsf{ADD}(x,y)$ outputs $x+y$, where x and y are integers.
2. Multiplication of an integer and a bit: $\mathsf{MULT}(x,y)$ outputs x if $y=1$, otherwise outputs 0, where x is an integer, y is a bit.
3. Comparison of two integers: $\mathsf{COMP}(x,y)$ outputs 1 if $x<y$, otherwise outputs 0, where x and y are integers.
4. AND of two bits
5. OR of two bits
6. NOT of a bit

In order to execute our protocol, some preliminary steps are required. First, the users' private inputs, which in this case are preference orders, are converted to lists of preference ranks. That is, m_i's list can be denoted by $L_{m_i} = (P_{m_i}(w_1), P_{m_i}(w_2), \ldots, P_{m_i}(w_n))$ and w_i's list can be denoted by $L_{w_i} = (P_{w_i}(m_1), P_{w_i}(m_2), \ldots, P_{w_i}(m_n))$. The private inputs of the list are encrypted element-by-element of the list, that is, the cyphertext from m_i to the verifier is $c_i^m = (c_{i,1}^m, c_{i,2}^m, \ldots c_{i,n}^m)$, where $c_{i,j}^m \leftarrow \mathsf{Enc}(\mathsf{pk}, P_{m_i}(w_j))$.

The matching result y sent to the verifier is represented by a pair (y^m, y^w) of lists of one-hot-vectors. If the partner of m_i is w_j (that is, $\mathcal{M}(m_i) = w_j$), let y_i^m be a n-bit strings where j th bit $y_i^m[j]$ is only 1 and all the others 0. In the same way, if $\mathcal{M}(w_i) = m_j$, let y_i^w be a n-bit strings where j th bit $y_i^w[j]$ is only 1 and all the others 0. This may seem redundant since y^w is just a permutation of y^m; however, it is necessary to construct our protocols using FHE.

We describe the algorithm that can return the same result as the stability checking algorithm by only $\mathsf{ADD}, \mathsf{MULT}, \mathsf{COMP}, \mathsf{AND}, \mathsf{OR}, \mathsf{NOT}$ in Algorithm 2.

The computation cost of this algorithm is $O(n^2)$.

5.4 Example

We show an example for a proof sketch of the proposed algorithm. Let $n = 4$. Each member's preference order is shown in Figure 1. Let $\mathcal{M} = \{(m_1, w_3), (m_2, w_1), (m_3, w_2), (m_4, w_4)\}$. In this case, this matching is not stable because (m_1, w_2) is blocking pair as shown in Figure 2.

In our protocol, the preference orders are converted to the lists of preference ranks shown in Figure 3. $\mathcal{M}$ is represented by $y^m = ((0,0,1,0),(1,0,0,0),(0,1,0,0),(0,0,0,1))$ and $y^w = ((0,1,0,0),(0,0,1,0),(1,0,0,0),(0,0,0,1))$.

Algorithm 2: Stability checking algorithm for FHE.

$r = 0$
for $i \leftarrow 1$ **to** n **do**
  $s^m = 0$
  $s^w = 0$
  for $j \leftarrow 1$ **to** n **do**
    $u^m = \mathsf{MULT}(P_{m_i}(w_j), y_i^m[j])$
    $s^m = \mathsf{ADD}(s^m, u)$
    $u^w = \mathsf{MULT}(P_{w_i}(m_j), y_i^w[j])$
    $s^w = \mathsf{ADD}(s^w, u)$
  end for
  for $j \leftarrow 1$ **to** n **do**
    $e_{i,j}^m \leftarrow \mathsf{COMP}(P_{m_i}(w_j), s^m)$
    $e_{i,j}^w \leftarrow \mathsf{COMP}(P_{w_i}(m_j), s^w)$
  end for
end for
for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow 1$ **to** n **do**
    $t_{i,j} \leftarrow \mathsf{AND}(e_{i,j}^m, e_{j,i}^w)$
    $r \leftarrow \mathsf{OR}(r, t_{i,j})$
  end for
end for
$r = \mathsf{NOT}(r)$
return r

	1	2	3	4
m_1	w_2	w_3	w_1	w_4
m_2	w_4	w_3	w_1	w_2
m_3	w_4	w_2	w_1	w_3
m_4	w_4	w_3	w_2	w_1

	1	2	3	4
w_1	m_2	m_1	m_3	m_4
w_2	m_1	m_3	m_4	m_2
w_3	m_1	m_2	m_4	m_3
w_4	m_4	m_3	m_2	m_1

Figure 1: An example of preference orders.

The ranks of the pair of each member can be calculated by multiplication and addition of the preference ranks and y^m, y^w. For example, for m_1, his preference rank is $(3,1,2,4)$ and his pair is w_3 (represented by $(0,0,1,0)$). w_3's rank of m_1's preference order is $3\cdot 0 + 1\cdot 0 + 2\cdot 1 + 4\cdot 0 = 2$.

Next, the ranks of each member in his/her preference list are compared with the preference rank of his/her pair. If a rank is smaller than the rank of the pair, 1 is assigned, otherwise, 0 is assigned as shown in Figure 4. Here, 1 means a member who is preferred to the current pair.

Finally, AND function between each row of the left table and each column of the right table is calculated as shown in Figure 5. Here, 1 means a blocking pair. Therefore, the matching is stable if all the elements of the table are 0, otherwise, it is not stable.

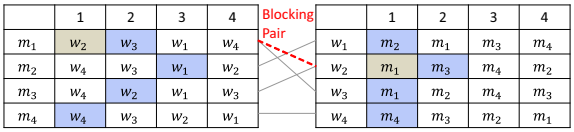

Figure 2: A blocking pair of the example.

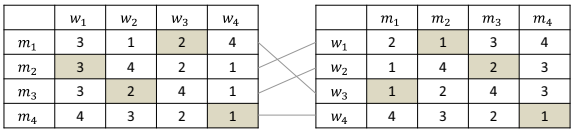

Figure 3: Preference ranks.

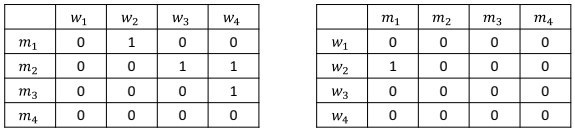

Figure 4: Comparison with the current pair.

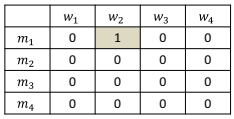

Figure 5: Existence of blocking pairs.

6 DISCUSSION

The proposed method assumes the use of a semi-honest verifier. Future directions include (1) the modification of the methods without the verifier and (2) the modification for stricter assumptions of an adversary such as a malicious adversary. For the methods without the verifier, constructing an MPC-based verification algorithm may be a promising direction.

Sharing FHE private keys among users can be a privacy risk; hence it is better to be able to use individual keys. Multi-key FHE or Secure Multiparty Computation (MPC) (Ohata, 2020) may be a potential solution for this issue; however, it requires all users to participate in the verification calculation.

We have shown that the computational complexity of our proposed algorithm is $O(n^2)$. On the other hand, the number of bits handled and the depth of operations also affect the computational cost in the implementation using FHE. We will implement and evaluate our protocol and optimize it as future work.

In this paper, we limit the target of the mathematical model to applications that can calculate results solely from users' inputs such as matchings. We will discuss the FHE implementation of the verification algorithm for a different problem from matching and the extension for more complex models where parameters other than user inputs affect the results.

7 CONCLUSION

In this paper, we focused on a system where participants provided preference orders to an assignee, and the assignee provided a stable matching. When considering a multi-stakeholder environment, not only the participants' requirements but also the assignee's intention should be respected, that is, the assignee should be given the discretion to select the matching that is the best for him/her among all the stable matchings, which satisfy the requirements from participants. In this paper, we proposed a protocol that satisfies the three requirements, verifiability, privacy, and selectability, with a semi-honest third-party server, called a verifier and fully homomorphic encryption (FHE). We first provided a general construction of this protocol which does not limit specific requirements from users but assumes that the verification algorithm of the requirement is available on FHE. Furthermore, we proposed a specific implementation of an FHE-based verification algorithm for two-sided matching that checks whether matching is stable while users' private inputs kept secret.

REFERENCES

Abdollahpouri, H., Adomavicius, G., Burke, R., Guy, I., Jannach, D., Kamishima, T., Krasnodebski, J., and Pizzato, L. (2020). Multistakeholder recommendation: Survey and research directions. *User Modeling and User-Adapted Interaction*, 30(1):127–158.

Acar, A., Aksu, H., Uluagac, A. S., and Conti, M. (2018). A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (Csur)*, 51(4):1–35.

Brakerski, Z. and Vaikuntanathan, V. (2011). Efficient fully homomorphic encryption from (standard) lwe. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 97–106.

Doerner, J., Evans, D., and shelat, a. (2016). Secure stable matching at scale. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 1602–1613, New York, NY, USA. Association for Computing Machinery.

Franklin, M., Gondree, M., and Mohassel, P. (2007). Improved efficiency for private stable matching. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 4377 LNCS:163–177.

Gale, D. and Shapley, L. S. (1962). College admissions and

the stability of marriage. *The American Mathematical Monthly*, 69:9–15.

Gentry, C. (2009). A fully homomorphic encryption scheme. STANFORD UNIVERSITY.

Golle, P. (2006). A private stable matching algorithm. In *Proceedings of the 10th International Conference on Financial Cryptography and Data Security*, FC'06, page 65–80, Berlin, Heidelberg. Springer-Verlag.

Gusfield, D. and Irving, R. W. (1989). *The Stable Marriage Problem: Structure and Algorithms*. The MIT Press.

Irving, R. W., Leather, P., and Gusfield, D. (1987). An efficient algorithm for the "optimal" stable marriage. *Journal of the ACM (JACM)*, 34(3):532–543.

Kato, A. (1993). Complexity of the sex-equal stable marriage problem. *Japan Journal of Industrial and Applied Mathematics*, 10:1–19.

Ohata, S. (2020). Recent advances in practical secure multiparty computation. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 103(10):1134–1141.

Parno, B., Howell, J., Gentry, C., and Raykova, M. (2013). Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE.

Riazi, M. S., Songhori, E. M., Sadeghi, A.-R., Schneider, T., and Koushanfar, F. (2017). Toward practical secure stable matching. *Proceedings on Privacy Enhancing Technologies*, 1:62–78.

Teruya, T. and Sakuma, J. (2015). Round-efficient private stable matching from additive homomorphic encryption. In *Information Security*, pages 69–86. Springer.