Privacy-Preserving Planarity Testing of Distributed Graphs

Guy Barshap and Tamir Tassa

The Open University, Ra'anana, Israel

Abstract. We study the problem of privacy-preserving planarity testing of distributed graphs. The setting involves several parties that hold private graphs on the same set of vertices, and an external mediator that helps with performing the computations. Their goal is to test whether the union of their private graphs is planar, but in doing so each party wishes to deny from his peers any information on his own private edge set beyond what is implied by the final output of the computation. We present a privacy-preserving protocol for that purpose which is based on the Hanani-Tutte Theorem. That theorem enables translating the planarity question into the question of whether a specific system of linear equations over the field $\mathbf{F}_2$ is solvable. Our protocol uses a diverse cryptographic toolkit which includes techniques such as homomorphic encryption, oblivious Gaussian elimination, and private set intersection. This is the first time that a solution to this problem is presented.

Keywords: secure multiparty computation · privacy-preserving distributed computations · distributed graphs · graph planarity.

1 Introduction

A planar graph $G = (V, E)$ is a graph that can be properly embedded in the two-dimensional plane $\mathbf{R}^2$ in the following sense: there exists a bijection φ from the vertex set V to $\mathbf{R}^2$ and a representation of each edge $e = (u, v) \in E$ as a continuous simple curve in $\mathbf{R}^2$ with $\varphi(u)$ and $\varphi(v)$ as its end points, such that no two curves intersect apart possibly at their end points.

Planar graphs constitute an attractive family of graphs, both in theory and in practice. In many applications where graph structures arise, it is needed to test the planarity of those graphs. A classical example is in the area of integrated circuit (IC) design. An IC consists of electronic modules and the wiring interconnections between them. It can be represented by a graph in which the vertices are the modules and the edges are the wires. An IC can be printed on the surface of a chip iff the graph is planar, because wires must not cross each other. Another setting in which planarity is a natural notion is in road maps. A set of cities and interconnecting roads can be thought of as a graph; the graph vertices are the cities while the edges are connecting roads. Such a map can be constructed with non-crossing roads (in order to avoid constructing bridges or obstructing the traffic flow by stop lights) iff the corresponding graph is planar.

Apart from the above motivating examples, there are cases in which the planarity of a graph can be exploited in order to simplify and expedite the solution of some computational problems. Examples include sub-graph isomorphism [1], maximal clique [2], and maximum cut [3].

In this study we consider a distributed version of the planarity testing problem. In that problem there are several parties, $P_1, \ldots, P_d$, each one holding a private graph on the same set of vertices; namely, P_i has a graph $G_i = (V, E_i)$ where V is publicly known and shared by all, while E_i is private, $1 \leq i \leq d$. They wish to determine whether the union graph $G = (V, E)$, where $E = \bigcup_{i=1}^{d} E_i$, is planar or not. As the edge sets E_i, $1 \leq i \leq d$, are private, the planarity testing should be carried out in a privacy-preserving manner. Namely, after the conclusion of the computational procedure P_i must not learn anything on E_j, $j \neq i$, beyond what is implied by G_i and the planarity of G. For example, two (or more) companies may wish to check the possibility of printing the ICs which implement their products on the same chip. They prefer not to disclose to each other their own IC design, before they are verified that they could collaborate in that manner. The algorithmic solutions which we propose herein for privacy-preserving planarity testing could be used in that application scenario.

The strict notion of perfect privacy-preservation is sometimes relaxed by allowing some leakage of information, if such a relaxation enables a more efficient computation and if the leakage of information is characterized (in order to decide, in any given application setting, whether the gain in efficiency justifies the reduction in privacy-preservation). There are many examples of studies that relax perfect privacy in order to allow practical solutions, from various domains such as distributed association rule mining [4, 5], anonymization of distributed datasets [6–8], collaborative filtering [9, 10], distributed graph mining [11], and distributed constraint optimization problems [12–14].

Well known characterizations for planar graphs were proposed by both Wagner [15] and Kuratowski [16]. For example, the Wagner's characterization states that a graph is planar iff it does not have K_5 (the complete graph over 5 vertices) or $K_{3,3}$ (the complete bipartite graph over 3 vertices in each part) as a minor (see Figure 1). Namely, K_5 or $K_{3,3}$ cannot be obtained from G by a sequence of these operations: contracting edges, deleting edges, and deleting isolated vertices. However, directly applying either Wagner's characterization, or the closely-related Kuratowski's characterization, in order to test the planarity of a given graph, yields exponential-time algorithms [17].

Optimal linear time planarity testing algorithms were proposed in [18, 19]. These algorithms are iterative and use a DFS-subroutine [17]. Alas, those features of these algorithms turn out to be significant obstacles when trying to devise corresponding privacy-preserving variants of these algorithms. Thus, none of the above-mentioned approaches seem to be adequate in order to base on them an efficient privacy-preserving planarity testing protocol.

In this study we propose a privacy-preserving protocol for planarity testing of distributed graphs, which is based on the Hanani-Tutte Theorem [20]. Our protocol is based on the mediated model that was presented in [21]. In that

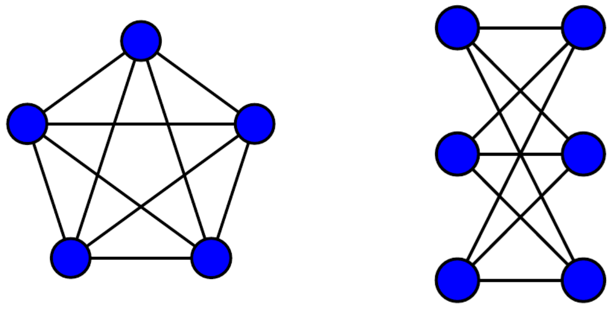

Fig. 1. The minors that cannot appear in a planar graph — K_5 and $K_{3,3}$.

model, there exists an external mediator T to which the parties may export some computations, but the mediator should not learn information on the private inputs of the parties or the final output. We assume that all interacting parties ($P_1, \ldots, P_d$ and T) are semi-honest. Namely, they follow the protocol correctly, and do not form coalitions, but they try to extract from their view in the protocol information on the private inputs of other parties. (All privacy-related studies that we mentioned earlier also make similar assumptions.)

Due to page limitation and for the sake of clarity, we focus here on the case $d = 2$. The extension to any d is deferred to the full journal version of this study.

The outline of this work is as follows. In Section 2 we provide the relevant background on planarity testing, while in Section 3 we describe the main cryptographic toolkit that we use in our solution. We overview our solution and the two main stages of which it consists in Section 4. The subsequent Sections 5 and 6 include the detailed description and analysis of each of the two stages in our solution. Finally, we conclude in Section 7.

2 Planarity testing using the Hanani-Tutte Theorem

In this section we state the Hanani-Tutte Theorem and then use it in order to translate the planarity question of a graph to the solvability of a system of linear equations over $\mathbf{F}_2$. To that end, we introduce the following definitions and notations:

– If $e \in E$ we let $a(e)$ and $b(e)$ denote the two vertices that e connects.

– A drawing D of a graph $G = (V, E)$ is an embedding of G in $\mathbf{R}^2$. Namely, it is a mapping $\varphi : V \to \mathbf{R}^2$ together with a representation of each edge $e \in E$ as a continuous simple curve that connects $\varphi(a(e))$ and $\varphi(b(e))$.

– Two edges $e, f \in E$ are called independent if $\{a(e), b(e)\} \cap \{a(f), b(f)\} = \emptyset$.

– The set of all pairs of independent edges in E is denoted E_2^{ind}.

– For a given drawing D of G and $\{e, f\} \in E_2^{ind}$, $parity_D(e, f)$ is the parity of the number of crossings between the curves representing e and f in D.

Theorem [Hanani-Tutte]. *A graph G is planar iff it has a drawing D in which*

$parity_D(e, f) = 0$ for all $\{e, f\} \in E_2^{ind}$.

Let D be a drawing of G, $e \in E$, and $v \in V \setminus \{a(e), b(e)\}$. An (e, v)-move consists of taking a small section of the curve that represents e in D and deforming it in a narrow tunnel to make it pass over v, while not passing over any other vertex. The effect of an (e, v)-move in a drawing D is that $parity_D(e, f)$ changes for all edges f that are adjacent to v, but it remains unchanged for all other edges f (see Figure 2).

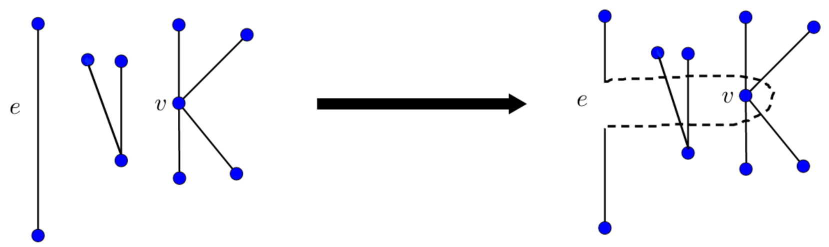

Fig. 2. An (e, v)-move. As a result, $parity_D(e, f)$ changes only for the four edges adjacent to v (from 0 to 1). For all other edges f, $parity_D(e, f)$ remains unchanged (0).

A remarkable corollary of this theorem is a planarity testing algorithm. It starts with an arbitrary drawing D of the input graph G, preferably a drawing in which $parity_D(e, f)$ can be computed efficiently for every pair of independent edges in G. Then, the algorithm tries to find another drawing D', by making a series of (e, v) moves, in which $parity_{D'}(e, f) = 0$ for all $\{e, f\} \in E_2^{ind}$. If it succeeds, then the graph is planar, otherwise it is not (see [20, Lemma 3.3]).

The existence of D' can be determined by considering the following system of linear equations. Define for each $e \in E$ and $v \in V \setminus \{a(e), b(e)\}$ a Boolean variable $x_{e,v}$; that variable equals 1 iff the transition from D to D' includes an (e, v)-move. It follows that for any pair of independent edges $\{e, f\} \in E_2^{ind}$,

$$parity_{D'}(e, f) = parity_D(e, f) + x_{e,a(f)} + x_{e,b(f)} + x_{f,a(e)} + x_{f,b(e)} \quad \text{in } \mathbf{F}_2 .$$

Hence, given the drawing D, there exists a drawing D' in which $parity_{D'}(e, f) = 0$ for all $\{e, f\} \in E_2^{ind}$ iff there exists a solution to the following system of linear equations over $\mathbf{F}_2$:

$$parity_D(e, f) + x_{e,a(f)} + x_{e,b(f)} + x_{f,a(e)} + x_{f,b(e)} = 0 \qquad \{e, f\} \in E_2^{ind} . \quad (1)$$

That is the Hanani-Tutte (HT hereinafter) system for the graph $G = (V, E)$ (with respect to the drawing D). It consists of $|E_2^{ind}|$ equations (one for each pair of independent edges) in $|E| \cdot (|V| - 2)$ unknowns ($x_{e,v}$ for all $e \in E$ and $v \in V \setminus \{a(e), b(e)\}$). The graph G is planar iff that system is solvable.

3 Cryptographic Toolkit

In this section we provide a birdseye view of the cryptographic primitives and procedures that we shall be using later on.

3.1 Homomorphic encryption

An encryption function $\mathcal{F}$ is called (additively) *homomorphic* if the domain of plaintexts is a commutative additive group, the domain of ciphertexts is a commutative multiplicative group, and for every two plaintexts, m_1 and m_2, $\mathcal{F}(m_1 + m_2) = \mathcal{F}(m_1) \cdot \mathcal{F}(m_2)$. When the encryption function is randomized (in the sense that $\mathcal{F}(m)$ depends on m as well as on a random string) then $\mathcal{F}$ is called *probabilistic*. Homomorphic encryption functions allow performing arithmetic computations in the ciphertext domain. The property of being probabilistic is essential for getting semantic security.

There are many well known ciphers that are probabilistic and additively homomorphic. A basic example of such a cipher over $\mathbf{F}_2$, which we use in our protocol, is the Goldwasser-Micali cipher [22].

3.2 Deciding the solvability of an encrypted linear system

Nissim and Weinreb [23] presented a method for obliviously deciding whether an encrypted system of linear equations is solvable or not. They considered a setting that involves two parties – T and P. T holds an encrypted matrix $\mathcal{F}(M)$, where M is a matrix of dimensions $k_a \times k_b$, and an encrypted vector $\mathcal{F}(\mathbf{b})$, where $\mathbf{b}$ is a column vector of dimension k_a. Both M and $\mathbf{b}$ are over the field $\mathbf{F} = \mathbf{F}_2$, while $\mathcal{F}$ is an additively homomorphic encryption over that field, for which P holds the private decryption key. Their protocol is Monte Carlo in the sense that its output may be wrong. Specifically, at the conclusion of the protocol T gets $\mathcal{F}(\beta)$ for some bit β. If the system $M\mathbf{x} = \mathbf{b}$ is not solvable then β will always be zero. Otherwise, if the system is solvable, then $\beta = 1$ with probability at least c for some positive constant c. Hence, by performing several independent runs of the protocol, it is possible to decide the solvability of the system with an error probability sufficiently small.

4 Overview of the proposed planarity testing protocol

Our planarity testing protocol has two stages. The first one consists of a preliminary check of the size of the unified edge set E. It is outlined in Section 4.1, and detailed in Section 5. The second stage includes the main protocol, in which the HT system of equations for the unified graph is constructed obliviously, and then its solvability is tested in a privacy-preserving manner. We provide a birdseye view of that stage in Section 4.2, and dive into its details in Section 6.

4.1 Testing the number of edges in the unified graph.

Let $V = \{v_1, \ldots, v_n\}$ denote the vertex set in the unified graph $G = (V, E)$. A well-known result (see e.g. [17]) states that G is planar only if

$$|E| = |E_1 \cup E_2| \leq 3n - 6\,. \tag{2}$$

Hence, in the first stage, the three parties, P_1, P_2 and T, engage in a secure protocol for checking whether inequality (2) holds or not. If it does not, they know that the unified graph G is not planar. If it does hold, they proceed to the second stage in the verification (Section 4.2).

Running this stage is optional. On one hand, it leaks to the interacting parties information on $|E|$ beyond the required output about the planarity of G (specifically, whether inequality (2) holds or not). On the other hand, it may enable the parties to detect non-planarity without running the costly computation of the second stage. Hence, if in the relevant application scenario the information regarding whether inequality (2) holds or not is deemed benign, it is recommended to run this stage.

4.2 A privacy-preserving implementation of the Hanani-Tutte planarity test.

The three parties P_1, P_2 and T construct the HT system of linear equations, Eq. (1), for the unified graph G. Towards that end, they begin by constructing the system of linear equations for the complete graph on V, denoted K_V (i.e., K_V is the graph on V that has all $\binom{n}{2}$ edges):

$$x_{e,a(f)} + x_{e,b(f)} + x_{f,a(e)} + x_{f,b(e)} = parity_D(e,f) \qquad \{e,f\} \in K_2^{ind}\,; \tag{3}$$

here, K_2^{ind} is the set of all pairs of independent edges in K_V. The number of equations in that system is

$$N_{K_V} := |K_2^{ind}| = \frac{1}{2} \cdot \binom{n}{2} \cdot \binom{n-2}{2}\,. \tag{4}$$

This step can be constructed publicly with no privacy risks, since the vertex set V is known to all, and K_V is the complete graph on V. The main effort is in letting the mediator T extract from the large system in Eq. (3) the subset of equations in Eq. (1). To protect the unified graph data from T, he will get only an encrypted version of the subset of linear equations corresponding to G. The last part of the protocol is dedicated to determining whether that system has a solution or not. The main difficulty here lies in the fact that no party actually sees the relevant system of equations: only T holds that system, but he holds an encryption of that system, where the corresponding decryption key is known only to P_1.

To allow this approach, we must start with some drawing of K_V, which, in turn, induces also a drawing of G. We consider the following embedding of V in $\mathbf{R}^2$. If $V = \{v_1, \ldots, v_n\}$, then v_j is mapped into the point

$$v_j \mapsto \varphi(v_j) := (\cos(2\pi j/n), \sin(2\pi j/n))\,, \quad 1 \leq j \leq n\,. \tag{5}$$

Namely, the vertices $v_1, \ldots, v_n$ are mapped to equi-distant points on the unit circle, in a counter-clockwise order according to their index. The edge $e_{i,j} = (v_i, v_j)$ is then represented by the straight line segment between $\varphi(v_i)$ and $\varphi(v_j)$. Figure 3 illustrates that basic drawing for a graph over $n = 6$ vertices.

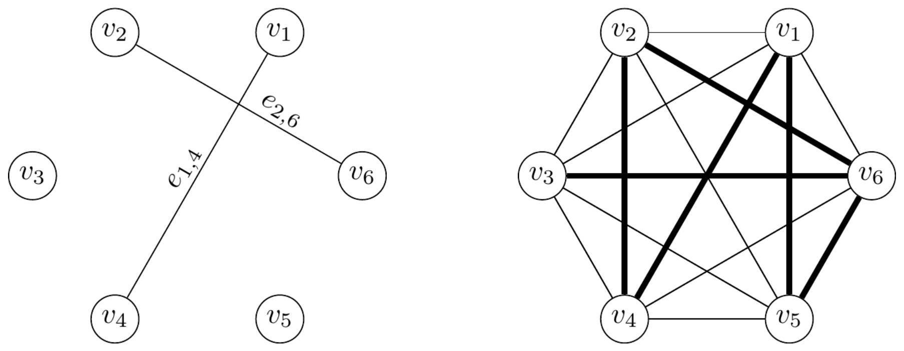

Fig. 3. Left: illustration of the embedding of $n = 6$ vertices, together with two connecting edges. Right: the corresponding drawing of K_V and of $G = (V, E)$ with $E = \{e_{1,4}, e_{1,5}, e_{2,4}, e_{2,6}, e_{3,6}, e_{5,6}\}$ (the six edges in E are marked by thicker lines).

Let us denote the above drawing of K_V (and the corresponding drawing of G, which is a sub-graph of K_V) by D. Consider now an arbitrary pair of independent edges $e_{i,j}$ and $e_{k,\ell}$; we may assume, without loss of generality, that $i < j$, $k < \ell$, and $i < k$. Then it is easy to see that $parity_D(e_{i,j}, e_{k,\ell}) = 1$ iff $i < k < j < \ell$.

The corresponding HT system (3) can be constructed publicly, by each of P_1, P_2 and T, for this drawing D of the complete graph K_V. As stated earlier, the main problem will be to identify, among those N_{K_V} equations, the $|E_2^{ind}|$ equations that relate to pairs of edges e and f that are both in E. Then, the graph $G = (V, E)$ is planar iff that sub-system of $|E_2^{ind}|$ equations has a solution. In Section 6 we provide the details of that computation.

5 First stage: Testing the size of the unified edge set

Let $V_2 := \{(v_i, v_j) : 1 \le i < j \le n\}$ denote the set of all possible $\binom{n}{2}$ edges in G. Since $E = E_1 \cup E_2$, we infer that $E^c = E_1^c \cap E_2^c$, where for any subset $A \subseteq V_2$, $A^c := V_2 \setminus A$ denotes its complement within V_2. Hence, by Eq. (2), the unified graph $G = (V, E)$ is planar only if

$$|E^c| = |E_1^c \cap E_2^c| \ge \binom{n}{2} - (3n - 6)\,. \tag{6}$$

In order to verify the latter inequality it is possible to invoke any of the multitude of protocols for private set intersection. The first such protocol was

Protocol 1 Testing the size of the unified edge set

```
1: $P_1$ and $P_2$ select a large multiplicative group $\mathbf{Z}_p^*$ ($p$ is prime) and a hash function
   $H$ whose range can be embedded in $\mathbf{Z}_p^*$.
2: $P_h$ selects a secret and random exponent $1 < \alpha_h < p-1$, $h = 1, 2$.
3: $P_1$ sends to $P_2$ a vector $\mathbf{x}_1$ of length $\binom{n}{2}$ where, for each edge $(v_i, v_j) \in E_1^c$, $i < j$,
   $\mathbf{x}_1$ includes an entry of the form $H(i,j)^{\alpha_1}$, while the remaining $\binom{n}{2} - |E_1^c|$ entries
   are randomly selected from $\mathbf{Z}_p^*$. The order of $\mathbf{x}_1$'s entries is random.
4: $P_2$ sends to $P_1$ a vector $\mathbf{x}_2$ of length $\binom{n}{2}$ where, for each edge $(v_i, v_j) \in E_2^c$, $i < j$,
   $\mathbf{x}_2$ includes an entry of the form $H(i,j)^{\alpha_2}$, while the remaining $\binom{n}{2} - |E_2^c|$ entries
   are randomly selected from $\mathbf{Z}_p^*$. The order of $\mathbf{x}_2$'s entries is random.
5: $P_1$ sends to $T$ the vector $\mathbf{y}_2$, where $\mathbf{y}_2(i) = \mathbf{x}_2(i)^{\alpha_1}$, $1 \le i \le \binom{n}{2}$.
6: $P_2$ sends to $T$ the vector $\mathbf{y}_1$, where $\mathbf{y}_1(i) = \mathbf{x}_1(i)^{\alpha_2}$, $1 \le i \le \binom{n}{2}$.
7: $T$ compares the two received vectors and finds out the number $z$ of matching entries
   in them.
8: If $z < \binom{n}{2} - (3n-6)$, $T$ notifies $P_1$ and $P_2$ that the union graph is not planar.
```

proposed in [24], and is based on the Diffie-Hellman protocol [25]. Protocol 1, which we present below, is based on the private set intersection protocol of [24].

As a result of Steps 1-6, which are self-explanatory, T receives two vectors, $\mathbf{y}_1$ and $\mathbf{y}_2$, each of which is of length $\binom{n}{2}$. The vector $\mathbf{y}_h$, $h = 1, 2$, includes the hash of all edges in E_h^c, raised to the exponent $\alpha_1\alpha_2$, while the remaining $\binom{n}{2} - |E_h^c|$ entries are random elements in $\mathbf{Z}_p^*$. The number z of matching entries in those two vectors (Step 7) satisfies $z \ge |E^c|$, while with very high probability $z = |E^c|$. Indeed, if $(v_i, v_j) \in E_1^c \cap E_2^c$ then both vectors $\mathbf{y}_1$ and $\mathbf{y}_2$ will include an entry that equals $H(i,j)^{\alpha_1\alpha_2}$; hence, those two entries will be identified by T in Step 7 as matching entries and, consequently, T will increment the counter z by 1. However, we note that T may wrongly increment the counter z due to random false matchings. False matchings can occur if there are collusions in H, namely, if there exist two pairs (i,j) and (i',j') such that $H(i,j) = H(i',j')$, or if P_1 and P_2 selected in Steps 3 and 4 random entries ξ_1 and ξ_2, respectively, so that $\xi_2^{\alpha_1} = \xi_1^{\alpha_2}$. By selecting a secure hash function with a sufficiently large range, the probability of such false matchings is negligible.

The security of Protocol 1 follows from the hardness of the Discrete Log problem. The protocol entails $O(n^2)$ hash function evaluations and exponentiations (for P_1 and P_2) and $O(n^2 \log n)$ comparisons for T. The protocol has only two rounds of communication in which $O(n^2 \log p)$ bits are transmitted.

Protocol 1 reveals to T the size of E. If P_1 and P_2 wish to prevent T from learning that information, they may modify Protocol 1 towards hiding that information. They can choose an integer $K > 0$ and then select at random an integer $k \in [0, K]$. Next, they will select $2K - k$ random and distinct elements from $\mathbf{Z}_p^*$: a_i, $1 \le i \le k$, and $b_{h,i}$, $1 \le i \le K-k$, $h = 1, 2$. Then, P_1 will add to $\mathbf{y}_2$ additional K entries, at random locations, with the values $a_1, \ldots, a_k, b_{1,i}, \ldots, b_{1,K-k}$, while P_2 will add to $\mathbf{y}_1$ additional K entries, at random locations, with the values

$a_1, \ldots, a_k, b_{2,i}, \ldots, b_{2,K-k}$. Given those modifications, T will recover in Step 7 a value z that equals $|E^c| + k$ (with high probability).

Hence, the inequality that needs to be verified now is whether $z < \binom{n}{2} - (3n-6) + k$. In the latter inequality, T knows the left hand side (z) while P_1 knows the right hand side, and the two parties need to verify the inequality without disclosing to each other information on the compared values beyond the information of whether the inequality holds or not. This is an instance of the celebrated Yao's millionaires' problem [26]. By invoking any of the many available protocols for solving that problem (e.g. [27]), the two parties may find out securely whether $|E^c| < \binom{n}{2} - (3n-6)$.

Such a modification of Protocol 1 prevents T from getting $|E|$. Higher values of K will imply higher levels of obfuscation, but at the same time also higher communication and computational costs. As implied by [28, Lemma 4], if $K > \binom{n}{2}$ then the probability of T not learning anything on $|E|$ from z is exactly $1 - \binom{n}{2}/(K+1)$; in all other cases (namely, in probability $\binom{n}{2}/(K+1)$, T will learn either a lower or an upper bound on $|E|$.

6 Second stage: Private planarity testing

Protocol 2 decides the planarity of the union graph G in a privacy-preserving manner. It begins with P_1 generating a key pair in a probabilistic additively homomorphic cipher, $\mathcal{F}$, over $\mathbf{F}_2$.

Next, the three parties execute a sub-protocol, called ConstructHTSystem (Step 2). The purpose of that sub-protocol is to construct the HT system for the union graph G, Eq. (1), in an oblivious manner. At the end of that sub-protocol T will hold an (entry-wise) $\mathcal{F}$-encryption of the coefficient matrix of that system. (We note that T will actually get an "inflated" version of that system, in the following sense: instead of an encryption of the HT system for G, Eq. (1), he will hold an encryption of the larger HT system for the complete graph K_V, Eq. (3), where all equations that relate to pairs of edges that are not in E_2^{ind} are zeroed.)

Then (Step 3), P_1 and T execute a sub-protocol, called DecideSolvability, which decides the solvability of the encrypted system that was constructed in the previous step. T holds the encryption of the coefficient matrix and the right hand side vector for that system, while P_1 holds the relevant decryption key. The sub-protocol DecideSolvability decides the solvability of the system in a privacy-preserving manner, that is – without decrypting the system. The Boolean flag that DecideSolvability returns indicates the solvability of the system. If it is **true** then the system is solvable and, consequently, the union graph G is planar. Otherwise, if it is **false**, then, with high probability (which may be tuned as desired), the system is not solvable and, hence, the union graph G is not planar.

In the next sub-sections (Sections 6.1 and 6.2) we discuss the implementation of the two main steps in Protocol 2.

Protocol 2 Privacy preserving HT planarity testing

```
1: $P_1$ generates a key pair in a probabilistic additively homomorphic cipher, $\mathcal{F}$, over
   $\mathbf{F}_2$. $P_1$ notifies $P_2$ and $T$ of the public encryption key in $\mathcal{F}$.
2: $P_1$, $P_2$ and $T$ execute ConstructHTSystem. At its conclusion, $T$ holds an $\mathcal{F}$-
   encryption of the HT system for the union graph, $(\mathbf{r}_{e,f} : \{e,f\} \in K_2^{ind})$.
3: $P_1$ and $T$ execute DecideSolvability$(\mathbf{r}_{e,f} : \{e,f\} \in K_2^{ind})$.
4: **if** DecideSolvability returns **true then**
5:    Output "The union graph is planar".
6: **else**
7:    Output "The union graph is non-planar".
8: **end if**
```

6.1 Constructing an $\mathcal{F}$-encryption of the HT system

Here we discuss the sub-protocol ConstructHTSystem, which is implemented in Protocol 3. Before starting to do so, we take a look at the HT systems for G and for the complete graph K_V. The HT system for the complete graph K_V, with respect to the drawing D described in Section 4.2, is given in Eq. (3). All parties can construct that system, since K_V is a public graph. The HT system for G, Eq. (1), which determines the planarity of $G = (V, E)$, is a sub-system (subset of equations) of (3). That sub-system includes only the equations relating to pairs $\{e, f\} \in E_2^{ind} \subset K_2^{ind}$ (namely, pairs of independent edges $\{e, f\}$ where both e and f are in E).

Let $V_2 := \{e_{i,j} = (v_i, v_j) : 1 \le i < j \le n\}$ denote the edge set in the full graph K_V (consisting of all possible pairs of vertices from V). The set K_2^{ind} consists of all pairs of independent edges in K_V. Its size is N_{K_V} (Eq. (4)), and it consists of all pairs of edges $\{e = (v_i, v_j), f = (v_k, v_\ell)\}$ where all four indices i, j, k, ℓ are distinct (as the two edges are independent), and $i < j$, $k < \ell$, and $i < k$. Our protocol, ConstructHTSystem, assumes that the set K_2^{ind} is ordered. We assume hereinafter that it is ordered lexicographically by the 4-tuple (i, j, k, ℓ).

For each edge $e \in V_2$ and $h \in \{1, 2\}$, let α_e^h be the Boolean variable denoting whether $e \in E_h$ or not. Then, $e \in E$ iff $\alpha_e^1 \vee \alpha_e^2 = 1$. Consequently, the equation that corresponds to the pair of independent edges $\{e, f\} \in K_2^{ind}$ appears in the HT system for G, Eq. (1), iff

$$\chi_{e,f} := (\alpha_e^1 \vee \alpha_e^2) \wedge (\alpha_f^1 \vee \alpha_f^2) = 1\,. \tag{7}$$

In the first part of ConstructHTSystem (Steps 1-2), T gets the $\mathcal{F}$-encryption of $\chi_{e,f}$ for all $\{e, f\} \in K_2^{ind}$, where $\chi_{e,f}$ is the Boolean flag indicating whether $\{e, f\} \in E_2^{ind}$ (Eq. (7)), and $\mathcal{F}$ is the cipher that P_1 selected in Step 1 of Protocol 2. Towards that end, we observe that

$$\chi_{e,f} := (\alpha_e^1 \vee \alpha_e^2) \wedge (\alpha_f^1 \vee \alpha_f^2) = (\alpha_e^1 + \alpha_e^2 - \alpha_e^1 \cdot \alpha_e^2) \cdot (\alpha_f^1 + \alpha_f^2 - \alpha_f^1 \cdot \alpha_f^2)\,. \tag{8}$$

Hence, by opening the brackets on the right hand side of Eq. (8) and then applying $\mathcal{F}$ on both sides of that equation, we get, using the homomorphism of

$\mathcal{F}$ and simple algebra, that $\mathcal{F}(\chi_{e,f}) = A \cdot B^{-1}$, where

$$A = \alpha^{\alpha_f^2} \cdot \beta^{\alpha_e^2} \cdot \gamma^{1+\alpha_e^2 \cdot \alpha_f^2} \cdot \mathcal{F}(\alpha_e^2 \cdot \alpha_f^2)\,, \quad B = (\alpha \cdot \beta)^{\alpha_e^2 \cdot \alpha_f^2} \cdot \gamma^{\alpha_e^2 + \alpha_f^2}\,, \tag{9}$$

and

$$\alpha := \mathcal{F}(\alpha_e^1)\,,\ \beta := \mathcal{F}(\alpha_f^1)\,,\ \gamma := \mathcal{F}(\alpha_e^1 \cdot \alpha_f^1)\,. \tag{10}$$

In view of the above derivations, P_1 sends to P_2 the three values α, β, and γ, for each of the N_{K_V} pairs $\{e,f\} \in K_2^{ind}$, where the triplets (α,β,γ) are ordered by the lexicographical order over K_2^{ind} (Step 1). P_2 can then use Eq. (9) in order to compute A and B, for each such pair. Note that all powers of α, β and γ in Eq. (9) are determined by Boolean variables owned by P_2, while $\mathcal{F}(\alpha_e^2 \cdot \alpha_f^2)$ can be computed by P_2 since he has the public encryption key of $\mathcal{F}$. P_2 then sends to T a vector of length N_{K_V}, in which each entry includes the value $\mathcal{F}(\chi_{e,f}) = A \cdot B^{-1}$ for the relevant pair $\{e,f\} \in K_2^{ind}$ (Step 2).

Protocol 3 CONSTRUCTHTSYSTEM: Constructing an encryption of the HT system

```
1: $P_1$ sends to $P_2$ the vector $((\alpha,\beta,\gamma) : \{e,f\} \in K_2^{ind})$ (see Eq. (10)).
2: $P_2$ sends to $T$ the vector $\mathbf{u} := (\mathcal{F}(\chi_{e,f}) : \{e,f\} \in K_2^{ind})$.
3: for all $\{e,f\} \in K_2^{ind}$, where $e = e_{i,j}$, $f = e_{k,\ell}$, $i < j$, $k < \ell$ and $i < k$ do
4:     $T$ allocates a vector $\mathbf{r}_{e,f}$ of dimension $N+1$ where $N := \binom{n}{2} \cdot (n-2)$.
5:     $T$ creates a bijection $\Phi : [N] \to \{(g,v) : g \in V_2, v \in V \setminus \{a(g), b(g)\}\}$.
6:     for $i \in [N]$ do
7:         $(g,v) \leftarrow \Phi(i)$.
8:         if $(g = e$ and $v \in \{a(f), b(f)\})$ or $(g = f$ and $v \in \{a(e), b(e)\})$ then
9:             $\mathbf{r}_{e,f}(i) \leftarrow \mathcal{F}(\chi_{e,f})$
10:        else
11:            $\mathbf{r}_{e,f}(i) \leftarrow \mathcal{F}(0)$
12:        end if
13:    end for
14:    if $i < k < j < \ell$ then
15:        $\mathbf{r}_{e,f}(N+1) \leftarrow \mathcal{F}(\chi_{e,f})$
16:    else
17:        $\mathbf{r}_{e,f}(N+1) \leftarrow \mathcal{F}(0)$
18:    end if
19: end for
```

The goal of the main loop in Steps 3-19 is to let T construct an entry-wise $\mathcal{F}$-encryption of the HT system for G, Eq. (1). That system has $|E_2^{ind}|$ equations over $|E| \cdot (n-2)$ unknowns. It is a sub-system of the full system for K_V, Eq. (3), which has N_{K_V} equations over $N := \binom{n}{2} \cdot (n-2)$ unknowns. The encrypted system that T constructs in the loop in Steps 3-19 will be of the same dimensions as the larger system for the full graph, but all rows in it that are not relevant for G will be zeroed. T will remain oblivious to the rows in the full system that he zeroes in this process. We proceed to explain how this is done.

Consider the augmented matrix that describes the HT system for K_V, Eq. (3). It has N_{K_V} rows, one for each pair $\{e, f\} \in K_2^{ind}$. Each row, $\mathbf{r}_{e,f}$, is a Boolean vector of length $N + 1$, where $N = \binom{n}{2} \cdot (n - 2)$, since it includes the coefficient of each unknown variable (and there are N such variables, one for each coupling of an edge and a non-adjacent vertex) plus the right hand side ($parity_D(e, f)$). In the linear equation corresponding to the pair $\{e, f\}$, the coefficients of all variables are zero, except for four of those variables (see Eq. (3)). Hence, in the inner loop in Steps 6-13, T goes over the first N entries of $\mathbf{r}_{e,f}$; in each of the four entries that should equal 1, T places the value $\mathcal{F}(\chi_{e,f})$ (those are values that T got from P_2 in Step 2), while in all the remaining ones he places the value $\mathcal{F}(0)$ (those are encryptions that T can compute on his own since he has the public encryption key of $\mathcal{F}$). In the last position in $\mathbf{r}_{e,f}$, corresponding to the right hand side of the equation for the pair $\{e, f\}$, T places the value $\mathcal{F}(\chi_{e,f})$ in case the two edges intersect in the basic drawing D, while otherwise he places the value $\mathcal{F}(0)$ (Steps 14-18). As a result, if $\chi_{e,f} = 0$, T constructs an encryption of the all-zero equation; but if $\chi_{e,f} = 1$, T constructs an encryption of the equation for the pair $\{e, f\}$, as in Eq. (3). In summary, T gets a system of N_{K_V} encrypted equations: $|E_2^{ind}|$ of those equations are an $\mathcal{F}$-encryption of the system (1), while the remaining ones are $\mathcal{F}$-encryptions of the trivial equation (the equation in which all coefficients and right hand side are zero).

6.2 Determining the solvability of an encrypted linear system

The sub-protocol DecideSolvability (Protocol 4) decides the solvability of a system of N_{K_V} linear equations over $N = \binom{n}{2} \cdot (n - 2)$ unknowns. Let M denote the $N_{K_V} \times N$ matrix of coefficients of that system, and $\mathbf{b}$ denote the right hand side vector (an N_{K_V}-dimensional column vector). The two parties that run the sub-protocol are P_1 and T. P_1 holds the decryption key in $\mathcal{F}$ (see Step 1 in Protocol 2) – a probabilistic additively homomorphic cipher over $\mathbf{F}_2$; T, on the other hand, holds $\mathcal{F}(M)$ and $\mathcal{F}(\mathbf{b})$. DecideSolvability determines whether the system $M\mathbf{x} = \mathbf{b}$ has a solution or not. It does so in a privacy-preserving manner, i.e., without revealing to neither of the two parties information on the underlying matrix M and right hand side $\mathbf{b}$.

To do so, the two parties execute the protocol due to Nissim and Weinreb [23] that we outlined in Section 3.2, which enables them to obliviously decide whether an encrypted system of linear equations is solvable or not. We refer to this protocol below by the name SolvabilityLinearSystem. The output of Protocol SolvabilityLinearSystem is $\mathcal{F}(\beta)$, where β is a bit that indicates the existence of a vector $\mathbf{x}$ for which $M\mathbf{x} = \mathbf{b}$.

We recall that the basic protocol due to Nissim and Weinreb is a true-biased Monte Carlo protocol. Namely, a **true** answer (i.e., the system is solvable) is always correct, while a **false** answer may be wrong. We assume herein that the procedure SolvabilityLinearSystem which is invoked in Step 1 of Protocol 4 executes the basic protocol due to Nissim and Weinreb a sufficient number of times so that the probability of a false answer to be incorrect is reduced to below some given desired threshold.

Sub-protocol 4 DecideSolvability

```
1: $T$ and $P_1$ run Protocol SolvabilityLinearSystem with inputs $\mathcal{F}(M)$ and $\mathcal{F}(\mathbf{b})$.
   The output $\mathcal{F}(\beta)$ goes to $T$.
2: $T$ sends to $P_1$ the value $\mathcal{F}(\beta)$.
3: $P_1$ decrypts and recovers $\beta$.
4: if $\beta = 1$ then
5:    return true
6: else
7:    return false
8: end if
```

6.3 Privacy analysis

The potential leakages of information to any party is due to messages that he receives from other parties. We proceed to discuss the security of each of the steps in Protocol 2 that involves exchange of messages.

In Step 1 of Protocol 3 (which is invoked by Protocol 2), P_2 receives from P_1 information relating to E_1. Specifically, for each pair of independent edges in K_V, P_2 receives the values $\alpha := \mathcal{F}(\alpha_e^1)$, $\beta := \mathcal{F}(\alpha_f^1)$, and $\gamma := \mathcal{F}(\alpha_e^1 \cdot \alpha_f^1)$. However, as that information is encrypted by $\mathcal{F}$, P_2 cannot extract information on E_1, assuming that the chosen cipher $\mathcal{F}$ is semantically secure (as is the case with the Goldwasser-Micali cipher [22] which we propose to utilize here). Similarly for Step 2 of Protocol 3 in which T receives information on E; as it is encrypted by $\mathcal{F}$, it is protected from T.

Finally, the security of Protocol 4 follows from the security of Solvability-LinearSystem that was established in [23].

6.4 Computational and communication costs

We begin by assessing the computational and communication costs of the first two steps in Protocol 3. The computational cost of Protocol 3 for P_1 is $2\binom{n}{2}$ encryptions (for computing α and β for all edges in K_V), and, in addition, N_{K_V} (Eq. (4)) encryptions, for computing γ for all pairs of edges in K_2^{ind} (Step 1). The computational cost of Protocol 3 for P_2 is dominated by the need to perform N_{K_V} encryptions ($\mathcal{F}(\alpha_e^2 \cdot \alpha_f^2)$) in order to compute A for for all pairs of edges in K_2^{ind}. The remaining operations for computing A and B (see Eq. (9)) in Step 2 are only multiplications (note that all exponents in Eq. (9) are 0, 1, or 2). The communication cost of Steps 1-2 in Protocol 3 is $O(N_{K_V})$ bits.

The computational costs for T due to Protocol 3 are negligible, as it has to perform no new encryptions, other than computing $\mathcal{F}(0)$ once. We note that the value $\mathcal{F}(0)$ appears in many entries in the encrypted HT linear system of equations. However, the procedure SolvabilityLinearSystem, which is executed in the next stage, is designed so that there is no need for T to generate here independent encryptions $\mathcal{F}(0)$ for each such entry.

The main computational bottleneck is Protocol 4. Indeed, as the underlying matrix has $k_a = N_{K_V} = O(n^4)$ rows and $k_b = \binom{n}{2} \cdot (n-2) = O(n^3)$ columns, the computational cost of running an oblivious Gaussian elimination on it is of order $O(k_a \cdot k_b^2) = O(n^{10})$. Such a computational cost severely limits the applicability of our protocol to very small graphs.

The main problem with Protocol 2 is that it runs the oblivious Gaussian elimination over an encrypted matrix that has N_{K_V} rows. We recall that the actual system, Eq. (1), has only $|E_2^{ind}|$ equations. Since $|E| \leq 3n-6$, as verified in the first stage, then $|E_2^{ind}| = O(n^2)$. Hence, one goal is to reduce the number of rows in the encrypted HT system from $N_{K_V} = O(n^4)$ to the exact number of relevant equations $|E_2^{ind}| = O(n^2)$. Moreover, the number of columns in Eq. (1) is $|E| \cdot (n-2) \leq (3n-6) \cdot (n-2) = O(n^2)$. Hence, another goal is to reduce the number of columns (unknowns) from $O(n^3)$ to $O(n^2)$. If we achieve both goals then the cost of the oblivious Gaussian elimination would reduce to $O(n^6)$. While this time complexity still limits the scalability of the protocol, it allows its execution on graphs with several hundreds of vertices. Such time complexity renders our protocol viable for application settings such as the two motivating examples that were considered in the introduction (IC design and road networks).

6.5 Reducing the size of the HT system

In the master thesis [29] on which this study is based, we present a variant of Protocol 2 that achieves the above mentioned goals of reducing the number of rows and number of columns in the HT system. Due to space limitations, we only outline the main ideas of that variant herein, and leave the detailed description and analysis to the full version of this study.

The main difference from Protocol 2 is that P_1 generates the encryption of the HT system for the full graph K_V, Eq. (3), and sends it to T. Then, T extracts from that large system the sub-system for G, Eq. (1). Specifically, P_1 performs a similar computation to the one that T does in Steps 3-19 of Protocol 3, where the only difference is that in Steps 9 and 15 P_1 inserts the value $\mathcal{F}(1)$ (and not $\mathcal{F}(\chi_{e,f})$ as done in Protocol 3) in the relevant entries. Before sending the encrypted matrix to T, P_1 applies on its rows and columns random permutations, which are selected jointly by P_1 and P_2 and are kept secret from T.

Recall that the matrix has $k_a = N_{K_V} = O(n^4)$ rows and $k_b + 1 = \binom{n}{2} \cdot (n-2) + 1 = O(n^3)$ columns. By examining the structure of the matrix, see Eq. (3), each of the k_a rows has either four or five 1-entries, while the remaining entries are 0. Hence, in the encrypted matrix that P_1 sends to T, there will be $O(n^4)$ entries that equal $\mathcal{F}(1)$ and $O(n^7)$ entries that equal $\mathcal{F}(0)$. P_1 cannot re-use encryptions, since it is necessary to prevent T from distinguishing between 0 and 1 entries. It is possible to generate those $O(n^7)$ encrypted entries by performing only $O(n^{3.5})$ encryptions and relying on the homomorphic property of $\mathcal{F}$ (which implies that $\mathcal{F}(0) \cdot \mathcal{F}(0) = \mathcal{F}(0)$ and $\mathcal{F}(0) \cdot \mathcal{F}(1) = \mathcal{F}(1)$).

In order to enable T to detect which rows in the encrypted matrix need to be discarded (since they relate to pairs of edges in $K_2^{ind} \setminus E_2^{ind}$), T generates a key pair in a probabilistic additively homomorphic cipher, $\mathcal{E}$, over $\mathbf{F}_2$. Then, P_1 and P_2 perform the computation in Steps 1-2 of Protocol 3 with $\mathcal{E}$ instead of

$\mathcal{F}$. Hence, P_2 sends to T the vector $\mathbf{u} := (\mathcal{E}(\chi_{e,f}) : \{e, f\} \in K_2^{ind})$, where the entries are permuted in accord to the selected order of rows in the encrypted matrix that P_1 had sent earlier to T. Subsequently, T computes $\mathbf{v} := \mathcal{E}^{-1}(\mathbf{u})$ and then he removes from the matrix all rows that correspond to 0-entries in $\mathbf{v}$. A similar procedure can be used to enable T to remove columns that are irrelevant for the HT system for G. After performing those two reductions, T gets an encryption of the system in Eq. (1). That is the system on which the procedure DecideSolvability is applied.

Such a variant of Protocol 2 has a computational cost of $O(n^6)$. It has larger communication costs than Protocol 2, as P_1 needs to transfer to T $O(n^7)$ bits. In addition, it enables T to infer $|E|$ (as the final number of columns equals $|E| \cdot (n-2)$). If the latter value is deemed sensitive, P_1 and P_2 can obfuscate it by sending to T information that will result in keeping unnecessary columns, i.e., columns relating to variables $x_{e,v}$ where $e \notin E$. Such course of action will increase the computational cost, but will prevent T from inferring $|E|$.

7 Conclusions

We introduced the problem of privacy-preserving planarity testing of distributed graphs. We presented a protocol that solves this problem. Our protocol, based on the Hanani–Tutte Theorem, protects the private edge sets of each of the parties, under the assumption that the parties are semi-honest and do not collude.

In the full version of this study [29] we present an extension of our protocol to any number of parties; we present in detail and analyze the more efficient variant of Protocol 2, which we outlined in Section 6.5; and we show how our protocol can be used in order to reduce the complexity of various privacy-preserving graph computations, such as testing 3-colorability or testing outer-planarity.

This study raises the following problems for future research:

(a) Improving scalability, either by devising more efficient ways to test the solvability of the HT system, or by designing a privacy-preserving version of another planarity testing algorithm.

(b) Devising privacy-preserving protocols for solving graph problems, which are known to have an efficient solution in cases where the underlying graph is planar, e.g. the sub-graph isomorphism problem, or the maximal clique problem.

(c) Enhancing the resiliency of the protocol to coalitions, and to stronger adversarial models (i.e., malicious parties).

References

1. Eppstein, D.: Subgraph isomorphism in planar graphs and related problems. Journal of Graph Algorithms and Applications **3**(3) (1999) 1–27
2. Papadimitriou, C.H., Yannakakis, M.: The clique problem for planar graphs. Information Processing Letters **13**(4/5) (1981) 131–133
3. Hadlock, F.: Finding a maximum cut of a planar graph in polynomial time. The SIAM Journal on Computing **4**(3) (1975) 221–225

4. Kantarcioglu, M., Clifton., C.: Privacy-preserving distributed mining of association rules on horizontally partitioned data. IEEE Transactions on Knowledge and Data Engineering **16** (2004) 1026–1037
5. Tassa, T.: Secure mining of association rules in horizontally distributed databases. Transactions on Knowledge and Data Engineering **26** (2014) 970–983
6. Jiang, W., Clifton, C.: A secure distributed framework for achieving k-anonymity. The VLDB Journal **15** (2006) 316–333
7. Tassa, T., Gudes, E.: Secure distributed computation of anonymized views of shared databases. Transactions on Database Systems **37, Article 11** (2012)
8. Tassa, T., Cohen, D.: Anonymization of centralized and distributed social networks by sequential clustering. Transactions on Knowledge and Data Engineering **25** (2013) 311–324
9. Jeckmans, A., Tang, Q., Hartel, P.: Privacy-preserving collaborative filtering based on horizontally partitioned dataset. In: CTS. (2012) 439–446
10. Shmueli, E., Tassa, T.: Secure multi-party protocols for item-based collaborative filtering. In: RecSys. (2017) 89–97
11. Asharov, G., Bonchi, F., García-Soriano, D., Tassa, T.: Secure centrality computation over multiple networks. In: WWW. (2017) 957–966
12. Grinshpoun, T., Tassa, T.: A privacy-preserving algorithm for distributed constraint optimization. In: AAMAS. (2014) 909–916
13. Léauté, T., Faltings, B.: Protecting privacy through distributed computation in multi-agent decision making. Journal of Artificial Intelligence Research **47** (2013) 649–695
14. Tassa, T., Grinshpoun, T., Zivan, R.: Privacy preserving implementation of the max-sum algorithm and its variants. Journal of Artificial Intelligence Research **59** (2017) 311–349
15. Wagner, K.: Über eine eigenschaft der ebenen komplexe. Mathematische Annalen **114** (1937) 570–590
16. Kuratowski, K.: Sur le problème des courbes gauches en topologie. Fundamenta Mathematicae **15** (1930) 271–283
17. Patrignani, M.: Handbook on graph drawing and visualization. CRC Press (2013) 1–42
18. Hopcroft, J.E., Tarjan, R.E.: Efficient planarity testing. Journal of the ACM **21**(4) (1974) 549–568
19. Boyer, J.M., Myrvold, W.J.: On the cutting edge: Simplified o(n) planarity by edge addition. Journal of Graph Algorithms and Applications **8**(2) (2004) 241–273
20. Schaefer, M.: Toward a theory of planarity: Hanani-tutte and planarity variants. Journal of Graph Algorithms and Applications **17**(4) (2013) 367–440
21. Alwen, J., Shelat, A., Visconti, I.: Collusion-free protocols in the mediated model. In: CRYPTO. (2008) 497–514
22. Goldwasser, S., Micali, S.: Probabilistic encryption and how to play mental poker keeping secret all partial information. In: STOC. (1982) 365–377
23. Nissim, K., Weinreb, E.: Communication efficient secure linear algebra. In: TCC. (2006) 522–541
24. Meadows, C.A.: A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party. In: IEEE Symposium on Security and Privacy. (1986) 134–137
25. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory **22**(6) (1976) 644–654
26. Yao, A.C.: Protocols for secure computations (extended abstract). In: FOCS. (1982) 160–164

27. Lin, H., Tzeng, W.: An efficient solution to the millionaires' problem based on homomorphic encryption. In: ACNS. (2005) 456–466
28. Grinshpoun, T., Tassa, T.: P-syncbb: A privacy preserving branch and bound DCOP algorithm. Journal of Artificial Intelligence Research **57** (2016) 621–660
29. Barshap, G.: Privacy-preserving planarity testing of distributed graphs. Department of Mathematics and Computer Science, The Open University of Israel. (2018) Master Thesis.