2G3-4

分散データベースからの プライバシー保護頻出飽和パターンマイニング

Privacy-preserving mining of frequent closed patterns from distributed databases

大滝啓介*1 山本章博*1
Keisuke OOTAKI Akihiro YAMAMOTO

京都大学大学院 情報学研究科*1
Graduate School of Informatics, Kyoto University

Privacy-preserving data mining is to discover knowledge from large-scale data that include private information and are stored in distributed places, without revealing data one's private information to other participants. In this paper, we treat the problem of finding frequent closed itemsets from horizontally distributed databases. Our solutions are based on expressing itemsets in ZDDs, with which the representation can be compact and communications will be efficient between databases. We propose two solutions for privacy-preserving mining, present some experimental results, and estimate them with the semi-honest model.

1. はじめに

近年，多様なデータが複数のサイトに分散して蓄積され，巨大な分散データベースを構成するようになった．分散データベースを所有する構成要素を**サイト**，各サイトが所持するデータベースを**部分データベース**，部分データベースを合わせて構成されるデータベースを**全体データベース**とよぶ．本研究では，全体データベースからデータマイニングを行う手法の開発を目標とする．そのようなマイニングにおいては，サイト間の通信量を小さくするだけでなく，各サイトが保持する個人情報の扱いに注意し，情報漏洩を防ぐ必要がある．

全体データベースから各サイトの秘密情報を漏洩することなく知識発見を行うことは**プライバシー保護データマイニング**とよばれ研究されている．本研究では頻出飽和パターンマイニングを対象として，各サイトの情報をコンパクトに表現するために **ZDD(Zero-suppressed BDD)**[5] を利用して通信量を抑制する．その上で，ZDD の処理に起因するプライバシーに関する問題に対処するために新たな手法を提案し，その評価と実験を行う．

2. 準備

本章では本研究で用いる基本的な概念について説明する．

2.1 頻出飽和パターンマイニング

データベースに出現する全ての要素の集合を$\mathcal{I} = \{i_1, \ldots, i_n\}$で表し，その部分集合 $X \subset \mathcal{I}$ を**パターン**とよぶ．データベース上の識別子とパターンの組を**トランザクション**，その集合を **(トランザクション) データベース**とよぶ．あるデータベース DB にパターン X が出現する回数を $X.sup$ と表し**サポート**とよぶ．ユーザの指定する**最小サポート** σ に対して $X.sup \geq \sigma \times |DB|$ が成立するパターンは**頻出である**という．

関数 Com を「トランザクションの組 T を受け取り，T に共通して含まれるパターンを返す」関数と定義し，関数 Occ を「パターン X を受け取り，X が含まれているトランザクションの組を返す」関数と定義する．このとき演算子 $c = Com \circ Occ$ を用いて $c(X) = X$ であるパターンは**飽和パターン**とよばれる [4]．頻出な飽和パターンから全ての頻出なパターンを復元できるという意味で，飽和パターンはデータベースの代表的なパターンであり，分散データベースにおいては，飽和パターンのみを扱うことで通信量の削減が期待できる．

連絡先: 〒 606-8501 京都市左京区吉田本町工学部 10 号館 401 号室山本研究室 ootaki@iip.ist.i.kyoto-u.ac.jp

ID	v_1	v_2	v_3	v_4	
1	1	1	0	0	
2	1	0	1	0	DB_1
3	1	0	0	0	
4	0	0	1	1	DB_2

図 1: $\mathcal{I} = \{v_1, v_2, v_3, v_4\}$ 上の水平分散型分散データベース

各サイトが所有する部分データベースを DB_i，サイト数を m とするとき，本研究が対象とする全体データベース DB は $DB = DB_1 \cup DB_2 \cup \ldots \cup DB_m$ を満たす．ただし $DB_i \cap DB_j = \emptyset (i \neq j)$．このような分散データベースは**水平分散型**とよばれる．このとき，パターン X のサイト i におけるサポートを $X.sup_i$ と表し，頻出であることの条件は $\sum_i X.sup_i \geq \sum_i \sigma \times |DB_i|$ と自然に拡張される．

本研究が対象とする分散データベースにおける頻出飽和パターンマイニングとは，サイト数 m，分散データベース $\{DB_i\}$，最小サポート σ について，全体データベース DB において飽和なパターンの集合 $C_{12\ldots m}$ を求め，それらから更に頻出であるパターンの集合 $FC_{12\ldots m}$ を求めることを目的とする．本研究では特に $C_{12\ldots m}$ を求める処理を考える．

2.2 パターンの集合の ZDD による表現

ZDD は BDD[2] を拡張したデータ構造であり，パターンの集合をコンパクトに表現することができる [6]．具体的な例として $\mathcal{I} = \{v_1, v_2, v_3, v_4\}$ 上のデータベースを考える．例えば $DB = \{v_1v_2, v_1v_3, v_1, v_3v_4\} = \{v_1v_2, v_1v_3, v_1\} \cup \{v_3v_4\} = DB_1 \cup DB_2$ は，図 1 のようなデータベースを表す．このとき，部分データベースは図 2 の ZDD で表される．

ZDD では，節点における 1 枝がアイテムの所持を表し，**1** への経路がパターンに対応する．ZDD は二分木なので，各節点における変数番号と，両枝のポインタを組にして (v, p_0, p_1) という 3 つ組のリストで表現できる．今，演算 $\otimes$ を 2 つの集合 A, B を受け取り $A \cup B \cup \{\alpha \cap \beta \mid \alpha \in A, \beta \in B\}$ を返すと定義する．サイト i, j が，それぞれのサイトにおける飽和パ

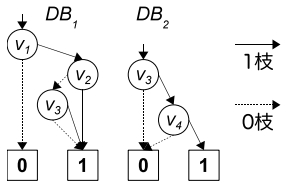


図 2: ZDD による図 1 のデータベースの表現

ターンの集合 C_i, C_j を持ち寄り，$C_{ij} = C_i \otimes C_j$ を求めることで，サイト i, j の両方で飽和なパターンの集合 C_{ij} を求めることができる [4]．本研究ではこの処理に ZDD を利用する．

2.3 分散処理とプライバシー保護

一般にプライバシーの保護に関しては，以下の要求を満足させる必要がある [7].

1. 漏洩した情報から個人を同定できない
2. 漏洩した情報から個人への攻撃を構成できない

本研究では，マイニングの分散処理に対して，次のような方針を取る．まず要求 2. については，処理への参加者が **semi-honest model**[1] に従うことを仮定した上で要求 1. に着目する．ZDD を用いた演算を実行する際には，節点情報を用いる [5] ため，その分散処理において，処理中に利用する**互いに共通して所持する変数**の情報や，**出力に共有する変数**の情報，また再帰処理の終点になる**終端節点**に関する情報が漏洩する可能性がある．そこでそれらの情報が漏洩しないように，また漏洩してもどのサイトから漏洩したものが区別できないようにするという方針から，新しい手法を提案する．なお以降の章では，$C_{12\ldots m}$ を求めることを基本として，後述するようなトーナメントという形の分散処理において基本となる 2 サイト間での演算 $C_i \otimes C_j$ について考察する．

3. プライバシー保護マイニングの具体的手法

本章では 2.3 節に示した方針の下，具体的なマイニング手法を提案する．

3.1 ZDD の変換に基づく手法

ZDD z は，アイテムの集合 $\mathcal{I}$ 上の論理関数 f_z と一対一に対応する．例えば図 2(左) で示した ZDD は，積和標準形の論理関数として $f(v_1, v_2, v_3) = v_1 v_2 \vee v_1 v_3 \vee v_1$ と表すことができる．本研究では，ZDD を用いた集合演算を安全に行うために，ZDD に対応した論理関数を表す真理値表 (2 進文字列) を用いた論理演算を安全に行うことで，プライバシー保護の要求を満たす手法を提案する．

具体的にはパターン同士の集合演算の $\cup, \cap$ に，論理演算の $\vee, \wedge$ を用いる．さらに $\vee$ は $\wedge, \neg$ を用いる．そこでこれらの演算を安全に行うために，**SFE(Secure Function Evaluation)**[8] という枠組みで提案されている安全な *and* と *not* を用いる．例えば ZDD P, Q から $P \cup Q$ を求めるには，P, Q に対応する真理値表 T_P, T_Q を用いて $T_P \vee T_Q$ を求め，その結果を用いて $P \cup Q$ に対応する ZDD を再構成することにより，安全に $P \cup Q$ を求めることができる．

しかしこのような変換では，アイテムの集合 $\mathcal{I}$ 上の論理関数から真理値表を求めるため，Naive な実装をすると真理値表の長さが指数的に爆発する欠点がある．

3.2 ダミーデータを利用する手法

3.1 節で示した ZDD の変換よりも実現可能性が高い手法として，ダミーデータを利用する手法を提案する．

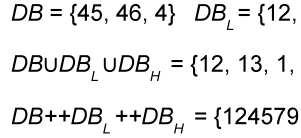


図 3: 水平挿入，垂直挿入による拡大データベース生成の例

3.2.1 概要と問題設定

ダミーデータを利用した手法の基本的な考え方は次の通りである．2 サイト S_1, S_2 が所有するデータベース DB_1, DB_2 に対し，**制約** c_1, c_2 を考える．また $p_i : 2^{2^{\mathcal{I}}} \times 2^{2^{\mathcal{I}}} \to 2^{2^{\mathcal{I}}} (i = 1, 2)$ であるような演算 p_1, p_2 を考える．ここで 2 サイトは c_i を利用して $D_i = \{X \mid c_i(X)\}$ のようなダミーデータ D_i を生成する．なお，各トランザクションの識別子は以降省略する．その後 2 サイトはダミーデータ D_i を利用して

$$DB'_i = p_i(DB_i, D_i)$$

のようなダミーが混入されたデータベースを生成する．この DB'_i を**拡大データベース**とよぶ．2 サイトは DB'_i を利用して $S' = DB'_1 \operatorname{op} DB'_2$ を求め，S' から，本来求めようとしている演算結果である $S = DB_1 \operatorname{op} DB_2$ を抽出する．その処理を関数 ES_i で表す．p_i と c_i によっては，$ES_i(S) = S$ のような恒等関数になる場合もある．ここで 2 サイト間での演算 $DB_1 \operatorname{op} DB_2$ を，プライバシー保護の要求を満たすように実行するために，$p_1, p_2, c_1, c_2, ES_1, ES_2$ をどのように設定するのかという問題を考える．なお以下では，$p_1 = p_2, c_1 = c_2, ES_1 = ES_2$ として，それぞれ p, c, ES と記述する．

3.2.2 本研究の提案手法

今 $i = 1, 2$ とし，問題を簡単にするため DB_i に出現するアイテムは全て整数であり，アイテムの集合 $\mathcal{I}$ を $a < b$ である整数 a, b を用いて具体的に $\mathcal{I} = \{a, a+1, a+2, \ldots, b\}$ であるとする．この事実を $range(DB) = [a, b]$ と閉区間 $[a, b]$ で記述し，データベース DB_i の定義域は $[a, b]$ であるとよぶ．

拡大データベース DB'_i の定義域として，b より大きい整数 M を用いて $[1, M]$ を設定する．つまり $range(DB'_i) = [1, M]$ で，$[1, M] = [1, a-1] \cup [a, b] \cup [b+1, M]$ である．このとき $[1, a-1]$ を L，$[b+1, M]$ を H と表す．L と H から作られるデータベースを DB_L, DB_H とする．つまり $DB_L \subseteq \{X \mid X \subseteq \{1, 2, \ldots, a-1\}\}, DB_H \subseteq \{X \mid X \subseteq \{b+1, b+2, \ldots, M\}$ である．このような背景の下で op, c, ES として以下を提案する．

関数 p 演算 $\cup$ 及び $++$ を利用する．ただし $++$ は対応する識別子同士のパターンの結合を行う．例えば $A = \{12, 13\}, B = \{45, 67\}$ のとき $A ++ B = \{1245, 1367\}$.

制約 c $D_1 = D_{L1} \cup D_{H1}$，$D_2 = D_{L1} \cup D_{H2}$ とする．ただし $D_{Li} \subseteq DB_L$，$D_{Hi} \subseteq DB_H$ で，$D_{L1} \cap D_{L2} = D_{H1} \cap D_{H2} = \emptyset$.

後処理 ES 恒等関数 $id(X) = X$．または定義域 $[1, M]$ 上のパターンから $[a, b]$ 上の部分だけを抽出する関数 choice.

以下でそれぞれについて説明する．$p = \cup$ による生成を**水平挿入**，$p = ++$ による生成を**垂直挿入**とよび，それぞれの例を図 3 に示す．2 つの生成法は独立しているため，同時に利用することも可能である．

制約 c と関数 ES は，求める演算の種類 op に合わせて設定する．$\cap, \cup, \sqcap$ についてはそれぞれ以下のように設定する．

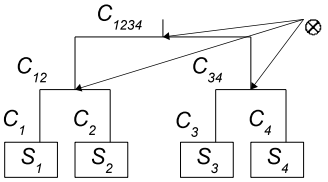


図 4: 演算の反復によって構成されるトーナメント

演算子 $\cap$ の場合．$p = \cup, ES = id$. 制約 c によって $DB'_1 \cap DB'_2 = (DB_1 \cup DB_{L1} \cup DB_{H1}) \cap (DB_2 \cup DB_{L2} \cup DB_{H2}) = DB_1 \cap DB_2$ となり，ES によって演算結果 $DB_1 \cap DB_2$ を求めることができる．

演算子 $\cup$ の場合．$p = \cup, ES =$ choice. 計算により $DB'_1 \cup DB'_2 = (DB_1 \cup DB_{L1} \cup DB_{H1}) \cap (DB_2 \cup DB_{L2} \cup DB_{H2}) = DB_1 \cup DB_2 \cup DB_{L1} \cup DB_{L2} \cup DB_{H1} \cup DB_{H2}$ が求まり，choice によって $DB_1 \cup DB_2$ を抽出できる．

演算子 $\sqcap$ の場合．$p = \cup, ES =$ choice. 計算により $DB'_1 \sqcap DB'_2 = (DB_1 \cup DB_{L1} \cup DB_{H1}) \sqcap (DB_2 \cup DB_{L2} \cup DB_{H2}) = (DB_1 \sqcap DB_2) \cup \{(DB_{L1} \cup DB_{H1}) \sqcap (DB_{L2} \cup DB_{H2})\}$ が求まり，choise によって $DB_1 \sqcap DB_2$ を抽出できる．

$p = ++$ については，プライバシー保護をより強固にするために利用できる．これらの提案手法の評価は 4 章でまとめる．

3.3 トーナメント実施する際の注意点

3.1 節と 3.2 節で述べた手法により，2 サイト i,j 間で $C_i \otimes C_j = C_{ij}$ を求めるとする．このとき演算を 2 サイトずつ繰り返すことで $C_{12\cdots m}$ を求めることができる．この反復処理は図 4 のようなトーナメントに基づいて行うものとする．図 4 では，サイト数 $m = 4$ である．このトーナメントを実施する際，各組合せの片方のサイトが空集合 $\emptyset$ を持つ場合，その処理中に相手のサイトの情報が漏洩するという問題が生じる．そのための対策として次のような手法を与える．

3.3.1 一方向性置換を利用したトーナメントの構成

関数 $f : X \to Y$ が入力 x に対する出力 $y = f(x)$ から，入力 x を逆計算することが困難であり，かつ $X = Y$ が成り立ち関数 f が置換であるとき，関数 f を**一方向性置換**とよぶ [8]．

今サイト数 $m = 2^n, (n \in \mathcal{Z} \backslash \{0\})$ の分散データベース $\{DB_i\}$ を考え，そこからランダムに構成されたトーナメント T を $\{(i, DB_i)\}$ と表す．ここに一方向性置換 $f : 2^m \to 2^m$ を用いることで，そのトーナメント実施上の番号をランダムに変更する．つまりトーナメント T に f を作用させ，

$$\{(f(1), DB_1), (f(2), DB_2), \ldots, (f(m), DB_m)\}$$

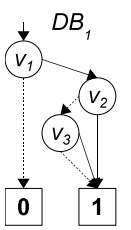
を生成し，$(f(i), f(j)) = (1, 2)$ となるサイト i, j から処理を行う．一方向性置換 f が十分に機能すれば，組合せの相手となるサイトを隠すことができると期待できる．

4. 実験と評価

本章では 3 章で提案した手法の理論的な評価と，実験による検証を行う．

4.1 手法の合成則による評価

複数のサイトが協調して，安全に処理を行うことは一般に **SMC**[3] とよばれる．SMC における処理の手順を**プロトコル**とよぶ．本研究が前提としている semi-honest model において，プロトコルが MPC の文脈で安全かどうかを評価するために，以下のプロトコルの合成定理が用いられる．



index	*var*	p_0	p_1	
0	-1	0	0	定数0
1	-1	1	1	定数1
2	1	0	3	節点v_1
3	2	4	1	節点v_2
4	3	1	1	節点v_3

図 5: ZDD DB_1 と，対応する節点表

定理 (semi-honest model におけるプロトコルの合成定理 [1])． プロトコル g が，より小さいプロトコル f に安全に縮小可能で，そのプロトコル f が安全に実行可能であるなら，プロトコル g は安全に実行可能である．

つまり，いくつかの安全なプロトコルを組み合わせて，より大きな安全なプロトコルを構築できる．この性質を用いて，プロトコルの処理ブロック毎に安全かどうか評価し，プロトコル全体を評価する．本研究では**安全である**ことを「**処理の入力と出力から知ることができる情報以上のことを知ることができない**」と設定して評価する．つまり ZDD とその変換やダミーの利用によって余計な情報が漏洩しないことを確認する．

ZDD の変換に基づく手法 (3.1 節)． ZDD の変換による手法では処理の基本要素として SFE による計算を利用する．SFE は暗号理論によって安全と評価されているため，それを基本要素として処理を構成するため，全体として安全に処理が可能である．

ダミーデータを利用する手法 (3.2 節)． この手法が安全かどうかは，ダミーデータが上手く混入されているかどうかに依存する．それはダミーデータの制約 c や，混入演算 p に依存して決まるため，それらを次の実験で評価する．

4.2 手法の実験とその結果

ZDD 処理系はその内部に節点情報を保存する表を保持している．ここでは ZDD の節点を以下の 3 種類に分類して考察する．なお定数節点とは ZDD 上の **0, 1** を指す．

1. **Single**: 両枝が定数節点に接続している節点
2. **S-Single**: 片方の枝が定数節点に接続している節点
3. **Complex**: 両枝が定数節点以外に接続している節点

図 2 中の ZDD DB_1 を表す節点表を図 5 に示す．ZDD 演算は，節点表の index を 2 つ受け取り，新しい index を返す処理に相当する．そこで ZDD 演算の安全性を節点表を通じて測定する手法を提案する．また元のデータベースから生じる節点を，ダミーデータによって上手く隠すことができるかどうかを確認することで，プライバシー保護の要求 1. を満たすことが可能かどうかを確認する．実験に際しては，Java 用の ZDD 処理系である JDD [*1] を利用した．データセットには FIMI Datasets[*2] から mushroom を選び，またランダムに生成したデータを使用した．実験は Mac OS X 10.6(Snow Leopard), Intel Core i5 2.8GHz, Memory 12GB で行った．

はじめに，上に挙げた節点の種類がどの程度の割合で ZDD 上に存在するのかを測定する．今，データベースのサイズを $n = 100$，データベースの全てのアイテムの集合を $\mathcal{I} = \{1, 2, \ldots, 100\}$ とする．1 つのトランザクションに

∗1 http://javaddlib.sourceforge.net/jdd/
∗2 http://fimi.cs.helsinki.fi/data/

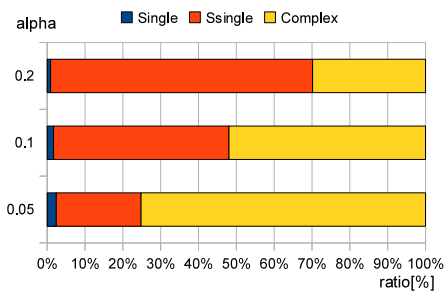


図 6: データベースの疎性パラメータ α と節点比率の変化

表 1: mushroom の結果

演算	真のデータ	ダミーデータ	中間データ
$\cup$	5,737	4,728	677
$\cup$(x2)	5,737	10,541	885
$\sqcap$	5,737	4,728	417,238
$\sqcap$(x2)	5,737	10,541	1,368,009

表 2: random-u の結果

演算	真のデータ	ダミーデータ	中間データ
$\cup$	6,025	12,562	1,829
$\cup$(x2)	6,025	35,509	2,601
$\sqcap$	6,025	12,562	9,998,089
$\sqcap$(x2)	6,025	35,509	27,932,954

$\alpha = 0.05, 0.1, 0.2$ の割合でアイテムが存在するとして，ランダムなデータを 8 個生成し，random-u とする．それらのデータベースを ZDD という形式に保存したとき，その ZDD を表現する節点表において，ZDD の節点数を種類毎に測定し平均したのものを図 6 に示す．これより，疎なデータであれば **complex** な節点が増加することが分かる．これは疎なデータであるほど節点共有が難しくなることを反映している．FIMI データセットの mushroom は似た多くのパターンからなるデータセットであり，ランダムなデータはホワイトノイズに近い．よって一般的なデータベースはこれらの 2 つのデータの中間に位置すると考えられる．よって以降の実験では，その両極端な 2 つのデータを利用する．

以上の考察を踏まえ実験を行う．mushroom と random-u の両方を利用して本研究の提案に基づき拡大データベースを生成する．それを利用した演算を行い，ノードの種類を測定する．そのとき演算に利用される節点 n が，ダミーデータ全ての集合 DM，通常のデータベースの DB に対して

$$f(n) = \frac{P(n \in DB|n)}{P(n \in DM|n)} < \frac{1}{2}$$

が成り立つような DM を実現可能かどうかを確認する．具体的には，任意の節点が真のデータである確率が，ダミーデータである確率より小さいかどうかを確認する．つまり $f(n)$ を $\frac{1}{2}$ より小さく保って処理を行うことができれば，処理中に通信を行う節点の情報に，ダミーのデータが十分に混ざっていると判断できると考えられる．そこで mushroom と random-u の両方に対しての各演算の実行を追跡する実験を行い，その最終段階における各データの割合を表 1，2 に示した．

実験における各データセットの大きさは 100 で，$p = \cup$ とする．つまり $DB'_i = DB_{Li} \cup DB_{Hi} \cup DB_i$ である．表中の (x2) はダミーの量を 2 倍にすることを意味していて，例えば表 1. から，mushroom に対して $p = \cup$ を実験するとき，ダミーデータを 2 倍にすることで 4728 個から 10541 個へとダミーデータによる節点の数が増えることが分かる．これはつまり，$|DB|$ に対して $|DB_{Li}| = 2|DB|$ かつ $|DB_{Hi}| = 2|DB|$ となるようにダミーデータの要素数を増やすことで，節点数を上から抑えることができることを意味している．このときのダミーデータの要素数は元のトランザクションデータベースの要素数の 4 倍となる．mushroom の定義域は $[1, 350]$ とし，真のデータは $[101, 220]$ 上に存在するとしている．random の定義域は $[1, 4000]$ とし，真のデータは $[1001, 3000]$ 上に存在するとしている．各トランザクションにおけるアイテム数の平均は，その定義域の大きさの $\alpha = 0.1$ 倍とする．

これらの実験結果から，十分な量のダミーデータを利用することで，真のデータベースの節点数を超えてダミーデータを挿入することが可能であることが分かる．水平挿入のみを利用する場合の必要なダミーデータの目安は，元のデータ量の 4 倍程度と想定できる．また，どれほどの中間データを許容するのかも，一つのパラメータになりうるため，今後，より詳細な検証が必要である．

5. まとめ

本研究では，水平分散型の分散データベースを対象としたマイニング手法の開発を目的とし，分散サイト間における通信量と，プライバシー保護の 2 つの問題を考慮した手法の提案を行った．頻出飽和パターンマイニングを対象として，サイト間における通信量を削減するために，飽和パターンを利用するだけではなく，ZDD を利用して処理を行う手法を導入した．その上で，プライバシー保護の問題を考慮して処理を行うために，ZDD の変換に基づく手法と，ダミーデータに基づく手法を提案し，その評価と実験を行った．同時にその実験結果から，ダミーデータに基づく手法を上手く作用させるための条件について考察を行った．今後の課題として，より一般的な参加者のモデルに対応したプロトコルの提案や，確率分布等を導入したダミーの生成手法の考案などがある．また今回利用した節点表による手法の評価についても，改善の余地があるため，今後の課題である．

参考文献

[1] Aggrawal, C. C. and Yu, P. S. *Privacy-Preserving Data Mining: Models and Algorithms*, Springer-Verlag (2008).

[2] Bryant, R. Graph-Based Algorithms for Boolean Function Manipulation, *IEEE Transactions on Computers*, **35** (1986), 677–691.

[3] Goldreich, O. Secure Multi-party Computation, *Workind Draft* (2000).

[4] Lucchese, C., Orland, S. and Pergo, R. Distributed Mining of Frequent Closed Itemsets: Some Preliminary Results, *International Workshop on High Performance and Distributed Mining* (2005).

[5] Minato, S. *Binary Decision Diagrams and Applications for VLSI CAD*, Springer (1996).

[6] Minato, S. and Arimura, H. Frequent Closed Item Set Mining Based on Zero-suppressed BDDs, *Transactions of the Japanese Socieety for Artificial Intelligence*, **22** (2007), 165–172.

[7] Vaidya, J., Zhu, Y. M. and Clifton, C. W. *Privacy Preserving Data Mining (Advances in Information Security)*, Springer-Verlag (2005).

[8] 岡本龍明, 山本博資 現代暗号, 産業図書 (1997).