Computer Security Symposium 2025
27-31 October 2025

浮動小数点数を用いたカードベース算術演算

小高 駿[1]　駒野 雄一[1,a)]

概要：カードベース暗号とは、物理的なカードを用いて秘密計算など暗号プロトコルを実現する技術である。我々は、CSS2024 において、整数コミットメントを用いた算術演算を提案して統計処理への応用を議論した。統計処理では対象となるデータの数が大きくなると統計値も大きくなるため、整数コミットメントでは処理に必要となるカードの枚数も増大する。この課題に対して、本稿は浮動小数点数表記したコミットメント（浮動小数点数コミットメントとよぶ）を提案する。浮動小数点数コミットメントを用いることで負の数や小数まで扱えるようになり、幅広い数を表現することが可能になる。また、従来の整数コミットメントから浮動小数点数コミットメントへの変換手法と、浮動小数点数コミットメントを入力とした四則演算を実行するプロトコルも提案し、カードベース暗号の統計処理への応用を議論する。

キーワード：カードベース暗号、整数コミットメント、浮動小数点数、統計演算

Card-based Arithmetic using Floating-point Numbers

Shun Odaka[1]　Yuichi Komano[1,a)]

Abstract: Card-based cryptography uses physical cards to realize cryptographic protocols such as secure multiparty computation. At CSS2024, we proposed card-based arithmetic operations using integer commitments and discussed their application to statistical processing. In statistical processing, the statistical value increases as the number of data points increases, so the number of cards required for processing increases with integer commitments. To address this issue, this paper proposes commitments expressed as floating-point numbers (called floating-point commitments). Floating-point commitments allow for the representation of a wide range of numbers, including negative and decimal values. We also propose a protocol for converting conventional integer commitments to floating-point commitments and protocols for performing arithmetic operations using floating-point commitments as input. We also discuss the application of card-based cryptography to statistical processing.

Keywords: card-based cryptography, integer commitment, floating-point numbers, statistical operations

1.　はじめに

カードベース暗号 [1] とは、トランプなどの物理的なカードに情報をのせて、カードを操作することで暗号プロトコルを実現する技術である。例えば、二つの入力ビットを秘匿したまま、それらの積や和を計算する秘密計算などが実現されている [3]。カードベース暗号は、暗号プロトコルの原理の理解を助けることから教育に応用されている。また、ブラックボックスとなる計算機を用いずに計算できるため、安心して暗号プロトコルを実行できることも利点である。

カードベース暗号における秘密計算では、入力となる整数を表現する二種類の方法が存在する。一つは、入力を二進数展開し、各桁のビットを $\boxed{\clubsuit}$ と $\boxed{\heartsuit}$ の 2 枚のカードで表現する方法（バイナリ整数コミットメント）である。この方法を用いてビットごとの積や和を計算する手法は既に知られており、それらを組み合わせれば任意の多倍長演算を行うことができる。この手法を用いた演算は、カードの枚数を扱う整数の対数オーダーに抑えることができるが、

[1] 千葉工業大学
Chiba Institute of Technology
a) yuichi.komano@p.chibakoudai.jp

ビットごとの積や和を用いて複雑な計算が必要となり、多大な回数のカード操作が必要となる。

もう一つの方法は、1 枚の $\boxed{\heartsuit}$ と $k-1$ 枚の $\boxed{\clubsuit}$ を 1 列に並べて、$\boxed{\heartsuit}$ の位置で入力を表現する方法（整数コミットメント [2], [9], [12]）である。この方法を用いると、必要なカードの枚数は増大するが、演算がシンプルであるためにカードの操作回数を減らすことができる。

1.1 関連研究

我々 [6], [7], [14], [15] は、整数コミットメントを入力とした四則演算や平方根、相関係数を効率よく計算する秘密計算プロトコルを提案し、統計処理へのカードベース暗号の応用を議論した。しかし、整数コミットメントを用いてこれらの処理を行うためには、多くの枚数のカードが必要となるという課題があった。

江利口と品川 [13] は、二進数表記でのバイナリ整数コミットメントを入力とした効率的な四則演算プロトコルを提案した。この方式では、演算する整数のビット長 ℓ に対してカード枚数は定数であり、加減算と乗除算の操作回数はそれぞれ $O(\ell)$ 回、$O(\ell^2)$ 回となることが示されている。

本稿は、浮動小数点数表記として符号部、仮数部、指数部を導入した整数の表記法と、それを入力とした演算をはじめて提案する。本稿で提案する浮動小数点数表記により、大きな整数の概数を少ないカード枚数で表現して演算できるほか、小数点以下の桁を含む数も同一の形式で表現して演算できる。

1.2 本稿の成果

本稿の成果は以下のとおり。

- まず、従来の浮動小数点数と同じように、符号部、仮数部、指数部から構成される浮動小数点数コミットメントを提案する。浮動小数点数コミットメントを用いることで正負の数を統一的に表現できるほか、指数部に負の数や正の数を割り当てることで大きな数から小数まで扱うことができる。
- 次に、整数コミットメントを桁表記を用いた整数コミットメント（桁表記整数コミットメント）や浮動小数点数コミットメントへと変換するプロトコルを提案する。これらのプロトコルを用いることで、従来の整数コミットメントを浮動小数点数コミットメントへと変換することができる。
- その後、浮動小数点数コミットメントを入力として、四則演算を実行するプロトコルを提案する。指数部が異なる二つの浮動小数点数コミットメントを入力とした演算を実行するためには、仮数部を調整して指数部を揃える必要がある。本稿では、指数部を揃えるプロトコルを提案した後、加減算と乗算、除算を実行するプロトコルを提案する。これらのプロトコルを用いることで、整数コミットメントを入力とした四則演算よりもカード枚数を抑えながら、任意の精度で演算を行うことが可能となる。
- 最後に、浮動小数点数コミットメントの統計処理への応用について考察し、整数コミットメントに比べて少ないカード枚数で計算ができることを示す。

本稿の構成は以下のとおり。2 節では、本稿で用いるカード組、整数コミットメントと桁表記された整数コミットメント、整数コミットメントを用いた四則演算について述べる。3 節では、整数コミットメントから桁表記や浮動小数点表記された整数コミットメントへ変換するプロトコルと、それを用いた四則演算について述べる。4 節では、浮動小数点数コミットメントを用いた統計処理について考察する。5 節では、まとめと今後の課題を述べる。

2. 準備

本節では、まず本稿で用いるカード組の物理的性質を説明し、次にカード組を用いて整数をエンコードする方法について述べる。その後、本稿で用いるシャッフル操作と、整数コミットメントを入力とした基数拡張や乗除算などの秘密計算プロトコルの概要を説明する。

2.1 カード組

本稿では、表面に $\clubsuit$ または $\heartsuit$ の記号が書かれた物理的なカード組 $\boxed{\clubsuit}\boxed{\heartsuit}$（黒赤カードとよぶ）を用いる。黒赤カードの裏面は同じ模様 $\boxed{?}$ が書かれており、裏面からは表面の記号の区別は付かないものと仮定する。

2.2 整数コミットメント

整数 $i \in [0, k-1]$ を、k 枚の黒赤カードを用いて、

$$\overset{0}{\boxed{\clubsuit}}\overset{1}{\boxed{\clubsuit}}\cdots\overset{i-1}{\boxed{\clubsuit}}\overset{i}{\boxed{\heartsuit}}\overset{i+1}{\boxed{\clubsuit}}\cdots\overset{k-1}{\boxed{\clubsuit}}$$

であらわす。これらのカードを裏返した状態を整数コミットメント [12],[9],[2] とよび、k を基数とよぶ。整数コミットメントのカードの状態を $\boxed{?}\boxed{?}\cdots\boxed{?}$ あるいは $\boxed{?}|||$ であらわし、$E_k(i)$ と表記する。

本稿では、$E_k(i)$ と $E_k(j)$（i と j は秘匿された整数の入力値）に対して、i と j を秘匿したまま $i+j$ の整数コミットメント $E_k(i+j)$ を求める必要がある。

文献 [9] は、pile-shifting シャッフルを利用した秘匿加算プロトコルを提案した。まず、$E_k(j)$ を逆順に並べて得られるカード列を $E_k(i)$ のカード列の下に並べて、$2 \times k$ の行列を得る。そして、後述する pile-shifting シャッフルを実行した後に、2 行目のカード列を逆順に並べなおす。このとき、1 行目は $E_k(i+r \bmod k)$ であり、2 行目は $E_k(j-r \bmod k)$ である。2 行目のカードをオープンして、$\boxed{\heartsuit}$ が現れた位置を左から $\ell+1$ 番目（$\ell = j - r \bmod k$）とするとき、1 行目のカードを右に ℓ だけ巡回シフトする。その結果、1 行

目のカード列は $E_k(i+j \bmod k)$ となる。

2.3 シャッフル

pile-shifting **シャッフル** [5], [10] は、同じサイズのカード束を、それらの束を崩さずにランダムに巡回シフトする操作である。

例えば、k 枚のカードからなる二つのコミットメントが裏向きで次のように並んでいるとしよう。上下のカードを一つの束として考えて pile-shifting シャッフルを適用すると、ランダムな整数 r が生じてカードが右側に r 列だけ巡回シフトし、カードは次のように遷移する。

$$\left(\begin{array}{c|c|c|c} \overset{0}{\boxed{?}} & \overset{1}{\boxed{?}} & \cdots & \overset{k-1}{\boxed{?}} \\ \overset{0}{\boxed{?}} & \overset{1}{\boxed{?}} & \cdots & \overset{k-1}{\boxed{?}} \end{array}\right) \rightarrow \begin{array}{cccc} \overset{-r \bmod k}{\boxed{?}} & \overset{1-r \bmod k}{\boxed{?}} & \cdots & \overset{k-1-r \bmod k}{\boxed{?}} \\ \overset{-r \bmod k}{\boxed{?}} & \overset{1-r \bmod k}{\boxed{?}} & \cdots & \overset{k-1-r \bmod k}{\boxed{?}} \end{array}$$

本稿では、上の例のように列で束ねてその順番をランダムに巡回シフトするシャッフルのほか、行で束ねてその順番をランダムに巡回シフトするシャッフルも利用する。

2.4 基数拡張プロトコル [7], [15]

整数 $x \in [0, k-1]$ の基数 k での整数コミットメント $E_k(x)$ が与えられたとしよう。このとき、$E_k(x)$ の右側に $(k'-k)$ 枚の $\boxed{\clubsuit}$ を伏せて並べると、基数 $k'(>k)$ での整数コミットメント $E_{k'}(x)$ を得ることができる。

このプロトコルに必要な追加カード枚数は $k'-k$ 枚である。また、操作ではシャッフルは使用しない。

2.5 秘匿 2 乗算 [7], [15]

整数 $i \in [0, k-1]$ の基数 k での整数コミットメント $E_k(i)$ を入力として、基数 $(k-1)^2+1$ での整数 $i^2 \in [0, (k-1)^2]$ の整数コミットメントを出力する。

秘匿 2 乗算では、整数コミットメントの左から j 番目（整数 $j-1 \geq 0$ に対応するカード）のカードが左から $\{(j-1)^2+1\}$ 番目のカードとなるように、それぞれのカードの間に $\boxed{\clubsuit}$ を挿入する。整数 i の整数コミットメントは、左から $(i+1)$ 番目に $\boxed{\heartsuit}$ を含んでいる。このとき、挿入操作により $\boxed{\heartsuit}$ は左から (i^2+1) 番目に移動するため、得られる整数コミットメントは整数 i^2 に対応する。

秘匿 2 乗算の出力は $(k-1)^2+1$ 枚の黒赤カードであり、プロトコルの途中で挿入される $\boxed{\clubsuit}$ の枚数は k^2-3k+2 枚である。また、操作ではシャッフルは使用しない。

2.6 秘匿定数乗算 [7], [15]

基数 k での整数 $i \in [0, k-1]$ の整数コミットメント $E_k(i)$ と、公開の整数 $n \geq 1$ を入力として、基数 $n(k-1)+1$ での整数 $ni \in [0, n(k-1)]$ の整数コミットメントを出力する。

秘匿定数乗算では、$n-1$ 枚分の $\boxed{\clubsuit}$ を、それぞれのカードの間に並べる。このプロトコルでは $(n-1)(k-1)$ 枚の $\boxed{\clubsuit}$ を追加カードとして使う。また、操作ではシャッフルを使用しない。

2.7 秘匿定数除算 [6], [7]

基数 k での整数 $i \in [0, k-1]$ の整数コミットメント $E_n(i)$ と、公開の整数 $n \geq 1$（必要に応じて $k = \ell n$ となるように 2.4 節の基数拡張を行っておく）を入力として、i を n で割ったときの商 q と余り r に関してそれぞれ基数 ℓ と n での整数コミットメントを出力する。具体的には、入力 $E_k(i)$ を $n \times \ell$ の行列に並び替え、$\boxed{\heartsuit}$ がある行と列を商と余りに関する整数コミットメントとしてシャッフルを利用しながら秘密裏に取り出す。

このプロトコルの出力は $\ell + n$ 枚の黒赤カードであり、プロトコルで必要なカードの枚数は $k + \ell + n$ 枚である。操作で使用するシャッフルの回数は 2 回である。

2.8 秘匿乗算 [7], [15]

基数 k での整数 $x, y \in [0, k-1]$ の整数コミットメント $E_n(x)$ と $E_n(y)$ を入力として、基数 $(k-1)^2+1$ での整数 $xy \in [0, (k-1)^2]$ の整数コミットメントを出力する。プロトコルのアイデアは式 (1) を利用する。

$$xy = \left\{(x+y)^2 - x^2 - y^2\right\}/2 \tag{1}$$

式 (1) の代わりに、x, y をそれぞれ 2 で割ったものを代入した式を用いると、シャッフル回数を増やす代わりにカード枚数を減らすこともできる。

2.9 秘匿絶対値 [7], [14]

文献 [11] の秘匿減算で得られた値を入力として、その絶対値と符号情報を出力する。このプロトコルは $k+2$ 枚の追加カードと 2 回のシャッフル操作が必要となる。

2.10 桁表記整数コミットメントの加算 [16]

各桁を 10 進数で表記した二つの桁表記整数コミットメントを入力として、それらの和の桁表記整数コミットメントを出力する。本稿では各桁の表記を n 進数に一般化して利用する。

3. 浮動小数点数コミットメントとプロトコル

本節では、まず、新たな数の表記法として、**浮動小数点数コミットメント**を提案する。

次に、入力として整数コミットメントが与えられたときに、n 進数で桁表記された整数コミットメント（**桁表記整数コミットメント**）を出力するプロトコルを提案する。そして、桁表記整数コミットが入力として与えられたときに、浮動小数点数表記した指数部と仮数部からなる整数コミットメント（**浮動小数点数コミットメント**）を出力するプロトコルを提案する。これら二つのプロトコルを実行することで、整数コミットメントを浮動小数点数コミットメント

へと変換できる。

その後、浮動小数点数コミットメントを入力として四則演算を行うプロトコルを提案する。

3.1 浮動小数点数コミットメント

浮動小数点数コミットメントは符号部、仮数部、指数部をあらわす 3 種類のカード列で表現される。

符号部は 2 枚のカードを用いて、♡♣ で負、♣♡ で正をあらわすこととする。ここで、減算がない場合や負の数があらわれないことが明確である場合には、符号部を省略してカードの枚数と操作回数を減らすことができる。

仮数部は、n 進数で有効桁数（精度）を p 桁とする整数であり、各桁が n 枚のカードからなる桁表記整数コミットメントで表現される。したがって、仮数部は pn 枚のカード列となる。

指数部は、$2k-1$ 枚のカードを用いて負の数まで拡張した整数コミットメント [11] で表現する。

3.2 桁表記整数コミットメントへの変換

入力として整数コミットメント $E_k(x)$ が与えられたときに、$x = \Sigma_{j=1}^{\ell} d_j^{(x)} n^{j-1}$ となる ℓ 組の n 進数表記された桁表記整数コミットメント $E_n(d_j^{(x)})$ を出力するプロトコルを提案する。提案するプロトコルは、k 枚のカードを入力として受け付け、$\ell = \lfloor \log_n(k-1) \rfloor + 1$ 桁からなる桁表記整数コミットメントとして ℓn 枚のカードを出力する。

プロトコル 1 (桁表記整数コミットメントへの変換)**.**

(1) 2.4 節の手法で $E_k(x)$ を $E_{n^\ell}(x)$ へと基数拡張する。
(2) $i = 1, 2, \cdots, \ell-1$ として、以下の操作を繰り返す。
  (a) $i = 1$ なら、$E_{n^\ell}(x)$ を n で定数除算する。$i \geq 2$ なら、$(i-1)$ 回目のステップ 2(a) の定数除算で得た商のコミットメントを n で定数除算する。
  (b) 余りを $E_n(d_i^{(x)})$ として保存する。
(3) $(\ell-1)$ 回目のステップ 2(a) の定数除算で得た商のコミットメントを、$E_n(d_\ell^{(x)})$ として保存する。

上述のプロトコルでは、$n^\ell + n^{\ell-1} + n$ 枚のカードと $2(\ell-1)$ 回のシャッフルが必要となる。

3.3 浮動小数点数コミットメントへの変換

$j = 1, 2, \cdots, \ell$ に対して、ℓ 組の $E_n(d_j^{(x)})$ からなる整数 x の桁表記整数コミットメントが入力として与えられたとする。このとき、n 進数で有効桁数（精度）が p 桁の浮動小数点数として表記された $x = \left(\Sigma_{j=1}^{p} f_j^{(x)} \times n^{j-p}\right) \times n^{(x_{exp})}$ に対応する、p 組の仮数部 $E_n(f_j^{(x)})$ と指数部 $E_{2\ell-1}(x_{exp})$ を出力するプロトコルを提案する。

プロトコル 2 (浮動小数点数コミットメントへの変換)**.**

(1) 指数部を保存するための ℓ 枚の ♣ を用意する。
(2) 二つのカード列 ♡♡ と ♣♣ を用意し、それぞれ左側のカードを裏返して ?♡ と ?♣ とする。
(3) $i = \ell, \ell-1, \cdots, 2$ として、以下の操作を繰り返す。
  (a) $E_n(d_i^{(x)})$ の左端のカードを 1 行目に並べ、それ以外のカード列を左右反転して 2 行目に並べる。その後、1 行目の左側に $(n-1)$ 枚の ♣ を裏返して並べ、2 行目の右側に 1 枚の ♣ を裏返して並べる。
  (b) ステップ 2 または $(\ell+1-i)$ 回目の繰り返し処理のステップ 3(g) で得たカード列のうち、右から 1 枚目が ♣ のカード列を 1 行目の右端に置く。同様に、右から 1 枚目が ♡ のカード列を 2 行目の右端に置く。そして、表向きのカードを裏返す。
  (c) 3 行目に、1 枚の ♡ と $(n-1)$ 枚の ♣ を裏返して並べる。
  (d) 上 2 行を pile-shifting シャッフルする。
  (e) 左 n 列を pile-shifting シャッフルする。
  (f) 上 2 行の左 n 列のカードを公開する。♡ が右端に来るように左 n 列のカードを巡回シフトし、3 行目の n 枚のカード列を保存する。このとき、3 行目は $E_n(d_i^{(x)})$ のコピーとなる。
  (g) ステップ 3(f) で ♡ があった行の右から 2 枚目のカードと、ステップ 1 で用意した指数部の左から i 枚目のカードとを交換する。
  (h) 右 2 列を行でまとめて pile-shifting シャッフルする。シャッフル後に、各行の右から 1 枚目のカードを表向きにする。
(4) $(\ell-1)$ 回目の処理のステップ 3(h) で得たカード列のうち、右から 1 枚目が ♡ となるカード列の右から 2 枚目のカードを、指数部の左から 1 枚目と交換する。
(5) 1 行目にステップ 3(f) で得た $E_n(d_i^{(x)})(i = \ell, \ell-1, \cdots, 2)$ のコピーと $E_n(d_1^{(x)})$ を左から順に並べる。2 行目に指数部を左右反転させて並べる。3 行目に $(\ell-1)$ 枚の ♣ と 1 枚の ♡ を裏返して並べる。
(6) 列でまとめて pile-shifting シャッフルする。
(7) 2 行目を公開して、♡ が左端に来るように全体を巡回シフトする。1 行目の左から p 組を仮数部 $E_n(f_p^{(x)}), E_n(f_{p-1}^{(x)}), \cdots, E_n(f_1^{(x)})$ として出力する。
(8) 3 行目の左側に $(\ell-1)$ 枚の ♣ を裏返して並べ、負まで拡張した指数部として出力する。
(9) 整数コミットメントから桁表記にする過程で絶対値プロトコルを実行している場合は、そのときの符号情報を符号部として出力する。それ以外の場合は、正を表す ♣♡ を裏返して出力する。

減算を行わない場合や入力に負がないことが明らかである場合は符号部を用いずに計算できる。上述のプロトコルでは、$\ell n + \ell + 2n + 4$ 枚のカードと $3\ell - 2$ 回のシャッフルが必要となる。

3.4 指数部を揃えるプロトコル

二つの浮動小数点数コミットメントを入力とし、指数が小さな方の値の指数部が指数の大きな方の値の指数部と揃うように仮数部を調整する。我々のプロトコルは、指数が大きな値の浮動小数点数コミットメントは数 x の浮動小数点数コミットメントとしてそのまま出力する。また、もう一方の数は指数部が揃うように仮数部を調整して、数 y の浮動小数点数コミットメントの符号部と仮数部を出力する。

プロトコル 3 (指数部を揃えるプロトコル).

(1) 文献 [8] の手法を用いて、$E_{2k-1}(x_{exp})$ と $E_{2k-1}(y_{exp})$ をそれぞれ二つに複製する。

(2) $E_{2k-1}(x_{exp})$ を 1 行目に並べ、$E_{2k-1}(y_{exp})$ を左右反転して 2 行目に並べる。それぞれの左右に $(k-1)$ 枚の $\boxed{\clubsuit}$ を裏返して並べる。

(3) 列でまとめて pile-shfiting シャッフルし、2 行目を公開する。$\boxed{\heartsuit}$ が左から $(2k-1)$ 枚目となるように巡回シフトし、1 行目を 2.9 節の秘匿絶対値プロトコルに入力して絶対値と符号情報を得る。

(4) x の符号部と仮数部と指数部を 1 行目、y の符号部と仮数部と指数部を 2 行目に並べる。その後、左端にステップ 3 で得た符号情報の左から 1 枚目を 1 行目、2 枚目を 2 行目に並べる。

(5) 行でまとめて pile-shifting シャッフルし、左端のカードを公開する。$\boxed{\heartsuit}$ があった行の符号部と仮数部と指数部を x として出力する。また、$\boxed{\clubsuit}$ があった行の符号部を y の符号部として出力する。

(6) $p > 2k-1$ なら、ステップ 3 で得た絶対値の右側に $p-(2k-1)$ 枚の $\boxed{\clubsuit}$ を裏返して並べる。

(7) ステップ 3 またはステップ 6 で得た絶対値を左右反転して、1 行目に並べる。

(8) 2 行目に、ステップ 5 で $\boxed{\clubsuit}$ があった行の仮数部の p 桁目から 1 桁目までを左から順に並べ、その右に $E_n(0)$ のコミットメントを 1 行目と同じ列数になるように並べる。

(9) 列でまとめて pile-shifting シャッフルし、1 行目のカードを公開する。

(10) 1 行目の $\boxed{\heartsuit}$ が右端に来るように巡回シフトし、2 行目の左から p 組を y の仮数部として出力する。

上述のプロトコルでは、$O((p+k)n)$ 枚のカードと 7 回のシャッフルが必要となる。

3.5 浮動小数点数コミットメントの加減算

x, y の浮動小数点数コミットメント（符号部、仮数部、指数部）を入力として受け付け、$x \pm y$ の浮動小数点数コミットメントを出力する。加減算内の論理演算では $\boxed{\heartsuit}\boxed{\clubsuit} = 0, \boxed{\clubsuit}\boxed{\heartsuit} = 1$ として計算する。計算内に減算がない場合や、負がないこと明らかである場合には、桁表記された整数コミットメントへの加算プロトコル [16] を用いることでシャッフル回数を減らすことができる。

プロトコル 4 (浮動小数点数コミットメントの加減算).

(1) 減算を行う場合は、y の符号部を左右反転する。

(2) x, y を入力として 3.4 節の指数部を揃えるプロトコルを実行する。

(3) 文献 [4] の手法で x, y の符号部の排他的論理和を計算した後に、それを三つに複製する。

(4) 文献 [8] の手法で $E_n(f_i^{(y)})(i = 1, 2, \cdots, p)$ を二つに複製する。

(5) ステップ 4 で複製した p 組の $E_n(f_i^{(y)})$ をそれぞれ 1 行目と 2 行目に並べ、各 $E_n(f_1^{(y)})$ の右端に $\boxed{\clubsuit}$ を裏返して並べる。

(6) 2 行目の各カード束を左右反転する。

(7) ステップ 3 で複製した排他的論理和の左から 1 枚目を 1 行目の右端に、2 枚目を 2 行目の右端に並べる。

(8) 行でまとめて pile-shifting シャッフルし、右端のカードを公開する。

(9) $E_n(f_i^{(x)})(i = 1, 2, \cdots, p)$ とステップ 8 で $\boxed{\heartsuit}$ があった行のカード列で桁表記整数コミットメントの加算を行う。桁表記加算内で 3 行目に $E_n(0)$ を並べる際に、4 行目にも $E_n(0)$ を並べておき、加算結果を二つに複製した状態で保存する。

(10) ステップ 9 で得た $E_n(f_{p+1}^{x+y})$ を文献 [8] の手法で三つに複製する。

(11) ステップ 3 で得た排他的論理和の左から 1 枚目を 1 行目、2 枚目を 2 行目に並べる。1 行目に $E_n(f_{p+1}^{x+y})$ を並べ、2 行目に $E_n(0)$ を並べる。行でまとめて pile-shifting シャッフルし、左端のカードを公開する。$\boxed{\heartsuit}$ があった行のカード列を $E_n(f_{p+1}^{x+y})$ として扱う。

(12) ステップ 11 で $\boxed{\clubsuit}$ があった行を左右反転させたカード列とステップ 3 で得た排他的論理和を入力として、文献 [4] の手法で論理積を計算する。

(13) ステップ 12 で計算した論理積を文献 [4] の手法で三つに複製する。

(14) x の符号部とステップ 13 で得た論理積を入力として、文献 [4] の手法で排他的論理和を計算し、$x \pm y$ の符号部として出力する。

(15) $E_n(f_i^{x+y})(i = 1, 2, \cdots, p+1)$ をそれぞれ 1 行目と 2 行目に並べる。2 行目の各カード列を左右反転する。左端にステップ 13 で得た論理積の左から 1 枚目を 1 行目、2 枚目を 2 行目に並べる。

(16) 行でまとめて pile-shifting シャッフルし、左端のカードを公開する。

(17) $\boxed{\heartsuit}$ があった行のカード列に、ステップ 13 で計算した論理積を 2.4 節の手法で基数を n に拡張したカード列を加算する。

(18) 加算結果を入力として浮動小数点表記への変換プロトコルを実行する。出力された仮数部を加算結果として

出力する。出力された指数部を左に $p-1$ 枚分巡回シフトする。

(19) ステップ 2 で得た指数部の左右に $\boxed{\clubsuit}$ を 1 枚ずつ裏返して並べ、1 行目に並べる。ステップ 18 で得た指数部の左右に $(k-p)$ 枚の $\boxed{\clubsuit}$ を裏返して並べ、2 行目に並べる。

(20) 列でまとめて pile-shfiting シャッフルし、2 行目を公開する。2 行目の $\boxed{\heartsuit}$ が $k+1$ 枚目に来るように巡回シフトして、1 行目を指数部として出力する。

上述のプロトコルでは、$O((p+k)n)$ 枚のカードと $O(p)$ 回のシャッフルが必要となる。

3.6 浮動小数点数コミットメントの乗算

x, y の浮動小数点数コミットメントを入力として受け付け、その乗算結果の浮動小数点数コミットメントを出力する。プロトコル内の秘匿乗算プロトコルで適用する式を変えることで、シャッフル回数とカード枚数のどちらを優先するか調整できる。

プロトコル 5 (浮動小数点数コミットメントの乗算).

(1) $i = 1, 2, \cdots, p$ として、以下の操作を繰り返す
  - (a) $j = 1, 2, \cdots, p$ として、以下の操作を繰り返す。
    - (i) $E_n(f_i^{(x)}) \times E_n(f_j^{(y)})$ を 2.8 節の秘匿乗算プロトコルで計算する。
    - (ii) 計算結果を n で定数除算する。商と余りの基数をそれぞれ np に拡張し、商を $E_{np}(d_{i+j}^{(r)})$ に、余りを $E_{np}(d_{i+j-1}^{(r)})$ とする。すでに該当する変数のカード組が存在する場合には、そのカード組に基数拡張した商または余りを秘匿加算する。

(2) $i = 1, 2, \cdots, 2p$ として、以下の操作を行う。
  - (a) $i \geq 2$ のとき、$E_{np}(d_i^{(r)})$ に $(i-1)$ 回目の繰り返し処理のステップ 2(c) で保存した $E_{np}(d_i^{(c)})$ を加算する。
  - (b) $E_{np}(d_i^{(r)})$ を n で定数除算する。余りを $E_n(d_i^{(xy)})$ として保存する。
  - (c) ステップ 2(b) で得た商の基数を np に拡張し、商を $E_{np}(d_{i+1}^{(c)})$ として保存する。

(3) $E_{np}(d_{2p+1}^{(c)})$ を $E_n(d_{2p+1}^{(xy)})$ として保存する。

(4) $E_n(d_i^{(xy)})(i = 1, 2, \cdots 2p+1)$ を入力として、3.3 節の浮動小数点コミットメントへの変換を実行する。

(5) ステップ 4 で得た指数部を左に $2p-1$ 枚分だけ巡回シフトする。

(6) $E_{2k-1}(x_{exp})$ を 1 行目に並べ、$E_{2k-1}(y_{exp})$ を左右反転して 2 行目に並べる。それぞれの左右に $(k-1)$ 枚の $\boxed{\clubsuit}$ を並べる。列でまとめて pile-shifting シャッフルし、$\boxed{\heartsuit}$ が $2k-1$ 枚目に来るように巡回シフトする。

(7) ステップ 6 で得た指数部の左右に $\boxed{\clubsuit}$ を 1 枚ずつ裏返して並べ、1 行目に並べる。ステップ 5 で得た指数部の左右に $\boxed{\clubsuit}$ を $(2k-p-1)$ 枚ずつ裏返して並べ、左右反転して 2 行目に並べる。

(8) 列でまとめて pile-shifting シャッフルする。2 行目を公開し、$\boxed{\heartsuit}$ が $2k$ 枚目に来るように巡回シフトする。1 行目を指数部として出力する。

(9) $\boxed{\heartsuit}\boxed{\clubsuit} = 1, \boxed{\clubsuit}\boxed{\heartsuit} = 0$ として、文献 [4] の手法で x, y の符号部の排他的論理和を計算し、乗算結果の符号部として出力する。

上述のプロトコルでは、$O(n^2 + pn)$ 枚のカードと $O(p^2)$ 回のシャッフルが必要となる。

3.7 浮動小数点数コミットメントの除算

x, y の浮動小数点数コミットメントを入力として受け付け、$q = x/y$ の浮動小数点数コミットメントを出力する。また、$y = 0$ であったときに 2 枚のカードによるコミットメントを出力する。

プロトコル 6 (浮動小数点数コミットメントの除算).

(1) $E_n(d_p^{(y)})$ の左から 1 枚目のカードと 2 枚目のカードの間に n 枚の $\boxed{\clubsuit}$ を裏返して並べる。

(2) ステップ 1 で得たカード列に対して、n で定数除算する。

(3) ステップ 2 で得た商を y が 0 であるか否かをあらわすフラグとして出力する。このとき、出力は、$y = 0$ ならば $\boxed{\heartsuit}\boxed{\clubsuit}$ となり、$y \neq 0$ ならば $\boxed{\clubsuit}\boxed{\heartsuit}$ となる。

(4) ステップ 2 で得た余りを再び $E_n(d_p^{(y)})$ として扱う。

(5) $E_n(f_i^{(y)})(i = 1, 2, \cdots, p)$ を文献 [8] の手法で二つに複製する。

(6) $i = 0, 1, \cdots, \lfloor \log_2 n \rfloor - 1$ として、以下の操作を繰り返す。
  - (a) $j = 1, 2, \cdots, p+1$ として、以下の操作を繰り返す。ただし、$i = 0$ のときには $j = p+1$ の処理をスキップする。
    - (i) $E_n(f_i^{(2^i y)})$ を 2 で定数乗算する。
    - (ii) $j \geq 2$ ならば $E_n(c_j)$ の基数を $2n-1$ に拡張し、ステップ 6(a)(i) で得たカード列に加算する。
    - (iii) ステップ 6(a)(ii) で得たカード列を n で定数除算する。このとき、4 行目にも 0 のコミットメントを追加して余りのコミットメントを二つ得る。余りを $E_n(f_i^{(2^{i+1} y)})$ として保存する。商を $E_k(c_j)$ として保存する。$i = 0$ かつ $j = p$ のとき、$E_n(f_{p+1}^{(2y)})$ として保存する。

(7) $i = p+1, p, \cdots, 1$ として以下の操作を繰り返す。
  - (a) 商のコミットメント $E_n(d_i^{(q)})$ を保存するための $E_n(0)$ を用意する。
  - (b) $i = p+1$ のとき、$E_n(f_{i+1}^{(x)})$ を保存するための $E_n(0)$ を用意する。$i \leq p$ のとき、$E_n(f_1^{(x)})$ を保

存するための $E_n(0)$ を用意する。

(c) $j=\lfloor\log_2 n\rfloor, \lfloor\log_2 n\rfloor-1, \cdots, 0$ として、以下の操作を繰り返す。

(i) $m=1,2,\cdots,p+1$ として、以下の操作を繰り返す。

(A) $E_n(f_m^{(2^j y)})$ と $E_2(c_{m-1})$ の基数をそれぞれ $n+1$ に拡張し、秘匿加算する。ただし、$m=1$ のときには $c_0=0$ とする。

(B) $E_n(f_m^{(x)})$ の基数を $n+1$ に拡張し、ステップ 7(c)(i)(A) で得たカード列による減算を文献 [11] の手法で計算する。

(C) カード列の右端のカードを取り除き、n で定数除算する。商を左右反転して $E_2(c_m)$ として得る。また、余りを $E_n(r_m)$ として保存する。

(ii) $E_2(c_{p+1})$ の左から 1 枚目を 1 行目、2 枚目を 2 行目に並べる。1 行目の右側に $E_n(r_i)(i=1,2,\cdots,p+1)$ を並べ、さらに右側に $E_n(2^j)$ を並べる。同様に 2 行目の右側に $E_n(f_i^{(x)})(i=1,2,\cdots,p+1)$ と $E_n(0)$ を並べる。

(iii) 行でまとめて pile-shifting シャッフルする。その後、左端のカードを公開する。

(iv) ステップ 7(c)(iii) で $\boxed{\heartsuit}$ があらわれた行の右端のカード列を、$E_n(d_i^{(q)})$ に加算する。

(v) ステップ 7(c)(iii) で $\boxed{\heartsuit}$ があらわれた行の中央のカード列を、$E_n(f_i^{(x)})(i=1,2,\cdots,p+1)$ として次の繰り返し処理で使用する。

(d) $E_n(d_i^{(q)})$ を商として出力する。

(e) ステップ 7(c)(v) で得た $E_n(f_i^{(x)})(i=1,2,\cdots,p+1)$ を新たに $E_n(f_{i+1}^{(x)})=E_n(f_i^{(x)})(i=1,2,\cdots,p+1)$ とする。その後、$E_n(f_{p+2}^{(x)})$ を取り除く。

(8) $E_{2k-1}(x_{exp})$ を 1 行目に並べ、$E_{2k-1}(y_{exp})$ を 2 行目に並べる。それぞれの左右に $(k-1)$ 枚の $\boxed{\clubsuit}$ を並べる。列でまとめて pile-shifting シャッフルし、$\boxed{\heartsuit}$ が $(2k-1)$ 枚目に来るように巡回シフトする。

(9) $E_n(d_i^{(q)})(i=1,2,\cdots,p+1)$ を入力として、浮動小数点数コミットメントへの変換プロトコルを行う。出力された仮数部 $E_n(f_i^{(q)})(i=1,2,\cdots,p)$ が商となる。

(10) ステップ 9 で得た指数部の左右に $(2k-p-1)$ 枚の $\boxed{\clubsuit}$ を裏返して並べ、左に p 枚分巡回シフトする。

(11) ステップ 8 で得た指数部の左右に 1 枚の $\boxed{\clubsuit}$ を並べ、1 行目に並べる。ステップ 10 で得た指数部を、左右反転して 2 行目に並べる。列でまとめて pile-shifting シャッフルし、2 行目を公開して $\boxed{\heartsuit}$ が $2k$ 枚目に来るように移動させる。そして、1 行目を指数部として出力する。

(12) $\boxed{\heartsuit}\boxed{\clubsuit}=1, \boxed{\clubsuit}\boxed{\heartsuit}=0$ として、x, y の符号部を文献 [4] の手法で排他的論理和を計算し、除算結果の符号部として出力する。

上述のプロトコルでは、$O(pn\log n)$ 枚のカードと $O(p^2\log n)$ 回のシャッフルが必要となる。

4. 考察

本節では、100 人の生徒が受験した 100 点満点のテストの平均点を求める場面を例に、浮動小数点数コミットメントの応用について考察する。

整数コミットメントを用いる場合は、2×10^4 枚のカードを用いてすべての学生の点数の合計点を計算したのち、定数除算を用いて小数点以下を繰り上げた平均値を求めることができる。

次に、浮動小数点数コミットメントを用いる場合を検討する。今回の例では、負の数を考慮する必要がない。そのため、まず、10 進数の桁表記整数コミットメントを用いて合計点を計算し、浮動小数点数表記に変換してから除算を行うことで、効率的に平均点を計算することができる。合計点の計算においては、最初に 100 人分の点数のカードを揃えるのではなく、一人ずつ順番にカードを用意して加算してゆくことで、必要な枚数を 1×10^2 枚程度に抑えることができる。その後、浮動小数点数表記に変換し、指数部を -2 とすることで、小数点以下まで平均を求めることができる。

以上のように、浮動小数点数コミットメントではカード枚数を整数コミットメントに比べて抑えることができ、小数点以下の桁まで表現できる。より大きな数を扱う場面では、浮動小数点数コミットメントを用いることによるカード枚数の削減の効果は大きくなる。また、負の数も統一的に扱うことができることも浮動小数点数コミットメントの利点である。その他、条件によってはシャッフルの回数を効率化できる場合もある。

一方で、定数乗算や二乗算のような整数コミットメントであればシャッフル操作が不要な演算に対して、桁表記整数コミットメントや浮動小数点数コミットメントではシャッフル操作が必要となる場合もある。

5. まとめ

本稿では、浮動小数点数コミットメントと浮動小数点数コミットメントを入力とした四則演算のプロトコルを提案し、整数コミットメントに比べてカード枚数を抑えながら任意の精度で四則演算を行うことができることを示した。今後は、浮動小数点数コミットメントを用いて、統計処理に必要となる平方根などを計算するプロトコルを構成し、カードベース暗号の統計分野への応用について報告する予定である。

謝辞: 本研究は JSPS 科研費 JP24K14951 の助成を受けたものです。

参考文献

[1] Bert Den Boer. More efficient match-making and satisfiability the five card trick. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *Advances in Cryptology—EUROCRYPT '89*, volume 434 of *LNCS*, pages 208–217, Berlin, Heidelberg, 1990. Springer.

[2] Daiki Miyahara, Yu-ichi Hayashi, Takaaki Mizuki, and Hideaki Sone. Practical card-based implementations of Yao's millionaire protocol. *Theor. Comput. Sci.*, 803:207–221, 2020.

[3] Takaaki Mizuki and Hiroki Shizuya. Computational model of card-based cryptographic protocols and its applications. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, E100.A(1):3–11, 2017.

[4] Takaaki Mizuki and Hideaki Sone. Six-card secure AND and four-card secure XOR. In *Frontiers in Algorithmics, LNCS 5598*, pages 358–369. Springer, 2009.

[5] Akihiro Nishimura, Yu-ichi Hayashi, Takaaki Mizuki, and Hideaki Sone. Pile-shifting scramble for card-based protocols. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 101(9):1494–1502, 2018.

[6] Shun Odaka and Yuichi Komano. Card-based arithmetic operations and application to statistical data aggregation. In *SecITC 2024, LNCS 15595*, pages 118–134. Springer, 2025.

[7] Shun Odaka and Yuichi Komano. Card-based arithmetic operations using integer commitments and their application to statistical data aggregation. *IEICE Transactions on Fundamentals*, (0):0–0, 2025.

[8] Suthee Ruangwises. Using five cards to encode each integer in Z/6Z. In Peter Y. A. Ryan and Cristian Toma, editors, *SecITC 2021, LNCS 13195*, pages 165–177. Springer, 2021.

[9] Suthee Ruangwises and Toshiya Itoh. Securely computing the n-variable equality function with $2n$ cards. *Theor. Comput. Sci.*, 887:99–110, 2021.

[10] Kazumasa Shinagawa, Takaaki Mizuki, Jacob Schuldt, Koji Nuida, Naoki Kanayama, Takashi Nishide, Goichiro Hanaoka, and Eiji Okamoto. Card-based protocols using regular polygon cards. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, E100.A(9):1900–1909, 2017.

[11] Mizuki Takaaki, Kuzuma Tomoki, Hirano Tomoya, Oshima Ririn, and Yasuda Momofuku. Gakmoro: An application of physical secure computation to card game. In ???, editor, *Unconventional Computation and Natural Computation*, volume ??? of *LNCS*, pages ???–???, Cham, 2025. Springer.

[12] Ken Takashima, Yuta Abe, Tatsuya Sasaki, Daiki Miyahara, Kazumasa Shinagawa, Takaaki Mizuki, and Hideaki Sone. Card-based protocols for secure ranking computations. *Theor. Comput. Sci.*, 845:122–135, 2020.

[13] 江利口 礼央, 品川 和雅. 四則演算に対する効率的なカードベースプロトコル. *2025* **年暗号と情報セキュリティシンポジウム** *(SCIS 2025), 4D2-4*, 2025.

[14] 小高 駿, 駒野 雄一. カードベース暗号における効率的な平方根計算と相関分析への応用. *2025* **年暗号と情報セキュリティシンポジウム** *(SCIS 2025), 3D2-2*, 2025.

[15] 小高駿, 駒野雄一. カードを用いた秘匿算術演算と秘匿統計演算. **コンピュータセキュリティシンポジウム** *(CSS)2024*, pages 2003–2010, 2024.

[16] 猪狩 紫雲, 小高 駿, 駒野 雄一, 水木 敬明. カードを用いる桁表記された整数コミットメントと加算プロトコル. *2025* **年暗号と情報セキュリティシンポジウム** *(SCIS 2025), 3D2-3*, 2025.