ペアリングを用いたラベル付きグラフにおける接続性のゼロ知識証明

吉岡 卓馬[1,a)]　中西 透[1,b)]　北須賀 輝明[1,c)]

概要：ネットワークで接続されたシステムは, グラフとして表現することができる. システムを利用するテナントは, 自らのシステムが正しく接続されていること (接続性), 自らのシステムが他のテナントのシステムから正しく分離されていること (分離性) について, システムの提供者であるプロバイダに確認する必要がある. プロバイダは複数のテナントのシステムを管理しているため, ネットワークトポロジーを開示せずに正しい情報を証明する手法が求められている. その解決策として, ペアリングアキュームレータを用いたグラフ情報のゼロ知識証明が提案されており, 検証時間や証明データサイズがグラフの点数, 辺数に依存しないという特徴がある. しかし, この方式はラベルを含めた証明が行われていないため, ネットワークの帯域やコストなどを考慮した接続性の証明ができない. 本研究ではこのペアリングベースの従来方式を拡張し, 各辺がラベルを持つグラフにおける接続性のゼロ知識証明を提案する. そして提案方式を PC 上で実装し, 処理時間を測定することによって評価を行う.

キーワード：ゼロ知識証明，グラフ，ペアリング，アキュームレータ

Zero-Knowledge Proof of Connectivity in Labeled-Graph Using Pairing-Based Accumulator

Takuma Yoshioka[1,a)]　Toru Nakanishi[1,b)]　Teruaki Kitasuka[1,c)]

Abstract: A networked system can be represented as a graph. Tenants who use the system need to check with the provider of the system that their systems are properly connected (connectivity) and that their systems are properly separated from the systems of other tenants (isolation). Since providers manage systems for multiple tenants, a method to prove the connectivity and isolation without disclosing the network topology is required. As a solution, a zero-knowledge proof of graph information using a pairing accumulator has been proposed, where the verification time and the size of the proof data do not depend on the number of graph vertexes and edges. However, since this scheme does not address labels in the graph, the provider cannot prove the connectivity w.r.t the network bandwidth and cost. In this study, we extend the previous pairing-based scheme and propose a zero-knowledge proof of the connectivity for graphs where each edge has labels. We implement the scheme on a PC and evaluate it by measuring the processing time.

Keywords: zero-knowledge proof, graph, pairing, accumulator

1. はじめに

ネットワークで接続されたシステムは, グラフを用いて表現することができる (図 1). システムを利用するテナントは, 自らのシステムが正しく接続されていること (接続性), 自らのシステムが他のテナントのシステムから正しく

[1] 広島大学大学院先進理工系科学研究科
Graduate School of Advanced Science and Engineering, Hiroshima University
[a)] m226526@hiroshima-u.ac.jp
[b)] t-nakanishi@hiroshima-u.ac.jp
[c)] kitasuka@hiroshima-u.ac.jp

分離されていること (分離性) について, システムの提供者であるプロバイダに確認する必要がある. しかし, プロバイダは複数のテナントのシステムを管理しているため, グラフの全ての情報を開示することができない. そのため, プロバイダはテナントに不必要な情報を一切与えずに, 接続性や分離性を満たしていることを証明する必要がある.

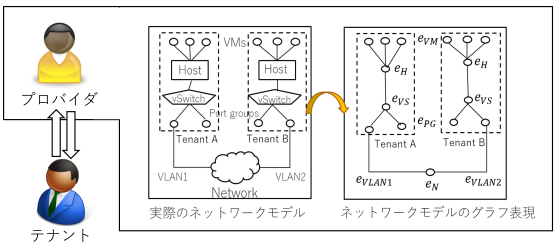

図 1 ネットワークモデル図

この問題の解決策として, ゼロ知識証明を用いた方式が提案されている [1]. ゼロ知識証明とは, ある命題が正しいということを, それ以外の情報を伝えることなく正しいと証明する手法である. ゼロ知識証明を利用することで, プロバイダはグラフの全ての情報を開示することなく, グラフ上の任意の 2 点における接続性, 分離性を証明することができる. このとき, グラフ情報は信頼できる監査人による署名が付与されることにより, その正しさが保証される.

従来方式 [1] では, RSA 暗号ベースの署名とゼロ知識証明を用いて接続性, 分離性の証明を行っている. しかし, この方式は検証時間がグラフの点数や辺数に依存して大きくなるという問題点がある. この依存を解消するために, ペアリングベースのアキュームレータを用いた方式 [2] が提案されている. この方式では, グラフ上の点集合, 辺集合をアキュームレータとしてそれぞれ 1 つの値に圧縮しているため, ゼロ知識証明にかかる検証時間がグラフの点数, 辺数に依存しないという利点がある. しかし, この方式ではグラフ中のラベルを考慮していないため, ラベルを用いて表現することができるネットワークの帯域やコストなどを考慮に入れた接続性の証明ができない.

本研究ではペアリングベースの従来方式 [2] を拡張し, 各辺がラベルを持つグラフにおける接続性のゼロ知識証明を提案する. 提案方式では, 全ての辺に対して M 個ずつ素数を割り当てることでラベルを表現する. 従来方式 [2] 同様, ペアリングベースのアキュームレータを利用することで, グラフの点数, 辺数, さらにラベル数への依存なしで接続性のゼロ知識証明を行うことができる. また, 接続性に関してネットワークの帯域などによる制約がある場面を想定し, テナントが作成する制約リストに含まれるラベルを持つ辺のみをパスに含むことを示す制約付き接続性のゼロ知識証明についても提案する. そして, これらの方式を PC 上で実装し, 処理時間を測定することによって評価を行う.

2. 数学的準備

2.1 双線形写像

本研究では, 双線形写像を構成可能な楕円曲線上の群を利用する. $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ を素数の位数 p の巡回群とし, $\mathbb{G}_1, \mathbb{G}_2$ の生成元をそれぞれ g_1, g_2 とする. このとき, 次に示す双線形写像 $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ を定義できる.

双線形性: 任意の $u \in \mathbb{G}_1, v \in \mathbb{G}_2$, および任意の $a, b \in \mathbb{Z}_p$ に対し, $e(u^a, v^c) = e(u, v)^{ab}$ が成り立つ.

非退化性: $e(g_1, g_2) \neq 1_{\mathbb{G}_T}$ ($1_{\mathbb{G}_T}$ は $\mathbb{G}_T$ 上の単位元)

このような双線形写像は楕円曲線上のペアリングにより実現できる.

2.2 安全性仮定

後述する AHO 署名は q-SFP 仮定に基づいており, ペアリングベースアキュームレータは双線形 q-SDH 仮定に基づいている.

定義 1. q-SFP(q-Simultaneous Flexible Pairing) 仮定

全ての多項式時間アルゴリズム $\mathcal{A}$ に対して, 確率

$$\begin{aligned}&\mathsf{Pr}[(z^*, r^*, s^*, t^*, u^*, v^*, w^*)\\ &\leftarrow \mathcal{A}(g_z, h_z, g_r, h_r, a, \tilde{a}, b, \tilde{b}, \{(z_j, r_j, s_j, t_j, u_j, v_j, w_j)\}_{j=1}^{q})\\ &\wedge e(a, \tilde{a}) = e(g_z, z^*)e(g_r, r^*)e(s^*, t^*)\\ &\wedge e(b, \tilde{b}) = e(h_z, z^*)e(h_r, u^*)e(v^*, w^*)\\ &\wedge z^* \neq 1_{\mathbb{G}_2} \wedge z^* \neq \{z_j\}_{j=1}^{q}]\end{aligned}$$

は無視できる. このとき, $(g_z, h_z, g_r, h_r, a, b) \in \mathbb{G}_1, (\tilde{a}, \tilde{b}) \in \mathbb{G}_2, (s_j, v_j) \in \mathbb{G}_1^2, (z_j, r_j, t_j, u_j, w_j) \in \mathbb{G}_2^5$ であり, これらは

$$\begin{aligned}&e(a, \tilde{a}) = e(g_z, z_j)e(g_r, r_j)e(s_j, t_j)\\ &\wedge e(b, \tilde{b}) = e(h_z, z_j)e(h_r, u_j)e(v_j, w_j)\end{aligned}$$

を満たす. また, $1_{\mathbb{G}_2}$ は $\mathbb{G}_2$ の単位元である.

定義 2. 双線形 q-SDH(q-Strong Diffie-Hellman) 仮定

全ての多項式時間アルゴリズム $\mathcal{A}$ に対して, $g_1, g_1^s, g_1^{s^2}, ..., g_1^{s^q} \in \mathbb{G}_1$, $g_2, g_2^s, g_2^{s^2}, ..., g_2^{s^q} \in \mathbb{G}_2$ を用いて, $(x, e(g_1, g_2)^{\frac{1}{s+x}})$ を出力する確率は無視できる. (ただし, $s \in \mathbb{Z}_p, x \in \mathbb{Z}_p^* \backslash \{-s\}$)

2.3 知識の証明

知識の証明とは, 証明者と検証者による対話型プロトコルであり, ある関係を満たす秘密情報を知っているということを, 秘密情報を開示することなく証明することである. 本研究では, 離散対数の秘密情報 x を知ることを示す知識の証明を用いる.

2.4 AHO 署名

AHO 署名 [3] とは複数の群要素メッセージに対して署名できる方式であり, 署名検証のペアリングの関係式をゼロ知識証明することである. num 個のメッセージに対する AHO 署名のアルゴリズムは以下のようになる.

- **AHOKeyGen:**
  ランダムに $G_r, H_r \in \mathbb{G}_1, \mu_z, \nu_z, \{\mu_i, \nu_i\}_{1 \leq i \leq \mathsf{num}} \in \mathbb{Z}_p$ を選び, $G_z = G_r^{\mu_z}, H_z = H_r^{\nu_z}, G_i = G_r^{\mu_i}, H_i = H_r^{\nu_i}$ を計算する. 次に, ランダムに $\alpha_a, \alpha_b \in \mathbb{Z}_p$ を選び, $A = e(G_r, g_2^{\alpha_a}), B = e(H_r, g_2^{\alpha_b})$ を計算する.
  AHO 署名の公開鍵, 秘密鍵をそれぞれ $\mathsf{pk_{AHO}} = (g_1, g_2, G_r, H_r, G_z, H_z, \{G_i, H_i\}_{1 \leq i \leq \mathsf{num}}, A, B), \mathsf{sk_{AHO}} = (\alpha_a, \alpha_b, \mu_z, \nu_z, \{\mu_i, \nu_i\}_{1 \leq i \leq \mathsf{num}})$ としてを出力する.

- **AHOSign:**
  ランダムに選んだ $\beta, \epsilon, \eta, \iota, \kappa \in \mathbb{Z}_p$ と, $\mathsf{pk_{AHO}}, \mathsf{sk_{AHO}}$ を用いて, 与えられたメッセージ $((M_1, ..., M_{\mathsf{num}}) \in \mathbb{G}_2^{\mathsf{num}})$ に, 以下のように署名 $\sigma = (\theta_1, ..., \theta_7)$ を生成し, 出力する.

$$\theta_1 = g_2^{\beta}, \theta_2 = g_2^{\epsilon - \mu_z \beta} \Pi_{i=1}^{\mathsf{num}} M_i^{-\mu_i}, \theta_3 = G_r^{\eta}, \theta_4 = g_2^{(\alpha_a - \epsilon)/\eta},$$
$$\theta_5 = g_2^{\iota - \nu_z \beta} \Pi_{i=1}^{\mathsf{num}} M_i^{-\nu_i}, \theta_6 = H_r^{\kappa}, \theta_7 = g_2^{(\alpha_b - \iota)/\kappa}$$

- **AHOVerify:**
  与えられたメッセージ $((M_1, ..., M_{\mathsf{num}}) \in \mathbb{G}_2^{\mathsf{num}})$ と署名 $\sigma = (\theta_1, ..., \theta_7)$ が以下の検証式を満たす場合に受理する.

$$A = e(G_z, \theta_1) \cdot e(G_r, \theta_2) \cdot e(\theta_3, \theta_4) \cdot \Pi_{i=1}^{\mathsf{num}} e(G_i, M_i),$$
$$B = e(H_z, \theta_1) \cdot e(H_r, \theta_5) \cdot e(\theta_6, \theta_7) \cdot \Pi_{i=1}^{\mathsf{num}} e(H_i, M_i)$$

AHO 署名は q-SFP 仮定に基づき, 存在的偽造不可能性 (existential unforgeability) が証明されている. また, AHO 署名は同じメッセージに対する別の unlinkable な署名にランダマイズ可能である.

2.5 ペアリングベースアキュームレータ

ペアリングベースアキュームレータ [4] は, ある要素が集合に含まれていることを効率よく証明できる手法である.

- **AccSetup:**
  $\mathbb{G}_1, \mathbb{G}_2$ の生成元をそれぞれ g_1, g_2 として, ランダムに $s \in \mathbb{Z}_p$ を選ぶ. acc 秘密鍵 $\mathsf{sk_{acc}} = s$, acc 公開鍵 $\mathsf{pk_{acc}} = (g_1, g_1^{s^1}, ..., g_1^{s^q}, g_2, g_2^{s^1}, ..., g_2^{s^q})$ をそれぞれ計算し, 出力する.

- **AccGen:**
  $\mathsf{sk_{acc}}, \mathsf{pk_{acc}}$ を用いて, $\mathbb{Z}_p^*$ の要素の集合 $\mathcal{X}$ のアキュームレータ $\mathsf{acc}_{\mathcal{X}}$ を以下のように計算し, 出力する.

$$\mathsf{acc}_{\mathcal{X}} = g_2^{\Pi_{x_i \in \mathcal{X}}(s + x_i)}$$

  また, $\mathsf{acc}_{\mathcal{X}} = g_2^{\Pi_{x_i \in \mathcal{X}}(s+x_i)}$ は変形すると, $\Pi_{x_i \in \mathcal{X}}(X + x_i) = \Sigma_{0 \leq k \leq |\mathcal{X}|} a_k X^k$ となる多項式の係数 $a_k \in \mathbb{Z}_p$ に対して, $\mathsf{acc}_{\mathcal{X}} = g_2^{\Sigma_{0 \leq k \leq |\mathcal{X}|} a_k s^k} = \Pi_{0 \leq k \leq |\mathcal{X}|} (g_2^{s^k})^{a_k}$ となるため, acc 秘密鍵 $\mathsf{sk_{acc}} = s$ を用いずに計算することができる.

- **AccWitGen:**
  $\mathsf{pk_{acc}}$ を用いて, 要素 $\tilde{x} \in \mathcal{X}$ に対する以下のような補助情報 W を計算し, 出力する.

$$W = g_1^{\Pi_{x_i \in \mathcal{X} \setminus \tilde{x}}(s + x_i)}$$

  この W も $\mathsf{acc}_{\mathcal{X}}$ と同様に $\mathsf{sk_{acc}}$ を用いずに計算できる.

- **AccVerify:**
  $\mathsf{pk_{acc}}, W$ を用いて, $\tilde{x} \in \mathcal{X}$ を以下の式で検証する.

$$e(W, g_2^s g_2^{\tilde{x}}) = e(g_1, \mathsf{acc}_{\mathcal{X}})$$

このアキュームレータは双線形 q-SDH 仮定に基づき安全である.

3. 先行研究

本章では, 先行研究として [2] で提案されている, グラフに対する接続性のゼロ知識証明について紹介する. この方式はペアリングに基づいており, 群演算は楕円曲線上で計算される. 従来方式 [1] と同様にグラフの各頂点に頂点識別子として素数 e_i を割り当て, 辺情報を両端の頂点識別子を掛け合わせた $e_i e_j$ と表現する. この時隣接している 2 点間の接続性は, 両端の点の値が辺の値を割り切ることを示すことで証明することができる. これを証明したい 2 点を結ぶパス上の全ての辺で行うことによって, 2 点間の接続性が証明される.

この方式では, 点情報・辺情報の正しさを証明するため, グラフ上の点集合 V, 辺集合 ε をそれぞれ以下のようにアキュームレータ $\mathsf{acc}_V, \mathsf{acc}_\varepsilon$ に圧縮した上で監査人が署名している. そして証明時には, アキュームレータと署名の正しさがゼロ知識証明される.

$$\mathsf{acc}_V = g_2^{\Pi_{i \in V}(s + e_i)}$$
$$\mathsf{acc}_\varepsilon = g_2^{\Pi_{(i,i') \in \varepsilon}(s + e_i e_{i'})}$$

アキュームレータを用いて検証を行うことができるため, 検証時間がグラフ内の点数や辺数に依存しないことが利点である.

さらに分離性の証明を行うため, グラフ内の点集合の内, 接続されているもの同士を 1 つの連結成分と定義し, L 個の連結成分に対して, それぞれの連結成分内の点集合 V_l についても点集合 V と同様に, アキュムレータ acc_{V_l} に圧縮している.

$$\mathsf{acc}_{V_l} = g_2^{\Pi_{i \in V_l}(s+e_i)}$$

そして分離性を証明する 2 点が異なるアキュームレータに存在することを示すことで, 2 点間の分離性が証明される.

この方式の問題点は, グラフでのラベルを想定していないため, ネットワークの帯域やコストなどを考慮した, ラベルの制約リストに対する接続性の証明ができないという点である.

4. ラベル付きグラフに対するゼロ知識証明のモデル

本論文では, 点集合 $\mathcal{V} = \{1, 2, ..., N\}$, ラベル付き辺集合 $\varepsilon = \{(i_1, i'_1, L_1), ..., (i_J, i'_J, L_J)\}$, 辺ラベル集合 $\Lambda = \{l_1, l_2, ..., l_{\tilde{N}}\}$ で構成されたグラフ $\mathcal{G} = (\mathcal{V}, \varepsilon, \Lambda)$ を考える. このとき, $N = |\mathcal{V}|, \tilde{N} = |\Lambda|$ であり, $L_j = (l_1, l_2, ..., l_M)$ は各辺 (i_j, i'_j) が持つ M 個のラベルの列である.

4.1 アルゴリズムの定義

グラフ署名のゼロ知識証明は以下のアルゴリズムで構成されている.

- **IssuerKeygen:**
  秘密鍵 isk, 公開鍵 ipk を生成し, 出力する.
- **Issue:**
  isk, ipk を用いて, 時刻 t に対してグラフ $\mathcal{G}$ に署名 σ を付与する.
- **ProofGen:**
  ipk, $\mathcal{G}, t, \sigma$ を用いて関係性 R の証明を行う. 本論文では, R は以下の二種類について考える.
  Connectivity(t, i, j, K, m)**:** 時刻 t のグラフにおいて, 点 i, j が接続されていることを示す. このとき, K は i, j 間のパスにおける点の数である.
  Constraint-connectivity$(t, i, j, K, m, \Lambda')$**:** 上記の Connectivity に加え, i, j 間における各辺の m 番目のラベルが全て, 検証者によって作成される制約リスト $\Lambda' = \{l_1, l_2, ...\}$ に含まれていることを示す.
- **ProofVerify:**
  証明の結果が有効なものであれば受理, そうでなければ拒否する.

4.2 安全性

グラフのゼロ知識証明の安全性について, 以下に記述する.

4.2.1 健全性

健全性とは、証明者が証明する関係 R を満たす時刻 t のグラフの署名を所有していない場合, その証明者による証明は検証者によって受理されないという性質である.

4.2.2 ゼロ知識性

ゼロ知識性とは, 検証者は公開パラメータと証明された関係以上の情報を得られないという性質である.

5. 提案方式

5.1 提案方式の概要

提案方式では, 各辺が M 個のラベルを持つグラフにおいて, 従来方式 [2] と同様ペアリングベースでの接続性のゼロ知識証明を行う. このとき, 従来方式 [1] のように, ラベルは頂点同様にラベル識別子として素数 e_l を割り当てることとする. また, 辺情報は従来方式 [2] において両端の点の頂点識別子の積 $e_i e_j$ と表現されていたが, 提案方式ではラベル付き辺情報を用いるため, さらにラベル識別子との積をとった $e_i e_j e_l$ と表現する.

まず, 点集合 V, ラベル付き辺集合 ε, およびラベル集合 Λ をそれぞれアキュームレータ $\mathsf{acc}_V, \mathsf{acc}_\varepsilon, \mathsf{acc}_\Lambda$ に圧縮する. そして, 生成したアキュームレータに AHO 署名を付与し, ラベルを含めたグラフ情報の正しさを保証する.

次に, 接続性を証明したい 2 点間のパス上の全ての頂点, 辺, および各辺の m 番目のラベルについて, アキュームレータによる包含関係を示すことでグラフに含まれていることを証明する. m の値はテナントによって指定される. そして, 従来方式 [2] と同様に両端の点の値とラベルの値が辺の値を割り切ることを示すことで, 接続性を証明する.

また, 従来通りの接続性に加えて, 特定の制約下での接続性のゼロ知識証明も行う. テナントはまず, 証明したい 2 点を結ぶパスに含めることを許容できるラベルの集合を制約リスト Λ' として設定し, プロバイダと共有するとともに. Λ' のアキュームレータを生成する. プロバイダは, 接続性を証明する 2 点間のパス上の辺ラベルが全て制約リスト Λ' に含まれていることをアキュームレータを用いて示すことで, 制約下での接続性を証明する.

5.2 グラフ符号化

従来方式 [1] と同様にグラフ情報を以下のように符号化する.

(1) 頂点 i に対し, 頂点識別子として素数 e_i を割り当てる.
(2) 各ラベル $l \in \Lambda$ に対しても素数 e_l を割り当てる. 辺 (i, j) の m 番目のラベル l に対し, $e_{(i,j,m)} = e_l$ を割り当てる.
(3) 辺 (i, j) について, ラベル付き辺として, 頂点識別子とラベル識別子の積 $e_i e_j e_{(i,j,m)} = e_i e_j e_l$ で表現する.

5.3 提案プロトコル

- **IssuerKeygen:**
  グラフ情報の署名に必要な秘密鍵や公開鍵などのパラメータを生成する.
  (1) $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ を素数位数 p の双線形群とし, g_1, g_2 を $\mathbb{G}_1, \mathbb{G}_2$ の生成元とする. また, 双線形写像 $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ として、param $=$

$(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g_1, g_2)$ とする.

(2) AHO 署名の秘密鍵 $\mathsf{sk}_{\mathsf{AHO}}$, 公開鍵 $\mathsf{pk}_{\mathsf{AHO}}$ をそれぞれ生成する.

(3) アキュムレータの秘密鍵 $\mathsf{sk}_{\mathsf{acc}}$, 公開鍵 $\mathsf{pk}_{\mathsf{acc}}$ をそれぞれ生成する.

(4) ランダムな $h_1 \in \mathbb{G}_1, h_2 \in \mathbb{G}_2$ を生成する.

(5) 秘密鍵 $\mathsf{isk} = (\mathsf{sk}_{\mathsf{AHO}}, \mathsf{sk}_{\mathsf{acc}})$, 公開鍵 $\mathsf{ipk} = (\mathsf{param}, \mathsf{pk}_{\mathsf{AHO}}, \mathsf{pk}_{\mathsf{acc}}, h_1, h_2,)$ を出力する.

- **Issue:**

$\mathsf{isk}, \mathsf{ipk}$ を用いて, 時刻 t におけるグラフ $\mathcal{G}$ をアキュムレータに圧縮し, AHO 署名を生成する.

(1) グラフ内の点集合 V は, $\mathsf{sk}_{\mathsf{acc}} = s$ を用いて以下のようにアキュムレータ acc_V に圧縮することができる.

$$\mathsf{acc}_V = g_2^{\Pi_{i \in V}(s+e_i)}$$

分離性の証明に関しては従来方式 [2] 同様に行えるようにするため, L 個の連結成分に対して, それぞれ連結成分内の点集合 V_l についても点集合 V と同様にアキュムレータ acc_{V_l} に圧縮する.

$$\mathsf{acc}_{V_l} = g_2^{\Pi_{i \in V_l}(s+e_i)}$$

(2) グラフ内のラベル集合 Λ についても同様にアキュームレータ acc_Λ に圧縮する.

$$\mathsf{acc}_\Lambda = g_2^{\Pi_{i \in \Lambda}(s+e_i)}$$

(3) グラフ内のラベル付き辺集合 ε については, すべての辺につき M 個のラベル付き辺情報を 1 つのアキュームレータ acc_ε に圧縮する.

$$\mathsf{acc}_\varepsilon = g_2^{\Pi_{(i,i',k) \in \varepsilon}(s+e_i e_{i'} e_{(i,i',k)})}$$

(4) $\mathsf{sk}_{\mathsf{AHO}}$ を用いて, 時刻 t に対して, $(g_2^t, \mathsf{acc}_V, \mathsf{acc}_{V_1}, ..., \mathsf{acc}_{V_L}, \mathsf{acc}_\Lambda, \mathsf{acc}_\varepsilon)$ をメッセージとする AHO 署名 $\sigma = (\theta_1, \cdots, \theta_7)$ を生成し, 出力する.

- **ProofGen, ProofVerify:**

$\mathsf{ipk}, \mathcal{G}, t, \sigma$ を用いて, AHO 署名とラベル付き接続性の知識証明を行う.

AHO 署名の知識証明

以下の手順で AHO 署名の知識証明を行う.

(1) [3] の手法により, AHO 署名を再ランダム化し, $\sigma' = (\theta'_1, \cdots, \theta'_7)$ を得る. これにより, 元の署名 σ とはリンクできないランダムな署名となる.

(2) $j = 1, 2, 5$ について, それぞれの θ'_j に対して, $\mathbb{Z}_p$ の中からランダムに選んだ $r_{\theta'_j}$ を用いて コミットメントを生成する.

$$C_{\theta'_j} = \theta'_j h^{r_{\theta'_j}}$$

(3) $\mathsf{acc}_V, \mathsf{acc}_\Lambda, \mathsf{acc}_\varepsilon$ に対して, $\mathbb{Z}_p$ の中からランダムに選んだ $r_{\mathsf{acc}_V}, r_{\mathsf{acc}_\varepsilon}, r_{\mathsf{acc}_\Lambda}$ を用いて以下のようにコミットメントを計算する.

$$C_{\mathsf{acc}_V} = \mathsf{acc}_V h_2^{r_{\mathsf{acc}_V}}$$
$$C_{\mathsf{acc}_\varepsilon} = \mathsf{acc}_\varepsilon h_2^{r_{\mathsf{acc}_\varepsilon}}$$
$$C_{\mathsf{acc}_\Lambda} = \mathsf{acc}_\Lambda h_2^{r_{\mathsf{acc}_\Lambda}}$$

同様に, 連結成分内の点集合 V_l に対しても同様に $\mathbb{Z}_p$ の中からランダムに選んだ $r_{\mathsf{acc}_{V_l}}$ を用いてコミットメントを生成する.

$$C_{\mathsf{acc}_{V_l}} = \mathsf{acc}_{V_l} h_2^{r_{\mathsf{acc}_{V_l}}}$$

(4) AHO 署名の検証式に上記のコミットメントを代入することで, 以下の検証式が得られる.

$$\begin{aligned}
&A^{-1} \cdot e(G_z, C_{\theta'_1}) \cdot e(G_r, C_{\theta'_2}) \cdot e(\theta'_3, \theta'_4) \cdot e(G_1, g_2^t) \\
&\cdot e(G_2, C_{\mathsf{acc}_V}) \cdot e(G_3, C_{\mathsf{acc}_\varepsilon}) \cdot e(G_4, C_{\mathsf{acc}_\Lambda}) \\
&\cdot e(G_5, C_{\mathsf{acc}_{V_1}}) \cdots e(G_{L+4}, C_{\mathsf{acc}_{V_L}}) \\
&= e(G_z, h_2)^{r_{\theta'_1}} \cdot e(G_r, h_2)^{r_{\theta'_2}} \cdot e(G_2, h_2)^{r_{\mathsf{acc}_V}} \\
&\cdot e(G_3, h_2)^{r_{\mathsf{acc}_\epsilon}} \cdot e(G_4, h_2)^{r_{\mathsf{acc}_\Lambda}} \\
&\cdot e(G_5, h_2)^{r_{\mathsf{acc}_{V_1}}} \cdots e(G_{L+4}, h_2)^{r_{\mathsf{acc}_{V_L}}} \qquad (1)
\end{aligned}$$

$$\begin{aligned}
&B^{-1} \cdot e(H_z, C_{\theta'_1}) \cdot e(H_r, C_{\theta'_5}) \cdot e(\theta'_6, \theta'_7) \cdot e(H_1, g_2^t) \\
&\cdot e(H_2, C_{\mathsf{acc}_V}) \cdot e(H_3, C_{\mathsf{acc}_\varepsilon}) \cdot e(H_4, C_{\mathsf{acc}_\Lambda}) \\
&\cdot e(H_5, C_{\mathsf{acc}_{V_1}}) \cdots e(H_{L+4}, C_{\mathsf{acc}_{v_L}}) \\
&= e(H_z, h_2)^{r_{\theta'_1}} \cdot e(H_r, h_2)^{r_{\theta'_5}} \cdot e(H_2, h_2)^{r_{\mathsf{acc}_V}} \\
&\cdot e(H_3, h_2)^{r_{\mathsf{acc}_\epsilon}} \cdot e(H_4, h_2)^{r_{\mathsf{acc}_\Lambda}} \\
&\cdot e(H_5, h_2)^{r_{\mathsf{acc}_{V_1}}} \cdots e(H_{L+4}, h_2)^{r_{\mathsf{acc}_{V_L}}} \qquad (2)
\end{aligned}$$

この 2 つの式に対して知識の証明を行う.

接続性の知識証明

$(i_1, \cdots, i_K)$ を頂点 $i(i_1 = i)$ から頂点 $j(i_K = j)$ までのパスとし, $1 \leq k \leq K-1$ としたときのそれぞれの辺 (i_k, i_{k+1}) に対して、二つの頂点の間に辺が存在することを証明する.

(1) $2 \leq k \leq K-1$ に対して, コミットされた e_{i_k} が acc_V に含まれていることを示す.

まず, 補助情報 W_{i_k} を以下のように計算する.

$$W_{i_k} = g_1^{\Pi_{t \in V, t \neq i_k}(s+e_t)}$$

補助情報 W_{i_k} に対して, $\mathbb{Z}_p$ の中からランダムに

選んだ $r_{W_{i_k}}$ を用いてコミットメントを生成する.

$$C_{W_{i_k}} = W_{i_k} h_1^{r_{W_{i_k}}}$$

頂点 e_{i_k} に対して, $\mathbb{Z}_p$ の中からランダムに選んだ $r_{e_{i_k}}$ を用いてコミットメントを生成する.

$$C_{e_{i_k}} = g_1^{e_{i_k}} h_1^{r_{e_{i_k}}}$$

ここで,

$$\alpha_k = e_{i_k} r_{W_{i_k}}$$

として, $\mathbb{Z}_p$ の中からランダムに選んだ r_{α_k} を用いてコミットメントを生成する.

$$C_{\alpha_k} = g_1^{\alpha_k} h_1^{r_{\alpha_k}}$$

さらに, β_k を以下のように計算する.

$$\beta_k = r_{\alpha_k} - r_{e_{i_k}} r_{W_{i_k}}$$

上記を用いると, 以下の検証式が得られる.

$$\begin{aligned} & e(C_{W_{i_k}}, g_2^s) \cdot e(g_1, C_{\mathsf{acc}_V})^{-1} \\ & = e(C_{W_{i_k}}, g_2^{-1})^{e_{i_k}} \cdot e(h_1, g_2)^{\alpha_k} \\ & \cdot e(h_1, g_2^s)^{r_{W_{i_k}}} \cdot e(g_1, h_2^{-1})^{r_{\mathsf{acc}_V}} \end{aligned} \tag{3}$$

$$C_{e_{i_k}} = g_1^{e_{i_k}} h_1^{r_{e_{i_k}}}, C_{\alpha_k} = g_1^{\alpha_k} h_1^{r_{\alpha_k}} \tag{4}$$

$$C_{\alpha_k} = C_{e_{i_k}}^{r_{W_{i_k}}} h_1^{\beta_k} \tag{5}$$

これらの式に対して知識の証明を行う.

(2) 各辺の m 番目のラベル $e_{(i,j,m)} = e_l$ について, 以下のいずれかを行う. このとき, m の値は検証者によって事前に指定されているものとする.

(a) 制約リストを用いない場合 ($R = \mathsf{Connectivity}(t, i, j, K, m)$)

$1 \leq k \leq K-1$ に対して, コミットされた e_l が acc_Λ に含まれていることを示す.

まず, 補助情報 W_{e_l} を以下のように計算する.

$$W_{e_l} = g_1^{\prod_{t \in \Lambda, t \neq l}(s+e_t)}$$

以下、頂点と同様に証明を行う.

(b) 制約リストを用いる場合 ($R = \mathsf{Constraint\text{-}connectivity}(t, i, j, K, m, \Lambda')$)

検証者は事前に, 点 i, j を結ぶパスに含めることを許容できるラベルの集合である, 制約リスト $\Lambda' = \{e_{l_1}, e_{l_2}, ...\}$ を作成し, 証明者と共有しているものとする.

また, 証明者, 検証者はともに, 制約リスト Λ' をアキュームレータ $\mathsf{acc}_{\Lambda'}$ に圧縮しておく.

$$\mathsf{acc}_{\Lambda'} = g_2^{\prod_{i \in \Lambda'}(s+e_i)}$$

このとき, $1 \leq k \leq K-1$ に対して, e_l が $\mathsf{acc}_{\Lambda'}$ に格納されていることを示す.

補助情報 W_{e_l} を以下のように計算する.

$$W_{e_l} = g_1^{\prod_{t \in \Lambda', t \neq l}(s+e_t)}$$

上記を用いると, 以下の検証式が得られる.

$$\begin{aligned} & e(W_{e_l}, g_2^s) \cdot e(g_1, \mathsf{acc}_{\Lambda'})^{-1} \\ & = e(g_1^{\prod_{t \in \Lambda', t \neq l}(s+e_t)}, g_2^s) \cdot e(g_1, g_2^{\prod_{t \in \Lambda'}(s+e_t)})^{-1} \\ & = e(g_1, g_2^{s+e_l})^{\prod_{t \in \Lambda', t \neq l}(s+e_t)} \\ & \quad \cdot e(g_1, g_2^{-e_l})^{\prod_{t \in \Lambda', t \neq l}(s+e_t)} \\ & \quad \cdot e(g_1, g_2)^{-\prod_{t \in \Lambda'}(s+e_t)} \\ & = e(W_{e_l}, g_2^{-1})^{e_l} \end{aligned} \tag{6}$$

これらの検証式に対して知識の証明を行う.

(3) $1 \leq k \leq K-1$ に対して, コミットされた $\epsilon_k = e_{i_k} e_{i_{k+1}} e_l$ が acc_ε に含まれていることを示す。

まず, 補助情報 W_{ϵ_k} を以下のように計算する.

$$W_{\epsilon_k} = g_1^{\prod_{(t,t') \in \varepsilon, (t,t') \neq (i_k, i_{k+1})}(s+e_t e_{t'} e_l)}$$

以下、頂点と同様に証明を行う.

(4) $2 \leq k \leq K-2$ に対して, $\epsilon_k = e_{i_k} e_{i_{k+1}} e_l$ が接続していることを示す.

(1), (2), (3) から, 以下のコミットメントが得られている.

$$\begin{aligned} C_{e_{i_k}} &= g_1^{e_{i_k}} h_1^{r_{e_{i_k}}} \\ C_{e_{i_{k+1}}} &= g_1^{e_{i_{k+1}}} h_1^{r_{e_{i_{k+1}}}} \\ C_{e_l} &= g_1^{e_l} h_1^{r_{e_l}} \\ C_{\epsilon_k} &= g_1^{\epsilon_k} h_1^{r_{\epsilon_k}} \end{aligned}$$

ここで, ラベルを情報を含まない, 両端の頂点の積のみで表現された辺情報 ϵ'_k を生成する.

$$\epsilon'_k = e_k e_{k+1}$$

この辺情報に対して, $\mathbb{Z}_p$ の中からランダムに選んだ $r_{\epsilon'_k}$ を用いてコミットメントを生成する.

$$C_{\epsilon'_k} = g_1^{\epsilon'_k} h_1^{r_{\epsilon'_k}}$$

γ'_k, γ_k を以下のように計算する.

$$\begin{aligned} \gamma'_k &= r_{\epsilon'_k} - r_{e_{i_k}} e_{k+1} \\ \gamma_k &= r_{\epsilon_k} - r_{\epsilon'_k} e_l \end{aligned}$$

上記を用いると, 以下の検証式が得られる.

$$C_{\epsilon'_k} = C_{e_{i_k}}^{e_{i_{k+1}}} h_1^{\gamma'_k} \tag{7}$$

$$C_{\epsilon_k} = C_{\epsilon'_k}^{e_l} h_1^{\gamma_k} \tag{8}$$

これらの検証式に対して知識の証明を行う.

(5) パスの両端, すなわち $\epsilon_1 = e_{i_1}e_{i_2}e_l, \epsilon_{K-1} = e_{i_{K-1}}e_{i_K}e_l$ が接続していることを示す.
(1), (2), (3) から, 以下のコミットメントが得られている.

$$C_{e_{i_2}} = g_1^{e_{i_2}} h_1^{r_{e_{i_2}}}$$
$$C_{e_{i_{K-1}}} = g_1^{e_{i_{K-1}}} h_1^{r_{e_{i_{K-1}}}}$$
$$C_{\epsilon_1} = g_1^{\epsilon_1} h_1^{r_{\epsilon_1}}$$
$$C_{\epsilon_{K-1}} = g_1^{\epsilon_{K-1}} h_1^{r_{\epsilon_{K-1}}}$$

ここで, ラベルを情報を含まない, 両端の頂点の積のみで表現された辺情報を生成する.

$$\epsilon'_1 = e_{i_1}e_{i_2}$$

$$\epsilon'_{K-1} = e_{i_{K-1}}e_{i_K}$$

この辺情報に対して, $\mathbb{Z}_p$ の中からランダムに選んだ $r_{\epsilon'_1}, r_{\epsilon'_{K-1}}$ を用いてコミットメントを生成する.

$$C_{\epsilon'_1} = g_1^{\epsilon'_1} h_1^{r_{\epsilon'_1}}$$
$$C_{\epsilon'_{K-1}} = g_1^{\epsilon'_{K-1}} h_1^{r_{\epsilon'_{K-1}}}$$

γ'_k, γ_k を以下のように計算する.

$$\gamma'_1 = r_{\epsilon'_1} - r_{e_{i_2}} e_{i_1}$$
$$\gamma'_{K-1} = r_{\epsilon'_{K-1}} - r_{e_{i_{K-1}}} e_{i_K}$$
$$\gamma_1 = r_{\epsilon_1} - r_{\epsilon'_1} e_l$$
$$\gamma_{K-1} = r_{\epsilon_{K-1}} - r_{\epsilon'_{K-1}} e_l$$

上記を用いると, 以下の検証式が得られる.

$$C_{\epsilon'_1} = C_{e_{i_2}}^{e_{i_1}} h_1^{\gamma'_1}, \ C_{\epsilon'_{K-1}} = C_{e_{i_{K-1}}}^{e_{i_K}} h_1^{\gamma'_{K-1}} \tag{9}$$

$$C_{\epsilon_1} = C_{\epsilon'_1}^{e_l} h_1^{\gamma_1}, \ C_{\epsilon_{K-1}} = C_{\epsilon'_{K-1}}^{e_l} h_1^{\gamma_{K-1}} \tag{10}$$

これらの検証式に対して知識の証明を行う.

5.4 安全性

健全性: i, j 間の接続性の証明において, 監査人の AHO 署名の検証により各アキュームレータは正しいことが保証される. また, アキュームレータの検証により, i, j は V に存在し, i, j 間の点 i_k, 辺 ϵ_k, およびラベル l はそれぞれ V, ϵ, Λ(または Λ') に含まれていることが保証される. さらに, $\epsilon_k = e_{i_k}e_{i_{k+1}}e_l$ を示しているため, 各辺がその端点 i_k, i_{k+1} と接続していること, そのラベルが l であることが保証されている. これにより, i, j 間にパスが存在することが証明され, そのラベルが m 番目であることもしくは Λ' に入っていることも証明される.

ゼロ知識性: コミットメントおよび知識の証明からは、証明していること以外の情報は洩れないため, ゼロ知識性は満たされる.

6. 実装・実験結果

本研究では, 表 1 に示す実装環境で提案方式の実装, および処理時間の測定を行った.

6.1 実装結果

表 1 実装環境

OS	WSL2(Ubuntu 20.04.4 LTS)
CPU	Intel(R)Core(TM)i7-11700(2.50GHz)
メモリ	16.0GB
プログラミング言語	C 言語
多倍長ライブラリ	GMP
ペアリングライブラリ	ELiPS [5]

証明時間, 検証時間, アキュームレータの補助情報の計算時間, 全体の処理時間 4 項目について計測を行った. 測定では, 辺数が点数より 1 つ少ない 3 分木状のグラフを用いており, 特に断りがない限り点数は 100 個, ラベル数は各辺に 1 つずつ含まれている. また, ラベルの値は重複を許しており, グラフ全体のラベル数 $|\Lambda|$ は $\frac{|\varepsilon|}{3}$ としている.

6.1.1 従来方式との比較

本研究で提案する方式は, ラベルを含まない方式である従来方式 [2] に比べ, 証明時間, 検証時間ともに増加している. しかし, 今回追加したラベルのアキュームレータに関してラベルの総数は点や辺に比べて少ないため, 増加量はわずかである.

6.1.2 点数の変化による処理時間の変化

グラフ全体の点数を 20 個から 200 個まで 20 個ずつ増加させたときの処理時間の変化を検証した (図 2).

点数の増加に伴い, 証明時間が増加していることがわかる. これは, 点情報, 辺情報, ラベル情報の補助情報 $W_{i_k}, W_{\epsilon_k}, W_{e_l}$ の計算時間がグラフ全体の点数, 辺数, ラベル数に依存しているためである. しかし, 検証時間は点数に関わらず一定である. これは, 従来方式 [2] と同様, 検証式にはグラフの点集合, 辺集合, ラベル集合をそれぞれアキュームレータとして 1 つの値にまとめたものを用いており, 検証式の数が点数, 辺数, ラベル数に依存しないためである.

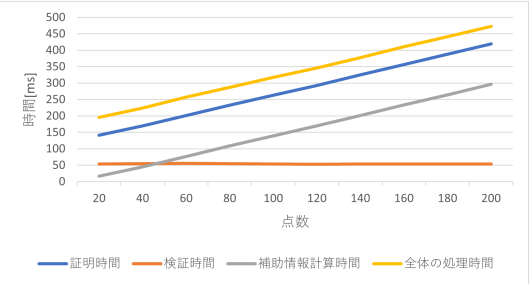

図 2 点数の変化による処理時間の変化

6.1.3 ラベル数の変化による処理時間の変化

各辺のラベル数を 1 個から 5 個まで, すなわちグラフ全体のラベル数を 33 個から 165 個まで増加させたときの処理時間の変化を検証した (図 3).

ラベル数の増加に伴い, 証明時間が増加していることがわかる. これは, ラベル情報の補助情報 W_{e_l} の計算時間がグラフ全体のラベル数に依存しているためである. しかし, 検証時間はラベル数に関わらず一定である. これは, 検証式にはグラフの全てのラベルをアキュームレータとして 1 つの値にまとめたものを用いているため, 検証式の数が各辺毎のラベル数に依存しないためである.

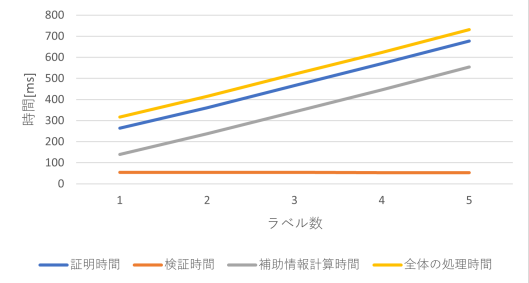

図 3 ラベル数の変化による処理時間の変化

6.1.4 制約リストのサイズの変化による処理時間の変化

制約を含む接続性の証明について, 制約リスト内のラベル数を 10 個から 50 個まで変化させたときの処理時間の変化を検証した (図 4).

制約リスト内のラベル数による処理時間の変化はほとんどないことがわかる. これは点数や辺数が約 100 個あるのに対し, 制約ラベル数が少ないため, 他の処理時間に対してラベルの処理時間の割合が小さいためである.

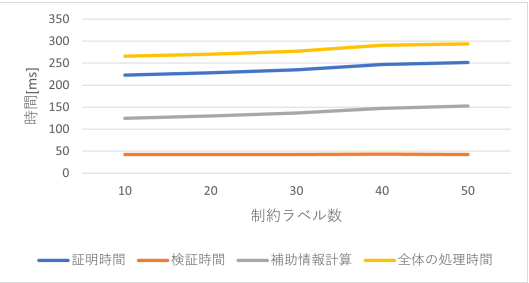

図 4 制約ラベル数の変化による処理時間の変化

6.1.5 証明する 2 点間の距離の変化による処理時間の変化

証明する 2 点間の点数を 2 個から 8 個まで増加させたときの処理時間の変化を検証した (図 5).

いずれの項目においても距離の増加に伴い処理時間が増加している. これは, **ProofGen** の内, 接続性の知識証明の 1 から 4 までの処理を, 証明する 2 点間の全ての辺に対して行う必要があるためである.

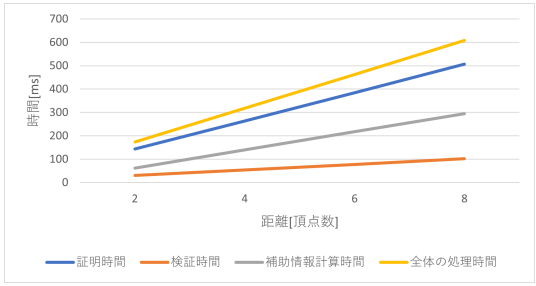

図 5 証明する 2 点間の距離の変化による処理時間の変化

7. まとめ

本研究では, 辺がラベルを持つグラフにおいて, 従来方式 [2] と同様にペアリング方式を用いた接続性のゼロ知識証明, および制約を持つ接続性のゼロ知識証明を提案した. いずれも従来方式 [2] 同様, テナント側で行われる検証にかかる処理時間と証明データサイズが, 点数, 辺数, ラベル数に依存していない. しかし, プロバイダ側で行われる証明にかかる処理時間の内, 補助情報の計算時間が点数, 辺数, ラベル数に依存しているため, この部分の改善は今後の課題の 1 つである. また, 全ての処理時間において, 証明する 2 点間の距離への依存の改善も今後の課題の 1 つである.

参考文献

[1] Gross, “Efficient Certification and Zero-knowledge proofs of knowledge on infrastructure topology graphs,” Proc. 6th ACM Workshop on Cloud Computing Security (CCSW’14), pp69-80, 2014.

[2] T. Nakanishi, H. Yoshino, T. Murakami, and G. Policharla, “Efficient zero-knowledge proofs of graph signature for connectivity and isolation using bilinear-map accumulator,” Proc. 7th ASIA Public-Key Cryptography Workshop(APKC@AsiaCCS 2020), pp.9–18, 2020.

[3] M. Abe, K. Haralambiev, and M. Ohkubo, “Signing on elements in bilinear groups for modular protocol design,” Cryptology ePrint Archive, Report 2010/133, 2010.

[4] C. Papamanthou, R. Tamassia, and N. Triandopoulos, “Optimal verification of operations on dynamic sets,” Advances in Cryptology — CRYPTO 2011, LNCS 6841, pp.91–110, Springer–Verlag, 2011.

[5] Y. Takahashi, Y. Nanjo, T. Kusaka, Y. Nogami, T. Kanenari, T. Tatara, “An Implementation and Evaluation of Pairing Library ELiPS for BLS Curve with Several Techniques,” 34th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC), 2019.