Protocols for Graph Isomorphism and Hamiltonicity Languages

Vadym Fedyukovych

Abstract

Constant-round protocols are introduced for deciding Graph Isomorphism and Hamiltonicity language membership. Protocols achieve negligible soundness error with Prover challenges chosen from a large finite field. Graphs are represented with characteristic polynomials.

1 Introduction

Interactive proof systems were introduced [9] for probabilistic verification of language membership, including deciding whether a pair of given graphs are isomorphic [8] or whether a Hamiltonian cycle exists in the given graph [2]. A number of interactive proof system (protocols) were designed with challenges chosen unpredictably from a set of two possible choices. A cheating Prover can guess the right answer with probability $\frac{1}{2}$ for such protocols. Parties can run such a protocol K times to reduce chances of a maybe cheating Prover to 2^{-K}. Protocols with challenges chosen from a large set were introduced for proving knowledge of a discrete logarithm [15] and for a root in a group of a hidden order [11]. Protocols of this class achieve small probability for a cheating Prover to convince a honest Verifier.

1.1 Our contribution

We consider interactive proof systems with challenges chosen from a large set for deciding validity of statements about graphs. We introduce a polynomial representation of graphs and re-state deciding isomorphism and Hamiltonicity as testing properties of polynomials. We introduce protocols of argument type to decide directed graph isomorphism and Hamiltonicity. Protocols achieve soundness error exponentially small in bitsize of group order of the group used to produce commitments and are honest verifier zero knowledge.

1.2 Related results

A protocol with binary challenges for deciding Graph Isomorphism was introduced by Goldreich, Micali and Wigderson [8]. Prover chooses an initial random permutation of vertices of one graph and sends such a permuted graph to Verifier. Verifier chooses one of two input graphs as his challenge. Prover responds with isomorphism from his initial random graph to the graph chosen by Verifier. Any cheating Prover with a pair of non-isomorphic graphs has at most $\frac{1}{2}$ probability to show isomorphism

from any graph send as his first message to the graph chosen by Verifier as his challenge. A honest Prover providing isomorphism from his initial random graph to just one of input graphs does not open isomorphism of input graphs by running this protocol. Any Verifier can produce a transcript which is indistinguishable from this protocol transcript with a honest Prover.

A protocol with binary challenges for deciding Hamiltonicity was introduced by Blum [2]. Prover chooses an initial permutation of vertices of the given graph and sends permuted graph. Verifier chooses whether Prover should show isomorphism from permuted graph to the graph tested, or show the cycle in the permuted graph.

A protocol with binary challenges for proving knowledge of a logarithm in discrete group with Prover responses that are polynomials linear in challenge was introduced by Chaum et al [3]. This protocol was further extended by Schnorr [15] to challenges chosen from a large set resulting in only a negligible error for a single run of protocol.

Polynomial set representation was introduced by Minsky, Trachtenberg and Zippel [12] in the context of two parties holding almost the same sets that need to find missing set elements. An alternative set representation was introduced with a protocol for deciding approximate set matching [6] and an associated signature scheme tolerating "small" changes to signing keys that are below a threshold. Polynomial sequence representation was introduced with a protocol for multiple substring matching [7].

Polynomial identity testing was introduced [16] as a fast probabilistic method of verifying whether a given polynomial is identically zero. Testing properties of polynomials without revealing them was developed for a protocol to decide whether error weight in a damaged codeword of Goppa code is below a threshold [5]. This technique was also used with protocols of Neff [13] and Groth [10].

2 Preliminaries

We consider polynomials defined over ring of residue classes $\mathbb{Z}_q$ modulo q, which is a field for a prime q. We commit to elements of this field with Pedersen commitment scheme [14]. We reduce testing language membership to testing properties of polynomials as described in this section.

2.1 Commitment scheme

We say an algorithm has an *advantage* of producing some data if probability to output that data is better than guessing. We say a function is *negligible* if it is asymptotically smaller than any power of it's argument.

We say a *commitment scheme* is a triple of algorithms to generate parameters of the scheme, produce and open a commitment such that:

1. Any commitment can always be opened with data it was produced from;
2. (*Binding*) For any given commitment, it is hard to find an alternative data producing the same commitment;
3. (*Hiding*) There is only a negligible advantage for any polynomial-time algorithm to output data any given commitment was produced from.

We use Pedersen commitment scheme [14] in this paper to commit to labels assigned to vertices of graphs and to coefficients of polynomials.

Generate algorithm of Pedersen commitment scheme outputs group description and two group elements g, h. We assume a subgroup of order q of multiplicative group of invertible elements of residue classes modulo p for prime q, p in this paper such that $q|(p-1)$. Group operation is product modulo p. A security parameter is an input to Generate algorithm to specify how large should be q and p.

Commit algorithm outputs group element

$$C = g^x h^r \pmod{p} \tag{1}$$

for input data $x \in \mathbb{Z}_q$ and for some r chosen at random from Z_q.

Open algorithm outputs a yes/no response depending on whether (1) holds for given commitment C, input x, random r and parameters g, h, q, p.

Discrete Logarithm Problem should be hard in the group chosen for binding property to hold, which means p should be at least 1500 bits long and q should be at least 160 bits long. Any party capable of producing alternative (x', r') such that $g^{x'} h^{r'} = g^x h^r$ can also output $\log_h(g) = \frac{r'-r}{x'-x}$. We assume Verifier generates parameters of the scheme such that Prover does not know $\log_h(g)$. We introduce a protocol of argument type in this paper such that a cheating Prover is required to find an alternative data to open zero commitment $g^0 h^0$ unless Verifier happen to choose a root of a polynomial as his challenge.

Hiding property of Pedersen commitment scheme is unconditional. Probability distribution of Commit algorithm output is flat over the group in case of distribution of r is flat and r is independent of x, which means zero advantage for any adversary algorithm with maybe unlimited resources.

We omit $\pmod{q}$ and $\pmod{p}$ in this paper in case it is clear from the context. We stress that standard group operation can be extended to multiplication of a group element by a number, which leads to residue classes modulo group order in case of a cyclic group, which is a finite field element in case of a group of prime order.

2.2 Polynomial identity testing

Let $f(x)$ be a nonzero polynomial with coefficients from a finite field $\mathbb{Z}_q$. According to fundamental theorem of algebra, this polynomial has at most $\deg(f(x))$ roots so there is at most $\frac{\deg(f(x))}{|S|}$ probability to choose one by choosing argument value at random from a set $S \subset \mathbb{Z}_q$ (Schwartz-Zippel lemma [16]):

$$\Pr\left[c \in_R S \mid f(c) = 0 \wedge f(x) \not\equiv 0\right] \leq \frac{\deg(f(x))}{|S|}$$

Probability estimate can also be given for the case of bivariate polynomial.

We introduce protocols in this paper such that Verifier is testing whether a polynomial chosen by Prover is identically zero.

2.3 Graph isomorphism and Hamiltonicity

We say two directed graphs are *isomorphic* if a bijection exists from vertices of one graph to vertices of the other such that there is an arc in one

graph between any pair of vertices if, and only if there is an arc between corresponding vertices in the other graph.

We say an alternating sequence of vertices and arcs $A, \overrightarrow{AB}, B, \ldots, E, \overrightarrow{EA}, A$ such that the first and the last vertex of the sequence are the same and all other vertices of the graph are listed exactly once is a *Hamiltonian cycle*. We also say such a graph is *Hamiltonian*.

We only consider directed graphs in this paper.

2.4 Interactive proof systems

We say a party that follows a protocol is *honest*, and we say *any* or *cheating* otherwise. Following [8], we say an *interactive proof system* (a protocol) for a language $\mathbb{L}$ is an interactive pair of Turing machines with a *word* x on common input tape such that Verifier always outputs an Accept/Reject decision in time polynomial in $|x|$ and completeness and soundness holds. We require a honest Verifier to always accept for a honest Prover and $x \in \mathbb{L}$, which is zero *completeness error*. We also require only a negligible probability for a honest Verifier to accept for any Prover and $x \notin \mathbb{L}$, which is negligible *soundness error*. We say a protocol is *zero knowledge* if a polynomial-time *simulator* algorithm exists producing simulated transcript that is indistinguishable from a transcript of a session with a Prover in case $x \in \mathbb{L}$. Following [1] we say a protocol is *special honest verifier zero knowledge* if simulator algorithm accepting Verifier challenge as input produces simulated transcript that is indistinguishable from transcripts of all sessions with a honest Verifier and with given challenge. We say a protocol is an *argument* if soundness only holds on some assumption like hardness of Discrete Logarithm problem. Both Prover and Verifier are polynomial time machines for argument protocols. Prover machine for argument protocol has a private input tape with *witness* to language membership (of polynomial size) on it.

2.5 Protocols with algebraic responses

A protocol with binary challenges to prove knowledge of a logarithm in a group was introduced by Chaum, Evertse, van de Graaf and Peralta [4]. A variant of this protocol was introduced later [3] such that Prover produces his response by evaluating a polynomial linear in challenge of Verifier. It was improved by Schnorr [15] by choosing challenges from finite field resulting in only a negligible soundness error in just 3 rounds.

A protocol to prove knowledge of a logarithm in a group is shown on Figure 1. Common input is group element y, witness is x, challenge is e, response is a linear polynomial $zx + \alpha$ evaluated at $z = e$.

Any Prover producing alternative acceptable responses with a non-negligible probability would break Discrete Logarithm problem.

3 Polynomial representation

We introduce a novel polynomial graph representation in this section. We also outline polynomial representation of multisets.

3.1 Set representation

Original polynomial representation was introduced in the context of "set reconciliation" problem of two sets with only a small difference. Consider

Common input: group generator g of a prime order q, group element y.
Private input of Prover: $x \in \mathbb{Z}_q$ such that $y = g^x$.

1. Prover chooses some α at random, produces W_0 and sends it to Verifier:

$$\alpha \in_R \mathbb{Z}_q, \qquad W_0 = g^\alpha \pmod p \tag{2}$$

2. Verifier chooses and sends a challenge:

$$e \in_R \mathbb{Z}_q \tag{3}$$

3. Prover produces and sends a response:

$$X = ex + \alpha \pmod q \tag{4}$$

4. Verifier accepts if

$$g^X y^{-e} W_0^{-1} = 1 \pmod p \tag{5}$$

Figure 1: A protocol to prove knowledge of a discrete logarithm

a set $S = \{s_j\}$ of elements of a finite field $\mathbb{F}_q$.

Definition 1 (Set characteristic polynomial [12])**.** *Characteristic polynomial* of a set S is

$$f(z) = \prod_{s_j \in S} (z - s_j)$$

3.2 Graph representation

Consider a weighted directed graph $\mathbf{\Gamma} = (\mathbf{V}, \mathbf{E})$ with arcs $\vec{a} = \overrightarrow{HT} \in \mathbf{E}$ and vertices $v \in \mathbf{V}$. We assign nonzero weights $w_v \in \mathbb{F}_q$ to vertices of the graph. We also assign binary flags $u_a \in \{0, 1\}$ to arcs of the graph.

Definition 2 (Characteristic polynomial)**.** Let $\{w_v, v \in \mathbf{V}\}$ be a set of labels assigned to vertices of a labelled directed graph $\mathbf{\Gamma} = (\mathbf{V}, \mathbf{E})$ and let $m = |\mathbf{V}|$. We say a map from a set of field elements to ring of bivariate polynomials with coefficients from $\mathbb{Z}_q$ is *graph characteristic polynomial* of the graph $\mathbf{\Gamma}$:

$$\begin{aligned} f_C : \quad & \mathbb{Z}_q^m \longrightarrow F[\mathbb{Z}_q^2] \\ f_C(x, y, \mathbf{\Gamma}) = & \prod_{\overrightarrow{HT} \in \mathbf{E}} (1 + x b_H + y b_T) \end{aligned} \tag{6}$$

We observe that labels w_v is witness of Prover such that he can efficiently test language membership and that Verifier has no access to witness while running a protocol. We also observe that Verifier has access to Prover responses (4) and that witness is "included" in Prover response of Chaum-Evertse-vdGraaf protocol.

Definition 3 (Verification polynomial)**.** Let $\{w_v, v \in \mathbf{\Gamma}\}$ be a set of labels assigned to vertices of a directed graph $\mathbf{\Gamma}$ from Definition 2. Let $\{\Theta_v(z), v \in \mathbf{\Gamma}\}$ be a set of univariate linear polynomials over $\mathbb{Z}_q$ defined

with labels $\{w_v, v \in \mathbf{\Gamma}\}$ and some $\{\alpha_v, v \in \mathbf{\Gamma}\}$ chosen by Prover while running a protocol shown on Figure1:

$$\Theta_v(z) = zw_v + \alpha_v$$

We say a map from a set of pairs $\{(w_v, \alpha_v), v \in \mathbf{V}\}$ of field elements to ring of polynomials with coefficients from $\mathbb{Z}_q$ is *graph verification polynomial*:

$$f_V : \quad (\mathbb{Z}_q^2)^m \longrightarrow F[\mathbb{Z}_q^3]$$
$$f_V(x, y, z, \mathbf{\Gamma}) = \prod_{\overrightarrow{HT} \in \mathbf{E}} (z + xB_H(z) + yB_T(z))$$

It is clear that $f_V()$ is a polynomial of power exactly $|\mathbf{V}|$ in z and that the top coefficient (in z) of verification polynomial is characteristic polynomial. We let Verifier test properties of characteristic polynomial by testing verification polynomial with our protocols. It is clear that Verifier can directly evaluate graph verification polynomial with responses of Prover at some point $z = e$ chosen as a challenge.

We observe that Verifier needs to test properties of a subgraph which is a Hamiltonian cycle, and such a subgraph is the witness to Hamiltonicity language membership. We assign '1' flags to arcs of the subgraph and '0' flags to the rest of arcs of the graph. We commit to flags with Pedersen scheme and prove knowledge of flags with protocol shown of Figure1.

Definition 4 (Marking polynomial)**.** Let $\{\Phi_a(z'), a \in \mathbf{E}\}$ be a set of univariate linear polynomials over $\mathbb{Z}_q$ defined with binary flags $\{u_a, a \in \mathbf{E}\}$ and some $\{\beta_a, a \in \mathbf{E}\}$ chosen by Prover while running a protocol shown on Figure1:

$$\Phi_a(z') = z'u_a + \beta_a$$

Let $n = |\mathbf{E}|$ and $\{\Theta_v(z), v \in \mathbf{\Gamma}\}$ be a set of polynomials from Definition 3.

We say a map from $\{(u_a, \beta_a), a \in \mathbf{E}\} \times \{(w_v, \alpha_v), v \in \mathbf{V}\}$ to ring of polynomials with coefficients from $\mathbb{Z}_q$ is *graph marking polynomial*:

$$f_M : \quad (\{0,1\} \times \mathbb{Z}_q)^n \times (\mathbb{Z}_q^2)^m \longrightarrow F[\mathbb{Z}_q^4]$$
$$f_M(x, y, z, z', \mathbf{\Gamma}) = \prod_{a = \overrightarrow{HT} \in \mathbf{E}} (z'z + \Phi_a(z')(xB_H(z) + yB_T(z)))$$

It is clear that $f_M()$ is a polynomial of power exactly $|\mathbf{V}|$ in z' and that the top coefficient (in z') of marking polynomial of a graph is verification polynomial of the subgraph defined with '1' flags.

In the following we may omit reminders for vertices and arcs while referring to a set of labels assigned to all vertices of the graph or to a set of flags assigned to all arcs.

4 Fixed-round protocols for graphs

4.1 Deciding Graph Isomorphism

Consider a case of two graphs with distinct labels assigned to vertices.

Theorem 1. *Let $\mathbf{\Gamma}' = (\mathbf{V}', \mathbf{E}')$ be a labelled directed graph with pairwise distinct nonzero labels from $\mathbb{Z}_q$ for a prime q and let $\mathbf{\Gamma} = (\mathbf{V}, \mathbf{E})$ be another labelled directed graph such that $|\mathbf{V}'| = |\mathbf{V}|$ and $|\mathbf{E}'| = |\mathbf{E}|$. Then $\mathbf{\Gamma}$ and $\mathbf{\Gamma}'$ are isomorphic if, and only if $\mathbf{\Gamma}$ admits a label assignment such that characteristic polynomials are the same:*

$$f_C(x, y, \mathbf{\Gamma}) - f_C(x, y, \mathbf{\Gamma}') \equiv 0$$

Proof. Let $\mathbf{\Gamma}, \mathbf{\Gamma}'$ be two isomorphic directed labelled graphs with nonzero pairwise distinct labels $\{b_{v'}, v' \in \mathbf{V}'\}$ assigned to $\mathbf{\Gamma}'$. Assign the same labels $\{w_v, v \in \mathbf{\Gamma}\}$ according to vertex mapping defined by isomorphism. Then for any arc $\overrightarrow{H'T'} \in \mathbf{E}'$ there is a multiplier $(1 + xb_{H'} + yb_{T'})$ in $f_C(x, y, \mathbf{\Gamma}')$. Since graphs are isomorphic, there is $(1 + xb_H + yb_T)$ multiplier in $f_C(x, y, \mathbf{\Gamma})$ if, and only if the same multiplier is there in $f_C(x, y, \mathbf{\Gamma}')$. It follows characteristic polynomials of isomorphic graphs are the same.

For the other way, consider a set of all bivariate polynomials of the form (6) with nonzero pairwise distinct labels w_v, which is a subset of the ring of polynomials $F[\mathbb{Z}_q^2]$ over the field. Observe that this set is unique factorization domain. It follows there is the only acceptable label assignment for graph $\mathbf{\Gamma}$ and this assignment defines a bijection between vertices of graphs, such that graphs are isomorphic. □

Verifier assigns arbitrary non-zero labels to vertices of $\mathbf{\Gamma}'$ such that labels assigned to any two different vertices are not the same. Prover assigns labels to vertices of $\mathbf{\Gamma}$ to be the same as labels assigned to corresponding vertices of $\mathbf{\Gamma}'$ according to isomorphism, commits to labels of $\mathbf{\Gamma}$ with Pedersen scheme before starting the protocol. Prover shows his knowledge of labels of $\mathbf{\Gamma}$ committed and assists Verifier testing that characteristic polynomials of given graphs are the same. Verifier test it at some point (x_c, y_c) chosen at random as a challenge. Verifier do it by evaluating verification polynomial at $z = e$ chosen as another challenge and testing whether top coefficient (in z) of verification polynomial is as expected. Protocol is shown on Figure 2, simulator is shown on Figure 4.

4.2 Deciding Hamiltonicity

Consider a case of a directed graph $\mathbf{\Gamma}$ with a Hamiltonian cycle. Let σ be a generator of a cyclic multiplicative group of order m. Such a σ can be selected by choosing some nonzero σ' at random from $\mathbb{Z}_q$, producing $\sigma = \sigma'^{\frac{q-1}{m}} \pmod q$ and testing $\sigma \neq 1$.

Theorem 2. *Let $\mathbf{\Gamma}' = (\mathbf{V}, \mathbf{E}')$ be a labelled directed graph with $|\mathbf{E}'| = |\mathbf{V}| = m$ for a prime m such that $m|(q-1)$ and let $\sigma \in \mathbb{Z}_q$ be a generator of a multiplicative group of order m in $\mathbb{Z}_q$. Then $\mathbf{\Gamma}'$ is Hamiltonian cycle if, and only if it admits label assignment such that characteristic polynomial is*

$$f_C(x, y, \mathbf{\Gamma}') = \prod_{j=0}^{m-1} (1 + x\sigma^j + y\sigma^{j+1}) \tag{7}$$

Proof. Let $\mathbf{\Gamma}' = (\mathbf{V}, \mathbf{E}')$ be a subgraph of a directed labelled graph $\mathbf{\Gamma} = (\mathbf{V}, \mathbf{E})$ such that $\mathbf{E}'$ is the set of arcs of the cycle. Let σ be a generator as described. Enumerate vertices along the cycle: $v_0, v_1, \ldots, v_{m-1}, v_0$.

Assign labels to vertices along the cycle: $\{\sigma^j \pmod q, j = 0 \dots m-1\}$. It follows graph characteristic polynomial is (7).

For the other way, all labels of the form σ^j are nonzero and pairwise distinct along the cycle except the first and the last: $\sigma^0 = \sigma^m = 1$. From unique factorization of graph characteristic polynomial into linear polynomials of the form $(1 + x\sigma^j + y\sigma^{j+1})$ it follows the only label assignment exists that gives characteristic polynomial of the form (7), and this assignment defines the cycle. □

Verifier tests that characteristic polynomial of some subgraph $\mathbf{\Gamma}'$ of the given graph $\mathbf{\Gamma}$ is (7) with our protocol. Verifier evaluates graph marking polynomial at some $z' = t$ chosen as third challenge and tests that top coefficient (in z') is as expected. In particular, top coefficient (in z) is (7) evaluated at (x_c, y_c) chosen as first challenge. Prover assigns labels and flags and commits to them before running the protocol. Prover shows his knowledge of flags and labels committed such that Verifier can use responses to evaluate marking polynomial.

Protocol is shown on Figure 3, simulator is shown on Figure 5.

4.3 Properties of protocols

It can be checked that protocols shown on Figure 2 and on Figure 3 have zero completeness error.

Theorem 3. *Protocol shown on Figure 2 is an argument on assumption of hardness of Discrete Logarithm problem.*

Proof. Consider a honest Verifier running a protocol with a cheating Prover and non-isomorphic graphs as common input. Let $\{\omega_v, v \in \mathbf{\Gamma}\}$ be some assignment of labels committed to by Prover. From Theorem 1 it follows that for any such an assignment

$$f_1(x, y) = f_C(x, y, \mathbf{\Gamma}) - f_C(x, y, \mathbf{\Gamma}') \not\equiv 0$$

From Schwartz-Zippel lemma it follows that there is at most $\frac{2n}{q}$ probability for Verifier to choose his challenge (x_c, y_c) that is a root of $f_1(x, y)$. Let $f_1(x_c, y_c) \neq 0$. Then for any $\{\zeta_k, k = 0 \dots n-1\}$ chosen by Prover it holds that

$$f_2(z) = f_V(x_c, y_c, z, \mathbf{\Gamma}) - z^n f_C(x_c, y_c, \mathbf{\Gamma}') - \sum_{k=0}^{n-1} z^k \zeta_k \not\equiv 0$$

From Schwartz-Zippel lemma it follows that there is at most $\frac{n}{q}$ probability for Verifier to choose his second challenge $z = e$ that is a root of $f_2(z)$. Now if $f_2(e) \neq 0$ then such a Prover can also produce a logarithm from his responses:

$$\log_h(g) = \frac{-\Psi + \sum_{k=0}^{n-1} \eta_k e^k}{f_2(e)}$$

It follows probability for Verifier to accept is at most $\frac{3n}{q}$, which is negligible in security parameter, on assumption that Discrete Logarithm problem is hard for Prover. □

Theorem 4. *Protocol shown on Figure 3 is an argument on assumption of hardness of Discrete Logarithm problem.*

Proof. Consider a honest Verifier running a protocol with a cheating Prover with a graph $\mathbf{\Gamma} = (\mathbf{V}, \mathbf{E})$ as common input such that no Hamiltonian cycle exists in $\mathbf{\Gamma}$. Let $\{\omega_v, v \in \mathbf{V}\}$ and $\{u_a, a \in \mathbf{E}\}$ be any assignment of labels and flags chosen by Prover. From Theorem 2 it follows that for any such an assignment and for any subgraph $\mathbf{\Gamma}' = (\mathbf{V}, \mathbf{E}')$ such that $|\mathbf{E}'| = |\mathbf{V}|$ it holds that

$$f_1(x, y) = f_C(x, y, \mathbf{\Gamma}') - \prod_{j=0}^{m-1} (1 + x\sigma^j + y\sigma^{j+1}) \not\equiv 0$$

From Schwartz-Zippel lemma it follows that there is at most $\frac{2n}{q}$ probability for Verifier to choose his challenge (x_c, y_c) that is a root of $f_1(x, y)$. Let $f_1(x_c, y_c) \neq 0$. Then for any $\{\zeta_k, k = 0 \dots n-1\}$ chosen by Prover it holds that

$$f_2(z) = f_M(x_c, y_c, z, \mathbf{\Gamma}') - z^n \prod_{j=0}^{m-1} (1 + x_c\sigma^j + y_c\sigma^{j+1}) - \sum_{k=0}^{n-1} z^k \zeta_k \not\equiv 0$$

From Schwartz-Zippel lemma it follows that there is at most $\frac{n}{q}$ probability for Verifier to choose his second challenge $z = s$ that is a root of $f_2(z)$. Let $f_2(s) \neq 0$. Then for any set of binary flags $\{u_a, a \in \mathbf{\Gamma}\}$, for any set of labels $\{w_v, v \in \mathbf{V}\}$ and for any set of coeffifients $\{d_i, i = 0 \dots n-1\}$ chosen by Prover it holds that

$$f_3(z') = f_M(x_c, y_c, s, z', \mathbf{\Gamma}) - (z')^n f_2(s) - \sum_{i=0}^{n-1} (z')^i d_i \not\equiv 0$$

From Schwartz-Zippel lemma it follows that there is at most $\frac{n}{q}$ probability for Verifier to choose his third challenge $z' = t$ that is a root of $f_3(z')$. Let $f_3(t) \neq 0$. Then such a Prover can also produce a logarithm from his responses:

$$\log_h(g) = \frac{-\Psi + t^n \sum_{k=0}^{n-1} \eta_k s^k + \sum_{i=0}^{n-1} \mu_i t^i}{f_3(t)}$$

Let $u_a \notin \{0, 1\}$ for some $a \in \mathbf{V}(\mathbf{\Gamma})$. Then for any (τ_a, ρ_a) chosen by Prover it holds that

$$\Xi_a(z') = \Phi_a(z')(\Phi_a(z') - z') - z'\tau_a - \rho_a \not\equiv 0$$

From Schwartz-Zippel lemma it follows that there is at most $\frac{2}{q}$ probability for Verifier to choose his challenge $z' = t$ that is a root of $\Xi_a(z')$. Let $\Xi_a(t) \neq 0$. Then such a Prover can also produce a logarithm from his responses:

$$\log_h(g) = \frac{-\Lambda_a + t\chi_a + \lambda_a}{\Xi_a(t)}$$

It follows optimal strategy for any Prover is to cheat at (14) maximizing probability for a honest Verifier to accept. It also follows probability for a honest Verifier to accept is at most $\frac{4n}{q}$, which is negligible in security parameter, on assumption that Discrete Logarithm problem is hard for Prover. □

Simulated transcripts produced by algorithms shown on Figure 4 and on Figure 5 have flat distribution. It follows protocols shown on Figure 2 and on Figure 3 are special honest verifier perfect zero knowledge.

5 Discussion

Protocol for deciding Hamiltonicity depends on label assignment such that labels are elements of a multiplicative group of a prime order. This protocol is only suitable for graphs with a prime number of vertices.

6 Remarks

Protocols described in this paper were presented at the 9th Central European Conference on Cryptography. Preliminary results were presented at the 7th "Mathematics and Security of Information Technologies" conference. Research on protocols for decision problems with sets was started while visiting National University of Singapore.

References

[1] Mihir Bellare, Silvio Micali, and Rafail Ostrovsky. The (true) complexity of statistical zero knowledge. In *STOC*, pages 494–502, 1990.

[2] Ran Canetti and Marc Fischlin. Universally composable commitments. In *CRYPTO*, pages 19–40, 2001.

[3] David Chaum, Jan-Hendrik Evertse, and Jeroen van de Graaf. An improved protocol for demonstrating possession of discrete logarithms and some generalizations. In *EUROCRYPT*, pages 127–141, 1987.

[4] David Chaum, Jan-Hendrik Evertse, Jeroen van de Graaf, and René Peralta. Demonstrating possession of a discrete logarithm without revealing it. In *CRYPTO*, pages 200–212, 1986.

[5] Vadym Fedyukovych. Argument of knowledge of a codeword of Goppa code and a bounded error (in Russian). *Applied Discrete Mathematics (PDM)*, (4):64–71, 2009. `http://mi.mathnet.ru/pdm151`.

[6] Vadym Fedyukovych. A signature scheme with approximate key matching (in Russian). In *Information Technology and Systems Conference*, pages 396–400, 2009. `http://www.iitp.ru/ru/conferences/539.htm`.

[7] Vadym Fedyukovych and Vitaliy Sharapov. A protocol for K-multiple substring matching (in Russian). In *Information Technology and Systems Conference (ITaS)*, pages 459–466, 2008. `http://www.iitp.ru/ru/conferences/340.htm`.

[8] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.

[9] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[10] Jens Groth. A verifiable secret shuffle of homomorphic encryptions. In *Public Key Cryptography*, pages 145–160, 2003.

[11] Louis C. Guillou and Jean-Jacques Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both trasmission and memory. In *EUROCRYPT*, pages 123–128, 1988.

[12] Y. Minsky, A. Trachtenberg, and R. Zippel. Set reconciliation with nearly optimal communication complexity. In *International Symposium on Information Theory*, page 232, 2001.

[13] C. Andrew Neff. A verifiable secret shuffle and its application to e-voting. In *ACM Conference on Computer and Communications Security*, pages 116–125, 2001.

[14] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO*, pages 129–140, 1991.

[15] Claus-Peter Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3):161–174, 1991.

[16] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.

Common input: graphs $\mathbf{\Gamma}$, $\mathbf{\Gamma}'$ group description and elements p, q, g, h, a set of pairwise distinct nonzero labels $\{w_v\}$ assigned to vertices of graph $\mathbf{\Gamma}'$.
Let $|\mathbf{E}(\mathbf{\Gamma})| = |\mathbf{E}(\mathbf{\Gamma}')| = n$.
Input of Prover: labels $\{w_v\}$ assigned to vertices of graph $\mathbf{\Gamma}$ according to graph isomorphism.

1. Prover chooses $\{r_v\}$, $\{\alpha_v\}$, $\{\gamma_v\}$ at random from $\mathbb{Z}_q$, produces and sends $\{W_v\}$, $\{R_v\}$:

$$W_v = g^{w_v} h^{r_v}, \qquad R_v = g^{\alpha_v} h^{\gamma_v} \pmod{p}$$

2. Verifier chooses (x_c, y_c) at random from $\mathbb{Z}_q$ and sends it to Prover
3. Prover chooses $\{\eta_k\}$ at random from $\mathbb{Z}_q$, produces $\{\zeta_k, k = 0 \dots n\}$, $\{M_k, k = 0 \dots n-1\}$, sends $\{M_k\}$:

$$\prod_{\overrightarrow{HT} \in \mathbf{E}(\mathbf{\Gamma})} (z + x_c(z w_H + \alpha_H) + y_c(z w_T + \alpha_T)) = \sum_{k=0}^{n} z^k \zeta_k, \quad M_k = g^{\zeta_k} h^{\eta_k}$$

4. Verifier chooses e at random from $\mathbb{Z}_q$ and sends it to Prover
5. Prover produces and sends $\{\Theta_v\}$, $\{\Omega_v\}$, Ψ:

$$\Theta_v = e w_v + \alpha_v, \qquad \Omega_v = e r_v + \gamma_v \qquad \Psi = \sum_{k=0}^{n-1} \eta_k e^k \pmod{q}$$

6. Verifier produces

$$F = \prod_{\overrightarrow{HT} \in \mathbf{E}(\mathbf{\Gamma})} (e + (x_c \Theta_H + y_c \Theta_T)) - e^n \prod_{\overrightarrow{HT} \in \mathbf{E}(\mathbf{\Gamma}')} (1 + (x_c w_H + y_c w_T)) \tag{8}$$

Verifier accepts if

$$g^{\Theta_v} h^{\Omega_v} W_v^{-e} R_v^{-1} = 1 \pmod{p} \tag{9}$$

$$g^F h^\Psi \prod_{k=0}^{n-1} M_k^{-e^k} = 1 \pmod{p} \tag{10}$$

Figure 2: An argument for Graph Isomorphism

Common input: graph $\mathbf{\Gamma}$, group descriptions and elements p, q, g, h, σ.
Let $n = |\mathbf{E}(\mathbf{\Gamma})|$, $m = |\mathbf{V}(\mathbf{\Gamma})|$ such that m is prime, $m|(q-1)$, $\sigma^{\frac{q-1}{m}} \neq 1 \pmod q$.
Input of Prover: labels $w_v = \sigma^j$ with j increasing along the cycle and flags $u_a \in \{0,1\}$ such that $u_a = 1$ for arcs of the cycle, $u_a = 0$ for all other arcs.

1. Prover chooses $\{r_v\}$, $\{\delta_a\}$, $\{\alpha_v\}$, $\{\beta_a\}$, $\{\gamma_v\}$, $\{\pi_a\}$ at random from $\mathbb{Z}_q$, produces and sends $\{W_v\}$, $\{U_a\}$, $\{R_v\}$, $\{Q_a\}$:

$$W_v = g^{w_v} h^{r_v}, \quad U_a = g^{u_a} h^{\delta_a}, \quad R_v = g^{\alpha_v} h^{\gamma_v}, \quad Q_a = g^{\beta_a} h^{\pi_a} \pmod p$$

2. Verifier chooses (x_c, y_c) at random from $\mathbb{Z}_q$ and sends it to Prover
3. Prover chooses $\{\eta_k\}$, produces $\{\zeta_k, k = 0 \dots n\}$, $\{M_k, k = 0 \dots n-1\}$, sends $\{M_k\}$:

$$\prod_{\overrightarrow{HT} \in \mathbf{E}(\mathbf{\Gamma})} (z + x_c(z w_H + \alpha_H) + y_c(z w_T + \alpha_T)) = \sum_{k=0}^{n} z^k \zeta_k, \quad M_k = g^{\zeta_k} h^{\eta_k}$$

4. Verifier chooses s at random from $\mathbb{Z}_q$ and sends it to Prover
5. Prover chooses $\{\mu_i\}$, $\{\chi_a\}$, $\{\lambda_a\}$ at random from $\mathbb{Z}_q$, produces $\{\Theta_v\}$, $\{\Omega_v\}$, $\{d_i, i = 0 \dots n\}$, $\{D_i, i = 0 \dots n-1\}$, $\{\tau_a\}$, $\{\rho_a\}$, $\{N_a\}$, $\{E_a\}$, sends $\{\Theta_v\}$, $\{\Omega_v\}$, $\{D_i\}$, $\{N_a\}$, $\{E_a\}$:

$$\Theta_v = s w_v + \alpha_v, \qquad \Omega_v = s r_v + \gamma_v \pmod q$$

$$(z' u_a + \beta_a)(z'(u_a - 1) + \beta_a) = \tau_a z' + \rho_a, \quad N_a = g^{\tau_a} h^{\chi_a}, \quad E_a = g^{\rho_a} h^{\lambda_a}$$

$$\prod_{a = \overrightarrow{HT} \in \mathbf{E}(\mathbf{\Gamma})} (z' s + (z' u_a + \beta_a)(x_c \Theta_H + y_c \Theta_T)) = \sum_{i=0}^{n} (z')^i d_i, \quad D_i = g^{d_i} h^{\mu_i}$$

6. Verifier chooses t at random from $\mathbb{Z}_q$ and sends it to Prover
7. Prover produces and sends $\{\Phi_a\}$, $\{\Delta_a\}$, $\{\Lambda_a\}$, Ψ:

$$\Phi_a = t u_a + \beta_a, \quad \Delta_a = t \delta_a + \pi_a, \quad \Lambda_a = t \chi_a + \lambda_a \pmod q$$

$$\Psi = t^n \sum_{k=0}^{n-1} \eta_k s^k + \sum_{i=0}^{n-1} \mu_i t^i \pmod q$$

8. Verifier produces

$$F = \prod_{a = \overrightarrow{HT} \in \mathbf{E}(\mathbf{\Gamma})} (ts + \Phi_a(x_c \Theta_H + y_c \Theta_T)) - t^n s^n \prod_{j=0}^{m-1} (1 + x_c \sigma^j + y_c \sigma^{j+1}) \tag{11}$$

Verifier accepts if

$$g^{\Theta_v} h^{\Omega_v} W_v^{-s} R_v^{-1} = 1 \pmod p, \qquad g^{\Phi_a} h^{\Delta_a} U_a^{-t} Q_a^{-1} = 1 \pmod p \tag{12}$$

$$g^{\Phi_a(\Phi_a - t)} h^{\Lambda_a} N_a^{-t} E_a^{-1} = 1 \pmod p \tag{13}$$

$$g^F h^\Psi \left(\prod_{k=0}^{n-1} M_k^{-s^k} \right)^{t^n} \prod_{i=0}^{n-1} D_i^{-t^i} = 1 \pmod p \tag{14}$$

Figure 3: An argument for Hamiltonicity

Input is graphs Γ, Γ', challenges $x_c, y_c, e \in \mathbb{Z}_q$, group description and elements p, q, g, h.
Let $n = |\mathbf{E}(\mathbf{\Gamma})|$.

1. Verifier chooses $\{\Theta_v\}, \{\Omega_v\}, \Psi$ at random from $\mathbb{Z}_q$;
2. Verifier chooses random group elements: $\{W_v\}, \{M_k, k = 1 \dots n-1\}$;
3. Verifier produces

$$R_v = g^{\Theta_v} h^{\Omega_v} W_v^{-e} \pmod{p}$$

$$F = \prod_{\overrightarrow{HT} \in \mathbf{E}(\mathbf{\Gamma})} (e + (x_c \Theta_H + y_c \Theta_T)) - e^n \prod_{\overrightarrow{H'T'} \in \mathbf{E}(\mathbf{\Gamma}')} (1 + (x_c w_{H'} + y_c w_{T'}))$$

$$M_0 = g^F h^\Psi \prod_{k=1}^{n-1} M_k^{-e^k} \pmod{p}$$

Simulated transcript is $\{W_v\}, \{R_v\}, (x_c, y_c), \{M_k\}, e, \{\Theta_v\}, \{\Omega_v\}, \Psi$.

Figure 4: Simulator for Graph Isomorphism protocol

Input is graph Γ, challenges $x_c, y_c, s, t \in \mathbb{Z}_q$, group descriptions and elements p, q, g, h, σ.
Let $n = |\mathbf{E}(\mathbf{\Gamma})|, m = |\mathbf{V}(\mathbf{\Gamma})|$ such that $m|(q-1)$.

1. Verifier chooses at random from $\mathbb{Z}_q$ for each vertex v and arc a

$$\{\Theta_v\}, \{\Omega_v\}, \{\Phi_a\}, \{\Delta_a\}, \{\Lambda_a\}, \Psi$$

2. Verifier chooses random group elements

$$\{W_v\}, \{U_a\}, \{N_a\}, \{M_k, k = 0 \dots n-1\}, \{D_i, i = 1 \dots n-1\}$$

3. Verifier produces

$$R_v = g^{\Theta_v} h^{\Omega_v} W_v^{-s} \pmod{p}$$

$$Q_a = g^{\Phi_a} h^{\Delta_e} U_a^{-t} \pmod{p}$$

$$E_a = g^{\Phi_a(\Phi_a - t)} h^{\Lambda_a} N_a^{-t} \pmod{p}$$

$$F = \prod_{a = \overrightarrow{HT} \in \mathbf{E}(\mathbf{\Gamma})} (ts + \Phi_a(x_c \Theta_H + y_c \Theta_T)) - t^n s^n \prod_{j=0}^{m-1} (1 + x_c \sigma^j + y_c \sigma^{j+1})$$

$$D_0 = g^F h^\Psi \left(\prod_{k=0}^{n-1} M_k^{-s^k} \right)^{t^n} \prod_{i=1}^{n-1} D_i^{-t^i} \pmod{p}$$

Simulated transcript is $\{W_v\}, \{U_a\}, \{R_v\}, \{Q_a\}, (x_c, y_c), \{M_k\}, s, \{\Theta_v\}, \{\Omega_v\}, \{D_i\}, \{N_a\}, \{E_a\}, t, \{\Phi_a\}, \{\Delta_a\}, \{\Lambda_a\}, \Psi$.

Figure 5: Simulator for Hamiltonicity protocol