一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report

紛失通信とアダマール変換を用いてポイズニング安全性を強化したLDP方式の提案

清水　正浩†　　菊池　浩明†

† 明治大学　総合数理学部　　〒164-8525 東京都中野区中野 4-21-1
E-mail: †{ev200592,kikn}@meiji.ac.jp

あらまし　近年, スマートデバイスからプライバシーを考慮してユーザの使用履歴を収集し, 利活用するため, 局所差分プライバシ（Local Differential Privacy, LDP）が盛んに用いられている. しかし, 局所差分プライバシはユーザ自身が局所的にノイズ処理を行うために, 悪意のあるユーザが意図的にデータを加工してサーバに送信することで, 集計結果を操作するポイズニング攻撃に対して脆弱であることが Cao らによって指摘されている. そこで, 本研究では, ポイズニング攻撃に対するロバスト性を向上させるために, 局所差分プライバシ方式 CMS に紛失通信プロトコルの適用を検討する. ハッシュ関数の値域に比例して送信量と処理コストが増加してしまうという課題に対して, アダマール変換を応用した Hadamard Count Mean Sketch(HCMS) を導入する. 提案方式を試験実装し, オープンデータを用いて提案方式の安全性と効率を評価する.

キーワード　局所差分プライバシ, Count Mean Sketch, Hadamard Count Mean Sketch, 紛失通信, アダマール変換

A poisoning-resilient LDP schema using oblivious transfer and Hadamard transform

Masahiro SHIMIZU† and Hiroaki KIKUCHI†

† School of Interdisciplinary Mathematical Sciences,　4-21-1 Nakano, Nakano-ku, Tokyo, 164-8525 Japan
E-mail: †{ev200592,kikn}@meiji.ac.jp

Abstract　In recent years, Local Differential Privacy (LDP) has been actively used to collect and utilize users' usage history from smart devices with privacy considerations. However, since LDP allows users to locally process noise themselves, it has been pointed out by Cao et al. that it is vulnerable to poisoning attacks, where malicious users can intentionally manipulate data and send it to servers, thereby tampering with the aggregation results. Therefore, this study examines the application of a Oblivious Transfer(OT) protocol to the LDP protocol CMS to improve robustness against poisoning attacks. Facing the challenge that the amount of data transmission and processing costs increase in proportion to the range of hash function values, we introduce the Hadamard Count Mean Sketch (HCMS) utilizing the Hadamard transform.The proposed method is experimentally implemented, and its security and efficiency are evaluated using open data.

Key words　Local Differential Privacy, Count Mean Sketch, Hadamard Count Mean Sketch, Oblivious Transfer, Hadamard transform

1. はじめに

近年, スマートデバイスの普及により, サービス事業者はユーザの使用履歴を盛んに収集し, 利活用している. しかし, サービス事業者はユーザの多くの機微なデータを収集するため, ユーザのプライバシーを十分に守ることができないという課題があった. その課題を解決するために Duchi らによって局所差分プライバシ（Local Differential Privacy Protocol, LDP）が提案された [6]. LDP はデータにユーザ自身でノイズを加えた後に, サービス事業者に送信する. サービス事業者はユーザから送信されたデータに脱ノイズ処理を施し, 集計を行う.

しかし, 局所差分プライバシはユーザが局所的にノイズ処理を行うために, 悪意のあるユーザが意図的なデータをサーバに送信して, 集計結果を操作するポイズニング攻撃に対して脆弱であることが Cao らによって指摘されている [1].

そこで, 本研究では, 2017 年に Apple から提案された局所差

分プライバシ方式 Count Mean Sketch(CMS) に対する, 3 つのポイズニング攻撃を検討する. ポイズニング攻撃に対するロバスト性を向上させるために, CMS に紛失通信プロトコル Oblivious Transfer [4] を適用した OT-CMS を提案する. しかし, OT-CMS は送信コストと計算コストがハッシュ関数の値域に比例して増加してしまうという問題点がある. そこで, その問題点を解決するためにアダナール変換に着目する. 紛失通信プロトコル Hadamard Count Mean Sketch(HCMS) [3] を導入し, 紛失通信プロトコルの通信コストを削減した OT-HCMS を提案する. 本研究の概要を図 1 に示す.

Apple Privacy Team は局所差分プライバシ方式 Count Mean Sketch [3] を提案した. CMS は 3 次独立なハッシュ関数の集合を用いて, 任意の長さのベクトルにエンコードし, 各ビットをランダマイズする. さらに, サーバは各ユーザから収集したランダマイズデータに脱ノイズ処理を行い, Sketch Matrix を作成し, 各アイテムの頻度を集計する.

Cao らは 3 つの Pure Protocol に対して 3 つのポイズニング攻撃 Maximal Gain Attack(MGA), Random Item Attack(RIA), Random Perturb Attack(RPA) [1] を提案し, MGA が最も強力なことを示した. 堀込らは key-value データに対する局所差分プライバシ方式 PrivKV, PrivKVM のポイズニング攻撃に対する強度を高めるために EM アルゴリズムを用いた推定方法を提案, 評価した [2]. また, PrivKV に対して 1-out-of-n Oblivious Transfer を用いた方式を提案した. しかし, OT を導入することで生じる通信コストや計算コストについては不明であった.

そこで本研究では, CMS に対して, Oblivious Transfer を導入して局所差分プライバシ方式を試験実装し, そのコストなどを多角的に評価する.

本研究の新規性は以下の 4 つである.

（1） CMS と HCMS に Oblivious Transfer を導入した OT-CMS, OT-HCMS を提案すること.

（2） CMS と HCMS に対するポイズニング攻撃の影響を評価したこと.

（3） Count Mean Sketch(CMS) と Hadamard Count Mean Sketch(HCMS) の精度に基づき, 提案方式のポイズニング攻撃に対する強度を評価したこと.

（4） 提案方式の Oblivious Transfer を導入したことによる計算コストを実験的に評価したこと.

2. 準　　備

2.1 基 本 定 義

使用する記号を表 1 に整理する.

各ユーザーは自身のプライベートなデータ d にノイズを付与し, 摂動化されたデータ $\tilde{d}$ をサービス事業者に送信する. サービス事業者は各ユーザから収集したデータを集計し, 頻度などを推定する. 各ユーザーはデータにノイズを付与する際, プライバシー予算を ε としたランダマイズアルゴリズム $M(d, \varepsilon)$ を用いる.

局所差分プライバシーは, 任意の異なる 2 つの入力に対して, ランダマイズアルゴリズム M の出力確率に区別がつかないことを保証する. これにより, サービス事業者は送信されたデータを用いてユーザの真の入力を知ることができず, ユーザのプライバシーが保証される. ランダマイズアルゴリズム M に対して局所差分プライバシーは以下のように定義される.

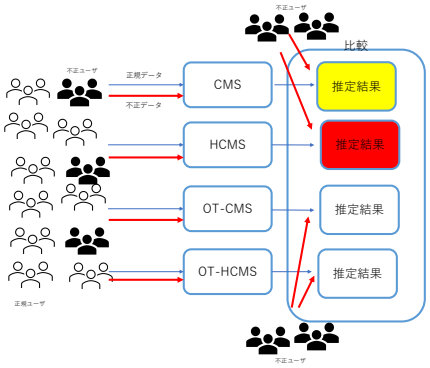

図 1　研究概要図

表 1　記 号 一 覧

記号	意味
D	アイテムの集合
ε	プライバシー予算
H	ハッシュ関数の集合
k	ハッシュ関数の集合の数
m	ハッシュ関数の値域の大きさ
n	正規ユーザの数
n'	不正ユーザの数
β	不正ユーザの割合
r	ターゲットアイテムの数

［定義 1］ D を入力の集合, Z を出力の集合とする. ランダマイズアルゴリズム M は入力 $d \in D$ を取り, $z \in Z$ を出力する. 任意の異なる 2 つの入力 $d_1, d_2 \in D$ の出力 $z \in Z$ に対して,

$$Pr(M(d_1, \varepsilon) = z) \leq e^{\varepsilon} Pr(M(d_2, \varepsilon) = z)$$

が成立するとき, ランダマイズアルゴリズム ε-局所差分プライバシを満たすという.

2.2 Count Mean Sketch

Count Mean Sketch(CMS) は 2017 年に Apple が提案した局所差分プライバシ方式の一つである [3]. CMS はユーザの使用歴を収集し, その頻度を推定する. ユーザとサーバでハッシュ関数を共有する.

入力: ハッシュ関数の集合を $H = \{h_j | h_j : D \to [m], j \in [k]\}$ とする. 各ユーザは H から一様に取得したハッシュ関数を用い, 自身のデータ $d \in D$ を次の M に従い, $\{1, -1\}$ の m 次元ベクトル $\boldsymbol{v}$ に変換する.

(例 1) あるサービスを使用しているユーザの性別の頻度を推定する場合を考える. $D = \{\text{“男”}, \text{“女”}\}, m = 2, k = 4$ とする. ユーザが $d = \text{“男”} \in D$ というデータを持っている場合, j を $\{1, 2, 3, 4\}$ から一様ランダムにサンプリングをして $j = 3 \in \{1, 2, 3, 4\}$ とする. $h_3(\text{“男”}) = 2$ を計算する. 2 次元ベクトル $\boldsymbol{v}$ の $h_3(\text{“男”}) = 2$ 番目の要素を 1 に設定し, そのほかの要素を -1 とする. 得られる 2 次元ベクトルは $\boldsymbol{v} = (-1, 1)$ である.

$\boldsymbol{v} = (-1, 1)$ に摂動化を施した $\tilde{\boldsymbol{v}}$ をサーバに送信する.

摂動:m 次元ベクトルを $(v_1, ..., v_m)$ とする. $i = 1, ..., m$ について確率 p で真の値 v_i を出力し, 確率 $q = 1 - p$ で $-v_i$ を出力する. すなわち,

$$\tilde{v_i} = \begin{cases} v_i & w./p. \quad p, \\ -v_i & w./p. \quad q. \end{cases}$$

ここで,

$$p = \frac{e^{\frac{\varepsilon}{2}}}{1 + e^{\frac{\varepsilon}{2}}}, \quad q = \frac{1}{1 + e^{\frac{\varepsilon}{2}}}$$

のとき, ε-局所差分プライバシを満たす.

集計:n 人のユーザからの出力を収集し, 各 $d_i \in D$ の頻度を推定する. CMS は収集したデータを用いて, Sketch Matrix と呼ばれる $k \times m$ の行列を作成する. ユーザから収集したデータの集合を $S = \{(\tilde{\boldsymbol{v}}^{(1)}, j^{(1)}), ..., (\tilde{\boldsymbol{v}}^{(n)}, j^{(n)})\}$, $c_\varepsilon = \frac{e^{\varepsilon/2}+1}{e^{\varepsilon/2}-1}$ と定義する. この時, $\tilde{\boldsymbol{v}}^{(i)}, k, m$ を用いて, $\tilde{\boldsymbol{x}}^{(i)} = k(\frac{c_\varepsilon}{2}\tilde{\boldsymbol{v}}^{(i)} + \frac{1}{2}\mathbf{1})$ を計算する. ただし, $\mathbf{1}$ は全ての要素が 1 の m 次元ベクトルである. $\tilde{\boldsymbol{x}}^{(i)}$ を累積して, Sketch Matrix SK を構築する. $i \in [n], \ell \in [k]$ とすると, M は $j^{(i)}$ 行 ℓ 列の要素 $SK_{j^{(i)},\ell}$ に $\tilde{x}_{\ell^{(i)}}$ の累積する. Sketch Matrix SK から,

$$\tilde{f}(d) = \left(\frac{m}{m-1}\right)\left(\frac{1}{k}\sum_{\ell=1}^{k} SK_{\ell,h_\ell(d)} - \frac{n}{m}\right)$$

として, アイテム d のハッシュエントリを平均化することによって, $\tilde{f}(d)$ を推定する.

(例 2) あるサービスを使用しているユーザの性別の頻度を推定する場合を考える. $n = 4, D = \{$“男”, “女”$\}, m = 2, k = 2, \varepsilon = \infty, S = \{((1,-1),1), ((1,-1),2), ((-1,1),1), ((-1,1),2)\}$ とする. この時, $\boldsymbol{x}^{(1)} =$ “男”, $\boldsymbol{x}^{(2)} =$ “男”, $\boldsymbol{x}^{(3)} =$ “女”, $\boldsymbol{x}^{(4)} =$ “女” とし, $\tilde{\boldsymbol{x}}^{(1)}, \tilde{\boldsymbol{x}}^{(2)}, \tilde{\boldsymbol{x}}^{(3)}, \tilde{\boldsymbol{x}}^{(4)}$ は以下のように計算される.

$$\tilde{\boldsymbol{x}}^{(1)} = 2\left(\frac{1}{2}(1,-1) + \frac{1}{2}(1,1)\right) = (2,0)$$

$$\tilde{\boldsymbol{x}}^{(2)} = 2\left(\frac{1}{2}(1,-1) + \frac{1}{2}(1,1)\right) = (2,0)$$

$$\tilde{\boldsymbol{x}}^{(3)} = 2\left(\frac{1}{2}(-1,1) + \frac{1}{2}(1,1)\right) = (2,0)$$

$$\tilde{\boldsymbol{x}}^{(4)} = 2\left(\frac{1}{2}(-1,1) + \frac{1}{2}(1,1)\right) = (0,2)$$

この時, Skecth Matrix は

$$SK = \begin{pmatrix} 4 & 0 \\ 2 & 2 \end{pmatrix}$$

と得られる. 従って, 推定頻度は

$$\tilde{f}(\text{“男”}) = \frac{2}{2-1}\left(\frac{1}{2}\sum_{\ell=1}^{2} SK_{\ell,h_\ell(\text{“男”})} - \frac{4}{2}\right) = 2$$

$$\tilde{f}(\text{“女”}) = \left(\frac{2}{2-1}\right)\left(\frac{1}{2}\sum_{\ell=1}^{2} SK_{\ell,h_\ell(\text{“女”})} - \frac{4}{2}\right) = 2$$

である. (誤差なし)

2.3 Hadamard Count Mean Sketch

HCMS は Apple によって提案された CMS の亜種である [3]. CMS では, ユーザは m 次元ベクトルを送信するため, ハッシュ関数の値域の大きさに比例して送信量が大きくなってしまう. その問題点を解決するために, CMS に改良を施したのが HCMS である. HCMS はアダマール行列を次のように適用することによって, 送信量を減らす. また, アダマール行列はユーザとサーバで共通しているものとする.

入力: ハッシュ関数の集合を $H = \{h_j | h_j : D \to [m] : j \in [k]\}$ とおく. 各ユーザは自身のデータ $d \in D$ を H から一様に取得したハッシュ関数を用いて m 次元ベクトル $\boldsymbol{v}$ に変換する. $m \times m$ のアダマール行列と積をとり m 次元ベクトル $\boldsymbol{w}$ に変換する. そのベクトル $\boldsymbol{w}$ の中から一様に 1 ビットを取得し, 摂動化する. その 1 ビットをサーバに送信する.

アダマール行列は, 以下のように再帰的に定義される. $m = 1$ について,

$$H_1 = \begin{pmatrix} 1 \end{pmatrix},$$

$m > 1$ について.

$$H_m = \begin{pmatrix} H_{m/2} & H_{m/2} \\ H_{m/2} & -H_{m/2} \end{pmatrix}.$$

(例 3)(例 1) と同様にユーザの性別の頻度を推定する場合を考える. $D =$ {“男”, “女”}, $m = 2, k = 4$ とする. この時, あるユーザが $d =$ “男” $\in D$ というデータを持っている場合, 次のような処理をする. j を $\{1,2,3,4\}$ から一様にサンプリングをして $j = 3 \in \{1,2,3,4\}$ とする. h_3(“男”) $= 1$ とする. m 次元ベクトル $\boldsymbol{v}$ の 1 番目の要素を 1 に, そのほかの要素は例 1 と異なり 0 とする. 結果として得られる 2 次元ベクトルは $v = (0,1)$ と表される. $\boldsymbol{w}$ は

$$\boldsymbol{w} = H_2\boldsymbol{v} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

となり, m 次元の要素の中から一様に $\ell = 1 \in \{1,2\}$ をサンプリングし, 次のように摂動化する.

摂動:$\boldsymbol{w}$ の中から一様に取得された 1 ビットを w とおく. 確率 p で真の値 v を出力し, 確率 $q = 1 - p$ で $-v$ を出力する.

$$\tilde{w} = \begin{cases} w & w./p. \quad p, \\ -w & w./p. \quad q. \end{cases}$$

ここで,

$$p = \frac{e^{\varepsilon}}{1 + e^{\varepsilon}}, \quad q = \frac{1}{1 + e^{\varepsilon}}$$

のとき, ε-局所差分プライバシを満たす. $\tilde{w}_\ell, j = 3, \ell = 1$ をサーバに送信する.

集計:n 人のユーザからの出力を収集し, 各 $d \in D$ の頻度を推定する. ユーザから収集した摂動化データを

$S = \left((\tilde{w}^{(1)}, j^{(1)}, \ell^{(1)}), ..., (\tilde{w}^{(n)}, j^{(n)}, \ell^{(n)})\right)$, ε をプライバシー予算, 定数 $c_\varepsilon = \frac{e^\varepsilon+1}{e^\varepsilon-1}$ と定義する. この時, $\tilde{w}^{(i)}, k, m$ を用いて, $\tilde{x}^{(i)} = kc_\varepsilon w^{(\tilde{i})}$ を求める. $\tilde{x}^{(i)}$ を用いて, Sketch Matrix SK を構築する. $i \in [n], \ell \in [m]$ とすると, SK の $j^{(i)}$ 行 $\ell^{(i)}$ 列の要素 $SK_{j^{(i)},\ell^{(i)}}$ に $\tilde{x}_{\ell^{(i)}}$ を累積する. Sketch Matrix に, アダマール行列を適用し, 正規化する. 各アイテムのハッシュエントリを平均化することによって,

$$\tilde{f}(d) = \left(\frac{m}{m-1}\right)\left(\frac{1}{k}\sum_{\ell=1}^{k} SK_{\ell,h_\ell(d)} - \frac{n}{m}\right)$$

と頻度推定を行う.

(例 4)$n = 4, D = \{$“男”,“女”$\}, m = 2, k = 2, \varepsilon = \infty, S = ((1,1,1),(1,1,2),(1,2,1),(1,2,2))$ とする. この時, $\tilde{x}^{(1)}, \tilde{x}^{(2)}, \tilde{x}^{(3)}, \tilde{x}^{(4)}$ は以下のように計算される.

$$\tilde{x}^{(1)} = 2 \cdot 1 \cdot 1 = 2$$

$$\tilde{x}^{(2)} = 2 \cdot 1 \cdot 1 = 2$$

$$\tilde{x}^{(3)} = 2 \cdot 1 \cdot 1 = 2$$

$$\tilde{x}^{(4)} = 2 \cdot 1 \cdot 1 = 2$$

この時, Skecth Matrix は以下のように表される.

$$SK = \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 2 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 2 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 2 \end{pmatrix} = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}$$

M に, 正規化を施すと,

$$MH_2^{-1} = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 4 & 0 \\ 4 & 0 \end{pmatrix},$$

$$\tilde{f}(\text{“男”}) = \left(\frac{2}{2-1}\right)\left(\frac{1}{2}\sum_{l=1}^{2} M_{\ell,h_\ell(\text{“男”})} - \frac{4}{2}\right) = 2$$

$$\tilde{f}(\text{“女”}) = \left(\frac{2}{2-1}\right)\left(\frac{1}{2}\sum_{\ell=1}^{2} M_{\ell,h_\ell(\text{“女”})} - \frac{4}{2}\right) = 2$$

このように頻度を推定する.

2.4 1-out-of-2 Oblivious Transfer

1-out-of-2 Oblivious Transfer [4] は, 受信者は送信者から送られた 2 つの情報のうち片方しか知ることができず, 送信者は受信者に送信した 2 つの情報のうちどの情報を得られたか知ることができないことを保証する 2 パーティの暗号プロトコルである. アルゴリズム 1 に 1-out-of-2 Oblivious Transfer の実装を示す.

Algorithm 1 1-out-of-2 Oblivious Transfer [4]

Require: message m_0, m_1

Sender generates RSA key pair private key d, public keys N, e

Sender sends public keys to Receiver

Sender has two random message x_0, x_1

1. Sender sends x_0, x_1 to Receiver

2. Receiver chooses $b \in \{0, 1\}$ and generates random k and computes $v = (x_b + k^e)\ mod\ N$ the encryption of k, blind with x_b. Receiver sends v to Sender.

3. Sender computes $k_0 = (v - x_0)^d\ mod\ N$, $k_1 = (v - x_1)^d$ and $m'_0 = (m_0 + k_0)\ mod\ N, m'_1 = (m_1 + k_1)\ mod\ N$ Sender send m'_0, m'_1.

4. Receiver computes $m_b = (m'_b - k)\ mod\ N$.

Ensure: m_b

2.5 ポイズニング攻撃

ポイズニング攻撃は, 悪意のあるユーザが意図的なデータをサーバに送信して推定結果を操作する不正行為である. 例えば, オンラインショッピングサービスの場合を考える. オンラインショッピングサービスの運営者はどの商品がよく売れているかを参考にして仕入れる商品を選択する. そのため, 商品を製造しているメーカは自社商品がよく売れていると偽装させることによって利益の向上を狙う動機がある. 不正なデータを送信することによって自社製品の売れ行きを操作する.

各不正ユーザは, 局所差分プライバシ方式を改ざんすることができ, 任意のデータをサーバに送信することができると想定する.

攻撃者は, システム上で n' 人の不正ユーザを操作することができるとする. 攻撃者が, 操作する r 個のアイテムをターゲットアイテムとし, その集合を $T = \{t_1, t_2, ..., t_r\}$ とする. サーバは n 人の真のユーザと n' 人の不正ユーザの出力から統計量を推定する.

n 人の真のユーザのアイテム t に対する頻度の推定値を $\hat{f}_t$, 不正ユーザを含めた $n+n'$ 人のアイテム t に対する頻度の推定値を $\tilde{f}_t$ とする. ポイズニング攻撃による頻度の推定値の変化量を Frequency Gain$\Delta\tilde{f}_t = \tilde{f}_t - \hat{f}_t$ とする.

[1] によるポイズニング攻撃には Random Perturb Attack(RPA), Random Item Attack(RIA), Maximal Gain Attack(MGA) がある. RPA は各不正ユーザが摂動されたデータ集合からランダムに一つ選びサーバに送信する攻撃である. RIA は各不正ユーザがターゲットアイテムの中からランダムに一つ選択し, そのデータを定められた方法で正しく摂動し, サーバに送信する. MGA は摂動されたデータを不正ユーザの意図したデータに置換してサーバに送信する.

3. 提 案 手 法

3.1 CMS, HCMS に対するポイズニング攻撃の評価

3.1.1 Random Perturb Attack

RPA は, 出力をランダムに選択する攻撃である. CMS では, 2^m の数だけ選択肢があり, 各選択肢を $\frac{1}{2^m}$ の確率で選び, サーバに送信する. 例えば, $m = 2$ の場合, 不正ユーザは $(1,1), (-1,1), (1,-1), (-1,-1)$ から一様ランダムに取得したデータをサーバに送信する. HCMS の場合では, 1 または-1 を $\frac{1}{2}$ の確率でサーバに送信する.

3.1.2 Random Item Attack

RIA は, 摂動対象を操作する攻撃である. 不正ユーザはランダムに $t \in T$ を選択する. また, 不正ユーザは t を CMS または HCMS を適用し, 得られた出力をサーバに送信する.

3.1.3 Maximal Gain Attack

MGA は, 出力を操作する攻撃である. アイテム t に対するポイズニング後の推定値を $\tilde{f}_t$, ポイズニング前の推定値を $\hat{f}_t$ とする. このとき, 不正ユーザは Frequency Gain(FG) の平均を

$$FG = \sum_{t \in T} \boldsymbol{E}[\tilde{f}_t - \hat{f}_t]$$

と定める. FG が最大になるように出力を生成する.

CMS に対する FG の期待値は次のように計算される.

$$\begin{aligned} FG &= \sum_{t \in T} \boldsymbol{E}[\hat{f}_t - \tilde{f}_t] \\ &= \sum_{t \in T} \left\{ \left(\frac{m}{m-1}\right) \left(\frac{1}{k} \sum_{\ell=1}^{k} \tilde{M}_{\ell,h_\ell(d)} - \frac{n}{m}\right) \right. \\ &\quad \left. - \left(\frac{m}{m-1}\right) \left(\frac{1}{k} \sum_{\ell=1}^{k} M_{\ell,h_\ell(d)} - \frac{n}{m}\right) \right\} \\ &= \frac{1}{k} \left(\frac{m}{m-1}\right) \sum_{t \in T} \boldsymbol{E} \left\{ \sum_{\ell=1}^{k} \left(\tilde{M}_{\ell,h_\ell(d)} - M_{\ell,h_\ell(d)} \right) \right\} \end{aligned} \tag{1}$$

このとき, $d \in D$ として, i 番目のユーザによる, Sketch Matrix の ℓ 行 $h_\ell(d)$ 列のエントリ $M_{\ell,h_i(d)}$ を $Y_\ell^{(i)}(d)$ とおく. また, i 番目の不正ユーザによる, Sketch Matrix の ℓ 行 $h_\ell(d)$ 列のエントリ $M_{\ell,h_i(d)}$ を $X_\ell^{(i)}(d)$ とおくと,

$$\begin{aligned} \tilde{M}_{\ell,h_\ell(d)} &= \sum_{i=1}^{n} Y_\ell^{(i)}(d) + \sum_{i=1}^{n'} X_\ell^{(i)}(d) \\ M_{\ell,h_\ell(d)} &= \sum_{i=1}^{n} Y_\ell^{(i)}(d) \end{aligned} \tag{2}$$

より, (2) を用いて (1) を変形すると,

$$FG = \frac{1}{k} \left(\frac{m}{m-1}\right) \sum_{t \in T} \left\{ \sum_{\ell=1}^{k} \sum_{i=1}^{n'} X_\ell^{(i)}(d) \right\} \tag{3}$$

これより, $X_\ell^{(i)}(d)$ を最大すればよい. つまり, 攻撃者は H から任意のハッシュ関数 h_j を選択し, $T = \{t_1, t_2, ..., t_r\}$ に対応するハッシュ関数 L の出力 $h_j(t_1), h_j(t_2), ..., h_j(t_r)$ を調べ, 得られた出力の位置を 1 に設定し, その他は -1 を設定して, $(\boldsymbol{v}, j)$ を送信する. 例えば, D = {“男”,“女”}, $m = 2, k = 2, H = \{h_j : D \to [2], j \in [2]\}$ のとき, 不正ユーザが男の集計結果を増加させるシナリオを考える. 攻撃者は任意のハッシュ関数 h_j を選択し, h_j(”男”) = 0 であれば, $(1, -1)$ とハッシュ関数番号 j を送信する.

一方, HCMS に対する MGA は複数の戦略が考えられる. 2.4 節で述べたように, あるアイテムの入力が他のアイテムの推定結果に影響を及ぼすことに起因している. ここでは最も簡単な戦略を述べる.

$d \in D$ として, i 番目のユーザによる, Sketch Matrix の ℓ 行 $h_\ell(d)$ 列のエントリ $M_{\ell,h_i(d)}$ を $Z_j^{(i)}(d)$ とおく. また, i 番目の不正ユーザによる, Sketch Matrix の ℓ 行 $h_\ell(d)$ 列のエントリ $M_{\ell,h_i(d)}$ を $\tilde{Z}_j^{(i)}(d)$ とおくと, (1) は次のように変形できる.

$$FG = \frac{1}{k} \left(\frac{m}{m-1}\right) \sum_{t \in T} \left\{ \sum_{l=1}^{k} \sum_{i=1}^{n'} \tilde{Z}_l^{(i)}(d) \right\} \tag{4}$$

この時,

$$\tilde{Z}_l^{(i)}(d) = k \cdot c_\varepsilon \cdot 1 \tag{5}$$

とすれば, FG が最大となる. つまり, 攻撃者は $w = 1, l = 0, j$ は任意 を送信すればよい.

3.2 OT-CMS, OT-HCMS

MGA は, 正規の摂動化プロセスの後, 不正ユーザが送信するデータを意図的なものに変更することによって実現される. つまり, 送信されたデータは摂動化が行われていないまま送信される.

そこで, Oblivious Transfer を用いて, ユーザに摂動化を強制させる OT-CMS と OT-HCMS を提案する. 本来, CMS は摂動化した m 次元ベクトル $\tilde{\boldsymbol{v}}$ とハッシュ関数の番号 j を送信するが, OT-CMS においては, ユーザは m 次元ベクトル $\boldsymbol{v}$ の摂動化を単独に行わず, 1-out-of-2 OT を用いてベクトルの要素をサーバの協力により選択する. 従って, 不正な摂動化が防止される. ハッシュ関数の番号は従来通り送信する.

例えば, ユーザが $\boldsymbol{v} = (1, -1)$ をサーバに送信する場合を考える. ユーザが 1 番目の要素 1 を送信するとき, ユーザは $1, -1$ のどちらも OT の送信候補とする. サーバはその内, 真のデータを確率 p, 偽データを確率 $q = 1 - p$ で取得する. このとき,

$$p = \frac{e^{\frac{\varepsilon}{2}}}{1 + e^{\frac{\varepsilon}{2}}}, \quad q = \frac{1}{1 + e^{\frac{\varepsilon}{2}}}$$

この試行を m 回繰り返す. この際, ユーザとサーバは 1-out-of-$1/p$ OT を用いて通信をしているため, サーバはどちらか片方の情報のみ得ることができ, ユーザはサーバがどちらの情報を取得したか知ることができない.

OT-HCMS では,

$$p = \frac{e^{\varepsilon}}{1 + e^{\varepsilon}}, \quad q = \frac{1}{1 + e^{\varepsilon}}$$

に変更し, 試行は 1 ビット分行えば十分である.

ここで、Oblivious Transfer 上で摂動を行うサーバはプロトコルは守るが, ユーザの入力値を知ろうとするセミオネストモデルを仮定する.

4. 実　　験

4.1 実験目的

アダマール行列を用いて削減する送信量と安全性を評価する. また, 提案手法の効果を評価する.

4.2 データセット

オンラインショッピングサービスの購入履歴の click stream データセット [7] を使用する. 図 2 にアイテムの購入頻度の分布を示す. 例えば, 商品 A2 は 3013 件ある. アイテム数は $|D| = 43$ である.

4.3 評価方法

実験で使用するパラメータを表 2 に示す. 真の分布と推定分

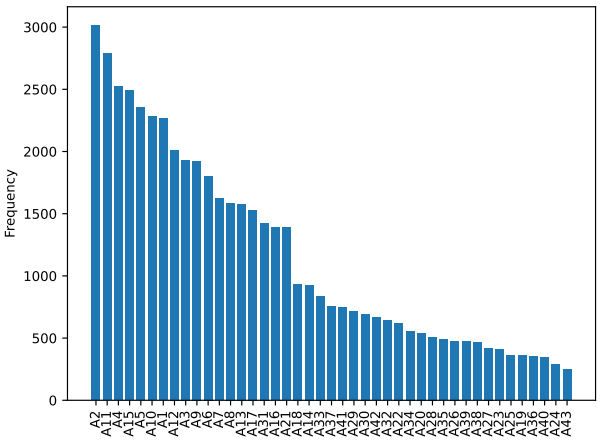

図 2　データセットのアイテム d の購入頻度 f_d の分布

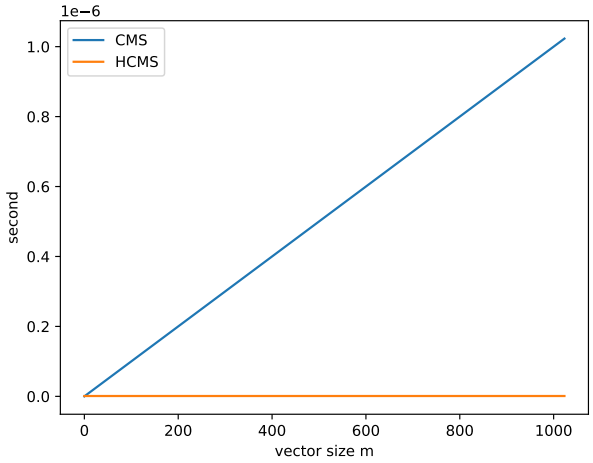

図 3　CMS と HCMS の送信時間

表 2　評価に用いるデフォルトパラメータ

パラメータ	
プライバシー予算 ε	1.0
ベクトル長 m	2^7
ハッシュ関数の数 k	2^{10}
不正ユーザの割合 β	0.01
ターゲットアイテムの数 r	1

布の平均二乗誤差 MSE を用いて有用性を評価する. FG を用いて安全性を評価する. 試行を 50 回繰り返しその平均値を評価値とする. ただし, OT-CMS, OT-HCMS の安全性評価では, 試行は 10 回行い, 標的アイテムは “$A18$”, “$A34$” について評価する. MGA ポイズニング攻撃を評価する.

また, OT-CMS と OT-HCMS の OT を導入したことによるコストを処理時間で評価する. 各ベクトル長で 50 回処理時間を計算し, その平均値を評価値とする.

4.4　実 験 結 果

有用性: 図 4, 図 5, 図 6, 表 3 にそれぞれプライバシー予算 ε, CMS の符号化ベクトル長 m, ハッシュ関数の数 k に対する CMS と HCMS の MSE を示す.

プライバシー予算については HCMS が CMS に比べ, 平均で 21.7% 推定誤差が大きい. 特に $\varepsilon = 1.0$ のとき, 47.3% だけ CMS の方が精度が良い. ベクトル長については全ての場合で CMS の方が HCMS よりも MSE が小さい. $m = 1024$ のとき, 52.8% だけ CMS の方が精度が良い. ハッシュ関数の数 k についても CMS, HCMS どちらも単調に MSE が減少している.

安全性: 図 7, 図 8, 図 9, 表 4, 表 5, 表 6 にそれぞれプライバシー予算 ε, 不正ユーザの割合 β, ターゲットアイテムの数 r に対する CMS と HCMS の FG を示す.

MGA については, ε, β, r の全ての条件で HCMS の方が CMS より FG が平均で 16.0% 小さい. RPA についても同様に, ε, β, r の全ての条件で HCMS の方が CMS より FG が小さい. 特に $\beta = 0.1$ の時, 最大で 40.35 小さい. 一方で, RIA では, ε, β, r における FG は CMS と HCMS の差は小さい.

OT-CMS と OT-HCMS の安全性: 図 10 と図 11, 表 7 に不正ユーザの割合に対する OT-CMS と OT-HCMS の FG を示す.

OT-CMS, OT-HCMS のどちらの場合でも, CMS と HCMS よりも FG が小さくなった. 特に, $\beta = 0.10$ のとき, OT-CMS は CMS より 267.83 安全であり, OT-HCMS は HCMS にくらべ 211.83 安全であった.

OT-CMS と OT-HCMS の計算コスト: 表 8 にベクトル長に対する OT-CMS と OT-HCMS の処理時間を示す.

どのベクトル長を比較しても OT-HCMS の方が処理時間が短い. また, ベクトル長が大きくなるほど処理時間の差は大きくなり, $m = 128$ のとき最大 9.10 秒, OT-HCMS の方が処理時間が短い.

4.5　考　　察

表 3 より, HCMS は CMS に比べて誤差が大きく, 有用性が低い. これは, ユーザが行う HCMS の処理に m 次元ベクトルから 1 ビットをランダムにサンプリングすることが原因だと考えられる. HCMS はサンプリングが一様ランダムになるとき CMS と同じ性能となることが Apple により示されている [3]. そのため, 実験的に行うとサンプリングにある程度の偏りが生じ, HCMS は CMS に比べ有用性が低下する.

MGA に対して, CMS に比べ HCMS の方が安全であった. その原因はノイズ除去の方法によって引き起こされている.

CMS はユーザから送信されたデータのノイズを除去するときに, $\frac{e^{\frac{\varepsilon}{2}}+1}{e^{\frac{\varepsilon}{2}}-1}$ と積をとる. 一方, HCMS は $\frac{e^{\varepsilon}+1}{e^{\varepsilon}-1}$ と積をとる [3]. ε が同じ値であれば, $\frac{e^{\frac{\varepsilon}{2}}+1}{e^{\frac{\varepsilon}{2}}-1}$ の方が大きな値をとる. そのため, 不正ユーザから MGA を用いて攻撃された際に, 攻撃の効果がより増幅される. RIA についてはほぼ同様の安全性を示した. HCMS の方が CMS よりも安全と言える.

OT-CMS と OT-HCMS では, 1-out-of-1/p OT を適用し, 摂動を強制的に行うため不正ユーザは意図的なデータを送信することができない. そのため, FG が小さくなり, 安全性が向上した. また, アダマール行列を適用した OT-HCMS はエンコードされたベクトルの長さにかかわらず, 1bit の送信のみなので結果と

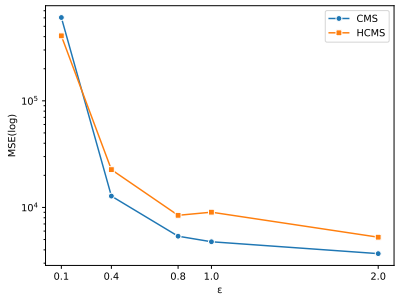

図 4　プライバシー予算 ε についての推定誤差

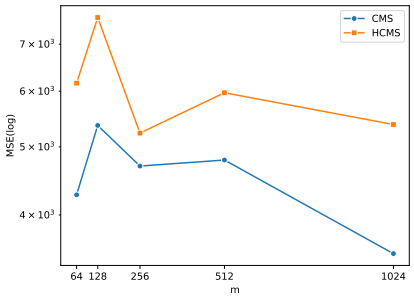

図 5　ベクトル長 m についての推定誤差

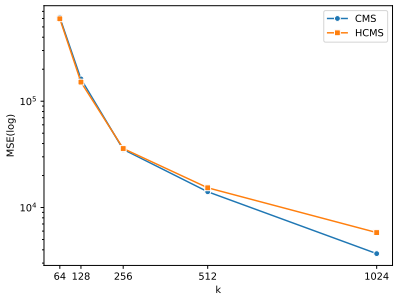

図 6　ハッシュ関数の数 k についての推定誤差

表 3　各パラメータによる MSE($\times 10^3$) の変化

ε	CMS	HCMS
0.1	604.66	407.07
0.4	12.78	22.63
0.8	5.37	8.43
1.0	4.76	9.03
2.0	3.69	5.26

m	CMS	HCMS
64	4.27	6.16
128	5.36	7.64
256	4.70	5.23
512	4.79	5.97
1024	3.52	5.38

k	CMS	HCMS
64	613.03	596.29
128	162.13	151.32
256	35.22	35.89
512	14.07	15.34
1024	3.68	5.81

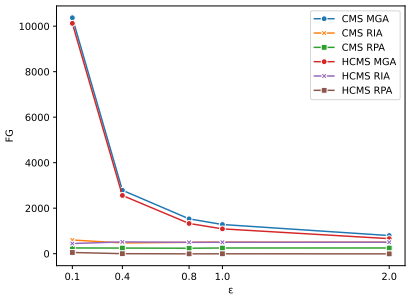

図 7　プライバシー予算 ε についての安全性 FG

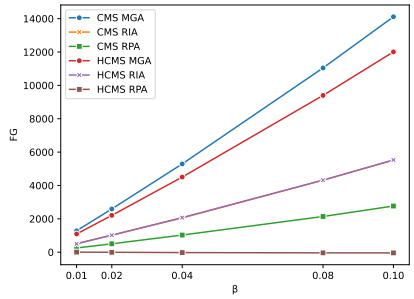

図 8　不正ユーザの割合 β についての安全性 FG

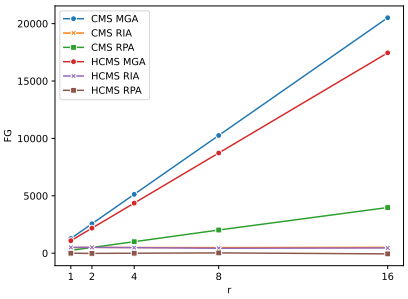

図 9　ターゲットアイテムの数 r についての安全性 FG

表 4　プライバシー予算 ε による FG($\times 10^2$)

ε	CMS RPA	HCMS RPA	CMS RIA	HCMS RIA	CMS MGA	HCMS MGA
0.1	2.51	0.53	6.01	4.46	103.70	101.24
0.4	2.47	0	4.27	5.17	27.87	25.58
0.8	2.40	−0.08	4.99	4.98	15.31	13.28
1.0	2.50	−0.05	5.08	5.00	12.82	10.91
2.0	2.50	−0.08	5.03	5.04	7.96	6.60

表 5　不正ユーザの割合 β による FG($\times 10^2$)

β	CMS RPA	HCMS RPA	CMS RIA	HCMS RIA	CMS MGA	HCMS MGA
0.01	2.51	0.04	5.18	5.01	12.82	10.91
0.02	5.03	−0.05	10.19	10.21	25.92	22.06
0.04	10.28	−0.24	20.52	20.73	52.91	45.03
0.08	21.39	−0.40	43.15	43.18	110.44	93.99
0.10	27.67	−0.43	55.19	55.30	141.11	120.09

して 1-out-of-1/p OT の暗号化と復号の必要回数が 1 回のみとなり送信時間が削減された.

4.6　今後の課題

OT-CMS の摂動をする際にベクトル長 m 回 1-out-of-1/p OT を行ったが, m が 2 のべき乗であるとき, 送信回数を減らすことが可能である. また, 提案方式はサーバがセミオネストモデルであることを仮定しているため, 本来の局所差分プライバシ方式のサーバを信頼しないという考え方に反している. サーバを完全に信頼しないままポイズニング攻撃に対してロバストにすることを今後の課題とする.

表 6　ターゲットアイテムの数 r による FG($\times 10^2$)

r	CMS RPA	HCMS RPA	CMS RIA	HCMS RIA	CMS MGA	HCMS MGA
1	2.50	−0.01	4.93	5.04	12.82	10.91
2	4.99	−0.17	5.09	5.07	25.64	21.82
4	10.02	−0.02	4.95	4.84	51.28	43.64
8	20.20	0.25	4.89	4.29	102.55	87.27
16	39.77	−0.58	5.10	4.56	205.11	174.54

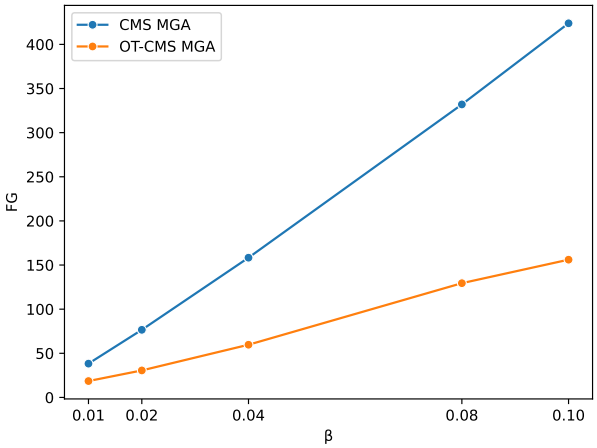

図 10　不正ユーザの割合 β についての OT-CMS の安全性 FG

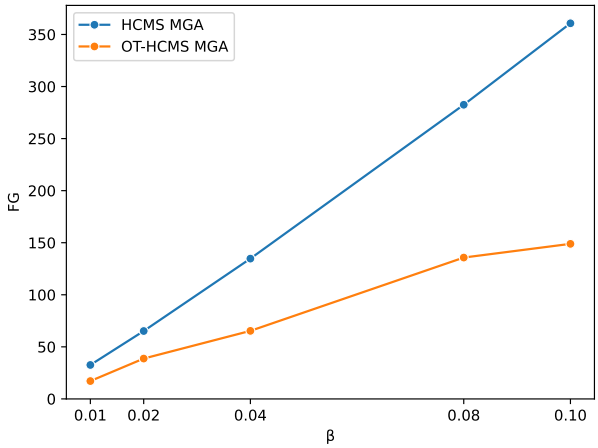

図 11　不正ユーザの割合 β についての OT-CHMS の安全性 FG

表 7　OT-CMS と OT-HCMS の
不正ユーザの割合 β による FG

β	CMS	OT-CMS	HCMS	OT-HCMS
0.01	38.30	18.56	32.60	17.15
0.02	76.61	30.59	65.19	38.75
0.04	158.33	59.70	134.73	65.34
0.08	331.97	129.44	282.51	135.75
0.10	423.90	156.07	360.74	148.91

表 8　OT-CMS と OT-HCMS のベクトル長 m に
よる処理時間 (秒)

m	OT-CMS	OT-HCMS
2^3	0.57	0.07
2^4	1.15	0.07
2^5	2.30	0.07
2^6	4.60	0.07
2^7	9.17	0.07

5. おわりに

本稿では, 局所差分プライバシプロトコル Count Mean Sketch(CMS), Hadamard Count Mean Sketch(HCMS) を用いて, ポイズニング攻撃に対する安全性を向上させた方式を提案した. ポイズニング攻撃に対するロバスト性を向上させるために, Oblivious Transfer を適用する手法を提案した. 実験に基づき, 提案方式では, 安全性が従来の CMS より 267.83 だけ高いことが示された. 提案方式の OT の処理時間は従来方式より最大 131 倍効率が向上した.

文　献

[1] X. Cao, J. Jia, N. Z. Gong, “Data poisoning attacks to local differential privacy protocols”, USENIX Security Symposium, pp. 947-964, 2021.

[2] Hikaru Horigome, Hiroaki Kikuchi and Chia-Mu, Yu, “Local Differential Privacy Protocol for Key-Value Data Robust against Poisoning Attacks”, Modeling Decisions for Artificial Intelligence, pp.241-252, Volume 13890, 2023.

[3] Differential Privacy Team, Leaning with Privacy at Scale(https://machinelearning.apple.com/research/learning-with-privacy-at-scale,).

[4] Even, Shimo Goldreich, Oded Lempel, Abraham, A Randomized Protocol for Signing Contracts, Communications of the ACM, pp.205-210, 1982.

[5] Gadotti, Andrea Houssiau, Florimond Annamalai, Meenatchi Montjoye, Yves-Alexandre. (2023). Pool Inference Attacks on Local Differential Privacy: Quantifying the Privacy Guarantees of Apple’s Count Mean Sketch in Practice, 31st USENIX Security Symposium (USENIX Security 22), 2022.

[6] J. C. Duchi, M. I. Jordan and M. J. Wainwright, ”Local Privacy and Statistical Minimax Rates,” 2013 IEEE 54th Annual Symposium on Foundations of Computer Science, Berkeley, CA, USA, pp. 429-438, 2013

[7] clickstream data for online shopping, (2019), UCI Machine Learning Repository.

表 9　従来方式と提案方式の比較

	CMS	HCMS	OT-CMS	OT-HCMS
精度	○	×	○	×
安全性 FG	×	△	○	○
通信コスト	△	○	×	○