情報処理学会論文誌 Vol. 51 No. 9 1563–1578 (Sep. 2010)

秘密分散データベースの構造演算を可能にするマルチパーティプロトコルを用いた関係代数演算

志村正法[†1] 宮崎邦彦[†2]
西出隆志[†3] 吉浦裕[†1]

個人情報が電子化されネットワーク上で授受されるに従い，その漏洩が社会問題となっている．個人情報の漏洩には様々な形態があるが，なかでもデータベースからの漏洩は大量の個人情報が 1 度に漏洩するので，きわめて甚大な被害をもたらす．データベースからの情報漏洩対策として，秘密分散法および暗号を用いてデータが漏洩しても読めないようにする方法がある．しかしこれらの従来方法を採用した場合，JOIN 演算など，複数のテーブルにまたがる構造演算を行うことができなかった．本論文では，関係データベースの構造演算が関係代数によってモデル化されることに着目する．マルチパーティプロトコルを用いた関係代数演算の実行を可能にし，秘密分散法によって分散されたデータベース上で，データを 1 度も復号することなくすべての構造演算を可能とする．

Relational Algebra in Multi-party Protocol to Enable Structual Operation in Secret Shared Databases

Masanori Shimura,[†1] Kunihiko Miyazaki,[†2]
Takashi Nishide[†3] and Hiroshi Yoshiura[†1]

As personal information comes to be in digital and transferred on networks, its leakage is becoming more and more serious social problem. Among various ways of personal information leakage, the leakage from databases is most serious because databases store vast amount of personal information. Methods of making data unreadable even if they have been copied outside are therefore studied actively using secret sharing and cryptography. With these previous methods, however, legal queries are limited, i.e., structural operations over multiple tables (such as JOIN) are impossible. In this paper, we take into account the fact that structural operations of relational databases are modeled by relational algebra. We then propose a method that can execute relational algebra in a multi-party protocol and thus can perform any structural operation over secret-shared databases without restoring plain text data.

1. はじめに

個人情報が電子化されネットワーク上で授受されるに従い，その漏洩が社会問題となっている．個人情報の漏洩には様々な形態があるが，なかでもデータベースからの漏洩は大量の個人情報が 1 度に漏洩するので，きわめて甚大な被害をもたらす．データベースからの情報漏洩対策として，秘密分散法および暗号化を用いてデータが漏洩しても利用できないようにする方法が提案されている[2)–5)]．秘密分散法を用いた方式では，データの復元に必要な数のシェアが漏洩しない限りデータは保護される．暗号化を用いた方式では，暗号化データと鍵が同時に漏洩しない限りデータは保護される．

しかし，従来の秘密分散法および暗号化を用いたデータベース保護方式は，関係データベースの本質であるテーブルの構造演算が検討されていなかった．関係データベースでは 1 つのテーブルにデータを格納するのではなく，複数のテーブルにデータを分散して格納することが多い．検索時には複数のテーブルを組み合わせた構造演算を行うことで，中間結果および，最終結果のテーブルを作成する．従来方式では 1 つのテーブルから 1 つのレコードを取り出すといった単純検索はできるが，上記のようなテーブルの構造演算はできなかった．

筆者らの研究の目的は，データベースサーバ内で 1 度もデータを復元することなく，テーブルの構造演算をすべて可能にすることである．関係データベースのすべての構造演算は関係代数によって数学的にモデル化されている．そこで提案方式では，秘密分散型のデータベースを基礎とし，その分散データベースサーバ上で関係代数のすべての演算をマルチパーティプロトコルで実行する．これにより，データベース内でデータを復元せずに任意の構造演算が可能となる[⋆1]．2 章ではデータベースの利用と攻撃について述べる．3 章では従来のデータベース保護方式について分析する．4 章では目標と方針をあげる．5 章では提案方式について述べる．6 章では関係代数による演算をマルチパーティプロトコルで実現するための関数の設計を行う．7 章では，想定する利用シーンの一例を示す．8 章で提案方式の評価

†1 電気通信大学
University of Electro-Communications
†2 株式会社日立製作所
Hitachi Ltd.
†3 九州大学
Kyushu University

⋆1 本論文の基本アイデアは第 42 回コンピュータセキュリティ研究発表会で提案した[1)]．本論文はこれを発展させたものである．

を行い，9 章で結論を述べる．

2. データベースの利用と攻撃

2.1 データベースの利用

関係データベースは，ユーザから見ると複数のテーブルによって構成されている．**図 1** の例では，データベースは「学生基本情報」,「指導教員」,「数学の成績」,「英語の成績」の 4 つの仮想的なテーブルによって構成されている．これらの仮想的なテーブルの 1 つの行は，1 つの対象に関するデータを表しており，レコードと呼ばれる．たとえば,「学生基本情報」の 1 つの行は，1 人の学生の基本情報を表すレコードである．仮想テーブルの列は対象の属性を表し，個々のレコードには属性の値が格納されている．たとえば「学生基本情報」の列は，ID，氏名，クラスなどの学生の属性を表し，個々のレコードには，ID，氏名，クラスの属性値が格納されている．なお，以下では，仮想テーブルのことを単にテーブルと呼ぶことにする．

データベースへの問合せは以下の 4 種類に大別される．本研究では③を可能にする．

①単純取出

指定したテーブル内のデータをレコード単位で取り出す．たとえば，ユーザは ID が 001 の学生基本情報を要求する．データベースは「学生基本情報」から属性「ID」の属性値が 001 であるレコードをユーザに返す．

②数値演算

1 つのテーブル内の数値データを用いて演算する．たとえば，ユーザは学生全員の数学の平均点を要求し，データベースはテーブル「数学の成績」の属性「点数」から平均点を演算し，結果をユーザに返す．

③構造演算

1 つあるいは複数のテーブルから新しいテーブルを構成する．たとえば，ユーザは数学の成績が D である学生とその指導教員に関するデータを要求する．データベースはテーブル「数学の成績」と「学生基本情報」を共通する属性「ID」で結合させ，属性「ID」,「氏名」,「クラス」,「点数」,「評価」からなる 1 つのテーブルを作成する．そのテーブルから属性「評価」の属性値が D であるレコードを取り出したテーブルと，テーブル「指導教員」を共通する属性「クラス」で結合させ，属性「ID」,「氏名」,「クラス」,「点数」,「評価」,「教員名」からなる 1 つのテーブルを作成する．そのテーブルから「ID」,「氏名」,「評価」,「教員名」を取り出したテーブルを作成し，ユーザに返す．

④数値演算と構造演算の組合せ

たとえば，ユーザは学生全員の数学と英語の成績，および両科目の合計点を要求する．データベースはテーブル「数学の成績」と「英語の成績」を属性「ID」で結合し，さらに数学と英語の合計点を計算して，属性「合計点」を加えたテーブルを作成し，ユーザに返す．

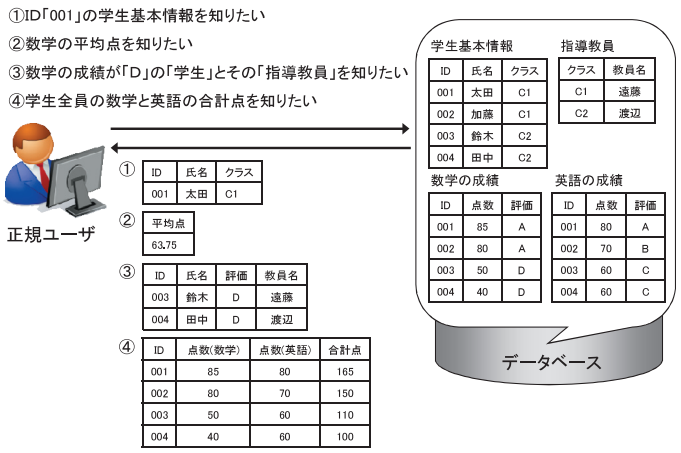

図 1　データベースの利用例
Fig. 1　Uses of database.

従来の方式では，①の単純取出しは可能である．②の数値演算については暗号化を用いた Privacy preserving data mining 手法をデータベースに応用することで解決の可能性がある[6]．③の構造演算，およびそれを用いた④が従来検討されていなかった．本研究では，③の構造演算を対象にする．

2.2 データベースへの攻撃と対策

データベースへの攻撃には，外部への漏洩以外に破壊，改ざんや DOS 攻撃など様々な形態があるが，本論文では外部への漏洩について議論する．ここでの関与者は外部不正者，内部不正者，管理者，正規ユーザである．

外部不正者は正規ユーザになりすまし，許可されないデータを取得する攻撃を行う（**図 2** ①）．これに対しては，認証などの技術的対策やパスワード管理などの人的対策がとられる[7]．また，データベースサーバに侵入し，データを盗む攻撃を行う（図 2 ②）．これに対しては，侵入検知システムやファイヤウォールなどの対策がとられる[7]．また，侵入後に盗難された

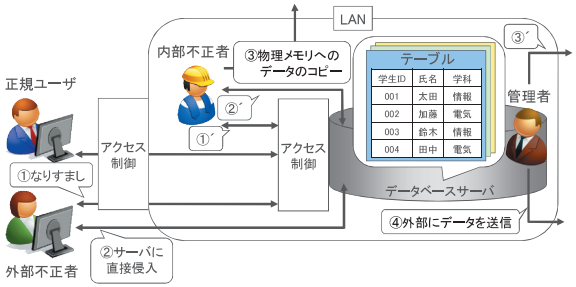

図 2　データベースへの攻撃
Fig. 2　Attacks to database.

データを利用できないようにするため秘密分散法や暗号化を用いた対策がとられている．これらの対策については 3 章で詳しく述べる．

内部不正者は，管理者になりすます場合（図 2 ①′）と，LAN 経由で侵入する場合（図 2 ②′）があるが，これらに対しては①，②と同様の対策がとられる．また USB などの外部メモリにデータを物理的にコピーする場合がある（図 2 ③）．①，②への対策に加え，ログ管理による事後的対策やセキュリティ教育などの人的対策がとられている．

管理者は USB などの外部メモリにデータを物理的にコピーする場合（図 2 ③′）とネットワーク経由で外部にデータを送信する場合がある（図 2 ④）．①〜③の対策に加え，電子メールチェックシステムなどの技術的対策がとられる[8]．

本研究では，外部不正者と内部不正者がデータベースサーバに直接侵入場合（②，②′），内部不正者と管理者が USB などの外部メモリにデータを物理的にコピーする場合（③，③′），管理者が外部にデータを送信する場合（④）を対象とする．

3. 先 行 研 究

3.1　秘密分散法と暗号化によるデータベース保護方式

文献 2) において，(k, n) 秘密分散法を用いたデータベース方式が提案されている．データベースは秘密分散されているが，データベースサーバには各レコードの ID を平文で保持している．ユーザは検索したいレコードの ID を各データベースサーバに送信する．ユーザはデータベースサーバから送信されたシェアを集め，元のデータを復元する．この方式では 2.1 節の①単純取出しは可能となるが，③構造演算は考慮されていない．

文献 3) においても，(k, n) 秘密分散法を用いた方式が提案されている．この方式では，平文テーブルをデータベースサーバ内で一時的に復元して検索を行う．すべての演算が可能となるが，サーバ内に平文のテーブルが生成されるため，2.2 節で述べたデータベースへの攻撃②，②′，③，③′ に弱い．

表 1　各方式のまとめ
Table 1　Features of previous methods.

方式	秘密分散方式		暗号化方式	ノイズ方式	提案方式の目標
参考文献	[2]	[3]	[4],[5]	[10]	
構造演算	×	○	×	×	○
復元不要性	○	×	○	○	○

暗号化されたデータベースに対する検索方法が文献 4) によって提案されている．この方式ではレコード単位でデータを暗号化し，それぞれの暗号化データに索引を付与する．索引は，当該レコードに格納されている個々の属性値をより粗い区間値に変換したものである．たとえば，テーブル「社員管理」に属性「月収」があるとする．このテーブルの 1 つのレコードにおける属性「月収」の属性値が 25 万である場合に，この属性値を 10 万〜30 万という区間値に変換し，索引とする．ユーザの検索に対しては，ユーザが指定した属性値（たとえば月収が 15 万〜20 万）を，各レコードの索引である区間値と比較することにより，結果の候補を返す．最終的な結果を得るためには，ユーザ側で候補となるレコードを復元した後に，再度検索をする必要がある．この方式でも暗号化したままで①単純取出しは可能だが，③構造演算は検討されていない．文献 9) では，暗号化を用いることでデータの機密性は保証されるが，JOIN などの構造演算が制限されることが述べられている．文献 5) ではテーブル内のデータの位置情報（行，列）に依存した鍵を生成し，その鍵でデータを暗号化することで保護している．この方式も単純取り出し以外は考慮されていない．

文献 10) ではデータにノイズを加えることで保護している．この方式は，一定の誤差範囲内で平均値や統計値を得ることができるが，構造演算は検討されていない．

各方式が達成できる機能と安全性について**表 1** にまとめる．ここで復元不要性とは，サーバ側でデータの復元が不要であることである．上述のように文献 3) では検索時に一時的に平文テーブルを復元するため，復元不要性を満たさない．従来，構造演算の機能と復元不要性を同時に達成できる方式はなかった．筆者らの研究の目標は構造演算と復元不要性を同時に達成する方式を提案することである．

表 2　科目 A の履修者
Table 2　Learners of subject A.

学生ID	氏名
001	太田
002	加藤
003	鈴木

表 3　科目 B の履修者
Table 3　Learners of subject B.

学生ID	氏名
001	太田
002	加藤
004	田中

表 4　和
Table 4　Union of table 2 and 3.

学生ID	氏名
001	太田
002	加藤
003	鈴木
004	田中

表 5　差
Table 5　Difference.

学生ID	氏名
003	鈴木

3.2 前提技術

3.2.1 関係データベース

2.1 節で述べたように，関係データベースは複数の仮想的なテーブル（「テーブル」と略記）によって構成されている．テーブルは $A(A_1, A_2, \ldots, A_n)$ で表される．ここで A はテーブル名で，個々の $A_1, A_2, \ldots, A_n$ はテーブル A の属性であり，それぞれ $1, 2, \ldots, n$ 列目である．それぞれの属性 $A_1, A_2, \ldots, A_n$ に格納できる属性値の集合を，その属性のドメインと呼ぶ．テーブル A は行すなわちレコードの集合と見なされ，$t \in A$ とは t がテーブル A の 1 つのレコードであることを示す．$t[A_j]$ とは任意のレコードの j 番目の属性の値を示す．

3.2.2 関係代数

関係代数とはテーブルに対する演算方法を定式化した代数であり，一般的な集合演算とのアナロジーにより定義される．関係データベースに対するすべての構造演算は関係代数によって定義される[11]．

和，差，共通，直積と，関係代数に特有な射影，選択，結合，商の計 8 つの演算子により定義される．和，差，共通は和両立を満たしている場合に演算が可能である．テーブル $A(A_1, A_2, \ldots, A_n)$ とテーブル $B(B_1, B_2, \ldots, B_m)$ が和両立とは，次の 2 つの条件を満たしていることをいう．

- A と B の列数が等しい $(n = m)$
- 各 $j(1 \leq j \leq n)$ について A_j と B_j のドメインが等しい

例として，和，差，結合について説明する．残りの演算は付録に記述する．ここで，**表 2** と**表 3** は和両立である．

①和

A，B の和 $A \cup B$ は，以下のように定義される．

定義：$A \cup B = \{t | t \in A \vee t \in B\}$

A のすべてのレコードと B のすべてのレコードで重複なく構成されるテーブルを返す．表 2 と表 3 の和を**表 4** に示す．

②差

A，B の差 $A - B$ は，以下のように定義される．

定義：$A - B = \{t | t \in A \wedge \neg (t \in B)\}$

A から，B に属するレコードを取り除いたテーブルを返す．表 2 と表 3 の差を**表 5** に示す．

③結合

テーブル $A(A_1, A_2, \ldots, A_n)$ とテーブル $B(B_1, B_2, \ldots, B_m)$ において，A_i のドメインと B_j のドメインの間に成り立つ任意の関係を θ とする．θ は比較演算子 $(=, \neq, <, >\leq, \geq)$ である．ここで A_i と B_j が θ 比較可能とは，次の 2 つの条件を満たすときである．

- A_i のドメインと B_j のドメインが等しい．
- A の任意のレコード t の A_i 属性と B の任意のレコード u の B_j 属性に対して，

表 6　学生
Table 6　Students.

学生ID	氏名	所属学科ID
001	太田	d001
002	加藤	d002
003	鈴木	d002
004	田中	d003

表 7　学科
Table 7　Departments.

学科ID	学科名	所属学部
d001	機械	工学部
d002	電気	工学部
d003	情報	工学部

表 8　表 6 と表 7 の結合（学生 [所属学科 ID]=学科 [学科 ID]）
Table 8　Join of tables 6 and 7.

学生ID	氏名	所属学科ID	学科名	所属学部
001	太田	d001	機械	工学部
002	加藤	d002	電気	工学部
003	鈴木	d002	電気	工学部
004	田中	d003	情報	工学部

$t[A_i]\theta\, u[B_j]$ の真偽が常に定義されている．

このとき結合 $A[A_i\theta B_j]B$ は以下のように定義される．

定義：$A[A_i\theta B_j]B = \{(t,u)|t \in A \wedge u \in B \wedge t[A_i]\theta\, u[B_j]\}$

A と B において関連する属性に注目し，2 つのテーブルを合わせて 1 つのテーブルを返す．**表 6** と**表 7** の結合を**表 8** に示す．これは表 6 の学生［所属学科 ID］と表 7 の学科［学科 ID］が等号となるときの結合である．

関係代数 8 つの演算子のうち，和，差，直積，選択，射影の演算子は互いに独立である．この 5 つの独立演子を組み合わせることによって共通，結合，商を表現できる（付録 A.1）．次章以降は，この 5 つの独立演算子について考える．

3.2.3　秘密分散法とマルチパーティプロトコル

秘密分散法には排他的論理和を用いた (n,n) 方式と多項式補間を用いた (k,n) しきい値方式がある[12]．(n,n) 方式では，元の情報を復元するためにすべての分散情報が必要であり，1 つでも紛失してしまうと復元できない．データベースを分散させる場合，破損などを考慮する必要があるため，ロバスト性が要求される．よって提案方式では，ロバスト性のある (k,n) しきい値方式を用いる[13]．秘密情報 s を定数項とするランダムな $(k-1)$ 次の多項式 $f(x)$ は以下の式で表される．

$$f(x) = s + a_1 x + a_2 x^2 + \ldots + a_{k-1} x^{k-1} \bmod p \tag{1}$$

このとき，秘密情報 s は $s = f(0)$ である．

管理者は各参加者 $U_i\,(i = 1,\ldots,n)$ に対して分散情報 $w_i = f(i)$ を配り，分散する．秘密情報 s は k 個の分散情報を集めることで復元することができる．

マルチパーティプロトコルとは，複数の参加者 $U_i\,(i = 1,\ldots,n)$ がそれぞれ秘密情報 s_i を持つ状況下で，秘密情報を秘匿したまま，秘密情報の関数値 $v = g(s_1,\ldots,s_n)$ を計算する方法である．式 (1) の (k,n) しきい値秘密分散法を用いた場合，各 U_i が自分の s_i を $w_i^1,\ldots,w_i^i,\ldots,w_i^n$ に分散し，$U_1,\ldots,U_n$ にそれぞれ分配する．各 U_i は $w_1^i,\ldots,w_i^i,\ldots,w_n^i$ を保持している状況下で，g に対応した関数 h を秘密情報を復元することなく計算する．すなわち，$h(w_1^1,w_1^2,\ldots,w_i^1,\ldots,w_i^j,\ldots,w_n^n) = (v^1,\ldots,v^i,\ldots,v^n)$ であり，$v^1,\ldots,v^i,\ldots,v^n$ は v の分散情報である．筆者らが知る限り，整数上の関数 g が加算，減算，乗算，大小比較，一致判定であるときに，$GF(p)$ 上の対応する関数 h として，加算プロトコル，減算プロトコル，大小比較プロトコル，一致判定プロトコルが存在することが分かっている[14)–17)]．そこで，整数上の加算，減算，乗算，大小比較，一致判定の組み合わせによって表される関数 g に対して，$GF(p)$ 上の対応するマルチパーティ関数 h を作ることができる．

4.　目標と方針

4.1　達 成 目 標

本研究で提案する方式では，以下の 3 つを同時に達成することを目標とする．

(i)　データベースにおけるすべての構造演算を行うことができる．

(ii)　復元不要性を満たす．

復元不要性とは，演算時にサーバ側でのデータの復元が不要なことである．上記を満たす

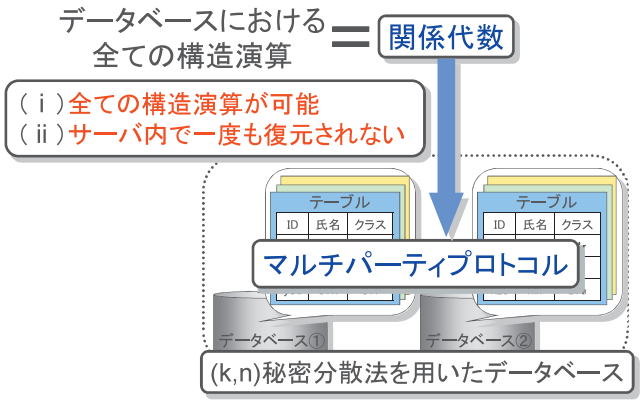

図 3　提案方式の基本となるアイデア
Fig. 3　Main ideas of proposed method.

自明な方式として，ユーザ側のコンピュータでテーブルを復元した後に検索を行う方法が考えられる．しかしユーザが検索対象以外の情報まで得られるという問題がある．よって上記 (i)，(ii) に以下の目標を加える．

(iii)　ユーザに返される結果は，ユーザが指定した検索結果のみであり，それ以外のデータベース内の情報はユーザに返されない．

4.2　基本アイデア

現在のデータベースの多くは関係データベース，およびその発展形である．関係データベースにおけるすべての構造演算は 3.2.2 項で述べた関係代数によって数学的にモデル化されている．上記の目標を達成するため，提案方式は (k,n) 秘密分散法を用いたデータベース方式を基礎とし，以下を方針とする（**図 3**）．

- (k,n) 秘密分散法を用いてデータをシェアに分散し，それぞれを個々のデータベースサーバに格納する．
- 個々のデータ・個々のデータベースサーバ管理者をそれぞれ，マルチパーティプロトコルの秘密情報・参加者と考える．
- 関係代数をマルチパーティプロトコルで実行可能にし，個々のデータベース管理者間のマルチパーティプロトコルによって演算する．

上記の演算により，サーバ内でシェアを 1 度も復元することなく，個々のシェアから関係代数を演算可能にし，構造演算を可能にする．

5. 提 案 方 式

5.1　データの格納

本方式では，関係データベースの仮想テーブルに格納するべき情報（たとえば図 1 の場合，学生の ID や英語の点数など）を，$Z_N - \{0\}$ すなわち N 以下の正整数で表現し，仮想テーブルの属性値として格納する．N は仮想テーブルの属性（列）によって異なる．なお，6 章の提案方式で，属性値が 0 のレコードを null として特別に扱うので，0 はデータの表現には使わないことにする．

3.2.3 項の (k,n) 秘密分散法を用いて，属性値をシェアに分散し，複数の関係データベースのテーブルに格納する．**図 4** は，$(2,3)$ 秘密分散法を用いてデータベースを 3 つのシェアに分散した例を示す．秘密分散後の値は，Z_p すなわち $\{0,1,2,\cdots,p-1\}$ の要素である．ここで，p は (k,n) 閾値法による秘密分散時に用いられる素数である．なお，p の値は，秘密分散前の仮想テーブルの属性値および検索処理の過程で現れる値のうち最大の値よりも大きい値とする．この条件を課する理由は，もし p が仮想テーブルの属性値および検索処理の過程で現れる値のうち最大の値以下であるならば，仮想テーブルの属性値や検索処理の過程で現れる値が mod p 演算によって変更される可能性があるからである．

秘密分散前の関係データベースと，秘密分散後の複数の関係データベースは下記のように対応する．

- 秘密分散前の関係データベース RDB は，秘密分散後の n 個の関係データベース $\mathrm{RDB}^1,\cdots,\mathrm{RDB}^h,\cdots,\mathrm{RDB}^n$ に対応する $(1 \le h \le n)$．ここで h は秘密分散データベースの番号を表す添字である．
- 秘密分散前の RDB のテーブル T は，秘密分散後の n 個のテーブル $T^1,\cdots,T^h,\cdots,T^n$ に対応する．テーブル T^h は，関係データベース RDB^h に格納される．
- 秘密分散前のテーブル T のレコード t_i は，秘密分散後の n 個のレコード $t_i^1,\cdots,t_i^h,\cdots,t_i^n$ に対応する．ここで t_i はテーブル T の i 番目のレコードであり，t_i^h はテーブル T^h の i 番目のレコードである．また，添字 i は 1 以上であり，T のレコード数以下である．
- 秘密分散前のレコード t_i の属性値 t_{ij} は，秘密分散後の n 個の属性値 $t_{ij}^1,\cdots,t_{ijj}^h,\cdots,t_{ij}^n$ に対応する．属性値 t_{ij}^h は，レコード t_i^h に格納される．ここで，t_{ij} はレコード t_i の j 番目の属性値であり，$t_{ij}^1,\cdots,t_{ij}^h,\cdots,t_{ij}^n$ は (k,n) 秘密分散法によって t_{ij} を分散したシェアである．添字 j は 1 以上であり，T の属性値数以下である．

上記の秘密分散データベースは，下記の手順によって生成することができる．

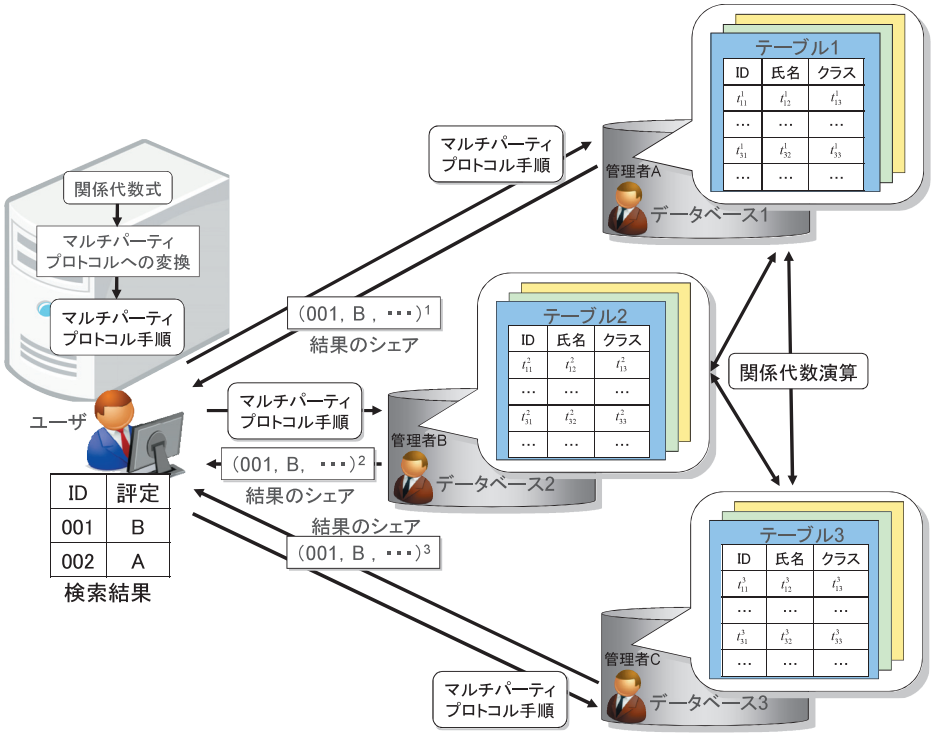

図 4　提案方式の概要
Fig. 4　Overview of proposed method.

- 関係データベース RDB のテーブル T について，T と同形のテーブル T^h を RDB^h 内に生成する．ここで同形とは行数と列数が等しく，格納するデータのドメインが等しいことである．T の i 行 j 列の属性値 t_{ij} を (k,n) 秘密分散法によって分散し，シェア t^h_{ij} を，T^h の i 行 j 列に格納する．以上を RDB のすべてのテーブル T のすべての属性値について実行する．

5.2　検　　索

(1)　ユーザは関係代数による任意の検索式（3.2.3 項における関数 g）を発行する．

(2)　この検索式はマルチパーティプロトコルで実行可能な手順（3.2.3 項における関数 h）に変換され，複数の分散データベースサーバに送られる．

(3)　データベースサーバはマルチパーティプロトコルによる関係代数演算を行い，検索結果のシェアを導出する．

(4)　ユーザは検索結果のシェアを取得し，検索結果を復元する．

(1)〜(3) によって 4.1 節の目標 (i)，(ii) を達成することができ，(4) によって目標 (iii) を達成することができる．

提案方式では検索式をマルチパーティプロトコルで実行可能な手順で記述するために，3.2.2 項の 5 つの独立演算子の関数を設計する必要がある．

6.　マルチパーティプロトコルによる関係代数演算

関係代数による検索式 g をマルチパーティプロトコルで実行可能な関数 h に変換する必要がある．そのためには検索式 g を 3.2.3 項で述べた，加算，減算，乗算，大小比較，一致判定によって表すことができればよい．

6.1　定　　義

- テーブル T は n_t レコード m_t 属性（すなわち n_t 行 m_t 列）であり，i 番目のレコードの j 番目の属性値を t_{ij} と表す．
- $[a=b]$ は，a と b の一致判定を行う．a と b が一致した場合には 1 を返し，不一致の場合には 0 を返す．
- $[a>b]$ は，a と b の大小比較を行う．a が b よりも大きい場合には 1 を返し，a が b 以下のときは 0 を返す．

6.2　1 レコード比較関数（comp_1line）

テーブル A の 1 レコードとテーブル B の 1 レコードが等しいかどうかを判定する関数を提案する．これは 3.2.3 項で述べた関数 g に相当するものである．このプロトコルは，3.2.2 項の和両立が前提となる．すなわち，$m_A=m_B=m$ が成立し，対応する属性のドメインが等しい．このとき 1 レコードの比較は以下のように行う．

入力：テーブル A の i_A 番目のレコード $a_{i_A,j}$（$1\le j\le m$）
　　テーブル B の i_B 番目のレコード $b_{i_B,j}$（$1\le j\le m$）

comp_1line
{
　$Sum=0$
　for$(1\le j\le m)$\{
　　$Sum=Sum+(1-[a_{i_A,j}=b_{i_B,j}])$

}
}
Sum を出力
$Sum = 0$：2 つのレコードの一致を表す．
$1 \leq Sum \leq m$：2 つのレコードの不一致を表す．

以上において，行どうしの比較を行う関数 g がマルチパーティプロトコルで実行可能な加算，減算，一致判定を用いて表される．これは 3.2.3 項で述べた関数 h に変換して実行される．関数 h において Sum の値（0，正値）は直接出力されるのではなく，そのシェアが出力される．次章ではこれに，乗算，大小比較を加えて設計し，5 つの独立演算子をマルチパーティプロトコルで実行可能な手順に設計する．

6.3　独立演算子の設計

$comp\_1line$ を用いて，差，和，直積，選択，射影について，マルチパーティプロトコルで実行可能な関数を提案する．

①差プロトコル

テーブル A とテーブル B の差 $A - B$ の結果をテーブル C へ出力する．ここで，C の行数は A と同じになる．また，C のある 1 レコードのすべての属性値が 0 であるとき，そのレコードは null として扱うことにする．

for($1 \leq i_A \leq n_A$){
$Mul = 1$
for($1 \leq i_B \leq n_B$){
$comp\_1line(a_{i_A,j},\ b_{i_B,j})$

A の i_A レコードと B の i_B レコードが一致した場合は $Sum = 0$ となり，不一致した場合は $Sum =$ 正値 となる．

$Mul = Mul \times Sum$

A の i_A レコードが B のどのレコードとも異なる場合，$1 \leq Mul \leq m^{N_B}$ となる．一方，A の i_A レコードが B のいずれかのレコードと一致した場合は $Mul = 0$ となる．

}
for($1 \leq j \leq m$){
$C_{i_A,j} = (1 - [Mul = 0]) \times a_{i_A,j}$

A の i_A レコードが B のどのレコードとも異なった場合，C の i_A レコードに A の i_A レコードが出力される．A の i_A レコードが B のいずれかのレコードと一致した場合，C の i_A レコードが null となる．3.2.2 項の表 2 と表 3 の例を用いると，**表 9** のように出力される．

表 9　出力結果の例
Table 9　Output for difference between tables 2 and 3.

学生 ID	氏名
null	null
null	null
003	鈴木

}
}

(k, n) 秘密分散に用いる素数 p の値は Mul の上限値 m^{N_B} 以下であってもかまわない（m より大きければいい）．なぜならば，p が m より大きい素数であれば，6.2 節の Sum は 0 以上 m 以下の整数となり，p と Sum は互いに素となる．Mul は複数回算出された Sum の積であるので，p と互いに素である．したがって，mod p 演算を加える前の Mul がゼロでなければ，mod p 演算を加えてもゼロにはならない．以上により，マルチパーティプロトコル関数 h の実行中に，Mul の値が p を超え mod p 演算を受けても，Mul がゼロか非ゼロかは変わらず，出力 C_{ij} の値は影響を受けないからである．

②和

Step1：テーブル A をテーブル C にコピーする．
for($1 \leq i \leq n_A$){
for($1 \leq j \leq m$){
$c_{i,j} = a_{i,j}$
}
}
Step2：$B - A$ を，差を用いて C に追加する
for($1 \leq i_B \leq n_B$){
$Mul = 1$
for($1 \leq i_A \leq n_A$){
$comp\_1line(b_{i_B,j}, a_{i_A,j})$
$Mul = Mul \times Sum$
}
for($1 \leq j \leq m$){

$c_{n_A+i_B,j} = (1 - [Mul = 0]) \times b_{i_B,j}$
}
}
③直積
- テーブル A はレコード m_A 属性，テーブル B は n_B レコード m_B 属性とする．
- 直積の結果のテーブル C は $n_A \times n_B$ レコード，$m_A + m_B$ 属性．

for($1 \le i_A \le n_A$){
for($1 \le i_B \le n_B$){
for($1 \le j_A \le m_A$){
$c_{(i_A-1)n_B+i_B,j_A} = a_{i_A,j_A}$
}
for($1 \le j_B \le m_B$){
$c_{(i_A-1)n_B+i_B,m_A+j_B} = b_{i_B,j_B}$
}
}
}
④選択
(i) 属性 l_1 と属性 l_2 の一致判定
テーブル A において，属性 l_1 の値と属性 l_2 の値が一致する行を取り出す．
for($1 \le i \le n_A$){
for($1 \le j \le m$){
$c_{i,j} = [a_{i,l_1} = a_{i,l_2}] \times a_{i,j}$
}
}
(ii) 属性 l と定数 α の一致判定
テーブル A の l 属性の値が，ある定数 α に一致するレコードを取り出す．
for($1 \le i \le n_A$){
for($1 \le j \le m$){
$c_{i,j} = [a_{i,l} = \alpha] \times a_{i,j}$
}
}
(iii) 列 l と列 l_2 の大小比較
テーブル A の属性 l_1 の値が属性 l_2 より大きいレコードを取り出す．
for($1 \le i \le n_A$){
for($1 \le j \le m$){
$c_{i,j} = [a_{i,l_1} > a_{i,l_2}] \times a_{i,j}$
}
}
(iv) 属性 l と定数 α の大小比較
テーブル A の l 属性の値が，ある定数 α より大きいレコードを取り出す．
for($1 \le i \le n_A$){
for($1 \le j \le m$){
$c_{i,j} = [a_{i,l} > \alpha] \times a_{i,j}$
}
}
⑤射影
テーブル A から指定された列を取り出す．テーブル A から取出す列のリストを $L = \{l_1, l_2, \ldots, l_r\}(r < m)$ と表す．たとえば 1，3，4 列目を取り出すとき，$L = \{1, 3, 4\}$ である．
for($1 \le i \le n_A$){
for($1 \le j \le r$){
$c_{i,j} = a_{i,lj}$
}
}

7. 想定する利用シーン

本章では，想定する利用シーンの一例を示す．筆者らの所属する学科では，学部 3 年生に対して就職指導を行う．5 名の就職指導教員が各学生に対して，11 月と 2 月の 2 回にわたって面談を行い，その記録，学業成績，本人の希望に基づいて指導する．指導する 3 年生の人数は 1 学年 70 名である．この指導のために，**図 5** に示す就職指導用データベースを利用する．

テーブル 1 は学生基本情報である．テーブルのレコード数は 70 名，各レコードは学生

テーブル1　学生基本情報

学生ID	姓	名	性別	年齢	電話番号

テーブル2　成績

学生ID	理系科目の平均点	文系科目の平均点	全科目の平均点

テーブル3　11月就職面談評価

出席学生ID	進路選択	進学への理解	就職への理解	総合評価

テーブル4　2月就職面談評価

出席学生ID	希望職種	業界分析	自己分析	総合評価

テーブル5　就職活動状況

学生ID	希望1	進捗1	希望2	進捗2	希望3	進捗3

図 5　就職指導用データベースのテーブル構成
Fig. 5　Database table structure for job finding advice.

ID，姓，名，性別，年齢，電話番号の属性を有する．テーブル 2 は成績であり，理系科目の平均点，文系科目の平均点，全科目の平均点の属性を有する．テーブル 3 は 11 月就職面談評価であり，レコード数は面談に出席した学生の数（50 名程度）である．各レコードは，出席学生 ID，進路選択，進学への理解，就職への理解，総合評価の属性を有する．進路選択は進学か就職かの二択とする．進学への理解，就職への理解，総合評価は各々5 段階評価とする．テーブル 4 は 2 月就職面談であり，出席学生 ID，希望職種，業界分析，自己分析，総合評価で構成される．希望職種は職種コードの番号である．業界分析，自己分析，総合評価は各々5 段階評価とする．テーブル 5 は就職活動状況であり，学生 ID，希望 1，進捗 1，希望 2，進捗 2，希望 3，進捗 3 で構成される．希望 i は第 i 希望の会社であり，会社コードを用いる．進捗 i は第 i 希望の会社に関する進捗状況であり，プレエントリから内定までの 8 過程のうちの 1 つである（$1 \leq i \leq 3$）．

これらの情報は個人情報を含むため，機密性が高い．そこで，テーブル 1〜5 を含む就職指導用データベースを，(2,3) 秘密分散し，3 つのサーバ上の分散データベースとして管理する（**図 6**）．3 つのサーバは，3 名の職員がそれぞれ管理している．ここでのユーザは 5 名の就職指導教員であり，6 章の提案方式によって検索を行う．以下検索の例を示す．

検索例 1：第 1 希望の内定が取れている学生について，学生の基本情報と内定状況を合わせた情報を取り出す．

1．テーブル 5 を「進捗 1」の値「8」で選択し，中間結果とする．
2．中間結果をテーブル 1 と「学生 ID」で結合し，中間結果とする．

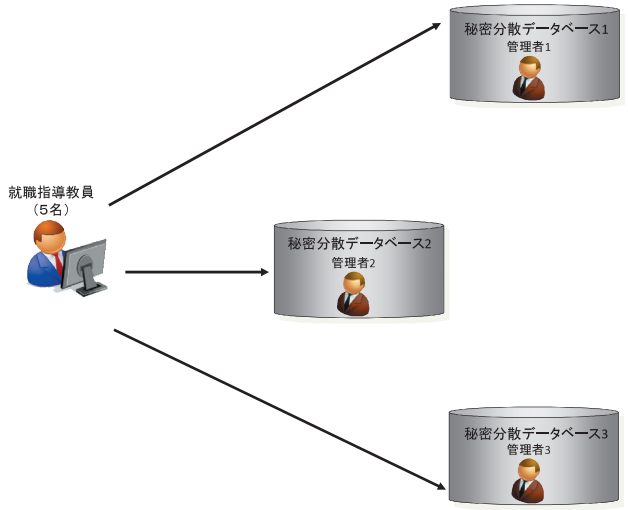

図 6　就職指導用データベースのシステム構成
Fig. 6　Database system structure for job finding advice.

検索例 2：成績の悪い学生の就職活動状況を取り出す．

1．テーブル 2 を「全科目の平均点」で選択し，その中間結果を出力する．ここでは選択の条件として，「60 点未満」とする．
2．中間結果とテーブル 5 を「学生 ID」で結合し，最終結果を出力する．

検索例 3：就職面談を 1 度も受けていない学生リストを取り出す．

1．テーブル 3 を「出席学生 ID」で射影し，中間結果とする．
2．テーブル 4 を「出席学生 ID」で射影し，中間結果とする．
3．1 の中間結果と 2 の中間結果の和をとり，3 の中間結果とする．
4．テーブル 1 を「学生 ID」で射影し，4 の中間結果とする．
5．4 の中間結果と 3 の中間結果の差（中間結果 4 − 中間結果 3）をとり，最終結果を出力する．

8. 評　　価

8.1 達成目標の評価

4.1 節の目標 (i)〜(iii) について評価する．

(i) 関係代数の 5 つの独立演算子をマルチパーティプロトコルで実行可能にし，すべての構造演算が可能となった．

(ii) 秘密分散されたシェアのまま演算を行うため，データベースサーバ上で 1 度も復元されない．

(iii) ユーザに返される結果は，ユーザが関係代数によって指定した検索結果のシェアのみである．それ以外のデータベース内の情報はユーザに返されない．

8.2 安　全　性

2.2 節で述べた外部不正者，内部不正者の直接侵入する場合（図 2 ②，②′），外部不正者と管理者が USB などの外部メモリにデータを物理的にコピーする場合（図 2 ③，③′），管理者が外部にデータを送信する場合（④）について安全性を評価する．

- 外部不正者，および内部不正者がデータベースサーバに直接侵入してデータを盗むためには，k 個以上のデータベースサーバに侵入する必要がある．
- 内部不正者，および管理者が外部メモリにデータを物理的にコピーして盗むためには，k 個以上のデータベースサーバからデータをコピーする必要がある．
- 管理者が外部にデータを送信して漏洩させるためには，k 個以上のデータベースサーバから送信する必要がある．

8.3 提案方式のコスト評価

6.3 節で提案した関係代数のマルチパーティ計算についてコストを評価する．提案方式は，加算，減算，乗算，大小比較，一致判定のプロトコルによって構成されている．そのうち，加算プロトコルおよび減算プロトコルは，参加者がローカルにシェアの加減算を行うので，そのコストは乗算プロトコルに比べると無視できるほど小さい．大小比較プロトコルおよび一致判定プロトコル[16),17)] のコストは，乗算プロトコルに分解して評価されている．そこで，提案方式のコストを下記の手順で評価する．

- 提案方式が乗算，大小比較，一致判定のプロトコルを各々何回含むか，について評価する．
- 文献 17) に基づき，大小比較および一致判定プロトコルのコストを乗算プロトコルに分解することにより示す．
- 乗算プロトコル[15)] の計算量を具体的に示す．

上記の手順をふまえて，提案方式の計算量を具体的に考察する．

8.3.1 差プロトコル

（1）乗算プロトコルと一致判定プロトコルへの分解

6.2 節および 6.3 節①のアルゴリズムは可読性を重視して単純化したアルゴリズムであるが，下記のように改良することができる．

・comp_1line の改良

$comp\_1line$
{
　$Sum = 0$
　for$(1 \le j \le m)$\{
　　$Sum = Sum + (a_{i_A,j} - b_{i_B,j}) \times (a_{i_A,j} - b_{i_B,j})$
　\}
}

・差プロトコルの改良

for$(1 \le i_A \le n_A)$\{
　$Mul = 1$
　for$(1 \le i_B \le n_B)$\{
　　$comp\_1line(a_{i_A,j},\ b_{i_B,j})$
　　$Mul = Mul \times Sum$
　\}
　$Temp = 1 - [Mul = 0]$
　for$(1 \le j \le m)$\{
　　$c_{i_A,j} = Temp \times a_{i_A,j}$
　\}
}

$comp\_1line$ の改良アルゴリズムは，(k, n) 閾値秘密分散に用いた素数 p が Sum のとりうる最大値より大きければ，6.2 節の改良前と同じ動作をする．差プロトコルの改良では，$[Mul = 0]$ の判定を for $(1 \le j \le m)$ の外に出した．この改良アルゴリズムにおける乗算プロトコルと一致判定プロトコルの回数は以下のとおりである．

- 乗算プロトコル $n_A n_B m + n_A n_B + n_A m = n_A(n_B m + n_B + m)$
- 一致判定プロトコル n_A 回

（2）一致判定プロトコルのコスト

文献 17) によると，一致判定プロトコルは，乗算プロトコル $2L$ 回に分解される．ここで，L は素数 p のビット数である．

（3）乗算プロトコルのコスト

(k,n) 閾値法によって分散された秘密データ上の乗算プロトコル[15] について考える．乗算プロトコルが可能であるためには $n \geq 2k-1$ であること，シェアを受け取った n 人のうち少なくとも $2k-1$ 人がプロトコルの実行に参加すること，が必要である．ここでは最小構成である $2k-1$ 人の参加を前提とする．また，この $2k-1$ 人はシェアを受け取った n 人のうち 1 から $2k-1$ 番目のパーティであることを，一般性を失うことなく，前提とする．

乗算プロトコルにおける参加者 i の処理の概要を下記に示す $(1 \leq i \leq 2k-1)$．なお，乗算したい 2 つの数 a，b は (k,n) 閾値法によって分散され，参加者 i は事前にシェア a^i，b^i を持っているとする．

① a^ib^i をローカルに計算する．

② a^ib^i を (k,n) 閾値法によって分散し，参加者 j へのシェア $(a^ib^i)^j$ を得る $(1 \leq j \leq n)$．

③ $(a^ib^i)^j$ のうち，$(a^ib^i)^i$ を自身が保持し，他は参加者 j に送る．

④ 自身が生成した $(a^ib^i)^i$ と，他の参加者 j から受信した $(a^jb^j)^i$ からなる $(2k-1)$ 次元ベクトル v と，$2k-1$ 次元のラグランジュ係数ベクトルとの内積 ip を計算する．この ip が $(ab)^i$ すなわち a と b の積のシェアとなる．なお，ラグランジュ係数ベクトルは事前に計算し再利用することが可能であり，乗算プロトコルのたびに計算する必要はない．

（3.1）乗算プロトコルの計算量 $MulCP(k,n)$

1 人の参加者のローカルな乗算回数によって評価する．①のローカル乗算回数は 1 回である．②は，(k,n) 閾値法の中で n 個のシェアを計算する．これは $(k-1)$ 次多項式をホーナー法により n 回評価するので，乗算回数は $n(k-1)$ である．④の乗算回数は $(2k-1)$ である．以上より，

$$MulCP(k,n) = n(k-1) + (2k-1) + 1$$

（3.2）乗算プロトコルの通信回数 $MulCM(k,n)$ と通信データ量

ネットワーク全体の通信量によって評価する．③では，$(2k-1)$ 人の参加者の各々 i が，$n-1$ 人の他の参加者 j に $(a^ib^i)^j$ を送る．以上より，

$$MulCM(k,n) = (2k-1)(n-1)$$

(k,n) 秘密分散に用いた素数を p とすると，$(a^ib^i)^j$ の値はたかだか p である．そこで，$(a^ib^i)^j$ のデータ量すなわち 1 回の通信データ量は $\lceil \log_2 p \rceil$ ビットとなる．

（4）提案した差プロトコルのコスト

上記（1）で提案方式を乗算プロトコル，一致判定プロトコルに分解した．(2) で一致判定プロトコルを乗算プロトコルに分解した．(3) で乗算プロトコルのコストを分析した．以上より，提案した関係代数演算の差プロトコルのコストは以下のとおりである．

・計算量

$$DifCP(k,n,n_A,n_B,m,L) = n_A(n_Bm + n_B + m + 2L)MulCP(k,n)$$
$$O(DifCP(k,n,n_A,n_B,m,L)) = n_An_Bmkn$$

・通信量

$$DifCM(k,n,n_A,n_B,m,L) = n_A(n_Bm + n_B + m + 2L)MulCM(k,n)$$
$$O(DifCM(k,n,n_A,n_B,m,L)) = n_An_Bmkn$$

（5）通常計算のコスト

6.2 節および 6.3 節①をマルチパーティプロトコルを用いずに実行した場合のコストは，乗算回数が $n_A(n_B+m)$，通信回数は 0 となる．なお，通常計算の場合は上記（1）より，改良前の方が乗算回数が少ないため，改良前のアルゴリズムで評価した．8.3.2 項以降の関係代数演算についても，通信回数は 0 である．

8.3.2　和

（1）提案した和プロトコルのコスト

8.3.1 項（1）で述べた *comp_1line* の改良アルゴリズムを用いる．また差プロトコルと同様に，$[Mul = 0]$ の判定を for（$1 \leq j \leq m$）の外に出した．この改良したアルゴリズムを用いた場合，和プロトコルでは以下のプロトコル実行を含む．

- 乗算プロトコル $n_B(n_Am + n_A + m)$ 回
- 一致判定プロトコル n_B 回

以上より，提案した関係代数演算の和プロトコルのコストは以下のとおりである．

・計算量

$$UniCP(k,n,n_A,n_B,m,L) = n_B(n_Am + n_A + m + 2L)MulCP(k,n)$$
$$O(UniCP(k,n,n_A,n_B,m,L)) = n_An_Bmkn$$

・通信量

$$UniCM(k,n,n_A,n_B,m,L) = n_B(n_Am + n_A + m + 2L)MulCM(k,n)$$
$$O(UniCP(k,n,n_A,n_B,m,L)) = n_An_Bmkn$$

（2）通常計算のコスト

乗算回数は，$n_B(n_A+m)$ となる．

8.3.3　直　　積

直積は，乗算プロトコル，一致判定プロトコル，大小比較プロトコルを用いない．よって本論文の評価手法ではコストは 0 となる．実際に直積はコピーのみであり，演算を含まない．

8.3.4 選　　択

(1) 提案した選択プロトコル (i) と (ii) のコスト

差プロトコルと同じように，6.3 節④ (i) は以下のように改良することができる．

for$(1 \leq i \leq n_A)$\{
　$Temp = [a_{i,l_1} = \ b_{i,l_2}]$
　for$(1 \leq j \leq m)$\{
　　$c_{i,j} = Temp \times a_{i,j}$
　\}
\}

このとき，選択プロトコルは以下のプロトコル実行を含む．

- 乗算プロトコル $n_A m$ 回
- 一致判定プロトコル n_A 回

以上より，提案した関係代数演算の選択プロトコル (i) のコストは以下のとおりである．

・計算量

$Sel\_1\_CP(k, n, n_A, m, L) = n_A(2L + m)MulCP(k, n)$

・通信量

$Sel\_1\_CM(k, n, n_A, m, L) = n_A(2L + m)MulCM(k, n)$

6.3 節④ (ii) も上記 (i) と同様である．

(2) 提案した選択プロトコル (iii) と (iv) のコスト

(1) と同様にアルゴリズムを改良することができる．このとき，選択プロトコルは以下のプロトコル実行を含む．

- 乗算プロトコル $n_A m$ 回
- 大小比較プロトコル n_A 回

文献 17) によると，大小比較プロトコルは，乗算プロトコル $3L - 4$ 回に分解される．以上より，提案した関係代数演算の選択プロトコル (iii) のコストは以下のとおりである．

・計算量

$Sel\_3\_CP(k, n, n_A, m, L) = n_A(3L - 4 + m)MulCP(k, n)$

・通信量

$Sel\_3\_CM(k, n, n_A, m, L) = n_A(3L - 4 + m)MulCM(k, n)$

6.3 節④ (iv) も上記 (iii) と同様である．

(3) 通常計算のコスト

6.3 節④ (i)〜(iv) を通じて，乗算回数は $n_A m$ となる．

8.3.5 射　　影

8.3.3 項と同様の理由により，射影のコストは 0 である．

8.4 既知の効率的アルゴリズムの利用についての考察

現在のデータベースでは，ハッシュテーブルや B 木を用いて検索を効率化している．そこで，これらの既知のアルゴリズムが 6 章の提案方式で実現可能であるかを考察する．具体例として，ハッシュテーブルの利用を取り上げる．

ハッシュテーブルによる検索の効率化について概要を説明する．データベースの任意のレコード（i 番目のレコード）を t_i，そのキー属性値（j 番目の属性の値）を t_{ij}，ハッシュ関数を H とする．また，レコード t_i の格納されたファイル上の場所（ページ）を P，P を含むいくつかのページの列（バケット）を B とする．このとき，B へのポインタがハッシュ値 $H(t_{ij})$ によって指定されるように，レコード t_i の記憶場所を定める．キー属性値が v であるようなレコードを検索する際には，$H(v)$ によってバケット B にアクセスし，B の中のページ P にアクセスして目的のレコードを読み出す．これにより，テーブルのすべてのレコードのキー属性値 t_{kj} を 1 つずつ v と照合することなく，目的のレコードを検索することができる．

6 章の提案方式にハッシュテーブルを取り入れる 1 つの方法としては，上記のハッシュテーブルアルゴリズムをマルチパーティ計算で実行し，提案方式に直接的に組み込むことが考えられる．その際，ハッシュテーブルアルゴリズムが加算，減算，乗算，大小比較，一致判定の組合せだけで実現できるかの検討，マルチパーティ計算のコストの検討が必要となる．また，マルチパーティ計算で実行する範囲と，ファイルシステムに任せる範囲の切り分けが必要である．

別の方法としては，3.1 節で述べた先行研究[2] の方式を提案方式に組み込むことが考えられる．すなわち，キー属性値だけは秘密分散せず，キー属性値 t_{ij} によってレコード t_i を検索する処理には，マルチパーティでなく従来の計算方法を用いる．たとえば，7 章の例題において学生 ID がキー属性とする．学生 ID は学内で公知であり，これを秘匿する必要はないことから，学生 ID の値だけは秘密分散することなく，各々の秘密分散データベースに平文値を格納する．学生 ID によるレコード検索は，上記のハッシュテーブルアルゴリズムを従来の計算方法によって実行する．この方法では，学生 ID 以外の属性値が漏洩しないかについて分析する必要がある．

8.5　提案方式の有用性に関する考察

7 章で述べた利用シーンを例とし，通常の計算と提案方式の計算とのコスト比較を通じて，提案方式の有用性を考察する．8.3 節で分析した関係代数演算のうち，一例として差演算を取り上げ，具体的にコストを比較する．

7 章の利用シーンでは，学生数が 70 であるから，テーブルのレコード数は $n_A = n_B = 70$ とする．テーブルの属性数 m は 4 から 7 であり，ここでは $m = 6$ とする．秘密分散のパラメータ k および n は各々2 および 3 である．L は，レコードの属性値および検索の過程で現れる値のうち最大の値を超える値とする必要がある．属性値の最大値を V とすると，$comp\_1line$（改良版）の Sum の最大値は，V の 2 乗の m 倍になり，これが検索処理の過程で現れる最大の値であるから，$L = \left\lceil \log_2 \left(V^2 \cdot m \right) \right\rceil$．学生の姓を 2 バイトコード 4 文字までとすると，8 バイト程度のデータ（2^{64}）となるので，$L = 131$ となる．

まず，計算量について比較する．通常計算の計算量は，8.3.1 項（5）で述べたように，乗算 $n_A\,(n_B + m)$ 回であり，上記のパラメータ値を代入すると 5,320 回となる．ただし，これはハッシュテーブルなどの効率化アルゴリズムを用いない場合のコストであり，効率化アルゴリズムを用いれば，より少ないコストになる．

提案方式については，8.3.1 項（4）で述べた評価式を用いると，371,420 回となる．以上から，提案方式の計算量は，ハッシュテーブルなどを用いない場合の通常計算と比べて，約 70 倍の増加となる．

次に通信量について考察する．通常計算では通信コストは発生しない．提案方式の通信コストは，8.3.1 項（4）で述べた評価式に上述のパラメータ値を代入することで求めると，318,360 回となる．1 回の通信データは 131 ビットであり，通信データの総量は 5.21 M バイトとなる（ただしパケットヘッダなどの付加情報は含んでいない）．

次に，提案方式の今後の効率化について考察する．効率化は以下の 3 層において可能である．

（1）提案方式自体の効率化

たとえば，上記 8.4 節で述べたハッシュテーブルの導入などが考えられる．

（2）部品となるプロトコルの効率化

乗算，一致判定，大小比較プロトコルの効率化が進んでいる．たとえば，文献 16) の大小比較プロトコルは乗算プロトコル（$354L + 6$）回相当であったが，文献 17) の大小比較プロトコルは $(3L - 4)$ 回相当である．このような部品の効率化を今後も取り入れる．

（3）実装レベルの効率化

文献 17) では，実装レベルの工夫により，一致判定および大小比較プロトコルを乗算プロトコル $\log_2 L$ ラウンドまで効率化できる可能性が示されている．

一方，提案方式の for ループは，基本的に独立な演算である．たとえば，レコード $a_{i_A,j}$ とレコード $b_{i_B,j}$ の一致判定と，レコード $a_{k_A,l}$ とレコード $b_{k_B,l}$ の一致判定は独立に実行できる．そのため，for ループを並列処理することで，高速化が可能である．また，1 回の通信データ量が 131 ビットであり小さいので，並列処理の通信を 1 回の通信にまとめ，通信回数を低減することが可能である．

以上の分析に基づき，提案方式の有用性について下記のように考える．

- 計算量，通信量が大きく，しかもテーブルサイズ（$n_A n_B m$）に比例することから，大規模データベースのリアルタイム検索は困難である．ハッシュテーブルや B 木の導入によりどこまで解決できるかは，現時点では判断できない．
- 7 章の利用シーンのような小規模データベースで，非リアルタイム検索であれば，提案方式を改良することで利用できる可能性がある．ただし，明確な見通しには実装評価が必要である．上述の様々な改良を重ねることで，データベースの規模とリアルタイム性において，適用範囲を拡大できる可能性がある．
- マルチパーティ計算による関係データベースの構造演算の可能性を示し，知見を示した点においての有用性は存在する．

9. 結　　論

データベースからの情報漏洩を防止する 1 つの有効な方法として，データを秘密分散し，複数のデータベースサーバ上に配置する方法があげられる．しかし，秘密分散法を用いた従来のデータベース保護方式では，関係データベースの本質的な処理であるテーブルの構造演算を行うことができなかった．本論文の提案方式では，関係データベースの構造演算が関係代数によってモデル化されることに着目した．秘密分散されたデータベースにおいて，マルチパーティプロトコルを用いて関係代数を実行可能にすることで，データを 1 度も復元することなく，関係データベースのすべての構造演算を実行可能とした．

提案方式では，ユーザの要求は関係代数で記述されなければならないが，通常は SQL などのデータベース言語で記述される．よってユーザの SQL などによる検索式を，マルチパーティプロトコルで実行可能な関数に変換するトランスレータの開発を検討する．

謝辞　電気通信大学太田和夫教授には本研究に関し，ご助言，ご討論をいただいた．また，株式会社東芝研究開発センター遠藤つかさ氏は，本研究の立ち上げ期に電気通信大学大

学院に所属し，ご助言，ご討論をいただいた．ここに，謹んで感謝の意を表する．

参 考 文 献

1) 志村正法，遠藤つかさ，宮崎邦彦，吉浦　裕：安全で機能制限のないデータベースを実現するマルチパーティプロトコルを用いた関係代数演算，情報処理学会研究報告，CSEC42, pp.187–193 (2008).
2) 桜井友二，齊藤泰一：情報漏えい防止データベースシステムの提案，*Proc. SCIS2006*, 1F2-5 (2006).
3) 守田泰博，宮本俊幸，熊谷貞俊：秘密分散共有法を用いた分散データベースシステム，電子情報通信学会技術研究報告，CST2004-47, pp.49–54 (2005).
4) Hacigumus, H., CLi, B.I. and Mehrotra, S.: Executing SQL over Encrypted Data in the Databases-Service-Provider Model, *Proc. 2002 ACM SIGMOD International Conference on Management of Data*, pp.216–227 (2002).
5) Denning, D.E.: Field Encryption and Authentication, *Proc. Advances in Cryptology* (*CRYPTO'83*), pp.231–247 (1984).
6) Lindell, Y. and Pinkas, B.: Privacy Preserving Data Mining, *Proc. Advances in Cryptology* (*CRYPTO'00*), pp.36–53 (2000).
7) 電子情報通信学会（編）：情報セキュリティハンドブック，オーム社 (2004).
8) 安　健司，赤羽泰彦，尾崎将巳，瀬本浩治，佐々木良一：暗号メールにおける個人情報不正送出チェックシステムの評価，情報処理学会論文誌，Vol.46, No.8, pp.1976–1983 (2005).
9) Sion, R.: Secure Data Outsourcing, *Proc. VLDB Conference*, pp.1431–1432 (2007).
10) Su, C., Zhou, J., Bao, F., Wang G. and Sakurai, K.: Privacy-Preservation Techniques in Data Mining, *Digital Privacy-Theory, Technologies and Practices*, Acquisti, A., Gritzalis, S., Lambrinoudakis, C. and Vimercati, S.D.C.D. (Eds.), pp.187–224, Auerbach Publications (2007).
11) Date, C.J.（著），藤原　譲（訳）：データベースシステム概論，丸善 (1997).
12) 岡本龍明，山本博資：現代暗号，産業図書 (1997).
13) Shamir, A.: How to Share a Secret, *Comm. ACM*, Vol.22, No.11, pp.612–613 (1979).
14) Ben-Or, M., Goldwasser, S. and Wigderson, A.: Completeness Theorem for Non-Cryptographic Fault-Tolerant Distributed Computation, *20th Annual ACM Symposium on Theory of Computing*, pp.1–10 (1988).
15) Gennaro, R., Rabin, M.O. and Rabin, T.: Simplified VSS and Fast-Track Multiparty Computations with Applications to Threshold Cryptography, *Proc. 17th ACM Symposium on Principles of Distributed Computing*, pp.101–110 (1998).
16) Nishide, T. and Ohta, K.: Constant-round Multiparty Computation for Interval Test, Equality Test and Comparison, *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, Vol.90-A, No.5, pp.960–968 (2007).
17) D9.2 Security Analysis, WP9 Cryptographic Aspects, SecureSCM. http://www.securescm.org/index.php?option=com_docman&task=cat_view&gid=22&&Itemid=24

付　　録

A.1 関 係 代 数

本文中で説明しなかった共通，直積，選択，射影，商について説明する．

①共通

テーブル A とテーブル B が和両立であるとき，共通 $A \cap B$ は，以下のように定義される．

定義：$A \cap B = \{t | t \in A \wedge t \in B\}$

A と B の両方に属するレコードで構成されたテーブルを返す．

②直積

テーブル A とテーブル B の直積 $A \times B$ は以下のように定義される．

定義：$A \times B = \{(s,t) | s \in A \wedge t \in B\}$

ただし，$s = (a_1, a_2, \ldots, a_m)$，$t = (b_1, b_2, \ldots, b_n)$ とするときに (s,t) を以下のように定義する．

$(s,t) = (a_1, a_2, \ldots, a_m, b_1, b_2, \ldots, b_n)$

③選択

テーブル A において，A_i と A_j を θ–比較可能な属性とする．このとき選択 $A[A_i \theta A_j]$ は以下のように定義される．

定義：

(i) $A[A_i \theta_j] = \{t | t \in A \wedge t[A_i] \theta t[A_j]\}$

(ii) $A[A_i \theta \alpha] = \{t | t \in A \wedge t[A_i] \theta\, \alpha\}$　※ α は定数

A において，指定したレコードのテーブルを返す．

④射影

テーブル A の全属性集合 $\{A_1, A_2, \ldots, A_n\}$ の部分集合を $X = \{A_{i_1}, A_{i_2}, \ldots, A_{i_k}\}$ $(1 \leq i_1 < i_2 < \ldots < i_k \leq n)$ とすると，射影 $A[X]$ は以下のように定義される．

定義：$A[X] = \{t[X] | t \in A\}$

A から指定した属性のテーブルを返す．

⑤商

n 次のテーブル $A(A_1, A_2, \ldots, A_{n-m}, B_1, B_2, \ldots, B_m)$ と m 次（$m < n$）のテーブル $B(B_1, B_2, \ldots, B_m)$ に対して，商 $A \div B$ は以下のように定義される．

定義：$A \div B = \{t | t \in A[A_1, A_2, \ldots, A_{n-m}] \wedge (\forall u \in B)((t, u) \in A)\}$

B のすべての属性の値を同時に満たす A のレコードを選び出し，B の属性を取り除いた列のテーブルを返す．

上記 8 つの演算子のうち，和，差，直積，選択，射影の演算子は互いに独立である．以下の式 (1)〜(3) で表せるように，この 5 つの演算子を組み合わせることによって共通，結合，商を表現できる．

・共通

$$A \cap B = A - (A - B) \tag{1}$$

・結合

$$A[A_i \theta B_j]B = (A \times B)[A.A_i \theta B.B_j] \tag{2}$$

A と B の直積 $(A \times B)$ は，A の属性と B の属性を継承している．$A.A_i$ は，A から継承した属性 A_i を表し，$B.B_j$ は，B から継承した属性 B_j を表す．$A.A_i$ と $B.B_j$ が同名の属性であるときに，2 つを区別するために “A.” と “B.” を用いる．

・商

$$A \div B = A[A_1, A_2, \ldots, A_{n-m}] - ((A[A_1, A_2, \ldots, A_{n-m}] \times B) - A)[A_1, A_2, \ldots, A_{n-m}] \tag{3}$$

(平成 21 年 11 月 30 日受付)
(平成 22 年 6 月 3 日採録)

志村　正法（正会員）

2008 年電気通信大学電気通信学部人間コミュニケーション学科卒業．2010 年電気通信大学大学院電気通信学研究科人間コミュニケーション学専攻博士前期課程修了．現在，（株）日立製作所公共システム事業部に勤務．

宮崎　邦彦（正会員）

1998 年東京大学大学院数理科学研究科修士課程修了．同年（株）日立製作所入社．現在に至るまで，同社システム開発研究所にて，暗号，情報セキュリティの研究に従事．2006 年東京大学大学院情報理工学系研究科博士課程修了．博士（情報理工学）．2004 年暗号と情報セキュリティシンポジウム（SCIS2004）論文賞受賞．電子情報通信学会会員．

西出　隆志（正会員）

1997 年東京大学理学部情報科学科卒業．同年日立ソフトエンジニアリング株式会社入社．セキュリティ，ネットワーク製品の設計開発に従事．2003 年南カリフォルニア大学修士課程修了．2008 年電気通信大学博士（工学）．2009 年より九州大学大学院システム情報科学研究院助教．公開鍵暗号，暗号プロトコル，コンピュータセキュリティ等の研究に従事．ACM，電子情報通信学会各会員．

吉浦　　裕（正会員）

1981 年東京大学理学部情報科学科卒業．日立製作所を経て，2003 年より電気通信大学電気通信学部人間コミュニケーション学科勤務．現在，同教授．情報セキュリティの研究に従事．理学博士．電子情報通信学会，日本セキュリティ・マネジメント学会，システム制御情報学会，人工知能学会，IEEE 各会員．2000 年日立製作所社長技術賞，2005 年情報処理学会論文賞，2005 年システム制御情報学会産業技術賞，2006 年 IEEE International Conference on Intelligent Information Hiding and Multimedia Signal Processing，Best Paper Award 各受賞．