情報を保護しながら活用する秘匿計算の研究—処理の効率化と開発の容易化

代表研究者　　吉浦□裕□　　電気通信大学　□情報理工学研究科　教授□□□□

1 はじめに

個人情報の活用による高度な電気通信サービスや企業情報を遠隔サーバに預けるクラウドサービスが発展する一方，これらの情報の保護が問題になっている．秘匿計算は，情報を秘匿したままの状態で情報処理を行う技術である．たとえば，1 万人の個人情報を秘匿したまま，各種の統計処理やデータマイニング，個人毎の異なるサービスを可能にする．暗号理論によって秘匿性を保証する一方，通常計算（秘匿しない計算）と同等範囲・精度の情報処理が原理的に可能であるため，情報の活用と保護を両立する切り札技術として期待されている．しかし，秘匿計算は処理効率が低く，特殊なプログラミングを必要とするため，実用化に至っていない．そこで，本研究では，秘匿計算の効率化とプログラム開発の容易化を目的とした．

2 研究方針

2-1 対象とする秘匿計算と研究課題

秘匿計算には，（1）情報を暗号化した状態で和を実行可能な加法準同型暗号および積を実行可能な乗法準同型暗号を利用する手法，（2）情報を暗号化した状態で和と積の両者を実行可能な完全準同型暗号を利用する方法，（3）情報を複数のサーバに分散した状態で和と積を実行可能なマルチパーティ計算を利用する方法がある．（1）の加法および乗法準同型暗号は計算可能な情報処理の範囲が通常計算より狭い，（2）の完全準同型暗号は基礎研究の段階にある．そこで，計算可能な範囲が通常計算と同じで実用に近い段階にある（3）のマルチパーティ計算を取り上げた．

マルチパーティ計算（以下 MPC と略す）では，情報を複数のサーバに分散することで秘匿し，サーバ間の通信処理により秘匿した状態で情報処理を行う．MPC の処理時間の大部分はサーバ間の通信時間であるため，処理の効率化では，通信コストの低減が課題となる．通信コストは，通信量（通信の対処となるデータのビット数）と，通信ラウンド数（最大限の並列化を行った場合の通信の回数）に依存するが，大規模計算の場合には通信量が支配的である．そのため，通信コスト特に通信量を低減することが重要である．

また，MPC では，和演算は通信を必要としない（ゼロコストである）が，積演算は全サーバ間の通信を必要とする，大小比較（a>b の判定）や等号判定（a=b の判定）は積演算の数百倍の通信を必要とする[1][2]．また，通常計算で一般的に行われる条件分岐処理は情報漏洩を引き起こす可能性がある．たとえば，a>b の結果によって異なる処理に分岐する場合，どちらの処理に分岐したかで，a と b の大小関係が漏洩する．このように MPC は通常計算と全く異なる性質を有するため，MPC を用いたプログラミングには特殊な技能を要するという問題がある．そこで，できるだけ特殊な技能を要しない MPC プログラミングを可能にすることが重要である．

2-2 課題を解決する方針

MPC は，（1）和，積，乱数生成，秘匿解除の 4 つの演算による最小要素層，（2）最小要素の組合せによる除算や大小比較等の基本部品層，（3）最小要素と基本部品の組合せによるプログラム層の 3 層から構成される．従来の効率化の研究は，主に基本部品層を対象とし，基本部品を構成する際の最小要素の組合せを最適化していたが，通常計算（秘匿しない計算）に比べて 1/1000 程度の効率に留まっていた[1][2][3][4]．そこで，本研究では，最小要素層と基本部品層の連携による効率化およびプログラム層の効率化を検討する．

また，一般のプログラマーによる効率的な MPC プログラムの開発を可能にするために，MPC の効率的なプログラミングを定型化してプログラマーに示すことができるようにすること，および，非効率な MPC プログラムを効率的な MPC プログラムに変換することを検討する．

3 処理の効率化

3-1 最小要素層と基本部品層の連携による効率化 [5][6]

全ての秘匿計算は，和，積，乱数生成，秘匿解除の 4 種類の最小要素の組合せにより構成される．そのう

ち和は通信を必要としないが，積，乱数生成，秘匿解除は通信を必要とし，その通信量は，これらの演算の対象となるデータのサイズ（ビット数）に比例する．たとえば，たとえば，a×b の MPC において，a および b が 64 ビットの整数である場合は 32 ビットの整数である場合に比べて，2 倍の通信量を必要とする．そのため，データのサイズを削減することで効率化が可能となる．

しかし，データサイズはそのデータの表現する対象によって定まるため，単純に削減することはできない．たとえば，上記の a というデータが機材の価格，b が会社で購入する機材の台数を表現する場合を想定する．機材の価格として最大 100 万円までの可能性がある場合には a は 100 万までの整数を表する必要があるため 20 ビットのサイズが必要となる．また，機材の代数として最大 25 台までの可能性がある場合には b は 25 までの整数を表現する必要があり，5 ビットのサイズが必要となる．それ以下のサイズの場合，必要な計算ができない可能性がある．また，実際に購入した機材が 20 万円，代数が 5 台であったとしよう．その場合，a は 18 ビット，b は 3 ビットで表現できる．しかし，そのように実際の値に合わせて a，b のサイズを小さくすると，そのプログラムから，購入した機材が 2^{18} ＝262,144 円以下であり，購入台数は 8 台以下であるという情報が漏洩する．したがって，想定される最大の値までを表現可能とし，実際の値を秘匿しながら，データサイズを削減する必要がある．

秘匿計算の多くの基本部品の入出力は整数であり，上述した理由から，そのデータサイズは削減できない．たとえば，大小比較（a>b）において，a と b が二つの機材の価格である場合，上述した理由から，a，b のサイズは削減できない．しかし，基本部品の内部処理の多くは，整数（たとえば 100）をビット列（1100100）に分解した状態で，ビット間の演算になっている．ビット間の演算は 1 桁の値しか取らない．そこで，プログラムを解析し，そのなかのビット間の演算の部分を小さいサイズで処理する手法を提案した．

ここでは基本部品のうち大小比較を例として，提案のサイズ削減方式を説明する．秘匿された二つの整数 a, b の大小を直接判定することは困難であり，a と b を各々ビット列に分解し，ビット列間の比較によって大小を判定する．しかし，整数をビットに分解する処理も秘匿した状態で直接実行することはできない．そこで，大小比較は下記の手順を取る[1]．①秘匿された整数 a, b に各々秘密の乱数 r1, r2 を加算，②a+r1, a+r2 の秘匿を解除し公開，③公開状態で a+r1, a+r2 をビット列 Ba, Bb に分解，④二つのビット列 Ba, Bb を秘匿した状態で乱数 r1, r2 の影響を除去し，a, b のビット列 Ba', Bb'を得る，⑤Ba'と Bb'を秘匿状態で比較し，結果を出力する．大小比較以外の基本部品についても分析し，等号判定（a=b），区間判定(a<b<c)も同様の処理を内包していることを見出した．また，詳細は省略するが，法変換（a の値を維持しながら a のビット数を変更）[7]も同様である．さらに，ビット分解（a をビット列に分解）[3]も，上記において手順⑤を省略した形と考えれば同様の処理になる．これらの基本部品は図 1 の共通パターンを内包している．

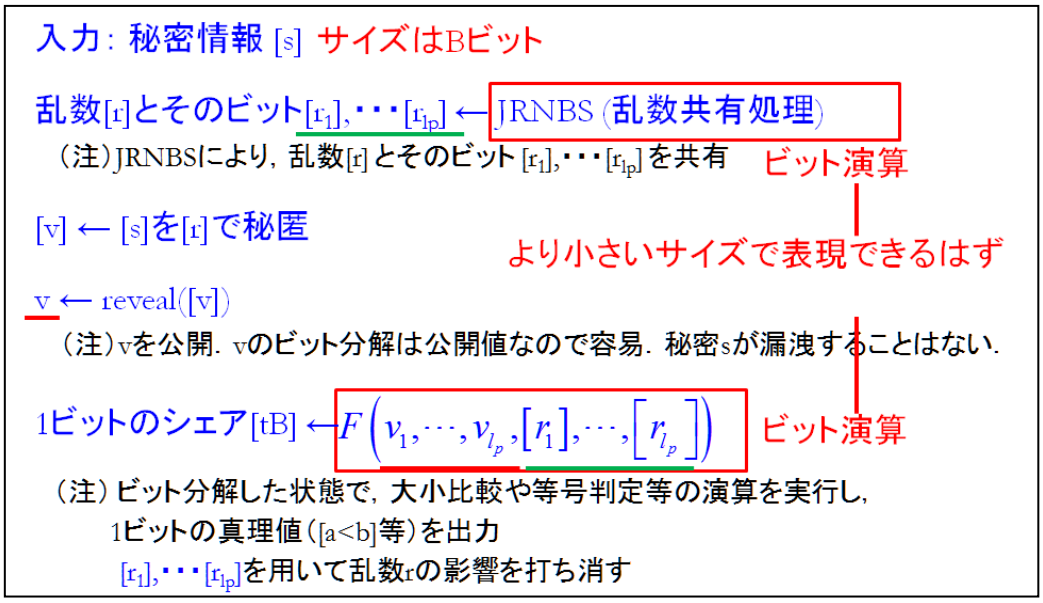
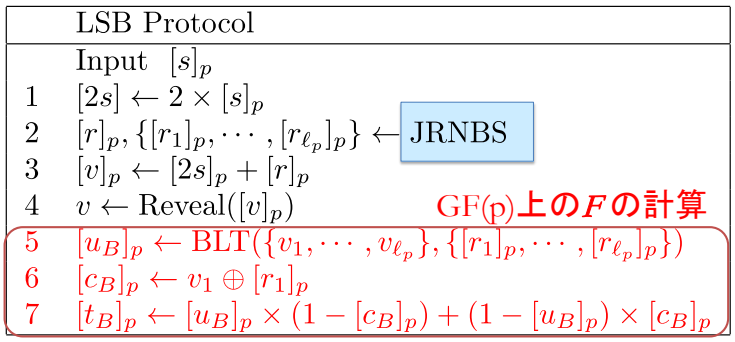


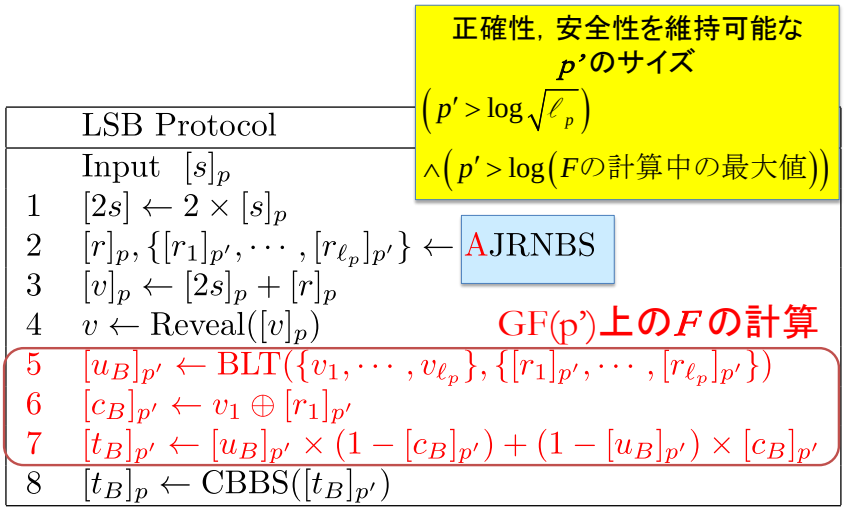
図 1　基本部品の共通パターン

図 1 のパターンにおいて，関数 F は各々の基本部品の機能に依存し，基本部品が大小比較の場合はビット列の間の大小比較関数，等号判定の場合はビット列間の等号判定である．このとき，内部処理における必要なデータサイズは，JRNBS に必要なデータサイズと F に必要なデータサイズの大きい方となる．入力情報のサイズが B の場合，JRNBS のデータサイズは $\log\sqrt{B}$ であり，F のデータサイズは基本部品によって異なるが多くの場合は $\log\sqrt{B}$ または $\log B$ であるため，内部処理のデータサイズは $\log\sqrt{B}$ または $\log B$ となる．たとえば B が 20 ビットの場合，$\log\sqrt{B}$ は 3 ビット，$\log B$ は 5 ビットとなり，入力データに比べて大幅なサイズ削減が可能である．

大小比較の内部処理（LSB protocol）におけるデータサイズ削減前のアルゴリズム（従来手法）と削減後のアルゴリズム（提案手法）を図 2 に示す．



(a) 従来手法



(b) 提案手法

図 2　大小比較の内部処理における従来手法と提案手法の比較

B＝32 ビットおよび 64 ビットにおける各基本部品の通信量の削減率（%）を表 1 にまとめる．なお，削減率 74%とは，提案手法により通信量が 74%減少する（従来の 26%になる）ことを意味する．

表 1　提案手法による基本部品の通信量の削減効果

入力データのサイズ（ビット数）	大小比較	等号判定	区間判定	法変換	ビット分解
32	74	60	76	82	70
64	69	72	81	79	79

3-2 プログラム層の効率化 [5]

(1)通信量のオーダー差を利用した効率化

N 個のデータに対する O(N)以上の処理を行う場合に，前処理として，個々のデータを当該処理の効率的な実行に適した形式に変換した後，当該処理を行う方式である．個々のデータに対する変換処理は O(N)であり，O(N)以上の当該処理を効率化できるので，全体として効率を向上させることができる．提案手法の概要を図 3 に示す．

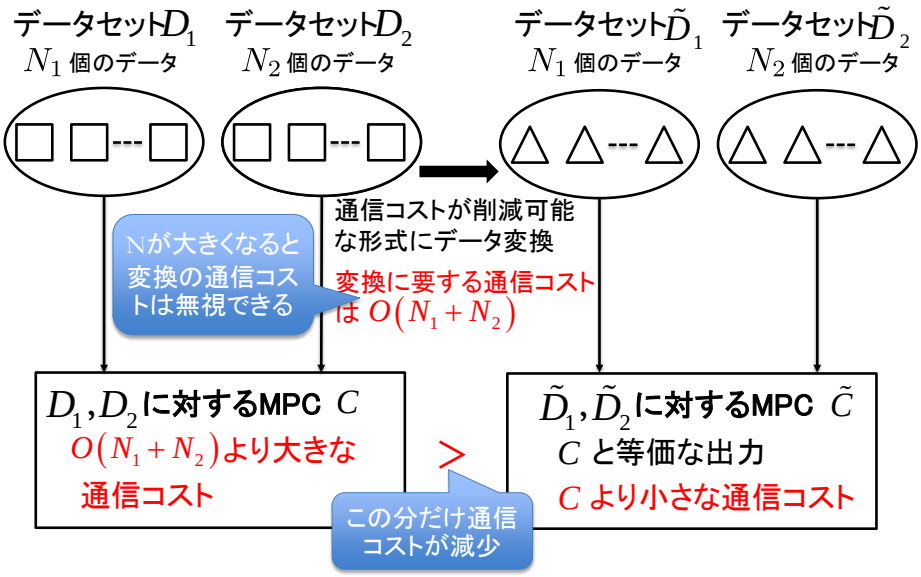


図 3　通信量のオーダー差に着目した通信量の削減手法

提案手法の考え方を，整列（N 個の数を昇順あるいは降順に並べる処理）を例として説明する．整列の場合は，データセットが 1 つ(図 3 の D_1 のみ)である．整列は極めて多用される処理である．また，NlogN 回の大小比較を必要とするため，その MPC による実行は非効率であり，効率化には大きな意義がある．上述したように，大小比較の内部処理はビットに分解した状態で行われる．そこで，大小比較を行う前に，その入力をビットに分解しておけば，大小比較を効率化できるはずである．さらに，ビットに分解する際にデータサイズを削減できるので，二重の効率化が可能となる．たとえば，a=100 と b=103 の大小を比較する場合，a=1100100, b=1100111 にビット分解した後に比較する．ビット分解処理は大小比較以上に非効率であるが，ビット分解の実行回数は N 回であり，大小比較の回数 NlogN より小さいので，N が大きくなるに従って，ビット分解の通信量は無視できるようになる．

同様の手法で，2 つの文字列から最長の共通部分を見出す処理および，2 つの信号列から最長の類似部分を見出す処理も効率化することができる．これらの 3 つの処理における通信量の削減率を表 2 にまとめる．

表 2　提案手法による代表的なプログラムの通信量の削減効果

入力データの個数 N	整列（N データ）	最長共通文字列の検出（N 文字と N 文字）	最長類似信号の検出（N 信号と N 信号）
100	98.3	91.0	98.8
1000	98.7	96.8	98.7

(2)アクセスパターンの効率的な秘匿 [8][9][10]

MPC では，データを暗号化した状態で計算を行うため，データの値が直接漏洩することはない．しかし，データにアクセスした回数やデータ間のアクセス順序に関する情報は漏洩する．これらのアクセスパターンからデータの値を推定される可能性が懸念されている．たとえば，薬のデータベースにおいて，ある病気が流行しているときに，データベースのあるデータが多数アクセスされたならば，そのデータが当該病気

の薬に関すると推定される．この推定を防止する方法としては，無関係のデータを含む全てのデータに同様にアクセスすることが考えられる．たとえば，全ての薬のデータを読み出し，アクセスしたい薬のデータの場合はそのまま，それ以外のデータの場合は空にして出力することが考えられる．しかし，このような全検索処理は，データ数 N に対して O(N)の通信量となるため非効率である．

これに対し，暗号学の別の分野で研究が進んでいる Oblivious Random Access Machine (ORAM)という手法[11][12]に着目した．ORAM はデータにアクセスするたびに，そのデータの配置場所を変更することでアクセスパターン（アクセス回数，アクセス順序）を秘匿する手法である．MPC の中に ORAM を取り込むことで，データの値だけでなく，データへのアクセスパターンを秘匿しながら計算を行うことを可能になることが，最近明らかになりつつある[13]．この ORAM と秘匿計算の組合せを用いることで，データベースから値を集計する処理を安全かつ効率的に実行する手法を提案した．提案手法は，データの値だけでなく，データへのアクセスパターンを秘匿し，通信量は $O((\log N)^3)$である．

4　プログラミングの容易化

上記 3.1 で述べたデータのビット数の削減は，大小比較や等号判定などの基本部品における図 1 のパターンの部分に適用可能であり，削減後のデータサイズも機械的に決定できる．そのため，このパターンをプログラム中から検出することにより，プログラマーにビット数削減の定型手順を示すことが可能である．また，パターン照合と変換処理を実装すれば，プログラムを自動変換することも可能である．また，3.2 の計算量のオーダー差に基づく効率化およびアクセスパターンの効率的な秘匿も，プログラム中の一定のパターンに基づいて効率化するため，プログラマーへのガイダンスや自動変換が可能である．これらのガイダンスや自動変換の実装・評価は今後の課題として取り組みたい．

5　まとめ

本研究では，秘密分散とマルチパーティ計算を用いる秘匿計算を取り上げ，マルチパーティ計算の処理の効率化について検討し，大小比較や等号判定などの基本部品において内部処理のデータサイズを削減することによる効率化，前処理と本処理のオーダーの差を利用する効率化，Oblivious RAM の利用によるアクセスパターンの効率的な秘匿手法を提案し，その効果を評価，確認した．これらの効率向上手法は，定型パターンに沿って機械的に実施できることから，マルチパーティ計算に習熟していない一般のプログラマーでも実施可能であり，また，プログラムの自動変換による人手を必要としない実施も可能である．

【参考文献】

[1] Takashi Nishide and Kazuo Ohta: Multiparty Computation for Interval, Equality, and Comparison without Bit-Decomposition Protocol. Proc. IPKC 2007, LNCS, vol.4450, pp. 343–360, Springer (2007)

[2] Octavian Catrina, Sebastiaan de Hoogh: Improved Primitives for Secure Multiparty Integer Computation. Proc. SCN'10, LNCS, vol.6280, pp.182-199, Springer (2010)

[3] Tomas Toft: Constant-Rounds, Almost-Linear Bit-Decomposition of Secret Shared Values, Prod. CT-RSA 2009. LNCS, vol. 5473, pp. 357-371. Springer (2009)

[4] SecureSCM: Security Analysis. Technical Report D9.2, (2009), http://www.securescm.org

[5] 加藤遼:法変換を用いたマルチパーティ計算の通信コストの削減，電気通信大学博士論文（2015）

[6] 加藤遼，西出隆志，吉浦裕:部分的に小さい法を用いたマルチパーティ計算のビット演算効率化，情報処理学会論文誌, Vol.55, No.9, pp.1971-1991 (2014)

[7] 加藤遼，桐淵直人，西脇雄高，吉浦裕: 定数ラウンド(k,n) 秘匿モジュロ変換プロトコルの提案，コンピュータセキュリティシンポジウム 2011, pp.427-431 (2011)

[8] 西田直央，和田紘帆，加藤遼，吉浦 裕:マルチパーティ計算を用いた Oblivious RAM の利便性及び安全性の向上(1) — 構想と方式概要 —，コンピュータセキュリティシンポジウム 2014, pp.527-534 (2014)

[9] 和田紘帆，西田直央，加藤遼，吉浦 裕:マルチパーティ計算を用いた Oblivious RAM の利便性及び安全性の向上(2) — プロトコルの設計と評価 —，コンピュータセキュリティシンポジウム 2014，pp.535-54 (2014)

[10] 奈良成泰, 天田拓磨, 西出隆志, 土井洋, 吉浦裕: 秘密計算を用いた時系列情報の安全な集計方法, コンピュータセキュリティシンポジウム 2016 (2016) 発表予定
[11] Oded Goldreich.: Towards a Theory of Software Protection and Simulation by Oblivious RAMs, Proc. STOC'87, pp.182-194 (1987).
[12] Emil Stefanov, Marten van Dijk, Elaine Shi, E. , et al.: Path ORAM: An Extremely Simple Oblivious RAM Protocol, Proc. CCS'13, pp.299-310 (2013).
[13] Marcel Keller, Peter Scholl: Efficient, Oblivious Data Structures for MPC, Proc. ASIACRYPT 2014, LNCS 8874, pp.506-525, Springer (2014)

〈発 表 資 料〉

題　　名	掲載誌・学会名等	発表年月
部分的に小さい法を用いたマルチパーティ計算のビット演算効率化	情報処理学会論文誌, Vol.55, No.9, pp.1971-1991	2014. 9
マルチパーティ計算を用いた Oblivious RAM の利便性及び安全性の向上(1) ― 構想と方式概要 ―	コンピュータセキュリティシンポジウム 2014 論文集, pp.527-534	2014.10
マルチパーティ計算を用いた Oblivious RAM の利便性及び安全性の向上(2) ― プロトコルの設計と評価 ―	コンピュータセキュリティシンポジウム 2014 論文集, pp.535-542	2014.10
法変換を用いたマルチパーティ計算の通信コストの削減	電気通信大学博士論文	2015. 3
秘密計算を用いた時系列情報の安全な集計方法	コンピュータセキュリティシンポジウム 2016 論文集（掲載確定）	2016.10