令和5年度

修　士　論　文

信頼できる鍵生成を必要としない
非中央集権型匿名評価システム

情報科学プログラム
計算機基礎学研究室
M222801　吾郷佳昭

指導教員

教授　中西　透
教授　岡村　寛之
准教授　北須賀　輝明
准教授　今井　克暢

2024年2月6日

広島大学大学院先進理工系科学研究科

概要

従来匿名でユーザの振る舞いの評価を行う匿名評価システムが提案されており, ID 管理者の信頼性のもとその証明書に基づいて安全性が保証されている. 一方, ブロックチェーンでの利用を想定して信頼できるサーバが不要な非中央集権型の匿名評価システムが提案されている. しかしその従来方式では, 匿名のままユーザ自身の評価値を証明する際に, ZK-SNARK と呼ばれる零知識証明が用いられており, 証明したい関係を示す回路に対する公開鍵を事前に生成するために, 信頼できるセットアップが必要となる.

本研究では, 信頼できる鍵生成を必要としない非中央集権型の匿名評価システムを提案する. 提案システムでは, 信頼できる鍵生成なしに群の生成元のみのパラメータで利用できる一対多の知識の証明を利用する. これにより信頼できるサーバなしで匿名のままユーザの評価値が分散台帳に存在することを証明できる. しかし, このシステムでは分散台帳中の N 個のデータに対して $O(N \log N)$ の証明生成時間が必要となる. そこで DualRing と呼ばれるリング署名を利用することにより $O(N)$ 時間となるシステムへの拡張を行う. さらに本研究では, 提案方式を実装し, 証明生成と検証の処理時間の評価を行う.

目次

第 1 章　はじめに

ID ベース認証とは, ID(ユーザ識別情報) を用いた認証方式であり, シンプルな仕組みであるため広く使われている. しかし, ID 認証ではユーザの全てのアクセスが ID に紐付いており, サービス提供者は, 各ユーザの利用履歴を取得できるため, ユーザのプライバシーが守られない.

そこで, ID 管理者による証明書を利用することで正当なユーザかどうかの検証はするが, サーバが正当なユーザのうちの誰であるかまでは特定できない匿名認証 [1] が提案されている. これにより, ユーザのプライバシーを保護した認証を実現できる. しかし, 匿名認証ではユーザを特定することができないため, あるユーザの振る舞いに対してそのユーザの信頼度を評価することができない. そこで, 匿名性を維持しつつ各ユーザーに対して評価を行うことができる匿名評価システム [2] が提案されている.

従来提案されている匿名評価システムでは, ID 管理者の信頼性のもとその証明書に基づいて安全性が保証されている. 一方, 信頼できるサーバが不要な分散型の匿名評価システム [3] が提案されている. これはブロックチェーンのような分散台帳の利用が想定されており, 登録時のみ身元を保証すればよく, それ以降は信頼できるサーバを利用することなくやり取りできる. この従来方式 [3] では, 匿名のままユーザ自身の評価値を証明する際に, ZK-SNARK と呼ばれる零知識証明が用いられている. ZK-SNARK では, 任意の回路で表現できる関係式を証明でき, 検証コストが一定という利点があるが, 証明したい関係を示す回路に対する公開鍵を事前に生成するために, 信頼できるセットアップが必要となる. しかし, これは想定される利用先であるブロックチェーンの目的の一つである信頼できる中央集権的な管理者を排除することに反する.

そこで本研究では, 信頼できる鍵生成を必要としない分散型の匿名評価システムを提案する. 提案システムでは, 信頼できる鍵生成なしに群の生成元のみのパラメータで利用できる一対多の知識の証明 [4] を利用する. これにより, 管理者なしで自身の評価値が分散台帳上に存在することを証明できる. この一対多の証明では, N 個のコミットメント中に自身

の評価値のコミットメントが存在することを, $O(\log N)$ のデータサイズで証明でき, これを提案システムで用いることにより, 証明サイズを小さくしつつ管理者なしで証明している. 一方証明生成時間が $O(N \log N)$ と比較的大きいという問題がある. そこで本研究では, DualRing[5] と呼ばれるリング署名を応用する. この方式では $O(N)$ 時間で公開鍵が N 個の集合に存在することを証明でき, これを提案システムに適用することにより $O(N)$ 時間の証明処理を実現する. さらに本研究では, 提案方式を PC 上で実装し, 証明と検証の処理時間の評価を行う.

第 2 章　数学的準備

本研究では, 素数位数 q の群 G を用いる. このような群は楕円曲線を用いて構築できる. また $g \in G$ を G の生成元とする.

2.1　DL 仮定

本研究で利用するコミットメントの安全性は DL 仮定に基づく. DL 仮定とは, g と $y = g^x$ が与えられた時, x を無視できない確率で求める確率的多項式時間アルゴリズムは存在しないという仮定である.

2.2　Pedersen コミットメント

本研究では, コミットメントとして Pedersen コミットメントを利用する. コミットメントでは送信者と受信者が参加し, まず送信者は コミットする入力値と乱数からコミットメントを計算して受信者に送る. また, 送信者は入力値と乱数を送信することにより, コミットした入力値を開示することができる.

コミットメントの安全性は, 秘匿性 (Hiding property) と拘束性 (Binding property) からなる. 秘匿性とは, コミットメントの受信者がコミットされた入力値の情報を知り得ないことを意味する. 拘束性とは, 送信者が開示時にコミットした入力値と違う値で開示できないことを意味する. Pedersen コミットメントは, 情報理論的秘匿性を持ち, DL 仮定の元で計算量的拘束性を持つ. Pedersen コミットメントは, 入力値 $m \in Z_q$ と乱数 $r \in Z_q$ を入力として, $C = g^m h^r$ と計算される. さらに, ベクトルを入力として取ることもでき, 生成元 $g_1, g_2, \cdots g_k$ を用いて, 入力 $m_1, \cdots, m_k \in Z_q$ に対して $C = g_1^{m_1} \cdots g_k^{m_k} h^r$ と計算される. このコミットメントの関数を $C = Com(m_1, ..., m_k) = Com(m_1, ..., m_k; r)$ と表記する. このコミットメントでは以下の準同型性が成り立つ.

準同型性: 入力 $\overrightarrow{m_1}, \overrightarrow{m_2}$ のコミットメント $Com(\overrightarrow{m_1})$, $Com(\overrightarrow{m_2})$ に対して,

$Com(\overrightarrow{m_1})\cdot Com(\overrightarrow{m_2})$ がもとの入力を加算したコミットメント $Com(\overrightarrow{m_1}+\overrightarrow{m_2})$ と一致する．提案方式ではこの性質を利用する．

2.3　知識の証明

知識の零知識証明とは証明者 P と検証者 V との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ことを，秘密情報を漏らすことなく証明する．本研究では, Schnorr 認証を拡張した離散対数の秘密情報を証明する知識の証明を用いる．この方式では, $y, g_1, g_2, \cdots \in G$, $x_1, x_2, \cdots \in Z_q$ に対して $y = g_1^{x_1} g_2^{x_2}, \cdots$ を満たすリプリゼンテーション $x_1, x_2, \cdots$ の知識を証明できる．さらに, 文献 [4] で提案されている一対多 (one-out-of-many) の証明を用いる．この知識の証明では, N 個のコミットメントに対してその 1 つが 0 ベクトルのコミットメントであり その乱数を知っていること, すなわち, $C_l = Com(\vec{0}; r)$ なる r の知識を l, r を明かすことなく証明できる．

2.4　リング署名

提案方式の一つで利用する DualRing はリング署名の一種である．DualRing を説明するために, 従来の AOS リング署名と DualRing の 2 つを説明する．リング署名は, 署名者が $n-1$ 個の公開鍵の集合を選び, 自身の公開鍵 1 つを加えた n 個の公開鍵を用いて署名を行い, n 個の公開鍵のうちどれを用いて署名をしたか分からないようにすることで署名者の匿名性を保証する署名方法である．

2.5　AOS リング署名

従来から存在する AOS リング署名 [6] は Schnorr 署名のような Three-move タイプの署名を元にした Type-T 署名と, RSA 署名のように Hash-and-one-way タイプの Type-H 署名の 2 つのタイプが存在する．本節は Type-T 署名を元にした AOS リング署名について述べる．AOS リング署名は, R, Z, V の 3 つの関数とハッシュ関数 H から構成される．Schnorr 署名を例に取ると，署名者が選択した公開鍵集合を $pk_1, ..., pk_n$ とし, pk_j を署名者の公開鍵とする．署名者はランダムに r_j を選択し, コミットメント $R_j = A(sk_j, r_j) = g^{r_j}$ を計算してハッシュに通し, チャレンジ $c_{j+1} = H(M, R_j)$ とレスポンス $z_{j+1} = r_j - c \cdot sk_j$ を計算して, $R_{j+1} = V(\cdot) = g^{z_{j+1}} pk_{j+1}^{c_{j+1}}$ を計算し, ハッシュに通して $\cdots$ と処理を繰り返す．最終的に $c_j = H(M, R_{j-1})$ が出力されて, $z_j = r_j$

– $c_j sk_j$ を計算して署名 $(c_1, z_1, \ldots, z_n)$ を出力する．検証者は入力としてチャレンジ c_1 とすべてのレスポンス z_i, 公開鍵 pk_i を受け取るためリングがどこから開始したか分からず, 結果的に署名に使われた公開鍵がどれであるか判別できない．このことにより実際の署名者を秘匿することができる．一方, 署名長が $O(n)$ と n に比例するため, n が大きいときにデータサイズが問題となる．

2.6 DualRing

DualRing[5] は, AOS リング署名におけるコミットメント R とチャレンジ c を別々のリングで計算することで署名サイズの軽量化を行ったリング署名である．AOS リング署名ではコミットメント R とチャレンジ c を交互に計算しているところを, DualRing では最初に R-ring 中で R を生成し, その後 C-ring で c を生成し, 署名 $(z, c_1, \ldots, c_n)$ を出力する．AOS リング署名とは違い, DualRing では R-ring と C-ring の 2 つのリングに分割したため, リングサイズ n に対し, AOS リング署名ではデータサイズが $O(n)$ であるところを, NISA を適用した DualRing ではデータサイズが $O(\log n)$ で済む利点がある．さらに, 署名生成および検証時間が $O(n)$ であり, 同様の証明を行う手法の一対多証明において証明生成時間が $O(n \log n)$ であること (検証時間は $O(n)$) と比較して高速に証明可能である．

2.7 Sum Arguments of Knowledge

まず, $\boldsymbol{g} = (g_1, ..., g_n) \in \mathbb{G}^n, \boldsymbol{a} = (a_1, ..., a_n) \in \mathbb{Z}_p^n$ に対して $\boldsymbol{g}^{\boldsymbol{a}}$ を，$\boldsymbol{g}^{\boldsymbol{a}} = \prod_{i=1}^{n} g_i^{a_i}$ と定義する．DualRing で用いられている NISA アルゴリズムで用いられる Sum Arguments of Knowledge について説明する．Sum Arguments of Knowledge を用いることで，DualRing における証明のデータサイズを $O(\log n)$ にできる．Sum Arguments of Knowledge は以下の関係を示す NIZK arguments である．

$$(\boldsymbol{g}, \boldsymbol{h} \in \mathbb{G}^n, P \in \mathbb{G}, c \in \mathbb{Z}_p; \boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}_p^n) : P = \boldsymbol{g}^{\boldsymbol{a}} \boldsymbol{h}^{\boldsymbol{b}} \wedge c =< \boldsymbol{a}, \boldsymbol{b} >$$

証明者（ユーザ）は検証者（サービス提供者）に対し，$P = \boldsymbol{g}^{\boldsymbol{a}} \boldsymbol{h}^{\boldsymbol{b}}$ かつ $c =< \boldsymbol{a}, \boldsymbol{b} >$ を満たす秘密情報 $\boldsymbol{a}, \boldsymbol{b}$ を知っていることを示す．NISA では, Sum Arguments of Knowledge の以下の特別な場合が用いられる．

$$(\boldsymbol{g} \in \mathbb{G}^n, P \in \mathbb{G}, c \in \mathbb{Z}_p; \boldsymbol{a} \in \mathbb{Z}_p^n) : P = \boldsymbol{g}^{\boldsymbol{a}} \wedge c = \sum \boldsymbol{a}$$

この場合，証明者（ユーザ）は検証者（サービス提供者）に対し，$P = \boldsymbol{g}^{\boldsymbol{a}}$ かつ $c = \sum \boldsymbol{a}$ なる秘密情報 $\boldsymbol{a}$ を知っていることを示す．これはもともとの場合で $b = 1$ であった場合に相当する．これを DualRing での $(c_1, ..., c_n)$ に対して $c = \sum c_i$ を証明する際に用いることにより, $O(\log n)$ のデータサイズで署名を生成できる.

第 3 章　先行研究とその問題点

3.1　先行研究の概要

文献 [3] では, ブロックチェーン (分散型台帳) を利用した分散型の匿名評価システムが提案されている. この方式では, 発行者を利用する代わりに, 各ユーザは秘密鍵と評価値に対するコミットメントを評価トークンとしてブロックチェーン上に提出する.
ユーザがあるトランザクションにおいて評価されると, 評価値をコミットメントのまま秘匿して更新し, ブロックチェーン上に提出する. ユーザが自身の評価値を示すために, ユーザーの評価値のコミットメントを集約しているブロックチェーン上の木構造に自身のコミットメントが確かに存在していることを知識の証明で示している. これにより, 匿名性を満たしたまま, 評価値の正当性が保証される. また, 具体的な評価値を示すと匿名性が弱くなるため, その代わりに特定の範囲に評価値が含まれていることを証明している. これらの証明において ZK-SNARK と呼ばれる任意の回路で表現された関係を証明できる方式を利用している.

3.2　先行研究のアルゴリズム

3.2.1　Setup(1^λ)

セキュリティパラメータ λ に対して, $q > 2^\lambda$ となる初期パラメータ $params = \{G, q, g, h, h_1, h_2\}$ を生成する. g, h, h_1, h_2 は q を素数位数とする巡回群 G の生成元である.

3.2.2　Register($params, U, R$)

ユーザー U とレジストラ R の間で実行される.
ユーザは長期的な秘密鍵 sk_U を生成し, レジストラはそれを保証する署名 $Sig_R(I(=$

$g^{sk}))$) を発行する．I はユーザの秘密鍵から一意に求まり, ユーザの公開識別子として利用される．

3.2.3　NymGen$(params, U, sk_U)$

ユーザ U が他のユーザ V とやりとりするためにその場限りの仮名を生成する．
乱数 $r \in Z_q$ と秘密鍵 $sk_U \in Z_q$ に対して, 仮名は $nym_U = g^r h^{sk_U}$ と計算される．sk_U から生成するため, U にしか生成できないが毎回異なる値となり, 仮名同士をリンクすることはできない．

3.2.4　MintRep$(params, sk_U, nym_U, Sig_R(I))$

ユーザは評価値の初期値 $v_{init} \in Z_q$, シリアル番号 $s \in Z_q$, 秘密鍵 $sk = sk_U$ と乱数 $r' \in Z_q$ を用いて, 台帳に記帳されるユーザ U の評価トークン $rep_U = g^{r'} h^{sk} h_1^s h_2^{v_{init}}$ を生成する．rep_U は (sk, s, v_{init}) に対するコミットメントとなっている．$Sig_R(I)$ および Rep_U の正しさを示す知識の証明も台帳に記帳される．

3.2.5　ShowRep$(params, nym_U^V, rep_U)$

ShowRep によりユーザ U はユーザ V に自身の評価値を示す．U は, 新しい仮名 nym_U^V を生成して, rep_U に対して以下を行う．まず $r_{new}, s_{new} \in Z_q$ と新しい評価値 v_{new} に対して, 新しい評価トークン $rep_U^{new} = g^{r_{new}} h^{sk} h_1^{s_{new}} h_2^{v_{new}}$ を生成する. 次に, 今回のトランザクションの仮名 nym_U^V, 元の評価トークン rep_U に対して, 以下の 4 つの関係を ZK-SNARK で証明する．
1. rep_U が台帳中の評価トークン集合のハッシュ木 $RepTree$ に属している．
2. rep_U, rep_U^{new} と nym_U^V が同じ秘密鍵 sk を共有している．つまり同じユーザに紐付いている．
3. 評価値 v がある指定された評価値の範囲 L_i に属している．
4. $v_{new} = v$

これにより, 評価値 v が L_i に含まれており, その評価トークンが以前に台帳へ提出されていることが保証される．この証明において, 元の評価トークンのシリアル番号 s は公開される．これにより同じ評価トークンが使用された場合は不正として検出される．

3.2.6　UpdateRep$(params, nym_U^V, rep_U, val)$

ユーザ U, V 間のインタラクションの後で, V が U を評価する. そして, **UpdateRep** により, 今回の評価値 val を反映した新しい U の評価トークンに更新する. U と V でそれぞれ **ShowRep** と同様の処理を行う. V も行う理由は, 同一人物がユーザ U とユーザ V 両方の処理を行うことで評価を操作する, self-promotion attack を防ぐためである. このために self-redeeming tag τ を利用する. τ は両者の仮名 nym_U^V, nym_V^U に加えて U は sk_U, V は sk_V をハッシュした値となる. $U = V$ なら両者の τ が等しくなり不正が検出される. τ の正しさは ZK-SNARK により証明される.

3.3　先行研究の問題点

利用している ZK-SNARK では, ハッシュ関数も含めた任意の回路を証明でき, 検証コストが一定という利点があるが, 事前に回路に対する公開鍵を生成するために信頼できるセットアップが必要となる. しかし, これは想定される利用先であるブロックチェーンの目的の一つである信頼できる中央集権的な管理者を排除するというものに反する.

第 4 章　提案方式

4.1　提案方式の概要

本章では, 従来の分散型匿名評価システム [3] に対して, ZK-SNARK の代わりに [4] の一対多の知識の証明を利用した方式を提案する. この証明方式では, N 要素のコミットメント集合中にゼロベクトルのコミットメントが存在することを効率的に $\mathrm{O}(\log N)$ の証明サイズで示すことができる. このとき, ユーザのコミットメント $C = Com(m_1, ..., m_k; r)$ に対して, コミットメント集合の各要素 C_i を $C'_i = C_i \cdot Com(m_1, ..., m_k; 0)^{-1}$ とすることにより, C がコミットメント集合に含まれている場合, ある l に対して C'_l がゼロベクトルのコミットメントとなる. こうして, C'_i の集合をコミットメント集合として一対多の証明をすることにより, C が元のコミットメント集合に存在することを証明できる. この方式では, 信頼できる鍵生成なしに生成元のみのパラメータで利用できる. これを用いて, **ShowRep** および **UpdateRep** での知識の証明において, 自身の評価点のコミットメントがブロックチェーン上のコミットメント集合に属していることを証明する.

4.1.1　提案方式のアルゴリズム

先行研究の方式 [3] に対して, **ShowRep** と **UpdateRep** における ZK-SNARK によるゼロ知識証明を一対多の知識の証明を用いるように変更する. 他のアルゴリズムは [3] と同様である.

4.1.2　Setup(1^λ)

セキュリティパラメータ λ に対して初期パラメータ $params = \{G, q, g, h, h_1, h_2\}$ を生成する. g, h, h_1, h_2 は $q > 2^\lambda$ を素数位数とする巡回群 G の生成元である.

4.1.3 Register($params, U, R$)

ユーザー U とレジストラ R の間で実行される.
ユーザは長期的な秘密鍵 sk_U を生成し, レジストラはそれを保証する署名 $Sig_R(I(=g^{sk}))$ を発行する. I はユーザの秘密鍵から一意に求まり, ユーザの公開識別子として利用される.

4.1.4 NymGen($params, U, sk_U$)

ユーザ U が他のユーザ V とやりとりするためにその場限りの仮名を生成する.
乱数 $r \in Z_q$ と秘密鍵 $sk_U \in Z_q$ に対して, 仮名は $nym_U = g^r h^{sk_U}$ と計算される. sk_U から生成するため, U にしか生成できないが毎回異なる値となり, 仮名同士をリンクすることはできない.

4.1.5 MintRep($params, sk_U, nym_U, Sig_R(I)$)

ユーザは評価値の初期値 $v_{init} \in Z_q$, シリアル番号 $s \in Z_q$, 秘密鍵 $sk = sk_U$ と乱数 $r\prime \in Z_q$ を用いて, 台帳に記帳されるユーザ U の評価トークン $rep_U = g^{r'} h^{sk} h_1^s h_2^{v_{init}}$ を生成する. rep_U は (sk, s, v_{init}) に対するコミットメントとなっている. $Sig_R(I)$ および Rep_U の知識の証明とともに台帳に記帳される.

4.1.6 ShowRep($params, nym_U^V, rep_U$)

自身の評価値を示す rep_U に以下を行う. まず $r_{new}, s_{new} \in Z_q$ と現在の評価値 v に対して, 新しい評価トークン $rep_U^{new} = g^{r_{new}} h^{sk} h_1^{s_{new}} h_2^v$ を生成する.

ユーザ U の評価トークン rep_U を含む台帳上の評価トークンの集合 rt を用意する. $rt = (rep_0, \cdots, rep_{N-1})$ とする.

次に rt を $RepTree$ の代わりとして, 3.2.5 で証明している関係 1〜4 を以下のようにして知識の証明を行う. 関係 4 については, $v_{new} = v$ として知識の証明がされる. ここで rt 中の rep_l がユーザ U の rep_U とする. すなわち, $rep_l = rep_U = g^{r'} h^{sk} h_1^s h_2^v$ である.

[4] の一対多証明では, l 番目に 0 ベクトルのコミットメントがあることを証明できるため, $rep'_l = g^{r\prime} h^0 h_1^0 h_2^0$ となるようにすべての rep_i に以下を行う.
$C = g^0 h^{sk} h_1^s h_2^v$ とおき, $rep'_i \leftarrow rep_i / C$ とする.
これにより, $rep'_l = g^{r\prime} h^0 h_1^0 h_2^0$ となり, 他の $rep'_i (i \neq l)$ は通常のコミットメントとなる.

また, C の正しさは関係 2 の証明とともに行う. このとき, シリアル番号 s は公開され, トークンの二重使用がチェックされる. そして, $(rep'_0, \cdots, rep'_{N-1})$ に対して一対多の証明プロトコルを実行する.

以下に, **ShowRep** における知識の証明の詳細を示す.

関係 1 の証明については以下を行う.

(1) 証明者は $j = 1, \cdots, N$ についてそれぞれ乱数 $r_j, a_j, s_j, t_j, \rho_k \in Z_q$ を生成し,

$c_{l_j} = Com(l_j; r_j)$

$c_{a_j} = Com(a_j; s_j)$

$c_{b_j} = Com(l_j a_j; t_j)$

$c_{d_k} = \Pi_i c_i^{p_{i,k}} Com(\vec{0}; \rho_k)$

を計算し検証者に送信する.

ここで $k = j - 1$ かつ $p_{i,k}$ は [4] における多項式 $P_i(\lambda)$ の $n-1$ 次以下の項の係数である.

(2) 検証者は乱数 $x \in Z_q$ を証明者に送り, 証明者は $j = 1, \cdots, N$ について

$f_j = l_j x + a_j$

$z_{a_j} = r_j x + s_j$

$z_{b_j} = r_j (x - f_j) + t_j$

$z_d = r' x^n - \sum_{k=0}^{n-1} \rho_k x^k$

を検証者に送信する.

(3) 検証者は以下の等式が成り立つか検証する.

$c_{l_1}, ..., c_{d_{n-1}} \in rt$

$f_1, ..., z_d \in Z_q$ であり

すべての $j = 1, \cdots, N$ に対して

$c_{l_j}^{x} c_{a_j} = Com(f_j; z_{a_j})$

$c_{l_j}^{x - f_j} c_{b_j} = Com(0; z_{b_j})$

$\Pi_i c_i^{\Pi_{j=1}^{n} f_{j,i_j}} \cdot \Pi_{k=0}^{n-1} c_{d_k}^{-x^k}$

ここで $f_{j,1} = f_j$, $f_{j.0} = x - f_j$　である.

関係 2 の証明は以下のように行う.

(1) 証明者はまず, 乱数 $\rho_r, \rho_{sk}, \rho_{\tilde{r}}, \rho_{\tilde{s}}, \rho_v \in_R Z_q^*$ を生成する.

次に,

$t_{nym} = g^{\rho_r} h^{\rho_{sk}}$
$t_{rep}^{new} = g^{\rho_{\tilde{r}}} h^{\rho_{sk}} h_1^{\rho_{\tilde{s}}} h_2^{\rho_v}$
$t_C = h^{\rho_{sk}} h_2^{\rho_v}$
$S_r = \rho_r + xr \bmod q$
$S_{sk} = \rho_{sk} + x \cdot sk \bmod q$
$S_{\tilde{r}} = \rho_{r'} + xr_{new} \bmod q$
$S_{\tilde{s}} = \rho_s + xs_{new} \bmod q$
$S_v = \rho_v + xv \bmod q$
を計算する.

(2) 検証者は乱数 x を返す. この x は関係 1 の (2) での x を再利用する.

(3) 検証者は以下の等式が成り立つか検証する.
$t_{nym} = g^{S_r} h^{S_{sk}} nym^{-x}$
$t_{rep}^{new} = g^{S_{\tilde{r}}} h^{S_{sk}} h_1^{S_{\tilde{s}}} h_2^{S_v} (rep_U^{new})^{-x}$
$t_C = h^{S_{sk}} h_2^{S_v} (C \cdot h_1^{-s})^{-x}$

関係 3 の証明は以下のように行う.
全ての評価値 V_i に対して, 属する範囲 L_i を示す $C_{L_i} = \{Com(v_j; 0) | v_j \in L_i\}$ を事前に決めておく.
証明したい rep_U の評価値 v' が範囲 L_i にあることを示す場合, $C'_{L_i} = \{Com(v_j - v'; 0 - r) = Com(v_j; 0)/Com(v'; r) | v_j \in L_i\}$ を新たに構成し, C'_{L_i} に対して関係 1 の証明と同様の処理を行う. 最後に, U の新しい評価トークンとして rep_U^{new} が台帳に記載される.

4.1.7　UpdateRep$(params, nym_U^V, rep_U, val)$

ユーザ U,V の秘密鍵をそれぞれ sk_U, sk_V とし, ユーザ V がユーザ U を評価する場合を考える.

U の行う処理:

まず, 新しいトークン $rep_U^{new} = g^{r_{new}} h^{sk_U} h_1^{s_{new}} h_2^{v_{new}}$ を作る. また $\tau_U = H(nym_U^V, nym_V^U)^{sk_U}$ を計算する. ここで H は任意の文字列から G の元へのハッシュ関数である. 次に, rep_U^{new} および τ_U の正当性を示す. rep_U^{new} については, **ShowRep** と同じ手順を用いる. τ_U の正当性は以下のように示す.

(1) 乱数 $\rho_{sk_U} \in Z_q$ を生成し,
$t_{\tau_U} = H(nym_U^V, nym_V^U)^{\rho_{sk_U}}$
を計算する.
(2) 検証者から乱数 x を受け取り,
$S_{sk_U} = \rho_{sk_u} + x \cdot sk_U \bmod q$
を計算する.
(3) 検証者は以下の等式が成り立つか検証する.
$t_{\tau_U} = H(nym_U^V, nym_V^U)^{S_{sk_U}} \cdot \tau_U^{-x}$

V の行う処理:

ここで val はユーザ V のユーザ U への評価値である.
まず, $rep_U^{new'} = rep_U^{new} \cdot h_2^{val}$ を計算する.
このとき, $rep_U^{new'} = g^{new'} h^{sk_U} h_1^{s_{new}} h_2^{v+val}$ となる.
次に $\tau_V = H(nym_U^V, nym_V^U)^{sk_V}$ を生成する.
U と同じように V のトークン rep_V^{new} が台帳にあることを証明して, 正当性を示す. さらに τ_V の正当性も示す. 検証者は $\tau_U \neq \tau_V$ をチェックすることにより, $U \neq V$ を確認する.
取引内容として nym_U^V, nym_V^U と各知識の証明および $rep_U^{new'}$ を台帳に提出する. $rep_U^{new'}$ が U の新しい評価トークンとなる.

4.2　安全性

安全性として, 不正な評価値の操作への耐性と匿名性を示す.

不正な評価値の操作への耐性: 各ユーザはインタラクションのために好きなだけ仮名を生成することができるが, それらは全て評価トークンと同じ秘密鍵を用いる必要があり, その正しさは知識の証明により保証されている. したがって, トークンはその正当な所有者によってのみ作成および使用される. またシリアル番号 s が公開されるため, 過去のトークンを使用することもできない. 各トークンは, **ShowRep**, **UpdateRep** において, 台帳に記載されたトークンであることが証明されており, コミットメントの準同型性から $v_{new} = v + val$ も保証されている. こうして, トークンでコミットされている評価値を不正に改ざんできない. そして **ShowRep** では, 評価値の範囲証明をしているため, U はトークンでコミットされたものより高い評価値の範囲を証明できない.

匿名性: 評価値は準同型性を用いることで匿名性を損なわない方法で更新される. またコミットメントと知識の証明を用いることで, 評価値を証明する際に 各ユーザと仮名を結び

つけることができないようにしている．これは同じユーザの仮名同士についても同様に結びつけることができない．

第 5 章　DualRing を用いた改善

5.1　方針

従来の分散型匿名評価システム [3] に対して, ZK-SNARK の代わりに 4 章では [4] の一対多の知識の証明を用いた．この証明方式では, N 要素のコミットメント集合中に自身のコミットメントが存在することを効率的に $\mathrm{O}(\log N)$ の証明サイズで示すことができるが, 6.2 節の図 1 のグラフで示すように評価トークンの個数 N に対して証明時間が $\mathrm{O}(N \log N)$ であり, N が大きくなると処理時間が大きい．そこで, DualRing を用いたリング署名 [5] の利用を考える．この署名では $\mathrm{O}(\log N)$ の署名サイズかつ $\mathrm{O}(\log N)$ 時間で署名生成が可能である．この署名では, 離散対数型の公開鍵集合に対して, 自身の公開鍵が含まれていることをゼロ知識証明している．そこで, 提案方式ではトークン集合に対して [4] の一対多の知識の証明を行っていた部分を DualRing による証明に同様の手法により置き換えることが可能である．これにより, $\mathrm{O}(\log N)$ 証明サイズ, $\mathrm{O}(N)$ 証明生成時間の方式が構成でき, 提案方式の証明時間を改善できる．

5.2　アルゴリズム

5.2.1　Setup(1^λ)

セキュリティパラメータ λ に対して λ ビットセキュリティの楕円曲線 E と初期パラメータ $params = \{G, q, g, h, h_1, h_2\}$ を生成する．g, h, h_1, h_2 は q を素数位数とし, 楕円曲線 E 上の有理点が成す群 G の生成元である．

5.2.2　Register($params, U, R$)

ユーザー U とレジストラ R の間で実行される．ユーザは長期的な秘密鍵 sk_U を生成し, レジストラはそれを保証する署名 $Sig_R(I(= g^{sk}))$ を発行する．I はユーザの秘密鍵から

一意に求まり, ユーザの公開識別子として利用される.

5.2.3　NymGen$(params, U, sk_U)$

ユーザ U が他のユーザ V とやりとりするためにその場限りの仮名を生成する. 乱数 $r \in Z_q$ と秘密鍵 $sk_U \in Z_q$ に対して, 仮名は $nym_U = g^r h^{sk_U}$ と計算される. sk_U から生成するため, U にしか生成できないが毎回異なる値となり, 仮名同士をリンクすることはできない.

5.2.4　MintRep$(params, sk_U, nym_U, Sig_R(I))$

ユーザは評価値の初期値 $v_{init} \in Z_q$, シリアル番号 $s \in Z_q$, 秘密鍵 $sk = sk_U$ と乱数 $r\prime \in Z_q$ を用いて, 台帳に記帳されるユーザ U の評価トークン $rep_U = g^{r'} h^{sk} h_1^s h_2^{v_{init}}$ を生成する. rep_U は (sk, s, v_{init}) に対するコミットメントとなっている. $Sig_R(I)$ および Rep_U の知識の証明とともに台帳に記帳される.

5.2.5　ShowRep$(params, nym_U^V, rep_U)$

自身の評価値を示す rep_U に以下を行う. まず $r_{new}, s_{new} \in Z_q$ と現在の評価値 v に対して, 新しい評価トークン $rep_U^{new} = g^{r_{new}} h^{sk} h_1^{s_{new}} h_2^v$ を生成する. ユーザ U の評価トークン rep_U を含む台帳上の評価トークンの集合 rt を用意する. $rt = (rep_0, \cdots, rep_{N-1})$ とする. 次に rt を $RepTree$ の代わりとして, 3.2.5 で証明している関係 1〜4 を以下のようにして知識の証明を行う. 関係 4 については, $v_{new} = v$ として知識の証明がされる. ここで rt 中の rep_l がユーザ U の rep_U とする. すなわち, $rep_l = rep_U = g^{r'} h^{sk} h_1^s h_2^v$ である. DualRing による証明を適用するために $rep_l' = g^{r\prime} h^0 h_1^0 h_2^0$ となるようにすべての rep_i に以下を行う. $C = g^0 h^{sk} h_1^s h_2^v$ とおき, $rep_i' \leftarrow rep_i / C$ とする. これにより, $rep_l' = g^{r\prime} h^0 h_1^0 h_2^0$ となり, 他の $rep_i'(i \neq l)$ は通常のコミットメントとなる. また, C の正しさは関係 2 の証明とともに行う. このとき, シリアル番号 s は公開され, トークンの二重使用がチェックされる. そして, 公開鍵集合 $(rep_0', \cdots, rep_{N-1}')$ に対して DualRing を用いた署名生成により証明を生成する.

以下に, **ShowRep** における知識の証明の詳細を示す.

関係 1 の証明については以下を行う.

DualRing を用いて以下を証明する.

$DualRing - EC(pk = rt/C, sk_i = r\prime\prime\prime, m = Nonce) \wedge C = g^0 h^{sk} h_1^s h_2^v$

関係 2,3 の証明は 4 章と同様に行う.

UpdateRep($params, nym_U^V, nym_V^U, val$)

ユーザ U,V の秘密鍵をそれぞれ sk_U, sk_V とし, ユーザ V がユーザ U を評価する場合を考える.

U の行う処理:

まず, 新しいトークン $rep_U^{new} = g^{r_{new}} h^{sk_U} h_1^{s_{new}} h_2^{v_{new}}$ を作る. また $\tau_U = H(nym_U^V, nym_V^U)^{sk_U}$ を計算する. ここで H は任意の文字列から G の元へのハッシュ関数である. 次に, rep_U^{new} および τ_U の正当性を示す. rep_U^{new} については, **ShowRep** と同じ手順を用いる. τ_U の正当性は以下のように示す.

(1) 乱数 $\rho_{sk_U} \in Z_q$ を生成し,

$t_{\tau_U} = H(nym_U^V, nym_V^U)^{\rho_{sk_U}}$

を計算する.

(2) 検証者から乱数 x を受け取り,

$S_{sk_U} = \rho_{sk_u} + x \cdot sk_U \mod q$

を計算する.

(3) 検証者は以下の等式が成り立つか検証する.

$t_{\tau_U} = H(nym_U^V, nym_V^U)^{S_{sk_U}} \cdot \tau_U^{-x}$

V の行う処理:

ここで val はユーザ V のユーザ U への評価値である.

まず, $rep_U^{new'} = rep_U^{new} \cdot h_2^{val}$ を計算する.

このとき, $rep_U^{new'} = g^{new'} h^{sk_U} h_1^{s_{new}} h_2^{v+val}$ となる.

次に $\tau_V = H(nym_U^V, nym_V^U)^{sk_V}$ を生成する.

U と同じように V のトークン rep_V^{new} が台帳にあることを証明して, 正当性を示す. さらに τ_V の正当性も示す. 検証者は $\tau_U \neq \tau_V$ をチェックすることにより, $U \neq V$ を確認する.

取引内容として nym_U^V, nym_V^U と各知識の証明および $rep_U^{new'}$ を台帳に提出する. $rep_U^{new'}$ が U の新しい評価トークンとなる.

5.3　安全性

4 章で示したものと同様である.

第 6 章　実装結果

6.1　実装環境

表 6.1　実装環境

OS	Ubuntu 18.04 LTS
CPU	AMD Ryzen 7 3800X 8-Core @ 3.90GHz
メモリ	32GB
プログラミング言語	C
暗号ライブラリ	OpenSSL 3.0.7

提案方式の有効性を評価するために, 表 6.1 の環境で実装を行い, 評価トークンのリストの要素数が $512, 1024, 2048, 4096$ の 4 つの場合で, システム中で頻繁に実行される **ShowRep** の証明時間と検証時間を計測した. 処理がほぼ同一である **UpdateRep** の処理時間も同等となる.

6.2　計測結果と評価

図 6.1 に要素数を変化させた際の 4 章の方式での **ShowRep** の証明時間と検証時間の変動を示す. 評価トークンの要素数に対して各処理時間はほぼ線形に推移している. 各処理において, 一対多の証明が支配的な処理となっている. その中で, 証明時には $O(N \log N)$ 回の冪乗算, 検証時には $O(N)$ 回の冪乗算が発生している. このため, 各処理時間はほぼ線形となりつつ, 証明時間が検証時間よりも大きくなっている.

図 6.2 には 4 章の方式による **ShowRep** の証明時間と 5 章の DualRing による方式の証明時間との比較を示す. 証明時に $O(N \log N)$ 回の冪乗算を行っていて処理全体のボトルネックとなっていた部分が解消されトークン数が 4096 の場合であっても 2 秒以下で証

明を生成することができている．検証にかかる時間については, 両方式ともに $O(N)$ であり, 処理時間はほぼ同様である．

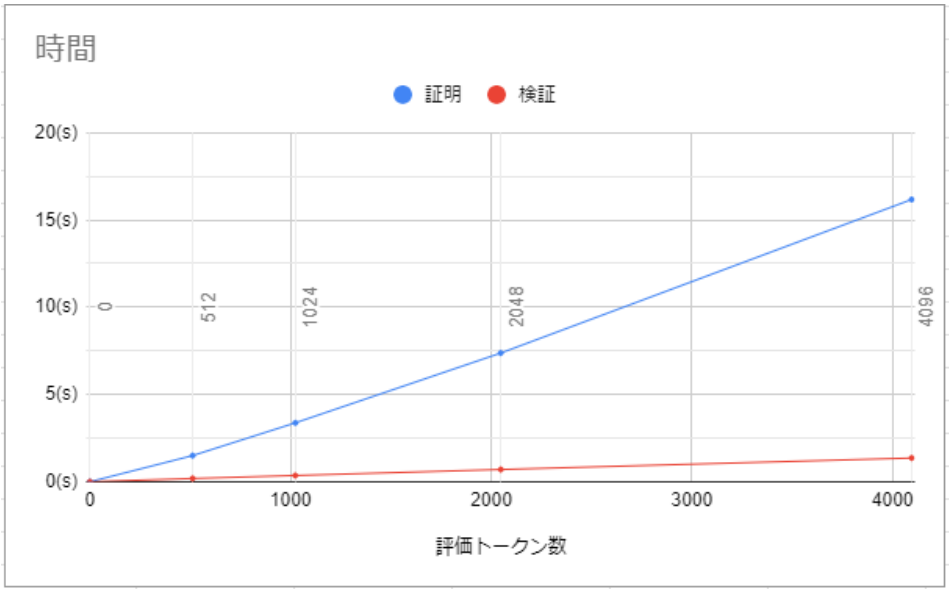

図 6.1　4 章の方式における ShowRep の処理時間

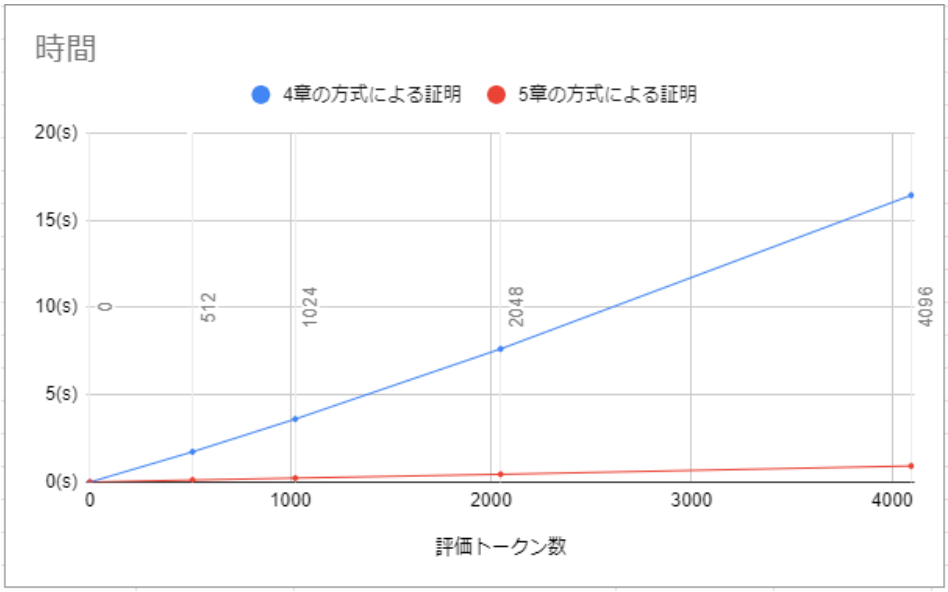

図 6.2　2 つの方式による処理時間の比較

第 7 章　まとめ

本研究では, 事前に信頼できる鍵生成を必要としない非中央集権型匿名評価システムを提案した. 従来方式での信頼できる鍵生成が必要となる点を解消するために, [4] の生成元のパラメータのみを用いた一対多の知識の証明を利用している. これにより, 評価トークンの要素数 N に対して, $O(\log N)$ の証明データサイズを達成している. そして, 提案方式を PC 上で実装し, 評価トークンのリストの要素数を 512 から 4096 まで変化させながら **ShowRep** の証明時間と検証時間を計測した. その結果, 4 章の方式では評価トークンの要素数 N に対して, 証明の処理時間が $\mathrm{O}(N \log N)$ であり N が大きい場合に 10 秒以上の処理時間となることを確認した. しかし DualRing を用いた方式では検証時間に大きな変化がない上で証明にかかる処理時間を $O(N \log N)$ から $O(N)$ へと大きく削減しており, トークン数が 4096 でも 2 秒以内で処理できている. 今後の課題としては, 提案方式の安全性の定式化と証明などが考えられる.

謝辞

本研究は広島大学大学院先進理工系科学研究科・計算機基礎学研究室において行ったものです. 本研究を進めるにあたり中西透教授，および研究室の大学院生から懇切丁寧かつ適切なご指導を賜りました．ここに感謝の意を表します.
また, 日頃から多くの助言, 知識等を頂きました．計算機基礎学研究室の北須賀輝明准教授, 福山大学の今井克暢准教授, ならびに大学院生, 学部生の皆様に心から感謝致します.

参考文献

[1] D. Chaum and E. van Heijst, “Group signatures,” Advances in Cryptology — EUROCRYPT’91, LNCS 547, pp.241–246, Springer-Verlag, 1991.

[2] E. Androulaki, S.G. Choi, S.M. Bellovin and T. Malkin, “Reputation systems for anonymous networks.” Proc. 8th International Symposium on Privacy Enhancing Technologies (PETS2008), LNCS 5134, pp.202 — 218, 2008.

[3] T. Dimitriou, “Decentralized reputation,” CODASPY2021, pp.119 — 130, 2021.

[4] J. Growth and M. Kohlweiss, “One-Out-of-Many Proofs:Or How to Leak a Secret and Spend a Coin,” EUROCRYPT2015, pp.253 — 280, 2015.

[5] T. H. Yuen, M. F. Esgin, J. K. Liu, M. H. Au and Z. Ding, “DualRing: Generic Construction of Ring Signatures with Efficient Instantiations,” CRYPTO 2021, pp 251 — 281, 2021.

[6] M. Abe, M. Ohkubo, K. Suzuki, “1-out-of-n Signatures from a Variety of Keys,” ASIACRYPT 2002, pp.415 — 432, 2002.