セキュアなコンテンツ配信におけるSD法に基づいた効率の良い鍵管理方式

菊池浩明[†] 中村雄一[††]

モバイルノードを含むワイヤレスアドホックネットワークなどにおいて，安全なグループ通信のための効率の良い鍵管理が1つの課題になっている．新規追加や退出などを生じる動的なグループの実現するために，Naorらは差分集合（SD）法を提案した．SD法は，疑似乱数生成器を用いることで送信する暗号文のサイズを，無効にするユーザ数 r に対して，$2r-1$ にまで大幅に削減する．SD法においては，ノード i の下の部分集合からノード j の部分集合を除いた差分集合の組で正規のユーザの部分集合を指定する．しかしながら，すべての正規ユーザを包含するのに必要なすべての差分集合（カバー）(i,j) を求めるためには，送信者は失効ユーザのすべての可能な組合せを検査する必要があった．このために，ナイーブな総当たりによる検索では $O(r^3)$ の時間がかかる．これがSD法実現の課題である．そこで我々は，このカバーをみつける課題を解決するため，必要な部分集合の中で処理する必要のあるノードのインデキシングのテクニックを導入する．これにより，計算のコストは $O(r\log r\log n)$ にまで削減される．さらに，本論文では提案アルゴリズムを実装し，その計算時間を報告する☆．

An Efficient Key Management Based on the Subset Difference Method for Secure Contents Distribution

Hiroaki Kikuchi[†] and Yuichi Nakamura[††]

An efficient key management for secure group communication is one of the current issue in wireless ad hoc network with mobile nodes. In order to address the dynamic receiver update operations such as leave or join, Naor, et al. proposed the subset difference (SD). The SD method allows senders to reduce drastically the size of ciphertext to be sent to $2r-1$ using a pseudo random number generator, where r is the number of revoked users (who leave the group). In the SD method, the subsets of authorized users are represented by some differences of two subsets such that i covers valid users and j excludes the revoked users in i. To have all subsets (i,j) necessary to cover all valid users in a tree, a sender has to test all possible combinations of revoked users. A naive exhaustive search for the purpose takes $O(r^3)$ time. This is a drawback of the SD method. Hence, to address the issue for finding the cover, we propose a new efficient algorithm to reduce the cost up to $O(r\log r\log n)$ by introducing the technique for indexing nodes to be dealt with in the necessary subsets. In addition, we implement the proposed algorithm and demonstrate the performance in terms of processing time in this paper.

1. はじめに

近年，高速通信回線の整備やDVDなどの大容量メディアの普及が急速に進んでいる．それにともない，ソフトウェアや映画，音楽などをデジタル化したコンテンツをインターネットなどで配信するサービスが広がってきている．その一方で，そうしたコンテンツの違法コピーが深刻化している☆☆．こうした違法コピーを防ぐため，コンテンツを事前に暗号化し，正当な対価を支払ったユーザ以外には内容が漏洩しないような仕組みを持ったコンテンツ配信サービスが求められている．

† 東海大学電子情報学部情報メディア学科
Department of Information Media Technology, School of Information Technology and Electronics, Tokai University
†† 東海大学大学院工学研究科
Graduate School of Engineering, Tokai University

☆ 本論文の初期のバージョンは，文献1)，2)で発表されている．

☆☆ コンピュータ・ソフトウェアの権利団体BSAの調査[3]によると，2003年1月〜12月に日本の違法コピーの状況は，前年比で違法コピー率，被害額ともに減少したものの，依然として被害額は世界的にみて高い水準にある．違法コピーによる被害額は約1,807億円にのぼり，米国，中国に続いて世界第3位であった．

コンテンツ配信者から n 人のユーザへの秘密通信を効率良く行う技術に，Fiat らによって初めて提案されたブロードキャスト暗号[4] があり，DVD-RAM ディスクなどのコンテンツ保護などに応用され始めている[5]．代表的なものに Wong らによる木構造に基づく方式 LKH[6] があり，IP マルチキャストをはじめとする各種のネットワーク分野に応用されている[7)~12)]．LKH 法では，ユーザ数 n に対して，各ユーザは $\log n + 1$ 個の秘密情報を管理し，失効ユーザ数 r のとき，$r\log(n/r)$ 個の暗号文で同報通信を実現する．この LKH 法の秘密情報の大きさを削減するために，Asano によるマスタ鍵を用いた方式[13] や野島らによる落とし戸付き一方向性関数を用いた方式[14] が提案され，改良されている．

Naor らは，LKH 法よりも暗号文の数を削減する Subset Difference（SD）法を提案している[15]．SD 法は，部分集合の差に基づいて失効ユーザを排除する方式であり，ユーザの秘密情報は $1/2\log^2 n + 1/2\log n + 1$ 個となり LKH 法よりも増加する代わりに，暗号文の数を $2r-1$ 個までに削減する．Naor らの提案以降，Halevy らによって階層を導入することによるさらなる削減を実現する方式[16]，奥秋らによるハイブリッド方式[17]，素因数分解の困難性に基づく鍵割当てを行う提案[18] など様々な改良が試みられている．

本研究では，SD 法の実現における次の問題点を指摘する．SD 法においては，ノード i の下の部分集合からノード j の部分集合を除いた差分集合（カバー）(i,j) で正規のユーザの部分集合を指定する．しかしながら，この i と j で定められる差分集合 (i,j) を導出するのは自明ではなく，送信者は失効ユーザのすべての可能な組合せを検査する必要がある．このために，ナイーブな総当たりによる検索では $O(r^3)$ の時間がかかる．失効者の数 r が大きな運用では無視できないオーバヘッドである．

そこで，本論文では，失効するユーザを一度に与えるのではなく，カバー候補となる集合のデータベースに逐次的に追加していく前処理を行うことで，検索範囲を狭めるインデキシングの手法をここに導入することを試みる．提案するアルゴリズム（インデックス法）では，$O(rn)$ の記憶容量を用いることで，ナイーブな方式の $O(r^3)$ 時間のカバー導出を，$O(r\log r\log n)$ 時間にまで削減する．さらに，提案方式の効率を示すため，Java による提案アルゴリズム（インデックス法）の実装を行い，実際の処理時間の計測結果を報告する．したがって，本論文の貢献は，従来方式の SD 法の実装に関する新しいアルゴリズムの提案である．

以下の構成は次のとおりである．2 章で，本研究の対象である SD 法について説明し，カバー導出における問題点を示す．3 章で，新しいカバー導出アルゴリズムを提案し，その効率を 4 章で評価する．計算量を理論的に評価し，試験実装とその実行結果を述べ，それらの結果に基づく考察を与える．最後に，5 章で提案方式をまとめ，今後の課題を示す．

2. Subset Difference 方式

2.1 SD 法アルゴリズム

ユーザとリーフが 1 対 1 になるような完全 2 分木を考える．このとき木の深さを d，ユーザ総数を n とする．すなわち，深さ d の部分木 T に属するユーザ u は，T のリーフに割り当てられた 2^d 個に対応している．これを $u \in T$ と書き表す．

SD 法はユーザをグループ化するとき，失効ユーザをその配下の部分木に含めてカバーする方式である．SD 法ではデバイス鍵は増加するものの，暗号化回数は小さく抑えられる．

カバー $S_{i,j}$ は，v_i をルートとする部分木から v_j をルートとする部分木を除いた残りの部分木とする．$S_{i,j}$ に属するユーザの集合を**図 1** に示す．ノードに割り当てるデバイス鍵は，ラベルという乱数から，次のように定める．SD 法でのカバー $S_{i,j}$ に対応してデバイス鍵 $K_{i,j}$ を定める．疑似乱数生成器を 3 種類用意し，それぞれ左計算用を H_L，右計算用を H_R，鍵生成用を H_M で表す．たとえば，部分木のルート v_i から部分木のルート v_j に，分岐点での選択が，左，左，右でたどり着くとき，

$$Label_{i,j} = H_R(H_L(H_L(Label_i)))$$

で $Label_{i,j}$ を定めて，

$$K_{i,j} = H_M(Label_{i,j})$$

を計算する．ユーザはラベルを保持し，H_M を用いて必要なデバイス鍵を作成する．暗号化回数はすべての

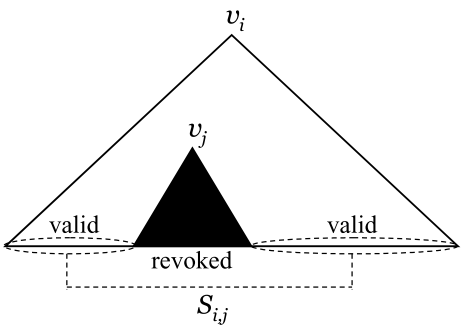


図 1 カバー $S_{i,j}$ と $S_{i,j}$ に属するユーザの集合
Fig. 1 Cover $S_{i,j}$.

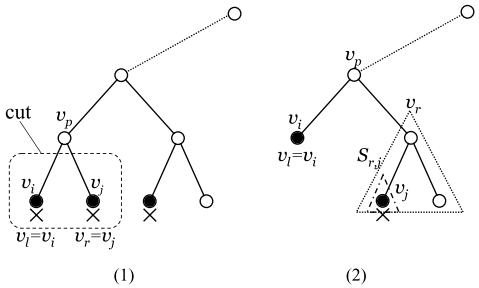


図 2 カバー導出アルゴリズム
Fig. 2 Cover-finding algorithm.

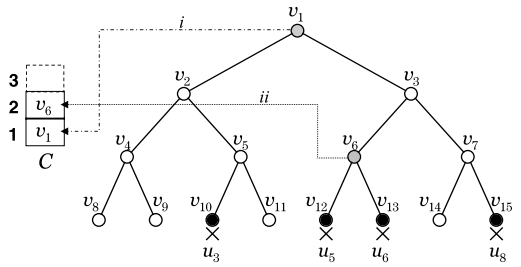


図 3 木 T とスタック C
Fig. 3 Tree T and stack C.

有効ユーザを包含するカバー $S_{i,j}$ の個数と等しい．

2.2 カバー導出アルゴリズム

部分木 T に r 個の失効ノードがあるとき，それらを除いて全有効ノードを包含する複数のカバー $S_{i,j}$ をすべて見つけるには，文献 15) による次のアルゴリズムが知られている．

1. 部分木 T の 2 つの失効ノード v_i，v_j の最も深い共通の親ノード（least-common-ancestor）を見つけそれを v_p とする．このとき，v_p をルートとする部分木の中に v_i，v_j 以外の失効ノードを含んでいてはならない（条件 1）．v_p の子ノード v_l の子孫には v_i を，v_r の子孫には v_j を含むようにそれぞれ v_l，v_r を定める（**図 2** (1)）．
2. $v_l \neq v_i$ ならば，部分木のカバーを $S_{l,i}$ を正しいカバーとして出力する．同様に，$v_r \neq v_j$ ならば，部分木のカバーを $S_{r,j}$ を出力する（図 2 (2)）．
3. v_p の子孫を T から削除し，v_p を新しい失効ノードとする．

このアルゴリズムを繰り返すことにより，すべてのカバー $S_{i,j}$ は一意に定まり，必ず求まる．

2.3 ナイーブなカバー導出アルゴリズム

しかしながら，条件 1（他の失効ノードを含まない）を満たした v_i と v_j を見つけるのは自明ではない．たとえば，**図 3** のように u_3，u_5，u_6，u_8 が失効ノードの場合，カバーを導出するための 2 つの失効ノードの組合せは (v_{10}, v_{12})，(v_{10}, v_{13})，(v_{10}, v_{15})，(v_{12}, v_{13})，(v_{12}, v_{15})，(v_{13}, v_{15}) の 6 組存在する．この中で，条件 1 を満たすものは (v_{12}, v_{13}) のみである．たとえば，隣り合っている v_{11} と v_{12} のようにノード間の距離が近くても，条件 1 を満たすとは限らない．この場合の共通親ノードはルートにまで遡ってしまう．

条件 1 を満たした 2 つのノードを探す最も単純で確実な方法は，r 個の失効ノードから総当たりで 2 個の組を検査することである．これをナイーブなカバー導出アルゴリズムと呼ぶ．この場合，1 回の検査に最悪

$$\binom{r}{2} = \frac{r(r-1)}{2} = O(r^2)$$

の試行が生じる．1 回の適用で 1 組 2 個のノードが削除され，新たな失効ノードが 1 個追加されるので，結局 1 個ずつ減っていく．この操作が最後の一組になるまで必ず $r-1$ 回繰り返されるため，ナイーブなカバー導出アルゴリズムの処理時間は，

$$(r-1)O(r^2) = O(r^3)$$

で与えられる．

3. 提 案 方 式

3.1 概　　要

最適な v_p を効率良く導出するためのカバー導出アルゴリズムを提案する．失効ユーザを木へ登録する過程で，各ノードにデータを溜めておき，v_p の検索範囲を制約する．

3.2 アルゴリズム（インデックス法）

入力：n 人のユーザをリーフとする木 T と，r 人の失効ユーザからなる集合 R.

リーフ以外の各ノードについて $n-1$ 個のノード集合 $L_1, L_2, \ldots, L_{n-1}$ を用意する．ノード集合は可変長のデータ構造とする．

Step 1. 各失効ユーザ $u \in R$ について，v_i を u の位置するノードとする☆．v_i からルートまでのパスにおける中間ノード $v_{i_1}, v_{i_2}, \ldots, v_1$ を求め，これらのノードに対応する各ノード集合 $L_{i_1}, L_{i_2}, \ldots, L_1$ のすべてに v_i を追加する．もしも，$|L_{i_*}| = 2$ となる i_* があれば，スタック C へノード v_* を積む．このとき，1 つのパス上で複数の i_* が条件を満たす場合はリーフに最も近いノードのみスタック C へ積む．

Step 2. スタック C から要素 v_k をポップし，対応するノード集合 $L_k = \{v_i, v_j\}$（たかだか 2 個の要

☆ ユーザ数 n の木において，ユーザ u_j はノード v_{n+j-1} に対応する．たとえば，図 3 のユーザ u_3 はノード v_{10} に位置している．

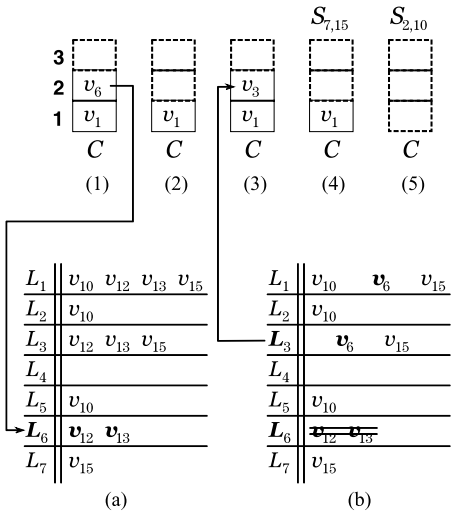


図 4 ノード集合 $L_1, \ldots, L_7$ の更新
Fig. 4 Updating node sets $L_1, \ldots, L_7$ and stack C.

素がある）に，2.2 節のカバー導出アルゴリズムを適用する．($|L_k| = 1$ のときはなにもしない)．カバー導出アルゴリズムの 2 の条件を満たした場合には，カバー $S_{i,j}$ を出力する．

Step 3. v_k から，ルートまでのパス上の中間ノード $v_{j_1}, v_{j_2}, \ldots, v_1$ のすべてのノード集合 L_{j_1}, $v_{j_2}, \ldots, L_1$ について，すべての v_i と v_j を v_k に置換する．このとき，$|L_{j_*}| = 2$ となる v_{j_*} があれば Step 1 と同様にスタック C へ積む．

Step 4. Step 2〜Step 3 を C の要素がなくなるまで繰り返す．このとき C の最後の要素である v_k がルート v_1 以外だった場合，カバー $S_{1,k}$ を出力する．

3.3 具 体 例

図 3 の $d = 3$, $n = 8$, u_3, u_5, u_6, u_8 が失効ユーザの場合を考える．

まず（Step 1），u_3 の位置するノード v_{10} からルートまでのパス上のノード集合 L_5, L_2, L_1 に v_{10} を追加する．同じように u_5, u_6, u_8 と順番に行っていき，要素数が 2 になった時点で C にノードを積んでいく．この場合は図 3 のように，v_1, v_6 の順に積まれる．u_6 に対応する v_{13} を積むと，L_6 と L_3 の要素数が 2 となりスタックの条件を満たすが，リーフに最も近い v_6 のみをスタックへ積むことに注意されたい．すべての R について行った結果を**図 4** の (a) に示す．

次に（Step 2），図 4 の (1) のように C の要素 v_6 をポップし，$L_6 = \{v_{12}, v_{13}\}$ を 2.2 節のカバー導出アルゴリズムに適用する (2)．

その後（Step 3），パス上の L_6, L_3, L_1 から v_{12}, v_{13} を親ノード v_6 に置換する (b)．このとき L_3 の要素数が 2 となるので C に積む (3)．同様にして，v_3 をポップしてカバー導出アルゴリズムにかけると，カバー $S_{7,15}$ を出力する (4)．

これを C の要素がなくなるまで繰り返し (5)，その過程で導出されたカバー $S_{7,15}$ と $S_{2,10}$ がアルゴリズムの出力である．

4. 評 価

4.1 提案アルゴリズムの完全性

完全性とは，提案アルゴリズムの出力するカバーがすべて「正しい」失効ユーザを包含しており，かつ，すべての失効ユーザは提案アルゴリズムで必ず出力されることである．いい換えれば，いかなる失効ユーザが与えられても，提案アルゴリズムが正しく処理できることを表している．ただし，正しい失効ユーザを冗長性なく包含するカバーを明確にするため次の定義を行う．

ユーザを表すリーフの集合を N，失効ユーザの集合 R をその部分集合とする．カバー S_p とは，ノード v_p をルートとする $N \setminus R$ の部分集合 $S_{p,q}$ である（失効ユーザの部分木のルートを v_q としている）．$N \setminus R$ を表すためには，複数のカバー $S_{p_1}, S_{p_2}, \ldots$ が必要である．カバーの間に自然に包含関係を定義することができる．たとえば図 3 において，$S_{5,10} \subset S_{2,10}$ であり，$\{v_8, v_9, v_{11}\}$ を表現するには，$S_{2,10}$ または，$S_{4,\phi} \cup S_{5,10}$ の 2 通りがある．これらの中で，包含関係において最大となるカバーの集合を極大カバー集合族とよぶ．

補題 4.1 $N \setminus R$ の極大カバー集合族は一意に決まる．

（証明） あるノード $v \in N \setminus R$ が異なる 2 つの極大カバー S_p, S_q について $v \in S_p \cap S_q$ であるとする．v から木のルートまでのパスは，木構造なので一意に決まり，v_p も v_q もこのパス上になくてはならない．仮定より $v_p \neq v_q$ なので，一般性を失うことなく v_p が v_q の祖先とすると，それは $S_q \subset S_p$ を表し，極大性の仮定に反する． □

補題 4.2 スタック C の先頭要素（最後に追加された要素）v は，つねに条件 1（2 個の失効ノードを含む）を満たす．

（証明） C の定義より，v は必ず 2 つ以上の失効ユーザの子孫を持つリーフに最も近い最下位層の親ノードである．3 つ以上存在するときは，それらより上に積まれるので，先頭要素は必ず 2 個の失効ノードを持つことを保証する． □

表 1 実装および評価環境
Table 1 System specification.

CPU	Pentium 4 2.4 GHz
Memory	512 MB
実行環境	Windows XP
開発言語	J2SDK 1.4.1

表 2 擬似乱数生成器の仕様
Table 2 Pseudo-random number generators.

擬似乱数生成器	アルゴリズム	ビット長
H_L	SHA-1	160 bit
H_R	SHA-256	256 bit
H_M	MD5	128 bit

定理 4.1 提案アルゴリズムの出力するカバー S ならば，$N \setminus R$ の極大カバー集合族であり，逆に，極大カバーならば必ず，アルゴリズムで出力される．

（証明） 補題 4.1 によりスタック C の先頭要素はつねに 2 個の失効ノードを含み，この条件 1 と Naor らのアルゴリズムの正当性により，出力が極大カバーであるのは明らか．極大カバーの一意性の補題 4.2 から，極大カバーならば必ず失効ノードを含み，すべての失効ノードはアルゴリズムのうえでノード集合の要素になっているので逆も示される． □

4.2 提案アルゴリズムの計算量

Step 1 では，ルートまでのパスに $O(\log n)$ のノードがあり，これが R の各ユーザについて r 回繰り返されるので，$O(r \log n)$ 時間かかる．

Step 2 のカバー導出は $O(1)$ 時間，Step 3 の置換はたかだか $\log n$ 回生じ，ノード集合 L は最悪 r 個の要素を持つ．(L の実装に 2 分探索木[19] などを仮定すると）1 回の置換は（v_i と v_j の）削除に $2 \log r$ と（v_k の）挿入に $\log r$ の時間で実行できる．したがって，Step 2 と 3 で

$$O(1) + (3 \log r) \log n = O(\log r \cdot \log n)$$

の時間がかかり，これがスタック C の要素分繰り返される．スタック C は 1 回のポップで R から 1 人ずつ減っていくので，正確に $r - 1$ 回生じる．

結局，全体では

$$O(r \log n) + (r-1)O(\log r \cdot \log n)$$
$$= O(r \log r \cdot \log n)$$

が提案アルゴリズムの計算量である．

4.3 試験実装

提案方式の有効性を実証するため Java で試験実装を行った．ノード集合とカバーリングの導出部分には，主に Stack と Vector クラスを用いた．Swing によるインタフェースを実装した．これによって失効ユーザの番号を入力するだけで簡単に鍵生成に必要なカバーの組合せを導出することができる．

本実装の仕様を**表 1** に，2.1 節で示した擬似乱数生成器の仕様を**表 2** に示す．3 種類のハッシュ関数は処理性能を評価する目的で，広く一般的に採用されているものから選んだ．ここでは乱数生成器として用いているので，安全性などの考慮はしていない．

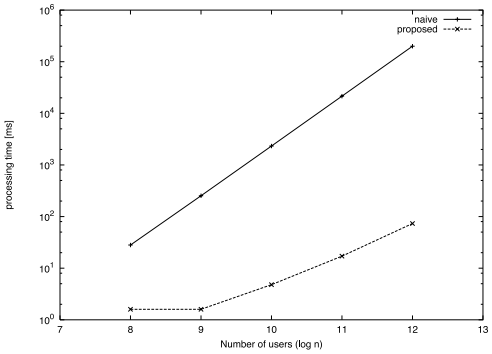


図 5 ユーザ規模による処理時間の変化（$r = 0.2n$）
Fig. 5 Processing time with respects to the depth of tree $d(= \log n)$ (with $r = 0.2n$).

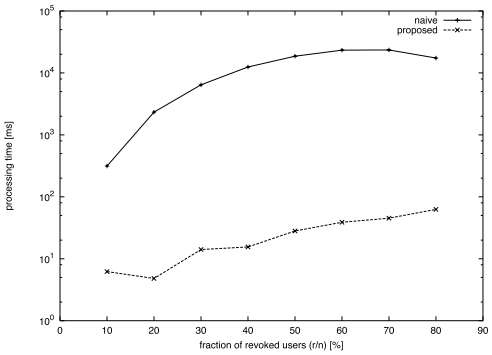


図 6 失効ユーザ数による処理時間の変化（$n = 1024$）
Fig. 6 Processing time with respects to a number of revoked users ($n = 1024$).

4.4 処理時間

図 5 に，$r = 0.2n$ 固定で $n(= 2^d)$ を変化させたカバーリングの処理時間を示す．2.3 節と 3.2 節の提案アルゴリズムを各々，“naive” と “proposed” で示す．**図 6** は $n = 1024$ のとき r の割合を変化させた処理時間を示す．図 5 と図 6 は，y 軸が対数スケールになっていることに注意されたい．また，測定値は R をランダムに変えて 10 回実行したときの平均である．また，**図 7** は深さを変えたときの処理時間を示している．図 5 では R を毎回ランダムに割り当てたのに対し，図 7 では初めに失効ユーザ R を定め，すべて同じ失効ユーザで測定を行っている．

本実験の処理速度が 4.2 節で見積もった提案アルゴリズムの計算量と一致しているかを検証するために，

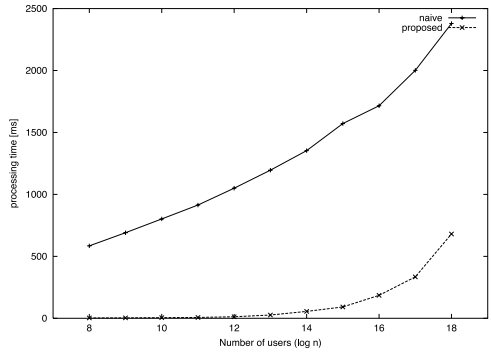


図 7 特定の失効ユーザによる処理時間の変化（$r = 128$）
Fig. 7 Processing time given revoked users ($r = 128$).

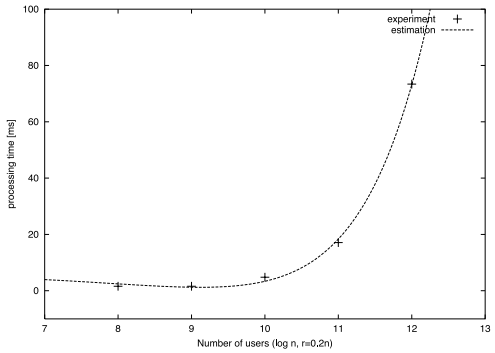


図 8 理論値との比較
Fig. 8 Comparison between the experiments and the estimation.

理論値との比較を行った．ユーザ数 n の増加に対して，失効ユーザ数は $r = 0.2n$ で一定の割合で推移することを仮定した．文献 3) による日本の平均不正コピー率が 29%であったので，その一部が失効されたと見なして，$0.2n$ と定めた．また，0.1 から 0.9 まで評価を行い，0.2 のときに提案方式が大きな差を生じることを確認している．このとき，$d = \log n$ とおくと計算量は

$$\begin{aligned} r \log r \log n &= 0.2n(\log 0.2 + \log n) \log n \\ &= a2^d \cdot d^2 + b2^d \cdot d + c \\ &= 0.83 \cdot 10^{-3} \cdot 2^d \cdot d^2 \\ &\quad +0.86 \cdot 10^{-2} \cdot 2^d \cdot d + 6.46 \end{aligned}$$

で与えられる．ここで，a，b，c は定数である．実測値に最小二乗法を適用して求めた結果を**図 8** に示す．

4.5 考　　察

以上の実験結果より，提案方式はナイーブな方式に比べて，処理時間に関して大きな改善を与えることが実証された．ただし，実際の処理時間は，ユーザ数 n，木の深さ d，失効ユーザ数 r に大きく依存するので，これらの値を変えて実測を行い，ほとんどの場合で顕著な差が生じることを観測した．たとえば，図 5 においては，深さ 12 のとき，ナイーブな場合の 2500 倍高速であった．

ただし，理論的に導いた処理時間に対しては，小さな誤差を生じた．たとえば図 8 における不一致や，図 6 や図 7 における不規則な処理時間の変動などである．これらの原因は，ランダムに選んだ失効ユーザの位置や評価環境における処理系のオーバヘッドなどが考えられる．特に，提案方式はノード集合に $r(n-1)$ のメモリを消費するので，これがメモリの再配置などを生じさせている可能性がある．

本評価では，10 回の測定の平均値を用いている．このときの標準偏差を見る限り，提案方式とナイーブな方式の間には有意な差があり，測定ミスや初期値の影響はほとんどないといえる．

5. おわりに

効率的な不正者の失効を実現する SD 法におけるカバー導出の課題を取り上げ，$O(r^3)$ の計算量を $O(r \log r \log n)$ にまで削減する効率的なアルゴリズムを提案した．Java による実装に基づき，実際の計算量の増加を計測し，その有効性を検証した．提案方式は理論的な計算量を低減できたが，インデキシングのためのメモリ消費量が大きくなる．これに対して，たとえば，付録に記載のノードの符号化に 2 進木を用いる方式を用いると，計算量は $O(r^2 \log n)$ 要するが，メモリ消費を提案手法より抑えられる可能性がある．条件によっては，このような 2 進木方式の方が適している場合も考えられる．これら理論的な計算量とメモリ消費などを考慮した実際の計算時間の比較・評価などについては今後の課題としたい．

謝辞 本研究の遂行にあたって多大な協力をいただいた東海大学電子情報学部中西祥八郎教授と匿名の査読者に感謝する．

参 考 文 献

1) 中村雄一，菊池浩明：セキュアなコンテンツ配信を行う Subset Difference 法の実装について，*Proc. DICOMO 2003*, pp.173–176, 情報処理学会 (2004).
2) Nakamura, Y. and Kikuchi, H.: Efficient Key Management Based on the Subset Difference Method for Secure Group Communication, *Proc. AINA2005*, pp.707–712, IEEE (2005).
3) 2004 Piracy Study（第 1 回世界ソフトウェア違法コピー調査），Business Software Alliance. http://www.bsa.or.jp/file/PiracyStudy_E.pdf

4) Fiat, A. and Naor, M.: Broadcast Encryption, LNCS 7734, pp.480–491, Springer (1994).
5) Content Protection for Recordable Media Specification. http://www.4centity.com/tech/cprm/
6) Wong, C.K., Gouda, M. and Lam, S.S.: Secure group communications using key graphs, *Proc. ACM SIGCOMM'98*, pp.68–79 (1998).
7) Kaya, T., Lin, G., Noubir, G. and Yilmaz, A.: Secure Multicast Groups on Ad Hoc Networks, *Proc. 1st ACM Workshop Security of Ad Hoc and Sensor Networks*, pp.94–103 (2003).
8) Wallner, D., Harder, E. and Agee, R.: Key Management for Multicast: Issues and Architectures, Internet RFC 2627 (1999).
9) Tseng, Y.-M.: A scalable key-management scheme with minimizing key storage for secure group communications, Vol.13, pp.419–4251 (2003).
10) Perrig, A., Song, D. and Tygar, J.D.: ELK, a New Protocol for Efficient Large-Group Key Distribution, *Proc. 2001 IEEE Symposium on Security and Privacy*, pp.247–262, IEEE (2001).
11) Kim, Y., Perrig, A. and Tsudik, G.: Simple and Foult-Tolerant Key Agreement for Dynamic Collaborative Groups, *Proc. ACM Conference on Computer and Communication Security* (2000).
12) Kogan, N., Shavitt, Y. and Wool, A.: A Practical Revocation Scheme for Broadcast Encryption Using Smart Cards, *Proc. 2003 IEEE Symposium on Security and Privacy*, pp.225–235, IEEE (2003).
13) Asano, T.: A Revocation Scheme with Minimal Storage at Reveivers, *Proc. ASIACRYPT 2002*, LNCS 2501, pp.433–450, Springer (2002).
14) 野島 良，楫 勇一：落とし度付き一方向性関数を利用した木構造鍵管理方式，2003 年暗号とセキュリティシンポジウム（SCIS）予稿集，pp.131–136 (2003).
15) Naor, D., Naor, M. and Lotspiech, J.: Revocation and Tracing Schemes for Stateless Receivers, in Advances in Cryptology *Crypto 2001*, LNCS 2139, pp.41–62, Springer (2001).
16) Halevy, D. and Shamir, A.: The LSD Broadcast Encryption Scheme, in Advances in Cryptology, *Crypto 2002*, LNCS 2442, pp.47–60, Springer (2002).
17) 奥秋清次，サントソバグス，太田和夫：Complete Subtree Method と Subset Difference Method を融合した Hybrid System の提案，2003 年暗号とセキュリティシンポジウム（SCIS）予稿集，pp.221–226 (2003).
18) Hwang, Y.H., Kim, C.H. and Lee, P.L.: An Efficient Revocation Scheme with Minimal Message Length for Stateless Receivers, *Proc. ACISP 2003*, LNCS 2727, pp.377–386, Springer (2003).
19) Cormen, T., Leiserson, C. and Rivest, R.: *Introduction to algorithms*, MIT Press (1990).

付　　録

A.1　2 進木方式☆

ユーザを 2 進木に割り当て，各ノードに**図 9** のように 2 進数のノード番号を割り当てる．失効ユーザの集合が与えられると，失効ユーザに対応するノード番号の 2 進表現の集合が得られる．ここで，2 進木の上層のノード番号については，2 進木の最下位層から数えただけの桁数を左シフトする．そして，求められたノード集合を大きい順にソートする．その後に，ソートされたノード番号を隣のノード番号と比較し，最上位ビットから数えて何番目まで一致するかを求め，その値を T とする．そして，さらにその隣のノード番号の T 番目までとノード番号が一致するかを調べる．一致しなければ，初めの 2 つのノードを含む最小部分木に他の失効ユーザが含まれることはない．よって，その 2 つのノードをカバー導出アルゴリズムに適応すればよい．

たとえば，$n = 8$ の木は図 9 のようになる．ここで，失効ユーザを $u0$，$u2$，$u3$，$u5$ とする．$u5$ と $u3$ を比較すると，101 と 011 より $T = 0$．一方，$u3(011)$ と $u2(010)$ を比較すると，01 まで一致して $T = 2$．次に，$u3(011)$ と $u0(000)$ を比較すると，$T = 1 < 2$

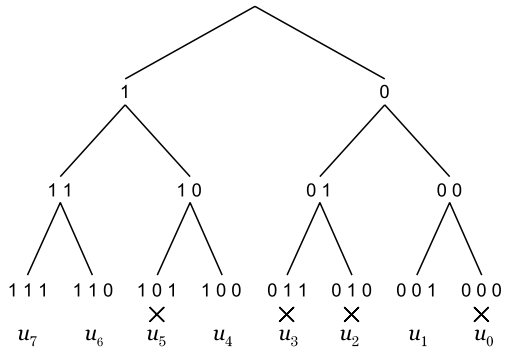


図 9　2 進木方式
Fig. 9　The binary tree scheme.

☆ 本方式は，文献 1) の投稿に対する匿名の査読者により提示されたものである．

なので，$u3$ と $u2$ を含む最小部分木に他の失効ユーザが含まれないことが分かる．よって，$u3$ と $u2$ をカバー導出アルゴリズムに適用する．

この方式では，ソートに $O(r\log r)$ かかり[19]，与えられた 2 つのノード番号から一致するビット数を求めるのに，$O(\log n)$ の計算量がかかる．この比較は $O(r)$ 回行われる．全体で，$r-1$ 回繰り返されるので，

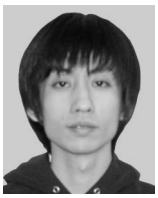
$$O(r\log r)+O(\log n)O(r)(r-1)$$
$$=O(r^2\log n)$$

の計算量になる．

(平成 17 年 5 月 10 日受付)
(平成 17 年 11 月 1 日採録)

菊池 浩明（正会員）

1988 年明治大学工学部電子通信工学科卒業．1990 年同大学院博士前期課程修了．1990 年（株）富士通研究所入社．1994 年東海大学工学部電気工学科助手．1995 年同専任講師．1999 年同助教授，1997 年カーネギーメロン大学計算機科学学部客員研究員．2000 年電子情報学部情報メディア学科助教授，現在に至る．博士（工学）．ファジィ理論，多値論理，ネットワークセキュリティに興味を持つ．1990 年日本ファジィ学会奨励賞，1993 年情報処理学会奨励賞，1996 年 SCIS 論文賞，2004 年情報処理学会研究開発奨励賞受賞．電子情報通信学会，日本ファジィ学会，IEEE，ACM 各会員．



中村 雄一

2004 年東海大学工学部電気工学科卒業．現在，東海大学大学院工学研究科博士課程前期在学中．ネットワークセキュリティ，コンテンツ保護の研究に従事．