情報を秘密にしたまま計算ができる秘密計算技術

東京理科大学　工学部　情報工学科　助教　Ahmad A. Aminuddin（アフマド　アクマル　アミヌディン）

1. はじめに

近年，ビッグデータやInternet of Things（IoT）技術の進歩に伴い，大量のデータを収集し，活用することで，様々な社会問題の解決や，新たな価値のあるサービスの開発などが可能になっている．一方，収集したデータには，検索履歴，医療データなど，個人に関連する情報が含まれるため，個人の情報漏洩や，プライバシー侵害などのリスクがあるという課題が存在する．

また，欧州連合の一般データ保護規制や日本の改正個人情報保護法などの法整備からも，秘密情報を保護できる技術がより重要視される．すなわち，自由なデータ流通と個人情報保護を両立できる技術が求められ，**個人情報保護を担保したデータ解析を含む情報処理が可能な技術**が必要となる．それによって，個人の賃金を秘匿したままの男女間賃金格差の検証や，検診情報を秘匿した自動健康診断サービスなどが実現できる．

個人のプライバシーを保護できる代表的な技術として，仮名化・匿名化，差分プライバシーと秘密計算があげられる．特に，**秘密計算**に関する研究は，国内外において盛んに行われ，注目を集めている．この数年では，秘密計算の定義化『ISO/IEC4922-1:2023』や秘密計算を実現する方式の定式化などの活動も進んでいて，2024年3月に発行された『ISO/IEC4922-2:2024』には，秘密分散を用いた秘密計算の基礎的な演算方式が規定された．

本稿では，なぜ秘密計算が必要になるのかを説明し，原理，モデルや，実現方法を紹介する．また，平均年収を求める問題を例とし，秘密計算を説明する．最後に，秘密計算の応用例を簡単に紹介する．

2. 秘密計算はなぜ必要なのか？

例えば，個人の医療データを安全に保管したい場合は，昔から知られている暗号技術を利用し，医療データを暗号化してクラウドサーバに保存すれば良い．**暗号化**とは，基となる意味がある秘密情報に対して，異なる意味を持つ暗号化した情報に変換し，解読できない状態にする処理である．また，暗号化した情報より，元の情報に戻す処理を**復号**という．

一方，集めた暗号化したデータに対して，解析処理を行い，何かの新たな知見を得るには，暗号化したデータを一旦復号してから行う必要があり，復号段階で秘密情報が漏洩するリスクが発生する．このようなリスクを回避するために，必要な計算を暗号化したままで実現できる，秘密計算という処理が必要となる．

3. 秘密計算とは？

秘密計算とは，**情報を秘密にしたまま計算を行う**技術である．1982年に，Yaoによって，2者間の金持ち比べ問題（お互いに自分の財産を秘密にして，どちらが金持ちかを知る問題）に対して，その概念の提案とともに，最初の秘匿回路を用いた手法が提案された．例えば，Aliceの財産をx円，Bobの財産をy円とする場合，どちらのほうが財産を多く持っているか？　を知りたいとき，Yaoの秘匿回路を実行し，xがyより大きい場合は0，そうでない場合は1，の2値だけで判断することができる．

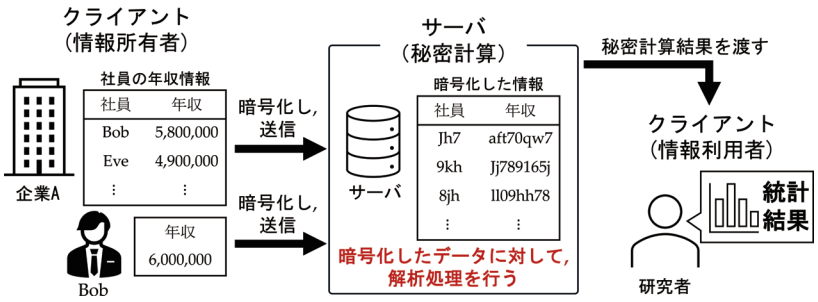
秘密計算によって，複数のユーザーが持つ個人に関連する情報をお互いに共有することなく，完全に信頼できない環境においても，情報の秘密性を保ったまま，生データと同等な統計解析の計算が可能になる．

4. 秘密計算のモデルと実現する方法

秘密計算では，想定する参加者によって，いくつかのモデルがあげられる．近年，AmazonのAWSをはじめとする，様々なクラウドサービスが普及している．クラウドサービスを利用することによって，物理的なサーバ設置が不要で，運営コストを削減したり，容量と仕様を柔軟に変更できたりするといった利点がある．

本稿では，計算資源やノウハウを持つクラウドサーバが，ユーザー（クライアント）から，個人に関連するデータを集めて，解析をする**クライアント・サーバ**モデルを想定する．このモデルでは，秘密情報を持つクライアント（情報所有者）は，自分の保持している情報を暗号化し，クラウドサーバにアウトソーシングする．サーバ（サービス提供者）は，収集したデータを

元に，暗号化したまま秘密計算を実行し，安全にデータ解析を行い，最終結果だけを秘密計算を依頼したクライアント（情報利用者，例えば，研究者や企業など）に返す【図】．後述する秘密計算の実現方法によって，サーバは 1 台だけで良い場合や，複数台必要な場合がある．



【図】クライアント・サーバモデル

秘密計算を実現する代表的な手法として，以下の 4 つがあげられる．

■秘匿回路（Garbled circuit）
■準同型暗号（Homomorphic encryption）
■秘密分散（Secret sharing）
■信頼できる実行環境（TEE: Trusted execution environment）

秘匿回路は，1982 年に，Yao が提案した，秘密計算を実現する手法であり，2 進数の 0，1 に対応するラベルを用いて，2 入力論理回路の計算を表によって実現できるが，秘匿回路の作成処理には時間がかかる．準同型暗号とは，1 台のサーバ内で暗号文を復号することなく暗号文のまま計算を行うことが可能な暗号方式のことであり，加法準同型暗号，乗法準同型暗号，Somewhat 準同型暗号（SHE）と完全準同型暗号に分類できるが，秘密分散に比べて，計算コストが大きいことが知られている．秘密分散とは，秘密情報を複数の異なる値（以降，分散値）に分割し，複数の独立に管理されるサーバに送信する手法であり，秘密計算を行う際に，複数サーバ間の通信が必要となる．TEE は，CPU に搭載される，隔離された信頼できる実行環境を指す．デバイスのメモリ上に保護された領域を構築し，秘密計算を行うことで，サーバ間の通信を削除できるが，サイドチャネル攻撃に弱い．

本稿では，最もわかりやすい秘密分散を用いて，秘密計算の原理や，計算方法などを紹介する．

5. 秘密分散を用いた秘密計算の仕組み

ここでは，「Charles が Alice と Bob の平均年収 $y=avg(a,b)$ を求めたい」場合を例として，秘密分散を用いた秘密計算の仕組みを解説する．秘密分散の例としては，2017 年 10 月に発行された『ISO/IEC19592-2:2017』に，(k,n) しきい値秘密分散，加法型秘密分散やランプ型秘密分散などが紹介されている．

ここでは，専門的にならず読者がイメージしやすい，連立方程式をベースとする Shamir の (k,n) しきい値秘密分散を用いる方法について例を交えて解説する．

5. 1. (k,n) しきい値秘密分散

(k,n) しきい値秘密分散とは，ある秘密情報 s を n 個の異なる分散値 $[s]_1,...,[s]_n$ に変換し，n 台の独立に管理されるサーバに保管する手法である．$[s]_i$ は，サーバ S_i（サーバの識別子 $id=x_i$）が保持する秘密情報 s の分散値を表している．(k,n) しきい値秘密分散により生成された n 個の分散値は，以下を満たす．

●任意の k 個以上の分散値を集めると，秘密情報 s を復元できる．

$$H(s|[s]_1,...,[s]_k)=0$$

● $k-1$ 個以下の分散値からは，秘密情報 s を一切復元できない．

$$H(s|[s]_1,...,[s]_{k-1})=H(s)$$

代表的な例として，Shamir によって提案された，ラグラジアン補間公式をベースとした手法がある（以降，Shamir 法）．分散と復元処理は次のように構成される．なお，D（ディーラ）は秘密情報を持つ，情報を分散するユーザーを指し，$x_i(i=1,...,n)$ を各サーバ S_i の識別子とし，公開されるとする．

Protocol 1：分散処理

1. D は，$k-1$ 個の係数 $a_1,...,a_{k-1}$ をランダムに選び，定数項を秘密情報 s とし，式（1）に示す $k-1$ 次の分散式を生成する．

$$f(x)=s+a_1x+a_2x^2+...+a_{k-1}x^{k-1} \qquad (1)$$

2. D は分散値 $[s]_i=f(x_i)(i=1,...,n)$ を計算し，サーバ S_i に $[s]_i$ を送信する．

Protocol 2：復元処理

1. n 台のサーバから任意の k 個の分散値とその識別子のペア $((x_1,[s]_1),...,(x_k,[s]_k))$ を収集し，ラグラジアン補間公式を利用し，分散式 $f(x)$ を計算し，秘密情報 $s=f(0)$ を得る．

$$f(x)=\sum_{i=1}^{k}\left([s]_i\prod_{1\le j\le k,\,i\ne j}\frac{x-x_j}{x_i-x_j}\right)$$

なお，上記の分散・復元処理では，秘密情報や乱数

などを整数とするが，通常の暗号の世界における演算処理は，有限体 GF(q) 上で実現されることが多い．ここで，q は位数とよばれ，位数が素数 p である有限体 GF(p) は素体とよばれる．有限体 GF(p) 上の加減算と乗算を実現するには，p 個の整数の集合 $\{0,1,\ldots,p-1\}$ に対して，剰余演算を行えばよい．例えば，入力 a,b がある場合，加減算と乗算を整数として計算してから，p で割ったあまりを取ればよい．除算は，逆元 b^{-1} 求めて（拡張ユークリッドの互除法）から，a と乗算する必要がある．

しかし，本稿では，説明を簡単にするために，有限体 GF(p) 上の演算を考えず，以降では，すべてを整数として，秘密計算の具体例を説明する．

5．2．秘密分散を用いた秘密計算の具体例

Shamir 法を利用し，分散値数 $n=3$，しきい値 $k=2$ の設定において，平均を求める問題を例として解説する．Alice, Bob は，自分の年収情報を $n=3$ 台のサーバに分散し，3 台のサーバはお互いに協力し，秘密計算を実行し，最終結果の分散値を Charles に返す．

(a) 分散処理

Alice と Bob の年収をそれぞれ $a=540$ 万円，$b=600$ 万円とする．Alice と Bob はそれぞれ Protocol 1 を実行し，以下の分散式 $f_a(x)$, $f_b(x)$ を決定する：

- Alice は $a_1=3$ を選び，$f_a(x)=540+3x$ を生成
- Bob は $b_1=5$ を選び，$f_b(x)=600+5x$ を生成

その後，各サーバ S_i の識別子 x_i に対する分散値 $[a]_i,[b]_i$ を計算し，送信する．例えば，サーバ S_2 の識別子，$x_2=2$ とすると，分散値 $[a]_2=540+3(2)=546$, $[b]_2=600+5(2)=610$ を計算し，送信する．各サーバ S_i は【表】に示す分散値を保持する．

(b) 秘密計算処理

各サーバ S_i は，保持している分散値の平均を求め，Charles に渡す．例えば，S_1 は，保持している分散値の平均値 $[c]_1=([a]_1+[b]_1)/2=574$ を計算し，Charles に渡す．残りの 2 台のサーバも同様な計算を行い，分散値 $[c]_2=578$，$[c]_3=582$ を Charles に渡す．

(c) 復元処理

しきい値 $k=2$ のため，Charles は，送られてきた 3 つの分散値のうち，2 つ，例えば，$(x_1,[c]_1)=(1,574)$ と $(x_2,[c]_2)=(2,578)$ を用いて，Protocol 2 を実行し，以下計算で平均年収を表している分散式 $f_c(x)$ を求め，平均年収 $c=f_c(0)=570$ 万円を求める．

$$f_c(x) = 574\frac{x-2}{1-2} + 578\frac{x-1}{2-1} = 570 + 4x$$

【表】(2,3) しきい値秘密分散を用いた秘密計算

	年収（万円）	各サーバが保持する分散値 $S_1(x_1=1)$	$S_2(x_2=2)$	$S_3(x_3=3)$
Alice	$a=540$	543	546	549
Bob	$b=600$	605	610	615
平均	$c=570$	574	578	582

5．3．秘密分散を用いた乗算について

秘密分散を用いた線形演算（加減算，定数倍乗算）は，サーバ間の通信を必要とせず，ローカル演算のみで簡単に実現できる．一方，乗算 ab は簡単に実現できず，$k-1$ 次式で表されている分散値同士を単純に乗算する場合，生成した分散値を表している**多項式の次数が $k-1$ から $2k-2$ に増加**し，乗算結果の復元に必要となる分散値の数も k から $2k-1$ に増える．

例えば，1 次式で表されている Alice と Bob の分散値同士を乗算すると，生成した乗算結果 ab を表している多項式 $f_{ab}(x)=f_a(x)f_b(x)$ の次数が 2 になり，2 次式の決定には 2 個ではなく，3 個の分散値が必要となるため，3 台のサーバから分散値を集め，ab を復元する必要がある．また，ab の分散値をさらに新たな入力 c の分散値と乗算すると，次数が $2k-2=3$ から $4k-4=4$ に変化し，必要となる分散値がさらに増える．この場合，分散値の数が足りないため，abc の乗算結果を復元できない．このように，最初に n 個の分散値だけを生成すれば，どこかで結果の復元ができなくなってしまうという問題が存在する．

乗算処理による次数変化という問題を回避するために，様々な方法が提案された．Ben-Or らは，$n>2k-1$ という仮定のもと，次数削減というステップを導入し，乗算処理により生成した多項式の次数を $2k-2$ から $k-1$ に戻す処理を提案した[1]．また，Beaver の multiplication triple をベースとする SPDZ 方式では，SHE を秘密分散と組み合わせて，SHE を用いて multiplication triple を事前に生成することで，最小 2 台のサーバで乗算を含む任意の演算を実現できる[2]．また，より高速な秘密計算を目指すために，加法型秘密分散と複製型秘密分散を利用する秘密計算法も多数検討されている．さらに，社会実装を目指すために，しきい値 k と分散値数 n を固定した，Bogdanov らによる Sharemind，補助乱数を利用する，NEC と NTT が独立に提案した秘密計算法なども知られている．著者は，攻撃者が知らない乱数の導入や，

1 台のサーバに複数分散値を安全に送信できる手法の検討などを行い，任意の $n,k(n\geq k>1)$ においても秘密計算ができるような研究を行っている[3]．さらに，元電気工学科の岩村恵市教授との共同研究で，秘密計算の高速化や，様々な応用を想定した場合の計算処理の最適化などの研究も行っている[4]．本稿では，これ以上言及しないが，興味のある方は，ぜひこの機会に，各論文を参照されたい．

6. 秘密計算の応用

秘密計算が提案されて以来，様々なところに応用されている．ここでは，活用事例をいくつか紹介する．

■セキュアオークション

秘密計算の最も有名な活用事例は，2008 年のデンマークの甜菜オークションである．農家の入札情報をデンマーク唯一の甜菜加工会社である Danisco 社にそのまま提供せず，農家連合，Danisco 社，研究機関の三者間でのオークションは秘密計算によって実現されることによって，お互いの入札金額を開示せず，セキュアなオークションができた．

■医療情報解析

医学領域において研究が進められているゲノム解析には，ゲノム情報の秘匿性を確実に守りながら，収集・解析ができるような技術が求められる．2019 年に，NEC と大阪大学の共同研究において，大阪大学が開発した，複数の医療・研究機関が保有するゲノム情報や疾病等に関する情報を統合して解析できるアプリケーションに，NEC の秘密計算を導入し，複数の機関が保持するゲノム情報を保護しながら，大量なゲノム情報を解析できるようにした．

■機械学習の安全性向上

機械学習の 1 つである連合学習では，分散しているデータを中央サーバに集約せず，安全なモデル構築を実現するが，学習過程で得られる情報から学習データが推測されるなどのプライバシー上のリスクがある．その対策法として，差分プライバシーが利用されているが，近年，秘密計算を利用する方法も注目を集めている．例えば，情報通信研究機構の DeepProtect では，連合学習に暗号技術を融合し，中央サーバに送られた，暗号化した差分パラメータを，秘密計算によって復号せずに暗号化したまま統合し，学習モデルを更新できる．そのほかに，創薬における予測モデルの構築に関する実証実験や，患者の情報を保護したまま炎症性腸疾患に関する観察研究なども，近年，盛んに行われる．

■秘匿検索

秘匿検索とは，データを秘匿したまま検索ができる技術である．クラウドストレージを利用することで，利用者は，いつでも，どこからでも，データにアクセスできるという利点があるが，保管されている情報が漏洩する可能性がある．著者は，秘密計算の基礎に加えて，クラウドの利点とプライバシー保護の両立ができる秘匿検索に関する研究も行っている．これまでに研究してきた秘密計算を活用し，最小 2 台のサーバで，低運営コストと高速な検索処理を実現する．これによって，IoT デバイスなどから収集したデータを秘密にしたままの解析だけでなく，クラウドに保管してあるデータを安全に検索でき，AI（人工知能）など種々の利活用が安価・安全・高速に実現できるようになる．

7. おわりに

近年，計算機やネットワークの性能向上に加え，秘密計算方法自体の高速化の研究も活発的に行われ，NTT の MEVAL や，東京理科大学の TUS 方式など，実世界の応用を目指す秘密計算法も多数提案されている．さらに，近年，AI の進化により，日常生活で取得・活用される，個人に関わる，または機密性の高いデータが増加し，データの活用とプライバシー保護の両立ができる秘密計算は不可欠である．このことから秘密計算は，今後のさらなる展開や様々な分野への活用が期待される技術であると考えられる．

謝辞

原稿にコメントをくださった，本学名誉教授 岩村恵市先生，情報工学科の藤沢匡哉先生，稲葉のぞみ氏，ならびに校正作業にご尽力いただきました本学広報課の皆様に厚く感謝を申し上げます．

[参考文献]

1）M. Ben-Or, S. Goldwasser, A. Wigderson, “Completeness theorems for non-cryptographic fault-tolerant distributed computation,” in Proceedings of ACM STOC ’88, pp. 1−10, 1988.

2）I. Damgård, *et al*. “Multiparty computation from somewhat homomorphic encryption,” in: Safavi-Naini, R., Canetti, R.（eds.）CRYPTO 2012. LNCS, vol. 7417, pp. 643−662, 2012.

3）A. Kamal, K. Iwamura, “Privacy preserving multi-party multiplication of polynomials based on（k,n）threshold secret sharing,” ICT Express, vol. 9, no. 5, pp. 875−881, 2023

4）K. Iwamura, A. Kamal, “Communication-efficient secure computation of encrypted inputs using（k, n）threshold secret sharing,” IEEE Access, vol. 11, pp. 51166−51184, 2023.