小さい法を用いたセキュアマルチパーティ計算のビット演算の効率化

加藤 遼† 吉浦 裕†

† 電気通信大学
182-0021 東京都調布市調布ケ丘１－５－１

あらまし セキュアマルチパーティ計算（SMC）の一つの問題点は参加者間で膨大な通信を必要とする点である．SMC の通信量は, シェアのビット数 ℓ, つまり秘密分散の法 p の対数に比例するので, 小さな法を用いることで通信量を低減する手法を提案する．大小比較や等号判定など多くのプロトコルは, ビットの状態の演算 (ビット演算) を多用している点に着目し, これらのビット演算の共通パターンを見出す．そして, プロトコルの入出力データの法 p に対して, このビット演算パターンを $\log p$ の法で実行し, 通信量を $\frac{\log \ell}{\ell}$ に削減する．この手法により, (2,3) 閾値法で法 p を 64 ビットと想定すると, 大小比較, 等号判定の通信量を各々75 パーセント, 70 パーセント削減できる．

Improving Efficiency of Secure Multi-Party Computation

Ryo Kato† Hiroshi Yoshiura†

†The University of Electro-Communications, Ltd.
1-5-1, Choufugaoka, Choufu, Tokyo 182-0021, JAPAN

Abstract It becomes more and more important to balance information usability and confidentiality. Secure multi-party computation based on secret sharing is expected to meet this challenge. It has serious problems, however, that it requires large amount of communications. In this paper, we describe a method to reduce the communication of secure multi-party computation based on secret sharing.

1 まえがき

個人情報や機密情報を保護しながら活用することが重要になっており, 情報を秘匿したままその関数値を計算する秘匿計算が期待されている．本論文では, 秘匿計算の一方法として, Shamir の (k, n) 閾値秘密分散法に基づくセキュアマルチパーティ計算 (SMC) を取り上げる．Shamir の (k, n) 閾値秘密分散法は, Z_p の元である秘密情報 s を, 分散情報（シェア）に分割し, n 人の参加者に配布する．k 個以上のシェアを集めないと元の秘密情報を復元できないという意味で, 情報の秘匿が可能である．SMC では, 秘密情報 a, b を分散した状態で, a のシェアと b のシェアから, a と b の和積の計算や大小比較を行うなどの様々なプロトコルが知られており, 幅広い応用が検討されている．しかし，SMC には大量の通信を必要とするという問題点がある．積プロトコルでは，n 人の参加者が相互に通信するため $n(n-1)$ 回の通信が必要となる．等号判定や大小比較プロトコルの処理コストは積プロトコルに換算した通信量と並列化を考慮したラウンド数で評価される．例えば，文献 [5] の等号判定プロトコルでは, 32 ビットの秘密情報 a と b に対する等号判定の場合，2592 回もの積プロトコルが必要になり，参加者が 5 人のときの通信量は 6480 バイトにも及ぶ．

SMC の通信量は, 素数 p のサイズ (ビット数)

ℓ に比例する．また, ℓ はプロトコルの入出力となる秘密情報を正しく表現するために, 秘密情報のサイズより大きく設定される．しかし, プロトコルの入出力が ℓ ビットであっても, プロトコルの処理途中に扱われる秘密情報 (中間秘密情報) は ℓ より小さい場合がある．そこで, 本論文では, プロトコルの処理途中で, ℓ より小さなビット数を用いる．つまり初期の秘密分散時に用いた素数 p より小さな素数 p 'を用いて秘密分散をやり直すことで通信量を削減する手法を提案する．しかしながら, この手法の実現には以下の問題が存在する．

- 中間秘密情報は秘匿されているので, 一般には, そのサイズは未知であり, p' を定めることが出来ない．
- 秘密分散時には法が公開される．そのため仮に p' を決めることができたとしても, その公開は中間秘密情報のサイズを公開することになり, 秘密漏洩につながる．
- 素数 p による秘密分散を p' による秘密分散に変換する処理に通信量が必要となり, 通信量の低減につながるとは限らない．

本研究では, 等号判定や大小比較, Bit Decomposition などの多くの基本的なプロトコルはビットの状態の演算 (ビット演算) を内部で多用している点に着目する．そしてこれらのビット演算を $\lceil \log p \rceil$ を超える小さな法 p' で実行する手法を提案する．提案手法が上記の問題を解決し, 提案手法が元のプロトコルの機能と安全性を保証していることを示す．

2 表記規則

- p : ℓ_p ビットで表現される素数．なお, $\ell_p = \lceil \log p \rceil$.
- s_B : $s_B \in \{0, 1\}$.
- $[s]_p$: s を素数 p で分散したシェア.
- s_i : s の i ビット目 ($s_i \in \{0, 1\}$).
- $[a]_p + [b]_p$: 秘密情報 $a, b \in \mathrm{Z}_p$ の和のシェア $[a + b \pmod p]_p$ を得る．
- $[a]_p \times [b]_p$: 秘密情報 $a, b \in \mathrm{Z}_p$ の積のシェア $[a \times b \pmod p]_p$ を得る．

3 先行研究

3.1 Shamir (k, n) 閾値秘密分散法 [6]

秘密情報 $s \in \mathrm{Z}_p$ を定数とし, $r_i \in \mathrm{Z}_p$ を乱数とする多項式

$$f(x) = s + r_1 x + r_2 x^2 + \cdots + r_{k-1} x^{k-1} \pmod p \quad (1)$$

をつくる ($1 \le i \le k-1$)．n 人の参加者 P_d に, シェア $f(d)$ を配布する ($1 \le d \le n$)．シェア $f(d)$ を k 個以上集めると $k-1$ 次の多項式が一意に定めるため，秘密情報 s が求まる．

3.2 セキュアマルチパーティ計算 (SMC)

秘密情報を秘匿したまま, 秘密分散法で分散されたシェアに対して, 関数を適用する手法である．和プロトコル, 積プロトコル, 乱数生成プロトコルが最も基礎的な SMC である．また, それらの基礎的な SMC を組み合わせて構成された上位プロトコルとして, 大小比較プロトコルや, 等号判定プロトコルなどがある．

3.2.1 和プロトコル

$[a]_p$ および $[b]_p$ から, $[c]_p = [a + b]_p$ を求める．$[c]_p$ を得るには各参加者が独立に Z_p 上でシェア $[a]_p$ と $[b]_p$ を足せばよい．Z_p 上の差 $[a - b]_p$ についても同様にして求めることができる．和プロトコルでは，参加者間の通信が不要であるため処理コストは無視できる．

3.2.2 積プロトコル [2, 4]

$[a]_p$ および $[b]_p$ から, $[a \times b]_p$ を求める．その処理コストは，参加者間の通信量として見積もられる．1 回の積プロトコルを実行するためには n 人の参加者間で相互に通信するため，$n(n-1)$ 回の通信が必要となる．ただし，$[a]_p$ と公開情報 $e \in \mathrm{Z}_p$ の積 $[c]_p = [a \times e]_p$ を得る場合，通信は不要である．

3.2.3 通信量とラウンド数

SMC には等号判定，大小比較，区間判定，Bit Decomposition などの上位レベルプロトコルが知られている．これらのプロトコルは主に和と積プロトコルで構成されるため，それらの処理コストは，必要な積プロトコルの回数で評価される．処理コストの評価には二つの評価指標がある．一つは通信量であり，積プロトコルの回数で表される．もう一つは，ラウンド数と呼ばれる並列処理を考慮した積プロトコルの実行回数である．ラウンド数の評価では，できる限り並列処理を行う．例えば，$[a]_p \times [b]_p \times [c]_p \times [d]_p$ という処理において，$[a]_p \times [b]_p$ と $[c]_p \times [d]_p$ を並列に実行し，それぞれの結果を掛け合わせることで，通信量は 3，ラウンド数は 2 となる．

3.3 上位プロトコル

3.3.1 Bitwise Less-Than(BLT)

$\{[a_1]_p, \cdots, [a_{\ell_p}]_p\}$ および $\{[b_1]_p, \cdots, [b_{\ell_p}]_p\}$ が与えられているとき，$[a < b]_p$ を得る．通信量は $19\ell_p$ であり，ラウンドは 8 回である．ただし，a か b のどちらかが公開されているとき，通信量は $17\ell_p$，ラウンドは 7 回となる [5]．

3.3.2 LSB

$[s]_p$ が与えられているとき，$[s > \frac{p}{2}]$ を得る．通信量は $93\ell_p + 1$ であり，ラウンドは 13 回である [5]．

3.3.3 Joint Random Bit Sharing(JRBS)

乱数のビットのシェア $[r_B]_p$ を得る．通信量は 2 回，ラウンドは 2 である [3]．

3.3.4 Joint Random Number Bitwise-Sharing(JRNBS)[5, 7]

乱数 $r \in \mathrm{Z}_p$ のビット単位のシェア $\{[r_i]_p \cdots [r_{\ell_p}]_p\}$ を得る．通信量は $76\ell_p$，7 ラウンドである [5]．JRNBS は内部で JRBS と BLT を 4 回繰り返す．

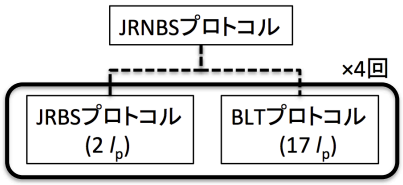

図 1: JRNBS プロトコル階層

大まかな手順はまず，JRBS により ℓ_p 個のビット単位のシェア $[r_i]_p$ を作成する $(1 \leq i \leq \ell_p)$．次に，Bitwise Less-Than プロトコルにより r を秘匿した状態で $(r < p)$ を判定し，真であれば $[r_i]_p$ を出力し，偽であれば最初からやり直す．

乱数共有プロトコル全体の通信量 $(76\ell_p)$ のうち，Bitwise Less-Than プロトコルの通信量は $68\ell_p$ であり，大半を占める．

3.3.5 大小比較プロトコル

$[a]_p$ および $[b]_p$ が与えられているとき，$[a < b]_p$ を得る．通信量は $279\ell_p + 5$ であり，15 ラウンドである [5]．

大小比較プロトコルは LSB プロトコルと積プロトコルで構成される．LSB プロトコルのサブプロトコルとして JRNBS プロトコルと，BLT プロトコルがある．また JRNBS プロトコルは，JRNBS プロトコルと BLT プロトコルで構成される．

大小比較プロトコルでは，BLT プロトコルが合計 15 回用いられている．その内訳としては大小比較プロトコル中では LSB プロトコルが 3 回用いられており，またその LSB プロトコルのサブプロトコルである JRNBS プロトコル中で，BLT プロトコルが 4 回用いられているためである $(15 = 3 \times 4 + 3)$．BLT プロトコル 15 回分の通信量 $(255\ell_p = 15 \times 17\ell_p)$ は，結果として，大小比較プロトコル全体の通信量の 90 パーセントを占めている．

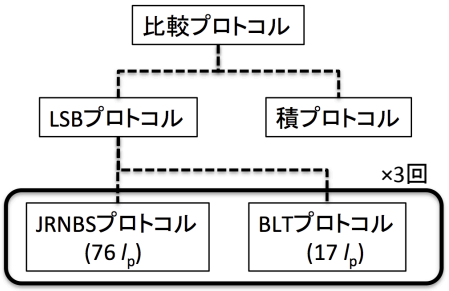

図 2: 大小比較プロトコル階層

3.3.6 等号判定プロトコル [3][5]

$[a]_p$ および $[b]_p$ が与えられているとき, $[a = b]_p$ を得る. 等号判定プロトコルの通信量は $81\ell_p$ であり, ラウンドは 5 回である [5].

等号判定プロトコルは JRNBS プロトコルと, ビット同士の演算で構成されるまた JRNBS プロトコルは BLT プロトコルと JRBS プロトコルで構成されているため, 等号判定プロトコル中の BLT プロトコルの回数は 4 回である.

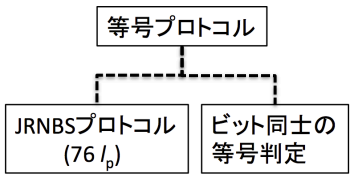

図 3: 等号判定プロトコル階層

3.3.7 Bit Decomposition プロトコル

$[s]_p$ が与えられているとき, $\{[s_1]_p, \cdots, [s_{\ell_p}]_p\}$ を得る [7]. Bit Decomposition プロトコルの通信量は $57\ell_p \log^* \ell_p + 71\ell_p + 14\sqrt{\ell_p} \log^* \ell_p + 30\sqrt{\ell_p}$ であり, ラウンドは 33 回である.

4 提案手法で用いるプロトコル

4.1 Random Bit Representation(RBR)

乱数のビットのシェア $[r_B]_p$ および $[r_B]_q$ を得る [1].

RBR プロトコル

1: Player$_i$ は乱数 r_B^i を生成し, $[r_B^i]_p$ および $[r_B^i]_q$ を分散する $(1 \leq i \leq k)$.
2: $[r_B]_p \leftarrow [r_B^1]_p \oplus \cdots \oplus [r_B^k]_p$
3: $[r_B]_q \leftarrow [r_B^1]_q \oplus \cdots \oplus [r_B^k]_q$

なお, 定数ラウンドにするために, k 個の要素の排他的論理和には, Unbounded Fan-In Xor プロトコルを用いる. 通信量は法 p での計算として $5k + 1$, 同様に法 q では $5k + 1$ である. ラウンドは 3 回である.

4.2 Conversion Between Bit Shares (CBBS)

$[s_B]_p$ から $[s_B]_q$ を得る [1].

CBBS プロトコル

Input: $[s_B]_p$

1: $[r_B]_p, [r_B]_q \leftarrow$ RBR
2: $[t_B]_p \leftarrow [s_B]_p \oplus [r_B]_p$
3: $t_B \leftarrow \text{Reveal}([t_B]_p)$
4: $[s_B]_q \leftarrow [r_B]_q \oplus t_B$

通信量は法 p での計算として $5k + 2$, 同様に法 q では $5k + 1$ である. ラウンドは 4 回である.

5 提案手法

5.1 基本アイディア

$[c]_p \leftarrow [a]_p \times [b]_p$ と $[c]_q \leftarrow [a]_q \times [b]_q$ では両者とも積プロトコル 1 回であるが, 通信量は異なる. なぜなら, 素数 p で分散されたシェアのサイズは ℓ_p ビットであり, 素数 q で分散されたシェアのサイズは ℓ_q ビットであるためである. 積プロトコルではプレイヤー間で, シェアを相互に通信するため, 通信量はシェアのサイズに比例する.

この点に着目し, プロトコルの中間処理で扱う秘密情報の分散に, 出来るだけ小さな素数 p' を用いることで, 通信量を削減する手法を考えた.

しかし, このアプローチの実践には以下の問題点がある.

p' の不確定性 中間処理において扱う秘密を s とすると, p' は s を超える最小の素数であることが望ましい. しかし, s は一般には秘匿された値であるため, p' を確定することができない.

情報漏洩 秘密分散時に p' が公開される. なんらかの方法で, p' を確定させることが出来ても, その公開によって, s のサイズが漏洩してしまう.

法変換のコスト　p から p' へ秘密情報を分散し直す必要があるため, ラウンド数は増大する. また p' の削減効果よりも, 分散し直すための通信量が大きい場合, 通信量も増大する.

5.2　通信量の分析

基本的なプロトコルを分析すると, その通信量の大部分はビット演算であると分かる.

例えば, 大小比較プロトコルの場合, 積プロトコルと JRBS プロトコルを除いた全ての演算が, ビット演算である. これは, 大小比較プロトコル全体の通信量のおよそ 90 パーセントにのぼる. また等号判定プロトコルについても同様で, JRBS プロトコル以外のプロトコルはすべてビット演算である. 等号プロトコルのビット演算の占める割合は 90 パーセントである.

従来方式では, ビットのシェアは, 1 ビットの秘密情報でありながら, プロトコルの入出力をなす秘密情報と同じく, 素数 p で分散されていた. しかし, ビットのシェアを p より小さな素数で分散し, なおかつ正確性と安全性を維持出来れば, ビット演算の通信量の削減が可能となる.

5.3　頻出パターン

上位プロトコルを分析した結果, 以下のビット演算が頻出することが分かった.

頻出プロトコル　JRNBS

1: $\{[r_1]_p, \cdots, [r_{\ell_p}]_p\} \leftarrow$ JRBS を ℓ_p回
2: $[v_B]_p \leftarrow \mathrm{BLT}(\{p_1, \cdots, p_{\ell_p}\}, \{[r_1]_p, \cdots, [r_{\ell_p}]_p\})$
3: $v_B \leftarrow \mathrm{Reveal}([v_B]_p)$
4: もしも, $r > p$ なら最初からやり直す.

第一のパターンは JRNBS である. JRBS を用いて, $\{[r_1]_p, \cdots, [r_{\ell_p}]_p\}$ を生成する. ビット乱数と, 公開された値 p を引数として, 関数 BLT を計算する. 結果を公開し, もしも, $r > p$ なら最初からやり直す.

第二のパターンは JRNBS を利用する, 次のようなパターンである.

JRNBS を用いて, $[r]_p$ および $\{[r_1]_p, \cdots, [r_{\ell_p}]_p\}$ を生成する. $[s]_p$ と $[r]_p$ を加算し, その結果を v として公開する. ビット乱数と, 公開された値 v を引数として, 関数 F を計算する. ただし, 関数 F は, 論理積, 論理和, 排他的論理和などで構成されたビット演算である. 第二のパターンを JRNBS 利用パターンと呼ぶことにする. 表 1 に示すように多くのプロトコルは JRNBS 利用パターンを内部に持つ.

頻出プロトコルパターン　JRNBS 利用パターン

Input: $[s]_p$
1: $[r]_p, \{[r_1]_p, \cdots, [r_{\ell_p}]_p\} \leftarrow$ JRNBS
2: $[v]_p \leftarrow [s]_p + [r]_p$
3: $v \leftarrow \mathrm{Reveal}([v]_p)$
4: $[u_B]_p \leftarrow \mathrm{F}(\{v_1, \cdots, v_{\ell_p}\}, \{[r_1]_p, \cdots, [r_{\ell_p}]_p\})$

表 1: プロトコル中の JRNBS 利用パターン

プロトコル	F 関数
大小比較 (LSB)	BLT
等号判定	ビット同士の等号判定
Bit Decomposition	排他的論理和

5.4　提案方式

小さな素数 p' を用いることで, 頻出パターンの通信量を削減する. なお p' は, 秘密分散に用いた p よりも小さな素数である. p' のサイズについては後述する.

まず素数 p' を用いて, JRNBS プロトコルのビット演算の通信量を削減する手法を示す.

Advanced JRNBS (AJRNBS) プロトコル

1: $\{[r_1]_{p'}, \cdots, [r_{\ell_p}]_{p'}\} \leftarrow$ JRBS を ℓ_p回
2: $[v]_{p'} \leftarrow \mathrm{BLT}(\{p_1, \cdots, p_{\ell_p}\}, \{[r_1]_{p'}, \cdots, [r_{\ell_p}]_{p'}\})$
3: $v \leftarrow \mathrm{Reveal}([v]_{p'})$
4: もしも v が真なら, 初めからやり直す.
5: $[r_i]_p \leftarrow \mathrm{CBBS}([r_i]_{p'})$ $(1 \le i \le \ell_p)$

このプロトコルは従来方式とは異なり, 法 p' のビットの乱数を生成する. p' の法のまま, 途中計算を行い, 最終結果を CBBS プロトコルにより, p で分散し直す.

JRNBS 利用パターンに対して, ビット演算の通信量を削減する手法を示す.

AJRBS 利用パターン

Input: $[s]_p$

1: $[r]_p, \{[r_1]_{p'}, \cdots, [r_{\ell_p}]_{p'}\} \leftarrow$ AJRNBS
2: $[v]_p \leftarrow [s]_p + [r]_p$
3: $v \leftarrow \mathrm{Reveal}([v]_p)$
4: $[u_B]_{p'} \leftarrow \mathrm{F}(\{v_1, \cdots, v_{\ell_p}\}, \{[r_1]_{p'}, \cdots, [r_{\ell_p}]_{p'}\})$
5: $[u_B]_p \leftarrow \mathrm{CBBS}([u_B]_{p'})$

JRNBS プロトコルを用いて, 乱数 $[r]_p$ および $\{[r_1]_{p'}, \cdots, [r_{\ell_p}]_{p'}\}$ を得る. $[s]_p$ に対して, $[r]_p$ を加算し, その結果を v として公開する. v と $\{[r_1]_{p'}, \cdots, [r_{\ell_p}]_{p'}\}$ を引数として, 関数 F を計算する. 関数 F の結果 $[u_B]_{p'}$ を, CBBS により, $[u_B]_p$ に変換する.

5.5 通信量の単位

一般的に SMC では, あるプロトコルの通信量は, そのプロトコルの内部で積プロトコルが何回用いられたかで, 見積もられる. 例えば, 前述の大小比較プロトコルの通信量は $279\ell_p + 5$ だが, これは大小比較プロトコルの内部で, 積プロトコルが $279\ell_p + 5$ 回相当用いられていることを意味していた. しかし, 前述したとおり, 積プロトコルの回数が同じでも, 素数のサイズに比例して, 通信量は異なる. そこで素数のサイズの影響を明示するために, 従来の通信量の指標を拡張する.

本指標では素数 p 上での積プロトコル一回 ($[c]_p \leftarrow [a]_p \times [b]_p$) を ℓ_p として見積もる. 同様に素数 q 上での積プロトコル一回 ($[c]_q \leftarrow [a]_q \times [b]_q$) は ℓ_q である.

本指標を用いて, 改めて大小比較プロトコルを評価すると, その通信量は $(179\ell_p + 5)\ell_p$ となり, また同様に等号判定プロトコルの通信量は $81\ell_p^2$ となる.

5.6 AJRNBS の効率, 正確性, 安全性

JRBS プロトコルを ℓ_p 回用いて, 乱数 $\{[r_1]_{p'}, \cdots, [r_{\ell_p}]_{p'}\}$ を得る ($2\ell_p\ell_{p'}$, 2 ラウンド). 乱数と p を引数として BLT プロトコルを計算する ($17\ell_p\ell_{p'}$, 7 ラウンド). 結果を公開し, もしも ($p \leq r$) のときは初めからやり直す. CBBS プロトコルを ℓ_p 回用いて, p' で分散された乱数を p で分散された乱数に変換する ($(5k+2)\ell_p^2 + (5k+1)\ell_p\ell_{p'}$, 4 ラウンド).

ラウンドと通信量について述べる. 文献 [3] によると, ($p \leq r$) のときのやり直しを考慮した場合, ステップ 1 から 4 を, 4 回繰り返すとみなしてよい. そこで通信量は $(5k+2)\ell_p^2 + (5k+77)\ell_p\ell_{p'}$ である. ラウンドは, JRBS プロトコルと, BLT 内部の JRBS プロトコル, CBBS プロトコル中の RBR プロトコルが並列して実行可能であるため, 8 ラウンドとなる.

ℓ_p のサイズについて考察する. BLT プロトコル内部の Unboud Fan In Or プロトコルにおいて, ラグランジュ方程式を作る際に, ℓ_p 個のビットの和を求める. その和の値をのぞき, AJRNBS プロトコル中の他の秘密情報は全て 0 か 1 のビットである. したがって, $\ell_{p'}$ を $\lceil \log \ell_p \rceil$ 以上の整数とすれば, AJRNBS プロトコルの中で値がモジュロ演算の影響を受けることはなく, その出力は JRNBS の出力に等しくなる. また AJRNBS の中の値の最大値が ℓ_p であることは, 秘密情報の値にかかわらない. したがって, $\ell_{p'}$(つまり p') を公開したとしても, 情報が漏洩することはない. 以上により, $\ell_{p'}$ を $\lceil \log \ell_p \rceil$ より大きくすれば, AJRNBS の正確性と安全性は保証される.

AJRNBS プロトコルは, 従来の JRNBS プロトコル ($76\ell_p^2$, 7 ラウンド) と比べると, 1 ラウンド増大する. p が 32 ビットの素数である場合の通信量と, 64 ビットの素数である場合の通信量の削減効果を, 表 2 に記す. なお表内部の比率は提案方式 (AJRNBS) の通信量を従来方式 (JRNBS) の通信量で割ったものである.

表 2: AJRNBS プロトコルの通信量の削減効果

$\ell_p, \ell_{p'}$	k	JRNBS	AJRNBS
$\ell_p = 32, \ell_{p'} = 7$	2	77824	31776(40%)
$\ell_p = 32, \ell_{p'} = 7$	4	77824	44256(55%)
$\ell_p = 64, \ell_{p'} = 8$	2	311296	93696(30%)
$\ell_p = 64, \ell_{p'} = 8$	4	311296	139776(45%)

5.7 AJRNBS 利用パターンの効率

F 関数の法 p 上の通信量を $C_F\ell_p$ とし, 法 p' 上の通信量を $C_F\ell_{p'}$ とする.

AJRNBS 利用パターンでは, AJRNBS プロトコルおよび F 関数, CBBS プロトコルを用いる. AJRNBS 利用パターンの通信量は $C_F\ell_{p'}+(5k+2)\ell_p^2+(5k+77)\ell_p\ell_{p'}+(5k+1)\ell_p+(5k+2)\ell_{p'}$ である.

6 効率化の実例

6.1 LSB プロトコル

LSB プロトコルは, 入力値 $[s]_p$ を 2 倍し, その結果の最下位ビットが 0 ならば $s \leq \frac{p}{2}$, 1 ならば $s > \frac{p}{2}$ と判定する. LSB プロトコルのビット演算部分に, 提案手法を用いる.

Advanced LSB (ALSB) プロトコル

Input: $[s]_p$

1: $[2s] \leftarrow 2 \times [s]_p$
2: $[r]_p, \{[r_1]_{p'}, \cdots, [r_{\ell_p}]_{p'}\} \leftarrow$ AJRNBS
3: $[v]_p \leftarrow [2s]_p + [r]_p$
4: $v \leftarrow \mathrm{Reveal}([v]_p)$
5: $[u_B]_{p'} \leftarrow \mathrm{BLT}(\{v_1, \cdots, v_{\ell_p}\}, \{[r_1]_{p'}, \cdots, [r_{\ell_p}]_{p'}\})$
6: $[c_B]_{p'} \leftarrow v_1 \oplus [r_1]_{p'}$
7: $[t_B]_{p'} \leftarrow [u_B]_{p'} \times (1 - [c_B]_{p'}) +$
$(1 - [u_B]_{p'}) \times [c_B]_{p'}$
8: $[t_B]_p \leftarrow \mathrm{CBBS}([t]_{p'})$

ALSB プロトコルのラウンドと通信量について述べる. AJRNBS プロトコル 1 回に加え BLT プロトコルが 1 回, CBBS プロトコルが 1 回, その他の計算 $(3\ell_{p'})$ がかかるため, プロトコル全体の通信量は $(5k+2)\ell_p^2+(5k+94)\ell_p\ell_{p'}+(5k+1)\ell_p+(5k+5)\ell_{p'}$ である. ただし, $[t_B]_{p'}$ を出力するときは, $(5k+2)\ell_p^2+(5k+94)\ell_p\ell_{p'}+3\ell_{p'}$ である.

また, ラウンドは AJRBS プロトコルと, BLT 内部の JRBS プロトコル, CBBS プロトコル中の RBR プロトコルが並列して実行が可能であるため, 15 ラウンドである.

従来の LSB プロトコル $(93\ell^2+\ell_p$, 13 ラウンド) と比べると, 提案方式の方が 2 ラウンド増大する. 通信量の削減効果を, 表 3 に記す.

表 3: LSB プロトコルの通信量の削減効果

$\ell_p, \ell_{p'}$	k	従来方式	提案方式
$\ell_p = 32, \ell_{p'} = 7$	2	95264	36041(35%)
$\ell_p = 32, \ell_{p'} = 7$	4	95264	48911(50%)
$\ell_p = 64, \ell_{p'} = 8$	2	380992	103224(30%)
$\ell_p = 64, \ell_{p'} = 8$	4	380992	150024(40%)

6.2 大小比較プロトコル

前述した LSB プロトコルを利用すると, 提案方式は $(15k+6)\ell_p^2+(15k+282)\ell_p\ell_{p'}+(5k+1)\ell_p+(5k+13)\ell_{p'}$, 17 ラウンドとなる. 従来方式と比べると, 提案方式の方が 2 ラウンド増大する. 通信量の削減効果を, 表 4 に記す.

表 4: 大小比較プロトコルの通信量の削減効果

$\ell_p, \ell_{p'}$	k	従来方式	提案方式
$\ell_p = 32, \ell_{p'} = 7$	2	285856	107265(35%)
$\ell_p = 32, \ell_{p'} = 7$	4	285856	145095(50%)
$\ell_p = 64, \ell_{p'} = 8$	2	1143104	308088(25%)
$\ell_p = 64, \ell_{p'} = 8$	4	1143104	447048(40%)

6.3 等号判定プロトコル

等号判定プロトコルのビット演算部分に, 提案手法を用いる.

Advanced 等号判定 (AEQ) プロトコル

Input: $[s^1]_p$, $[s^2]_p$

1: $[r]_p, \{[r_1]_{p'}, \cdots, [r_{\ell_p}]_{p'}\} \leftarrow$ AJRNBS
2: $[v]_p \leftarrow [s^1]_p - [s^2]_p + [r]_p$
3: $v \leftarrow \mathrm{Reveal}([v]_p)$
4: **if** $v_i = 0$ **then**
5: $\quad [c_i]_{p'} \leftarrow 1 - [r_i]_{p'}$
6: **else**
7: $\quad [c_i]_{p'} \leftarrow [r_i]_{p'}$
8: **end if**
9: $[t_B]_{p'} \leftarrow \mathrm{UnboudFanInAnd}(\{[c_1]_{p'}, \cdots, [c_{\ell_p}]_{p'}\})$
10: $[t_B]_p \leftarrow \mathrm{CBBS}([t_B]_{p'})$

AEQ プロトコルのラウンドと通信量について述べる. AJRNBS プロトコル 1 回に加え UnboudFanInAnd プロトコルが 1 回, CBBS プロトコルが 1 回かかるため, プロトコル全体の通信量は $(5k+2)\ell_p^2+(5k+82)\ell_p\ell_{p'}+(5k+1)\ell_p+(5k+2)\ell_{p'}$ である. また, ラウンドは JRNBS プロトコル内部の JRBS プロトコルと, UnboudFanInAnd プロトコル内部の JRBS プロトコル, CBBS プロトコル内部の RBR が並列して実行可能であるため 10 ラウンドとなる.

従来の等号判定プロトコル ($81\ell_p^2$, 8 ラウンド) と比べると, 提案方式の方が 2 ラウンド増大する. 通信量の削減効果を, 表 5 に記す.

表 5: 等号判定プロトコルの通信量の削減効果

$\ell_p, \ell_{p'}$	k	従来方式	提案方式
$\ell_p = 32, \ell_{p'} = 7$	2	82944	33332(40%)
$\ell_p = 32, \ell_{p'} = 7$	4	82944	46202(55%)
$\ell_p = 64, \ell_{p'} = 8$	2	331776	97056(30%)
$\ell_p = 64, \ell_{p'} = 8$	4	331776	143856(45%)

6.4 Bit Decomposition プロトコル

詳細は省くが, ラウンドは 35, 通信量は $(10k+3)\ell_p^2+(10k+57\log *\ell_p+87)\ell_p\ell_{p'}+14\sqrt{\ell_p}\ell_{p'}$ である. 通信量の削減効果を, 表 6 に記す.

表 6: Bit Decomposition の通信量の削減効果

$\ell_p, \ell_{p'}$	k	従来方式	提案方式
$\ell_p = 32, \ell_{p'} = 7$	2	200576	73644(35%)
$\ell_p = 32, \ell_{p'} = 7$	4	200576	98604(50%)
$\ell_p = 64, \ell_{p'} = 8$	2	208256	97056(25%)
$\ell_p = 64, \ell_{p'} = 8$	4	300416	143856(40%)

7 まとめ

秘密情報を少ないビット数で表現し, 通信量を低減する手法を提案した. SMC の多くの基本的なプロトコルは, 内部にビット演算を含むことから, その頻出パターンを 2 つ見出した. プロトコルの入出力情報のビット数を ℓ_p とすると, これらの頻出パターン中で扱われる情報のビット数の上限は, 入出力情報に依存せず, $\lceil \log p \rceil$ となる. そこで, 頻出パターンの計算時にサイズ ℓ_p の法からサイズ $\lceil \log \ell_p \rceil$ の法への法変換を行うことで, 正確性と安全性を維持しながら, 当該部分の通信量を $\frac{\log \ell_p}{\ell_p}$ に低減できる. この手法を大小比較, 等号判定などのプロトコルに適用し, 通信量が従来の 25% から 55%に削減できることを示した.

参考文献

[1] Aleksandr, Y.: Efficient Cryptographic Tools for Secure Distributed Computing. A Dissertation Presented to the Faculty of the Graduate School of Yale University

[2] M. Ben-Or, S. Goldwasser, and A. Wigderson.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. 20th Annual ACM Symposium on Theory of Computing, pp.1–10, ACM, 1988.

[3] Damgård, I., Fitzi, M., Kiltz, E., Nielsen, J.B., Toft, T.: Unconditionally Secure Constant-Rounds Multi-party Computation for Equality, Comparison, Bits and Exponentiation. TCC 2006. LNCS, vol. 3876, pp. 285–304. (2006)

[4] R. Gennaro, M.O. Rabin, and T. Rabin.: Simplified VSS and fast-track multiparty computations with applications to threshold cryptography. 17th Annual ACM Symposium on Principles of Distributed Computing, pp.101–111, ACM, 1988.

[5] Nishide, T., Ohta, K.: Multiparty Computation for Interval, Equality, and Comparison Without Bit-Decomposition Protocol. PKC 2007. LNCS, vol. 4450, pp. 343–360. (2007)

[6] Shamir, A.: How to Share a Secret. CACM 22(11), 612–613 (1979)

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