(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2024年8月29日(29.08.2024)

WIPO | PCT

(10) 国際公開番号
WO 2024/176307 A1

(51) 国際特許分類:
G09C 1/00 (2006.01)

(21) 国際出願番号： PCT/JP2023/006016

(22) 国際出願日： 2023年2月20日(20.02.2023)

(25) 国際出願の言語： 日本語

(26) 国際公開の言語： 日本語

(71) 出願人: 日本電信電話株式会社 (**NIPPON TELEGRAPH AND TELEPHONE CORPORATION**) [JP/JP]; 〒1008116 東京都千代田区大手町一丁目５番１号 Tokyo (JP).

(72) 発明者: 市川 敦謙 (**ICHIKAWA, Atsunori**); 〒1808585 東京都武蔵野市緑町３丁目９－１１ ＮＴＴ知的財産センタ内 Tokyo (JP).

(74) 代理人: 伊東 忠重, 外 (**ITOH, Tadashige** et al.); 〒1000005 東京都千代田区丸の内二丁目１番１号 丸の内 ＭＹ ＰＬＡＺＡ （明治安田生命ビル） １６階 Tokyo (JP).

(81) 指定国(表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR,

(54) **Title:** DATA MANAGEMENT SYSTEM, METHOD, AND PROGRAM

(54) 発明の名称： データ管理システム、方法、及びプログラム

[図5]

Algorithm 4 $(k, v) \leftarrow \mathrm{Access}(\mathrm{op}, k, v')$

Require: $\mathrm{op} \in \{0, 1\}$; 0 = 'read', 1 = 'write'.

1: クライアントは $\ell = 0, \ldots, L-1$ で $\Phi_\ell.\mathrm{membership}(k)$ を実行し，ターゲットレベル t; $\mathrm{argmin}_t(k \in \Phi_t)$ を得る．ただし，全ての ℓ で $k \notin \Phi_\ell$ の場合，$t = L$ とする．

2: クライアントは t 番目の値が 1 となる長さ $L+1$ の One-hot ベクトル $\vec{d}$ を生成し，これをシェア $[\![\vec{d}]\!] = ([\![d_0]\!], \ldots, [\![d_L]\!])$ として，$[\![\mathrm{op}]\!], [\![k]\!], [\![v']\!]$ と共にサーバへ分散する．

3: サーバは各 $\ell = 0, \ldots, L$ において**並列に**
$[\![a_\ell]\!] \leftarrow \mathrm{Lookup}([\![k]\!], [\![d_\ell]\!], [\![T_\ell]\!], R_\ell, [\![s_\ell]\!], \mathrm{ctr}_\ell)$ を実行する
$(a_\ell = (k_\ell, v_\ell)$ は キー・バリューペア).

4: サーバはベクトル $[\![\vec{a}]\!] = ([\![a_0]\!], \ldots, [\![a_L]\!])$ と $[\![\vec{d}]\!]$ の内積 $[\![\tilde{a}]\!] \leftarrow \mathrm{Prod}([\![\vec{a}]\!], [\![\vec{d}]\!])$ を計算する．

5: サーバは $[\![\tilde{a}]\!]$ をクライアントへ送信する．

6: クライアントは $(k, v) \leftarrow \mathrm{Reveal}([\![\tilde{a}]\!])$ を出力として得る．

7: サーバは $[\![\vec{a}]\!] \leftarrow \mathrm{IfElse}([\![\mathrm{op}]\!], ([\![k]\!], [\![v']\!]), [\![\vec{a}]\!])$ として $\mathrm{Reshuffle}([\![\vec{a}]\!])$ を実行する．

1: Client executes $\Phi\ell.\mathrm{membership}(k)$ where ℓ=0, ..., L-1, and obtains:
target level t; and argmint ($k \in \Phi t$), however in the case of $k \notin \Phi\ell$ for all ℓ, t= L.
2: Client generates One-hot vector $\rightarrow$d of length L+1 with a t-th value of 1,
One-hot vector $\rightarrow$d is distributed to a server together with [[op]], [[k]], [[v']] as share [[$\rightarrow$d]]=([[d0]], ..., [[dL]]).
3: Server executes each of [[aℓ]]$\leftarrow$Lookup([[k]],[[dℓ]],[[Tℓ]],Rℓ,[[sℓ]],ctrℓ) (aℓ=(kℓ, vℓ)
is key value pairs) in parallel in ℓ=0, ..., L.
4: Server calculates inner product [[~a]]$\leftarrow$Prod([[$\rightarrow$a]],[[$\rightarrow$d]]) of vector [[$\rightarrow$a]]=([[a0]], ..., [[aL]]) with [[$\rightarrow$d]].
5: Server transmits [[~a]] to client.
6: Client obtains (k,v)$\leftarrow$Reveal([[~a]]) as output.
7: Server executes Reshuffle([[~a]]) with [[~a]]$\leftarrow$IfElse([[op]],([[k]],[[v']]),[[~a]]).

(57) **Abstract:** A data management system according to one aspect of the present disclosure in which a multi-server ORAM is realized by a client and a plurality of servers that mutually execute a secure computation protocol, wherein: the aforementioned client includes a specifying unit that specifies a layer t $\in$ {0,..., L-1} to which a key k of the data to be accessed belongs, using L data structures Φ_0,..., Φ_{L-1} representing dictionaries, and a distribution unit that distributes, to the plurality of servers, a secret value [[d]] of a One-hot vector d of length L+1 where the t-th value is 1, a secret value [[op]] of op that indicates whether the access operation is read or write, and a secret value [[k]] of the key k; and the server includes an acquisition unit that, using arrays [[T_0]],..., [[T_L]] that store anonymized data represented by key-value pairs, plaintext arrays R_0,..., R_L that store auxiliary information for referring to [[T_l]], the secret value [[d]], the secret value [[op]], and the secret value [[k]], acquires a secret value [[a_l]] of a_l=(k_l, v_l) that matches the key k in parallel at each l=0,..., L from the secret value [[T_l]], and an inner product calculation unit that transmits, to the client, secret values [[~a]] of the inner product of the secret values [[a]] = ([[a_0]],..., [[a_L]]) with the

WO 2024/176307 A1

［続葉有］

LS, LU, LY, MA, MD, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) 指定国(表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, RU, TJ, TM), ヨーロッパ (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

添付公開書類:
一　国際調査報告（条約第21条(3)）

secret value [[d]].

(57) 要約：本開示の一態様によるデータ管理システムは、クライアントと互いに秘密計算プロトコルを実行する複数のサーバとによりマルチサーバＯＲＡＭを実現するデータ管理システムであって、前記クライアントは、辞書を表すＬ個のデータ構造Φ_0，・・・，Φ_{L-1}を用いて、アクセス対象のデータのキーｋが所属する階層ｔ∈｛０，・・・，Ｌ－１｝を特定する特定部と、ｔ番目の値が１となる長さＬ＋１のＯｎｅ－ｈｏｔベクトルｄの秘密値［［ｄ］］と、アクセス操作がｒｅａｄ又はｗｒｉｔｅのいずれであるかを表すｏｐの秘密値［［ｏｐ］］と、前記キーｋの秘密値［［ｋ］］とを前記複数のサーバに分散する分散部と、を有し、前記サーバは、キー・バリューペアで表される秘匿化されたデータを格納する配列［［T_0］］，・・・，［［T_L］］と、［［T_l］］を参照するための補助情報を格納する平文の配列R_0，・・・，R_Lと、前記秘密値［［ｄ］］と、前記秘密値［［ｏｐ］］と、前記秘密値［［ｋ］］とを用いて、各ｌ＝０，・・・，Ｌにおいて並列に前記キーｋに合致するa_l＝（k_l，v_l）の秘密値［［a_l］］を秘密値［［T_l］］から取得する取得部と、秘密値［［ａ］］＝（［［a_0］］，・・・，［［a_L］］）と、前記秘密値［［ｄ］］との内積の秘密値［［~ａ］］を前記クライアントに送信する内積計算部と、を有する。

明　細　書

発明の名称 ： データ管理システム、方法、及びプログラム

技術分野

[0001] 本開示は、データ管理システム、方法、及びプログラムに関する。

背景技術

[0002] Ｏｂｌｉｖｉｏｕｓ　ＲＡＭ（以下、ＯＲＡＭという。）は、データの持ち主であるクライアントが、データの預け先であるストレージサーバに対して安全にアクセスを行うためのアルゴリズムである。ＯＲＡＭは、データへの安全なアクセス（検索）を保証するという性質から高安全なクラウドストレージへの応用等が期待されている。その一方で、クライアントとサーバの間で大きな通信量、通信ラウンド及びクライアント側の計算資源を要するため、必ずしもクライアントの通信環境や端末性能が優れないクラウドサービスでの利用は容易ではないという問題がある。

[0003] 上記の問題を解決するため、クライアントに或る程度の空間コストを要求する代わりに通信量と通信ラウンドを削減する手法（非特許文献１）と、複数台のサーバを用いる手法（非特許文献２）との２つの手法が提案されている。

先行技術文献

非特許文献

[0004] 非特許文献1：William Holland, Olga Ohrimenko, and Anthony Wirth. Single round-trip hierarchical oram via succinct indices, 2022.

非特許文献2：Thang Hoang, Ceyhun D. Ozkaptan, Attila A. Yavuz, Jorge Guajardo, and Tam Nguyen. S3oram: A computation-efficient and constant client bandwidth blowup oram with shamir secret sharing. In CCS, pp. 491-505, 2017.

発明の概要

発明が解決しようとする課題

[0005] しかしながら、非特許文献１に記載されている手法は、通信ラウンドを定数まで削減することができるが、クライアントの記憶容量が大きく、かつ、クライアントとサーバ間の通信量も大きいという問題がある。一方で、非特許文献２に記載されている手法は、クライアントとサーバ間の通信量及び通信ラウンドはともに定数で、かつ、クライアントに大きな記憶容量は不要だが、サーバ間の通信ラウンドが大きいという問題がある。

[0006] 本開示は、上記の点に鑑みてなされたもので、効率的なＯＲＡＭを実現する技術を提供する。

課題を解決するための手段

[0007] 本開示の一態様によるデータ管理システムは、クライアントと互いに秘密計算プロトコルを実行する複数のサーバとによりマルチサーバＯＲＡＭを実現するデータ管理システムであって、前記クライアントは、辞書を表すＬ個のデータ構造Φ_0，・・・，Φ_{L-1}を用いて、アクセス対象のデータのキーｋが所属する階層ｔ∈｛０，・・・，Ｌ－１｝を特定する特定部と、ｔ番目の値が１となる長さＬ＋１のＯｎｅ－ｈｏｔベクトルｄの秘密値［［ｄ］］と、アクセス操作がｒｅａｄ又はｗｒｉｔｅのいずれであるかを表すｏｐの秘密値［［ｏｐ］］と、前記キーｋの秘密値［［ｋ］］とを前記複数のサーバに分散する分散部と、を有し、前記サーバは、キー・バリューペアで表される秘匿化されたデータを格納する配列［［T_0］］，・・・，［［T_L］］と、［［T_l］］を参照するための補助情報を格納する平文の配列R_0，・・・，R_Lと、前記秘密値［［ｄ］］と、前記秘密値［［ｏｐ］］と、前記秘密値［［ｋ］］とを用いて、各ｌ＝０，・・・，Ｌにおいて並列に前記キーｋに合致するa_l＝（k_l，v_l）の秘密値［［a_l］］を秘密値［［T_l］］から取得する取得部と、秘密値［［ａ］］＝（［［a_0］］，・・・，［［a_L］］）と、前記秘密値［［ｄ］］との内積の秘密値［［~ａ］］を前記クライアントに送信する内積計算部と、を有する。

発明の効果

[0008] 効率的なＯＲＡＭを実現する技術が提供される。

図面の簡単な説明

[0009] [図1]本実施形態に係るデータ管理システムの全体構成の一例を示す図である。

[図2]Ｂｕｉｌｄアルゴリズムの一例を示す図である。

[図3]Ｌｏｏｋｕｐアルゴリズムの一例を示す図である。

[図4]Ｅｘｔｒａｃｔアルゴリズムの一例を示す図である。

[図5]Ａｃｃｅｓｓアルゴリズムの一例を示す図である。

[図6]Ｒｅｓｈｕｆｆｌｅアルゴリズムの一例を示す図である。

[図7]コンピュータのハードウェア構成の一例を示す図である。

発明を実施するための形態

[0010] 以下、本発明の一実施形態について説明する。

[0011] ＜準備＞

まず、本実施形態の説明に必要な技術について説明する。

[0012] ≪ＯＲＡＭ≫

ＯＲＡＭは、データの持ち主たるクライアントと、データを預かる（信頼できない）サーバとの間で実行されるデータアクセスプロトコルである。ＯＲＡＭは、主に、データアクセス時にサーバ側で観測されるメモリ参照位置（これはアクセスパターンとも呼ばれる。）の秘匿を目的としている。

[0013] ＯＲＡＭの機能は次の関数F_{RAM}で定義される。

[0014] F_{RAM}（ｏｐ，ｋ，ｖ’）：内部状態として容量Ｎの配列Ｘを持ち、ｏｐの値に応じて以下の（１）又は（２）を実行する。

[0015] （１）ｏｐ＝ｒｅａｄである場合、配列Ｘから組（ｋ，ｖ）を探し、出力する。配列Ｘに組（ｋ，ｖ）が存在しないときはダミー（⊥，⊥）を出力する。

[0016] （２）ｏｐ＝ｗｒｉｔｅである場合、配列Ｘから組（ｋ，ｖ）を削除し、組（ｋ，ｖ’）で置き換える。配列Ｘに組（ｋ，ｖ）が存在しないときは単に配列Ｘに組（ｋ，ｖ’）を追加する。

[0017] なお、（ｋ，ｖ）はキー・バリューペアやブロック等とも呼ばれ、データ

の１単位として扱われる。

[0018] アクセスの正当性のため各キーｋは相異なる必要があるため、データの総数をＮとすれば、ブロックサイズはＢ＝Ω（ｌｏｇＮ）ビット必要である。ＯＲＡＭではしばしば１回のアクセス（すなわち、１ブロックの取得）の際にアルゴリズムが必要とするブロック操作回数をオーバーヘッドと呼び、これを計算効率の指標としている。

[0019] ≪Ｃｏｍｐｌｅｔｅ ｄｉｃｔｉｏｎａｒｙ≫

Ｃｏｍｐｌｅｔｅ ｄｉｃｔｉｏｎａｒｙ Φは、或る全体集合｛１，・・・，Ｎ｝に対して、その任意の部分集合Ｓ⊆｛１，・・・，Ｎ｝を保持し、また所属判定クエリΦ．ｍｅｍｂｅｒｓｈｉｐ（ｋ）＝［ｋ∈?Ｓ］に対して応答できるデータ構造である。ここで、所属判定クエリΦ．ｍｅｍｂｅｒｓｈｉｐ（ｋ）とは、任意のｋ∈｛１，・・・，Ｎ｝に対して、Ｃｏｍｐｌｅｔｅ ｄｉｃｔｉｏｎａｒｙ Φが保持する集合Ｓにｋが含まれるか否かの判定を要求するクエリのことである。

[0020] また、Ｃｏｍｐｌｅｔｅ ｄｉｃｔｉｏｎａｒｙ Φの空間コスト（つまり、データ構造のビット長）は、Ｓがｍ個のデータを保持する場合にはＯ（ｍ×ｌｏｇ（Ｎ／ｍ））で抑えることができる。

[0021] 以上のようなデータ構造は、例えば、参考文献１等に記載されている方法で実現することができる。ただし、これに限られるものではなく、同等の機能を持つ方法であれば任意の方法で代替可能である。

[0022] ≪秘密計算≫

秘密計算は、データを暗号化したまま様々な計算を行う技術である。本実施形態では、以下で説明する積和、条件付き選択、置換、逆置換、疑似ランダム置換が少なくとも計算可能な秘密計算を用いる。これらの計算を可能とする秘密計算の一例としては、秘密分散法（参考文献２）を利用した手法等が挙げられる。以下では、秘密分散法を利用したマルチパーティ計算（ＭＰＣ：multi-party computation）を想定する。ただし、これに限られるものではなく、同等の機能を持つ手法であれば任意の手法を用いることが可能であ

る。なお、秘密分散法を利用したマルチパーティ計算は、例えば、秘匿マルチパーティ計算等と呼ばれてもよい。

[0023] 以下、或る値ａを秘匿化した状態を

[0024] [数1]

$$[[a]]$$

で表すものとし、これを「ａのシェア」又は単に「シェア」と呼ぶ。また、明細書のテキスト中では、ａのシェアを［［ａ］］と表記することにする。

[0025] また、ａから［［ａ］］に変換する処理［［ａ］］←Ｓｈａｒｅ（ａ）を「分散」と呼び、［［ａ］］からａに変換する処理ａ←Ｒｅｖｅａｌ（［［ａ］］）を「復元」と呼ぶ。更に、配列Ａ＝（a_1，・・・，a_n）のシェア（［［a_1］］，・・・，［［a_n］］）は、簡単のため、単に［［Ａ］］と記述する。

[0026] ・積和

［［Ａ］］、［［Ｂ］］をそれぞれ要素数ｎの配列のシェアとしたとき、両者の積和（内積）を以下のように表す。

[0027] ［［ｃ］］←Ｐｒｏｄ（［［Ａ］］，［［Ｂ］］）

これには、例えば、参考文献３に記載されている方法を用いることができる。

[0028] ・条件付き選択

条件付き選択を以下のように記述する。

[0029] ［［ｚ］］←ＩｆＥｌｓｅ（［［ｅ］］，［［ｘ］］，［［ｙ］］）

ここで、入力［［ｘ］］、［［ｙ］］はｕビットのシェア、［［ｅ］］は１ビットのシェアとする。出力はｅ＝１ならばｚ＝ｘ、そうでなければｚ＝ｙを満たす。これには、例えば、参考文献４に記載されている方法を用いることができる。

[0030] ・置換と逆置換

［［Ａ］］を要素数ｎの配列のシェアとし、［［π］］を置換π：［ｎ］→［ｎ］のシェアとする。このとき、［Ａ］の置換と逆置換をそれぞれ以下で表す。

[0031] ［［πＡ］］←Ｐｅｒｍ（［［π］］，［［Ａ］］）

［［π^{-1}Ａ］］←Ｕｎｐｅｒｍ（［［π］］，［［Ａ］］）

これには、例えば、参考文献５に記載されている方法を用いることができる。

[0032] ・疑似ランダム置換

秘密の鍵［［ｓ］］とシェア［［ｘ］］から疑似乱数値のシェアを得る処理を以下で表す。

[0033] ［［ｒ］］←ＰＲＰ（［［ｓ］］，［［ｘ］］）

これには、例えば、参考文献６に記載されている方法を用いることができる。

[0034] 以上の準備の下、以下、効率的なＯＲＡＭを実現するデータ管理システム１について説明する。

[0035] ＜データ管理システム１の全体構成例＞

本実施形態に係るデータ管理システム１の全体構成例を図１に示す。図１に示すように、本実施形態に係るデータ管理システム１には、クライアント１０と、複数のサーバ２０とが含まれている。また、クライアント１０と各サーバ２０は、例えば、インターネット等の通信ネットワーク３０を介して相互に通信可能に接続される。なお、図１に示す例では、サーバ２０_{1}、サーバ２０_{2}、・・・、サーバ$２０_{M'}$が存在する場合を示している。ここで、Ｍ’（≧２）はサーバ２０の総数である。以下では、各サーバ$２０_{m'}$（m’∈｛１，・・・，Ｍ’｝）を総称して単に「サーバ２０」と表記する。

[0036] クライアント１０は、ＯＲＡＭのクライアントに相当するコンピュータ又はコンピュータシステムである。クライアント１０には、例えば、ＰＣ（パーソナルコンピュータ）、スマートフォン、タブレット端末、ウェアラブルデバイス、ゲーム機器、車載器、産業用機器等を用いることができる。

[0037] サーバ２０は、ＯＲＡＭのサーバに相当するコンピュータ又はコンピュータシステムである。サーバ２０には、例えば、データベースサーバやクラウドストレージサーバ等として機能する汎用サーバ等を用いることができる。

[0038] ここで、本実施形態に係るクライアント１０は、演算部１０１と、記憶部１０２とを有する。演算部１０１は、例えば、クライアント１０にインストールされた１以上のプログラムが、ＣＰＵ（Central Processing Unit）等の演算装置に実行させる処理により実現される。記憶部１０２は、例えば、ＨＤＤ（Hard Disk Drive）、ＳＳＤ（Solid State Drive）、フラッシュメモリ等のストレージ装置により実現される。

[0039] 演算部１０１は、関数F_{RAM}をシミュレートするアルゴリズム（後述するＡｃｃｅｓｓアルゴリズム）に含まれる一部の手順を実行する。

[0040] 記憶部１０２は、辞書（辞書群）として利用される後述するデータ構造Φ_0，・・・，Φ_{L-1}を記憶する。

[0041] また、本実施形態に係るサーバ２０は、演算部２０１と、記憶部２０２とを有する。演算部２０１は、サーバ２０にインストールされた１以上のプログラムが、ＣＰＵ等の演算装置に実行させる処理により実現される。記憶部２０２は、例えば、ＨＤＤ、ＳＳＤ、フラッシュメモリ等のストレージ装置により実現される。なお、図１に示す例では、サーバ$２０_{m'}$（$m' \in \{1, \cdots, M'\}$）の演算部２０１及び記憶部２０２をそれぞれ「$演算部２０１_{m'}$」及び「$記憶部２０２_{m'}$」と表している。

[0042] 演算部２０１は、後述するデータ構造の構築や参照を行うための各種アルゴリズム（後述するＢｕｉｌｄアルゴリズム、Ｌｏｏｋｕｐアルゴリズム、Ｅｘｔｒａｃｔアルゴリズム）に含まれる手順を実行する。また、演算部２０１は、上記のＡｃｃｅｓｓアルゴリズムに含まれる一部の手順を実行する。

[0043] 記憶部２０２は、秘匿化されたキー・バリューペアを保持するための後述するデータ構造$[[T_0]], \cdots, [[T_L]]$と、補助情報を保持するための後述するデータ構造$R_0, \cdots, R_L$とを記憶する。また、これら以

外にも、記憶部２０２は、後述するランダム置換や秘密鍵、クエリカウンタ等も記憶する。

[0044] ＜データ構造＞

クライアント１０が持つデータ構造とサーバ２０が持つデータ構造について説明する。

[0045] ≪クライアント１０が持つデータ構造≫

クライアント１０は、Ｌ個のデータ構造Φ_0，・・・，Φ_{L-1}を持つ。すなわち、クライアント１０は、Ｌ段の階層型データ構造Φ_l（ｌは小文字の「Ｌ」）を持つ。ただし、

[0046] [数2]

$$L = \lceil \log N \rceil$$

である。

[0047] また、各Φ_lは最大で2^l個の要素を保持するものとする。このため、各Φ_lの空間コストはＯ（2^l×ｌｏｇ（Ｎ／2^l））となる。以下では、各Φ_lはＣｏｍｐｌｅｔｅ　ｄｉｃｔｉｏｎａｒｙであるものとする。ただし、これに限られるものではなく、同等の機能を持つデータ構造であれば任意のデータ構造で代替可能である。

[0048] ここで、Φ_0，・・・，Φ_{L-1}の合計空間コストに関して以下の式が成り立つ。

[0049]

[数3]

$$O\left(\sum_{\ell=0}^{L-1} 2^{\ell} \log \frac{N}{2^{\ell}}\right) = O\left(\sum_{\ell=0}^{L-1} 2^{\ell}(L-\ell)\right)$$
$$= O\left(\sum_{i=0}^{L-1} \sum_{\ell=0}^{i} 2^{\ell}\right)$$
$$= O(N)$$

このため、クライアント１０に要求される記憶部２０２の記憶容量（ストレージサイズ）はＯ（Ｎ）ビットである。

[0050] ≪サーバ２０が持つデータ構造≫

サーバ２０は、秘匿化されたＬ＋１個の配列［［T_0］］，・・・，［［T_L］］；｜T_l｜＝2^{l+1}と、平文のＬ＋１個の配列R_0，・・・，R_L；｜R_l｜＝2^{l+1}とを持つ。すなわち、サーバ２０は、Ｌ＋１段の階層型データ構造である配列［［T_l］］及び配列R_lを持つ。

[0051] 配列［［T_l］］には秘匿化されたキー・バリューペア（［［ｋ］］，［［ｖ］］）が格納され、また配列R_lには［［T_l］］を参照するための補助情報が格納される。更に、サーバ２０は、配列［［T_l］］を構築した際の情報として秘密のランダム置換［［π_l］］と秘密鍵［［s_l］］とクエリカウンタｃｔｒとを持つ。

[0052] ＜配列［［T_l］］、R_lの構築及び参照方法＞

本実施形態では、サーバ２０が保持する階層型データ構造はランダム置換を利用したシンプルなアルゴリズムにより管理できる。以下で説明するＢｕｉｌｄアルゴリズム、Ｌｏｏｋｕｐアルゴリズム及びＥｘｔｒａｃｔアルゴリズムはいずれもクライアント１０との通信に基づく補助入力や補助計算を必要とせず、サーバ２０間での秘密計算（つまり、例えば、秘密分散法を利用したＭＰＣ）で実現される。

[0053] ≪Ｂｕｉｌｄアルゴリズム≫

以下では、表記の簡単のため或るｌに関する［［T_l］］、R_lをそれぞれ［［Ｔ］］、Ｒとして、［［Ｔ］］、Ｒを構築するためのＢｕｉｌｄアルゴリズム（Ａｌｇｏｒｉｔｈｍ１）を図２に示す。図２に示すＢｕｉｌｄアルゴリズムは、［［Ａ］］を入力として、［［Ｔ］］，Ｒ，［［ｓ］］，［［π］］，ｃｔｒを出力する。ただし、［［Ａ］］＝（［［a_1］］，・・・，［［a_n］］）、［［a_i］］＝（［［k_i］］，［［v_i］］）、$k_i \in [N]$、かつ、全てのｉ，ｊ≠ｉについて$k_i \neq \perp$ならば$k_i \neq k_j$であるものとする。また、このＢｕｉｌｄアルゴリズムの実行によってＡ⊆Ｔが保証される。なお、［［Ａ］］は、例えば、クライアント１０から与えられる。

[0054] まず、サーバ２０の演算部２０１は、ｎ個のダミーデータのシェア［［Ｄ］］＝（（⊥，⊥），・・・，（⊥，⊥））；｜Ｄ｜＝ｎを生成し、

[0055] [数4]

$$[[\tilde{T}]] = [[A]] \| [[D]]$$

とする（１行目）。ここで、|| は配列の連結を表す。例えば、要素数ｎの配列をＡ，Ｂとすれば、Ｃ＝Ａ||Ｂは、先頭のｎ要素が配列Ａの要素、残りのｎ要素が配列Ｂの要素である要素数２ｎの配列となる。以下、明細書のテキスト中では、文字の上に付与された「～」を文字の左上に記載する。例えば、上記の数４の左辺を明細書のテキスト中に記載する場合、［［~Ｔ］］と記載する。

[0056] 次に、サーバ２０の演算部２０１は、ランダムなシェア［［ｓ］］を生成し、［［~Ｒ］］＝（［［$\tilde{r}_1$］］，・・・，［［$\tilde{r}_{2n}$］］）；［［$\tilde{r}_i$］］＝ＰＲＰ（［［ｓ］］，［［k_i］］）　ｆｏｒ　１≦ｉ≦ｎ，［［$\tilde{r}_j$］］＝ＰＲＰ（［［ｓ］］，［［⊥＋ｊ－ｎ］］）　ｆｏｒ　ｎ＋１≦ｊ≦２ｎを計算する（２行目）。

[0057] 次に、サーバ２０の演算部２０１は、ランダムな置換のシェア［［π］］を生成し、［［Ｔ］］←Ｐｅｒｍ（［［π］］，［［~Ｔ］］）と、［［Ｒ］

]←Ｐｅｒｍ（[[π]]，[[~R]]）とを計算する（３行目）。

[0058] 次に、サーバ２０の演算部２０１は、R←Ｒｅｖｅａｌ（[[R]]）とし、またクエリカウンタｃｔｒ＝０を準備する（４行目）。

[0059] そして、サーバ２０の演算部２０１は、[[T]]、R、[[s]]、[[π]]、ｃｔｒを自身の記憶部２０２に保存する（５行目）。

[0060] このように、Ａｌｇｏｒｉｔｈｍ１では、入力であるサイズｎの配列に同数のダミーを追加した上でランダム置換を行うことでアクセス位置を秘匿可能にしている。これによって構築される配列[[T]]は、”各要素a_iごとに””重複なく一度まで”、最大ｎ回の参照を許容すると共に、適宜ダミーの参照も（最大ｎ回まで）可能としている。また、参照のための情報として疑似ランダム置換を用いたタグ付け（つまり、配列Ｒ）を行い、同一の置換πの元で配列Ｔと要素の並び順をリンクさせている。

[0061] ≪Ｌｏｏｋｕｐアルゴリズム≫

クエリｋに合致するデータを[[T]]から取得するためのＬｏｏｋｕｐアルゴリズム（Ａｌｇｏｒｉｔｈｍ２）を図３に示す。図３に示すＬｏｏｋｕｐアルゴリズムは、[[k]]，[[d]]，[[T]]，R，[[s]]，ｃｔｒを入力として、（[[k]]，[[v]]）を出力する。ただし、d∈{0，1}であるものとする。なお、ｄは、クエリがダミーか否かを表すフラグであり、参照位置に制御に用いられる。

[0062] まず、サーバ２０の演算部２０１は、クエリカウンタをｃｔｒ＝ｃｔｒ＋１と更新する（１行目）。

[0063] 次に、サーバ２０の演算部２０１は、[[~k]]←ＩｆＥｌｓｅ（[[d]]，[[⊥＋ｃｔｒ]]，[[k]]）を計算する（２行目）。

[0064] 次に、サーバ２０の演算部２０１は、[[r]]←ＰＲＰ（[[s]]，[[~k]]）を計算し、r←Ｒｅｖｅａｌ（[[r]]）を得る（３行目）。

[0065] 次に、サーバ２０の演算部２０１は、配列Ｒ＝（r_1，・・・，r_{2n}）からr_i＝ｒなる値を探す。該当するr_iが存在しなければ停止（つまり、Ａｌｇ

ｏｒｉｔｈｍ２を終了）する（４行目）。

[0066] そして、サーバ２０の演算部２０１は、［［Ｔ］］＝（［［t_1］］，・・・，［［t_{2n}］］）から（［［ｋ］］，［［ｖ］］）＝［［t_i］］を取得し、これを出力する。また、サーバ２０の演算部２０１は、［［t_i］］を（［［⊥］］，［［⊥］］）で上書きする（５行目）。

[0067] このように、Ａｌｇｏｒｉｔｈｍ２では、クエリｋに対応するタグ（疑似乱数値）ｒを計算・公開し、同一の値となるr_iを配列Ｒから探索する。このとき、データt_i＝（ｋ，ｖ）が配列Ｔに含まれる場合には必ずインデックスを持つr_iが存在する。また、ダミークエリである場合にはアクセス毎に別のダミーを参照する必要があるため、クエリカウンタにより参照位置を制御している。

[0068] ≪Ｅｘｔｒａｃｔアルゴリズム≫

配列［［Ａ］］を取り出すためのＥｘｔｒａｃｔアルゴリズム（Ａｌｇｏｒｉｔｈｍ３）を図４に示す。図４に示すＥｘｔｒａｃｔアルゴリズムは、［［Ｔ］］，［［π］］を入力として、［［Ａ］］を出力する。

[0069] サーバ２０の演算部２０１は、［［~Ｔ］］←Ｕｎｐｅｒｍ（［［π］］，［［Ｔ］］）を計算し、［［~Ｔ］］の先頭のｎ要素を配列［［Ａ］］として出力する（１行目）。

[0070] このように、Ａｌｇｏｒｉｔｈｍ３では、配列［［Ｔ］］の構築で利用した置換πの逆を行うことで、入力配列Ａを取り出す。これにより、［［Ｔ］］に残る全ての非ダミーデータを取り出すことができる。なお、Ｌｏｏｋｕｐアルゴリズムで参照されたデータはダミーによって上書きされていることに留意されたい。

[0071] ≪Ｂｕｉｌｄ、Ｌｏｏｋｕｐ及びＥｘｔｒａｃｔアルゴリズムの効率≫

上記のＡｌｇｏｒｉｔｈｍ１，３はそれぞれＯ（ＮＢ）ビットの通信量とＯ（１）ラウンドで実行できる。また、上記のＡｌｇｏｒｉｔｈｍ２はＯ（Ｂ）ビットの通信量とＯ（１）ラウンドで実行できる。

[0072] ＜データ構造全体のアクセス＞

次に、クライアント１０が、サーバ２０が保持するデータ構造にアクセスするためのアルゴリズム（Ａｃｃｅｓｓアルゴリズム）について説明する。このＡｃｃｅｓｓアルゴリズムは、上記のＢｕｉｌｄアルゴリズム、Ｌｏｏｋｕｐアルゴリズム、Ｅｘｔｒａｃｔアルゴリズムをサブルーチンとして用いる。また、このＡｃｃｅｓｓアルゴリズムは、配列［［Ｔ］］のアクセス回数の進んだ階層の解体・再構築を行うためのＲｅｓｈｕｆｆｌｅアルゴリズムもサブルーチンとして用いる。以下、Ａｃｃｅｓｓアルゴリズム、Ｒｅｓｈｕｆｆｌｅアルゴリズムについて説明する。なお、以下では、キー・バリューペアがサーバ２０側のデータ構造で保持されており、またクライアント１０側はサーバ２０のデータ構造とリンクしたデータ構造（辞書群）を有しているものとして説明する。

[0073] ≪Ａｃｃｅｓｓアルゴリズム≫

クライアント１０がサーバ２０のデータ構造にアクセスするためのＡｃｃｅｓｓアルゴリズム（Ａｌｇｏｒｉｔｈｍ４）を図５に示す。図５に示すＡｃｃｅｓｓアルゴリズムは、ｏｐ，ｋ，ｖ’を入力として、（ｋ，ｖ）を出力する。

[0074] まず、クライアント１０の演算部１０１は、ｌ＝０，・・・，Ｌ－１でΦ_l．ｍｅｍｂｅｒｓｈｉｐ（ｋ）を実行し、ターゲットレベルｔ；ａｒｇｍｉｎ$_t$（ｋ∈Φ_t）を得る（１行目）。ただし、ｋ∈Φ_lとなるｌが存在しない場合、クライアント１０の演算部１０１は、ｔ＝Ｌとする。なお、ターゲットレベルｔとは、キーｋが含まれる集合Ｓを保持するデータ構造Φ_lの階層（レベル）のことである。

[0075] 次に、クライアント１０の演算部１０１は、ｔ番目の値が１となる長さＬ＋１のＯｎｅ－ｈｏｔベクトル

[0076] [数5]

$$\vec{d}$$

を生成し、シェア

[0077] [数6]

$$[[\vec{d}]] = ([[d_0]], \ldots, [[d_L]])$$

として、［［ｏｐ］］，［［ｋ］］，［［ｖ’］］と共にサーバ２０に分散する（２行目）。以下、明細書のテキスト中では、文字の上に付与された「→」を文字の左上に記載する。例えば、上記の数５を明細書のテキスト中に記載する場合、→ｄと記載する。

[0078] 次に、サーバ２０の演算部２０１は、各Ｉ＝０，・・・，Ｌにおいて並列に［［a_I］］←Ｌｏｏｋｕｐ（［［ｋ］］，［d_I］，［［T_I］］，R_I，［［s_I］］，ctr_I）を実行する（３行目）。なお、a_I＝（k_I，v_I）はキー・バリューペアである。

[0079] 次に、サーバ２０の演算部２０１は、ベクトル［［→ａ］］＝（［［a_0］］，・・・，［［a_L］］）と［［→ｄ］］の内積［［~ａ］］←Ｐｒｏｄ（［［→ａ］］，［［→ｄ］］）を計算する（４行目）。

[0080] 次に、サーバ２０の演算部２０１は、［［~ａ］］をクライアント１０に送信する（５行目）。

[0081] そして、クライアント１０の演算部１０１は、（ｋ，ｖ）←Ｒｅｖｅａｌ（［［~ａ］］）を出力として得る（６行目）。

[0082] また、サーバ２０の演算部２０１は、［［~ａ］］←ＩｆＥｌｓｅ（［［ｏｐ］］，（［［ｋ］］，［［ｖ’］］），［［~ａ］］）としてＲｅｓｈｕｆｆｌｅ（［［~ａ］］）を実行する（７行目）。

[0083] このように、Ａｌｇｏｒｉｔｈｍ４は、サーバ２０が保持するデータ構造にクライアント１０がアクセスするためのアルゴリズムであり、サーバ２０がクライアント１０の補助入力（つまり、ベクトル→ｄ）を受けて実行する秘密計算プロトコルである。クライアント１０は自身が持つ辞書群Φ_0，・・・，Φ_{L-1}からターゲットとなるデータが存在する階層（レベル）を特定し、こ

れをベクトル$\vec{d}$という形でサーバ２０に与える。これにより、サーバ２０は（$\vec{d}$の値は知らないままでも）秘密計算により効率的なデータアクセスが可能となる。

[0084] ≪Ｒｅｓｈｕｆｆｌｅアルゴリズム≫

配列［［Ｔ］］のアクセス回数の進んだ階層の解体・再構築を行うためのＲｅｓｈｕｆｆｌｅアルゴリズム（Ａｌｇｏｒｉｔｈｍ５）を図６に示す。図６に示すＲｅｓｈｕｆｆｌｅアルゴリズムは、［［~ａ］］を入力する。ここで、以下では、Ｒｅｓｈｕｆｆｌｅアルゴリズムが実行された時点で、或るレベルｐにおいて配列［［T_p］］が未構築（つまり、データを保持していない状態）であり、かつ、全ての［［T_0］］，・・・，［［T_{p-1}］］が構築済みであるものとする。

[0085] まず、サーバ２０の演算部２０１は、全てのｌ＝０，・・・，ｐ－１において［［A_l］］←Ｅｘｔｒａｃｔ（［［T_l］］，［［π_l］］）を並列に実行し、［［A_p］］＝［［A_0］］||・・・||［［A_{p-1}］］||（［［~ａ］］）とする（１行目）。これにより、各［［T_l］］は未構築状態に戻る。

[0086] そして、サーバ２０の演算部２０１は、［［T_p］］，R_p，［［s_p］］，［［π_p］］，ｃｔｒ←Ｂｕｉｌｄ（［［A_p］］）によりレベルｐのデータ構造を構築する（２行目）。

[0087] このように、Ａｌｇｏｒｉｔｈｍ５は、アクセス回数の進んだ階層を解体・再構築するためのアルゴリズムであり、アクセス済み要素の配置のランダマイズを行う。すなわち、構築済みの上層のデータ構造を全て解体し、下層へと押し込めることで、再度上層のデータ構造を利用可能としている。本アルゴリズムによるレベルｐの再構築は２ｐ回のアクセスごとに一度実行される。また、サーバ２０側の秘密計算プロトコルである本アルゴリズムと並行して、クライアント１０側では、辞書群Φ_0，・・・Φ_{p-1}の解体とΦ_pへの統合が行われる。この辞書群Φ_0，・・・Φ_{p-1}の解体とΦ_pへの統合は、上記の非特許文献１に記載されている方法と同様に行うことができる。この解体・統合の対象となるレベルｐはアクセスの実行回数から一意であるため、クラ

イアント１０は、サーバ２０と通信することなく、サーバ２０のデータ構造とリンクしたデータ構造（辞書群）に更新することができる。

[0088] ≪Ａｃｃｅｓｓ及びＲｅｓｈｕｆｆｌｅアルゴリズムの効率≫

まず、サーバ２０側に掛かるコストとしては、Ａｌｇｏｒｉｔｈｍ５では２^{p}に一度Ｏ（２^{p}Ｂ）ビット、Ｏ（１）ラウンドの通信を要するため、償却通信量はＯ（Ｂ×ｌｏｇＮ）ビット、償却ラウンド数はｏ（１）となる。したがって、Ａｌｇｏｒｉｔｈｍ４全体の効率は償却通信量Ｏ（Ｂ×ｌｏｇＮ）ビット、償却ラウンド数ｏ（１）となる。一方、クライアント１０側に掛かるコストは、Ｏ（Ｌ＋Ｂ）ビットの通信量とＯ（１）通信ラウンドとなる。ブロックサイズＢ＝Ω（ｌｏｇＮ）ビットすれば、本実施形態のクライアント１０－サーバ２０間のオーバーヘッドはＯ（１）、サーバ２０間のオーバーヘッドはＯ（ｌｏｇＮ）となる。

[0089] ＜ハードウェア構成例＞

本実施形態に係るデータ管理システム１に含まれるクライアント１０及びサーバ２０は、例えば、図７に示すコンピュータ５００のハードウェア構成により実現することができる。図７に示すコンピュータ５００は、入力装置５０１と、表示装置５０２と、外部Ｉ／Ｆ５０３と、通信Ｉ／Ｆ５０４と、ＲＡＭ（Random Access Memory）５０５と、ＲＯＭ（Read Only Memory）５０６と、補助記憶装置５０７と、プロセッサ５０８とを有する。また、これらの各ハードウェアは、それぞれがバス５０９を介して通信可能に接続されている。

[0090] 入力装置５０１は、例えば、キーボード、マウス、タッチパネル、物理ボタン等である。表示装置５０２は、例えば、ディスプレイ、表示パネル等である。なお、コンピュータ５００は、例えば、入力装置５０１及び表示装置５０２のうちの少なくとも一方を有していなくてもよい。

[0091] 外部Ｉ／Ｆ５０３は、記録媒体５０３ａ等の外部装置とのインタフェースである。コンピュータ５００は、外部Ｉ／Ｆ５０３を介して、記録媒体５０３ａの読み取りや書き込み等を行うことができる。記録媒体５０３ａとして

は、例えば、フレキシブルディスク、ＣＤ（Compact Disc）、ＤＶＤ（Digital Versatile Disk）、ＳＤメモリカード（Secure Digital memory card）、ＵＳＢ（Universal Serial Bus）メモリカード等が挙げられる。

[0092] 通信Ｉ／Ｆ５０４は、コンピュータ５００を通信ネットワークに接続させるためのインタフェースである。ＲＡＭ５０５は、プログラムやデータを一時保持する揮発性の半導体メモリ（記憶装置）である。ＲＯＭ５０６は、電源を切ってもプログラムやデータを保持することができる不揮発性の半導体メモリ（記憶装置）である。補助記憶装置５０７は、例えば、ＨＤＤ、ＳＳＤ、フラッシュメモリ等のストレージ装置（記憶装置）である。プロセッサ５０８は、例えば、ＣＰＵ等の演算装置である。

[0093] 本実施形態に係るデータ管理システム１に含まれるクライアント１０及びサーバ２０は、例えば、図７に示すコンピュータ５００のハードウェア構成を有することにより、上述した各種処理を実現することができる。ただし、図７に示すコンピュータ５００のハードウェア構成は一例であって、これに限られるものではない。例えば、コンピュータ５００は、複数の補助記憶装置５０７や複数のプロセッサ５０８を有していてもよいし、図示したハードウェアの一部を有していなくてもよいし、図示したハードウェア以外の様々なハードウェアを有していてもよい。

[0094] ＜既存技術との比較＞

上記の非特許文献１に記載されている既存技術は、単一サーバ、かつ、クライアント－サーバ間の通信ラウンドが定数（Ｏ（１））であるが、クライアントにＯ（Ｎ＋$\sqrt{}$（Ｎ）Ｂ）ビットの記憶容量が必要であり、かつ、クライアント－サーバ間にＯ（ｌｏｇＮ）の通信量が必要である。

[0095] 一方で、上記の非特許文献２に記載されている既存技術は、複数台のサーバが必要だがクライアント－サーバ間の通信量・通信ラウンドが共に定数、かつ、クライアントに大きな記憶容量が不要であるが、サーバ同士での通信量・通信ラウンドが共にＯ（ｌｏｇＮ）となる。

[0096] 上記の既存技術に対して、本実施形態に係るデータ管理システム１では、

既存技術のそれぞれの要素を複合しており、定数ラウンドのマルチサーバＯＲＡＭを実現している。既存技術に対する本実施形態に係るデータ管理システム１の貢献は主に以下の２点である。

[0097] ・定数ラウンドとクライアント負荷軽減の両立

本実施形態に係るデータ管理システム１では、クライアント１０のストレージサイズをＯ（Ｎ）ビットに削減し、かつ、クライアント１０－サーバ２０間のラウンド数・通信量が共に定数（Ｏ（１））となる。

[0098] ・サーバ間で定数ラウンドかつ通信量Ｏ（ｌｏｇＮ）のマルチサーバＯＲＡＭ

本実施形態に係るデータ管理システム１では、サーバ２０間の通信量がＯ（ｌｏｇＮ）、ラウンド数は定数（Ｏ（１））となり、アクセスプロトコル全体で通信ラウンド数がＯ（１）である。

[0099] ＜まとめ＞

以上のように、本実施形態に係るデータ管理システム１は、既存技術と比較して効率的なマルチサーバＯＲＡＭを実現することができる。このため、本実施形態に係るデータ管理システム１を用いて、安全かつ効率的な分散データベースを構築することが可能になり、例えば、安全かつ効率的なクラウドストレージサービス等への応用が期待できる。

[0100] 本発明は、具体的に開示された上記の実施形態に限定されるものではなく、請求の範囲の記載から逸脱することなく、種々の変形や変更、既知の技術との組み合わせ等が可能である。

[0101] ［参考文献１］

参考文献１：Rasmus Pagh. Low redundancy in static dictionaries with constant query time. SIAM Journal on Computing, Vol. 31, No. 2, pp. 353-363, 2001.

参考文献２：Adi Shamir. How to share a secret. Commun. ACM, Vol. 22, No. 11, pp. 612-613, 1979.

参考文献３：Koji Chida, Daniel Genkin, Koki Hamada, Dai Ikarashi, R

yo Kikuchi, Yehuda Lindell, and Ariel Nof. Fast large-scale honest-majority mpc for malicious adversaries. In CRYPTO, pp. 34-64, 2018.

参考文献４：Marcel Keller and Peter Scholl. Efficient, oblivious data structures for mpc. In ASIACRYPT, pp. 506-525, 2014.

参考文献５：Koji Chida, Koki Hamada, Dai Ikarashi, Ryo Kikuchi, Naoto Kiribuchi, and Benny Pinkas. An efficient secure three-party sorting protocol with an honest majority. Cryptology ePrint Archive, 2019.

参考文献６：Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. Ciphers for mpc and fhe. In EUROCRYPT, pp. 430-454, 2015.

符号の説明

[0102]

１	データ管理システム
１０	クライアント
２０	サーバ
３０	通信ネットワーク
１０１	演算部
１０２	記憶部
２０１	演算部
２０２	記憶部
５００	コンピュータ
５０１	入力装置
５０２	表示装置
５０３	外部Ｉ／Ｆ
５０３ａ	記録媒体
５０４	通信Ｉ／Ｆ
５０５	ＲＡＭ
５０６	ＲＯＭ
５０７	補助記憶装置

５０８　　プロセッサ

５０９　　バス

請求の範囲

[請求項1] クライアントと互いに秘密計算プロトコルを実行する複数のサーバとによりマルチサーバＯＲＡＭを実現するデータ管理システムであって、

前記クライアントは、

辞書を表すＬ個のデータ構造Φ_0，・・・，Φ_{L-1}を用いて、アクセス対象のデータのキーｋが所属する階層ｔ∈｛０，・・・，Ｌ－１｝を特定する特定部と、

ｔ番目の値が１となる長さＬ＋１のＯｎｅ－ｈｏｔベクトルｄの秘密値［［ｄ］］と、アクセス操作がｒｅａｄ又はｗｒｉｔｅのいずれであるかを表すｏｐの秘密値［［ｏｐ］］と、前記キーｋの秘密値［［ｋ］］とを前記複数のサーバに分散する分散部と、を有し、

前記サーバは、

キー・バリューペアで表される秘匿化されたデータを格納する配列［［T_0］］，・・・，［［T_L］］と、［［T_l］］を参照するための補助情報を格納する平文の配列R_0，・・・，R_Lと、前記秘密値［［ｄ］］と、前記秘密値［［ｏｐ］］と、前記秘密値［［ｋ］］とを用いて、各ｌ＝０，・・・，Ｌにおいて並列に前記キーｋに合致するa_l＝（k_l，v_l）の秘密値［［a_l］］を秘密値［［T_l］］から取得する取得部と、

秘密値［［ａ］］＝（［［a_0］］，・・・，［［a_L］］）と、前記秘密値［［ｄ］］との内積の秘密値［［~ａ］］を前記クライアントに送信する内積計算部と、

を有するデータ管理システム。

[請求項2] 前記クライアントは、

前記秘密値［［~ａ］］を復元して前記キーｋと前記アクセス対象のデータｖとのキー・バリューペア（ｋ，ｖ）を計算する復元部、を更に有する請求項１に記載のデータ管理システム。

[請求項3] 前記分散部は、

前記ｏｐが表すアクセス操作がｗｒｉｔｅである場合に書き込む値ｖ’の秘密値［［ｖ’］］を更に前記複数のサーバに分散し、

前記サーバは、

前記秘密値［［ｏｐ］］と、前記秘密値［［ｋ］］と、前記秘密値［［ｖ’］］とを用いて、秘匿化されたキー・バリューペアが格納されているｐ個の前記配列［［T_0］］，・・・，［［T_{p-1}］］と、（ｋ，ｖ’）の秘密値とをキー・バリューペアが未だ格納されていない前記配列［［T_p］］に格納する格納部、を更に有する請求項１又は２に記載のデータ管理システム。

[請求項4] 前記格納部は

前記配列［［T_0］］，・・・，［［T_{p-1}］］に格納されている秘匿化されたキー・バリューペアの配列をそれぞれ［［A_0］］，・・・，［［A_{p-1}］］として、［［A_0］］，・・・，［［A_{p-1}］］と（ｋ，ｖ’）を要素とする配列とを結合した配列の秘密値［［A_p］］を計算し、

前記秘密値［［A_p］］と、前記秘密値［［A_p］］の要素数と同数のダミーデータを要素とする配列の秘密値［［Ｄ］］とを結合した配列の秘密値［［~Ｔ］］を計算し、

前記秘密値［［~Ｔ］］と、前記秘密値［［~Ｔ］］と同数の疑似乱数値を要素とする配列の秘密値［［~Ｒ］］とをそれぞれランダム置換し、

前記ランダム置換した前記秘密値［［~Ｔ］］を配列［［T_p］］、前記ランダム置換した前記秘密値［［~Ｒ］］を復元した値ＲをR_pとして記憶部に保持する請求項３に記載のデータ管理システム。

[請求項5] 前記取得部は、

前記秘密値［［ｄ］］＝（［［d_0］］，・・・，［［d_{L+1}］］）を用いて、［［d_I］］により前記キーに合致するデータがダミー

データであるか否かを判定し、ダミーデータでない場合に前記キーｋに合致するａ$_{I}$＝（ｋ$_{I}$，ｖ$_{I}$）の秘密値［［ａ$_{I}$］］を秘密値［［Ｔ$_{I}$］］から取得する、請求項４に記載のデータ管理システム。

［請求項6］ クライアントと互いに秘密計算プロトコルを実行する複数のサーバとによりマルチサーバＯＲＡＭを実現するデータ管理システムに用いられる方法であって、

前記クライアントが、

辞書を表すＬ個のデータ構造Φ$_{0}$，・・・，Φ$_{L-1}$を用いて、アクセス対象のデータのキーｋが所属する階層ｔ∈｛０，・・・，Ｌ－１｝を特定する特定手順と、

ｔ番目の値が１となる長さＬ＋１のＯｎｅ－ｈｏｔベクトルｄの秘密値［［ｄ］］と、アクセス操作がｒｅａｄ又はｗｒｉｔｅのいずれであるかを表すｏｐの秘密値［［ｏｐ］］と、前記キーｋの秘密値［［ｋ］］とを前記複数のサーバに分散する分散手順と、を実行し、

前記サーバが、

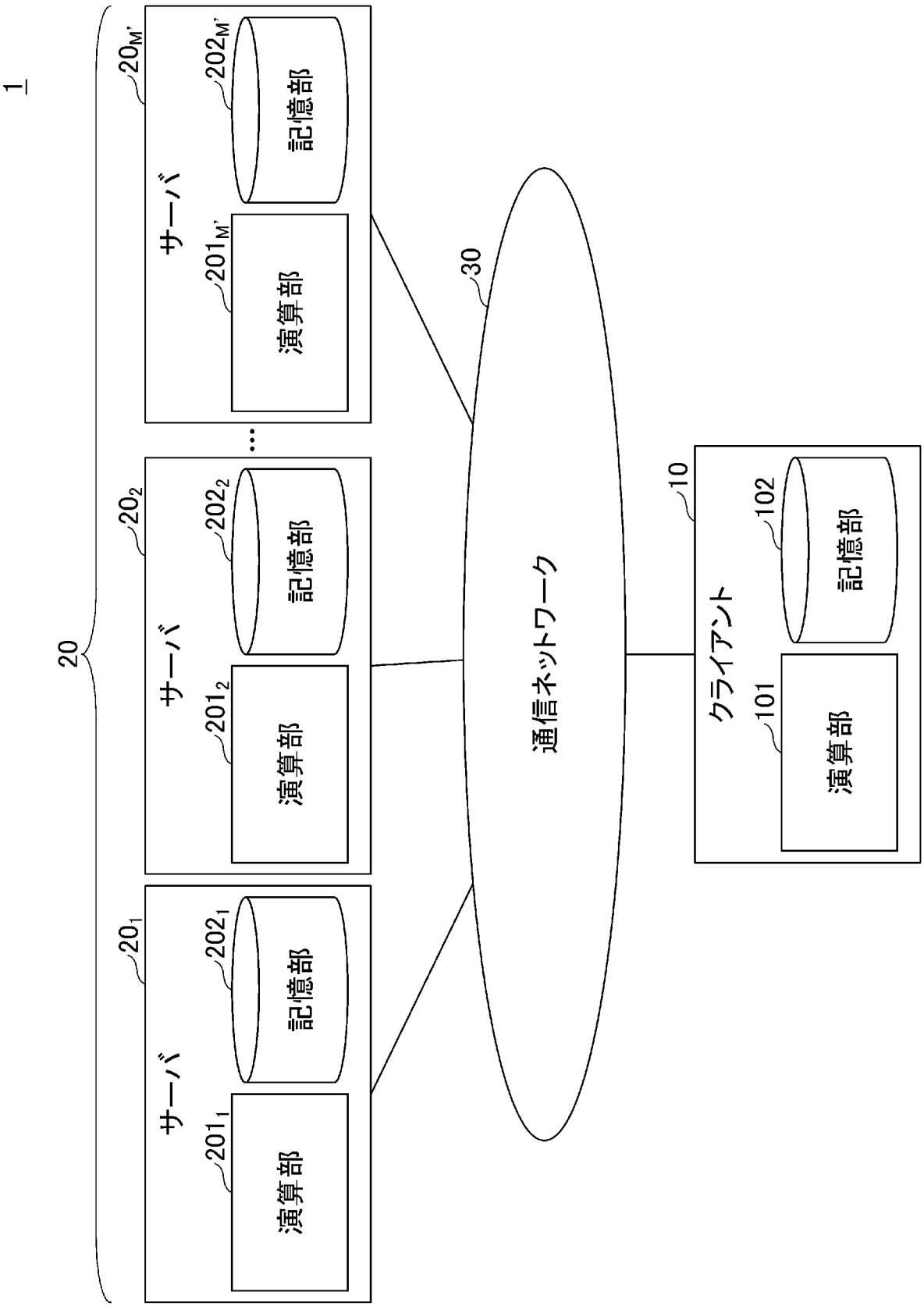
キー・バリューペアで表される秘匿化されたデータを格納する配列［［Ｔ$_{0}$］］，・・・，［［Ｔ$_{L}$］］と、［［Ｔ$_{I}$］］を参照するための補助情報を格納する平文の配列Ｒ$_{0}$，・・・，Ｒ$_{L}$と、前記秘密値［［ｄ］］と、前記秘密値［［ｏｐ］］と、前記秘密値［［ｋ］］とを用いて、各Ｉ＝０，・・・，Ｌにおいて並列に前記キーｋに合致するａ$_{I}$＝（ｋ$_{I}$，ｖ$_{I}$）の秘密値［［ａ$_{I}$］］を秘密値［［Ｔ$_{I}$］］から取得する取得手順と、

秘密値［［ａ］］＝（［［ａ$_{0}$］］，・・・，［［ａ$_{L}$］］）と、前記秘密値［［ｄ］］との内積の秘密値［［~ａ］］を前記クライアントに送信する内積計算手順と、

を実行する方法。

［請求項7］ コンピュータを、請求項１に記載のデータ管理システムに含まれるクライアント又はサーバとして機能させるプログラム。

[図1]


1
20
20_1
サーバ
201_1
演算部
202_1
記憶部
20_2
サーバ
201_2
演算部
202_2
記憶部
$20_{M'}$
サーバ
$201_{M'}$
演算部
$202_{M'}$
記憶部
30
通信ネットワーク
10
クライアント
101
演算部
102
記憶部

[図2]

Algorithm 1 $[\![T]\!], R, [\![s]\!], [\![\pi]\!]$, ctr $\leftarrow$ Build($[\![A]\!]$)

Require: $[\![A]\!] = ([\![a_1]\!], \ldots, [\![a_n]\!]), [\![a_i]\!] = ([\![k_i]\!], [\![v_i]\!]), k_i \in [N]$ かつ全ての $i, j \neq i$ について $k_i \neq \perp$ ならば $k_i \neq k_j$.

Ensure: $A \subseteq T$.

1: サーバは n 個のダミーデータのシェア $[\![D]\!] = ((\perp, \perp), \ldots, (\perp, \perp))$; $|D| = n$ を生成し，$[\![\tilde{T}]\!] = [\![A]\!] \,\|\, [\![D]\!]$ とする．

2: サーバはランダムなシェア $[\![s]\!]$ を生成し，$[\![\tilde{R}]\!] = ([\![\tilde{r}_1]\!], \ldots, [\![\tilde{r}_{2n}]\!])$; $[\![\tilde{r}_i]\!] = \mathsf{PRP}([\![s]\!], [\![k_i]\!])$ for $1 \leq i \leq n, [\![\tilde{r}_j]\!] = \mathsf{PRP}([\![s]\!], [\![\perp + j - n]\!])$ for $n+1 \leq j \leq 2n$ を計算する．

3: サーバはランダムな置換のシェア $[\![\pi]\!]$ を生成し，$[\![T]\!] \leftarrow \mathsf{Perm}([\![\pi]\!], [\![\tilde{T}]\!])$ および $[\![R]\!] \leftarrow \mathsf{Perm}([\![\pi]\!], [\![\tilde{R}]\!])$ を計算する．

4: サーバは $R \leftarrow \mathsf{Reveal}([\![R]\!])$ とし，またクエリカウンタ ctr $= 0$ を準備する．

5: サーバは $[\![T]\!], R, [\![s]\!], [\![\pi]\!]$, ctr を保存する．

[図3]

Algorithm 2 $([[k]], [[v]]) \leftarrow \mathsf{Lookup}([[k]], [[d]], [[T]], R, [[s]], \mathsf{ctr})$

Require: $d \in \{0,1\}$

1: サーバはクエリカウンタを $\mathsf{ctr} = \mathsf{ctr} + 1$ と更新する．

2: サーバは $[[\tilde{k}]] \leftarrow \mathsf{IfElse}([[d]], [[\perp + \mathsf{ctr}]], [[k]])$ を計算する．

3: サーバは $[[r]] \leftarrow \mathsf{PRP}([[s]], [[\tilde{k}]])$ を計算し，$r \leftarrow \mathsf{Reveal}([[r]])$ を得る．

4: サーバは配列 $R = (r_1, \ldots, r_{2n})$ から $r_i = r$ なる値を探す．該当する r_i が存在しなければ停止する．

5: サーバは配列 $[[T]] = ([[t_1]], \ldots, [[t_{2n}]])$ から $([[k]], [[v]]) = [[t_i]]$ を取得し，これを出力とする．また，配列の $[[t_i]]$ を $([[\perp]], [[\perp]])$ で上書きする．

[図4]

Algorithm 3 $[\![A]\!] \leftarrow \mathsf{Extract}([\![T]\!], [\![\pi]\!])$

1: サーバは $[\![\tilde{T}]\!] \leftarrow \mathsf{Unperm}([\![\pi]\!], [\![T]\!])$ を計算し，$[\![\tilde{T}]\!]$ の先頭 n 要素を配列 $[\![A]\!]$ として出力する．

[図5]

Algorithm 4 $(k, v) \leftarrow \mathsf{Access}(\mathsf{op}, k, v')$

Require: $\mathsf{op} \in \{0, 1\}$; $0 =$ 'read', $1 =$ 'write'.

1: クライアントは $\ell = 0, \ldots, L-1$ で $\Phi_\ell.\mathsf{membership}(k)$ を実行し，ターゲットレベル t; $\mathrm{argmin}_t(k \in \Phi_t)$ を得る．ただし，全ての ℓ で $k \notin \Phi_\ell$ の場合，$t = L$ とする．

2: クライアントは t 番目の値が 1 となる長さ $L+1$ の One-hot ベクトル $\vec{d}$ を生成し，これをシェア $[\![\vec{d}]\!] = ([\![d_0]\!], \ldots, [\![d_L]\!])$として，$[\![\mathsf{op}]\!], [\![k]\!], [\![v']\!]$ と共にサーバへ分散する．

3: サーバは各 $\ell = 0, \ldots, L$ において**並列に**
$[\![a_\ell]\!] \leftarrow \mathsf{Lookup}([\![k]\!], [\![d_\ell]\!], [\![T_\ell]\!], R_\ell, [\![s_\ell]\!], \mathsf{ctr}_\ell)$ を実行する
$(a_\ell = (k_\ell, v_\ell)$ は キー・バリューペア$)$.

4: サーバはベクトル $[\![\vec{a}]\!] = ([\![a_0]\!], \ldots, [\![a_L]\!])$ と $[\![\vec{d}]\!]$ の内積 $[\![\tilde{a}]\!] \leftarrow \mathsf{Prod}([\![\vec{a}]\!], [\![\vec{d}]\!])$ を計算する．

5: サーバは $[\![\tilde{a}]\!]$ をクライアントへ送信する．

6: クライアントは $(k, v) \leftarrow \mathsf{Reveal}([\![\tilde{a}]\!])$ を出力として得る．

7: サーバは $[\![\tilde{a}]\!] \leftarrow \mathsf{IfElse}([\![\mathsf{op}]\!], ([\![k]\!], [\![v']\!]), [\![\tilde{a}]\!])$ として $\mathsf{Reshuffle}([\![\tilde{a}]\!])$ を実行する．

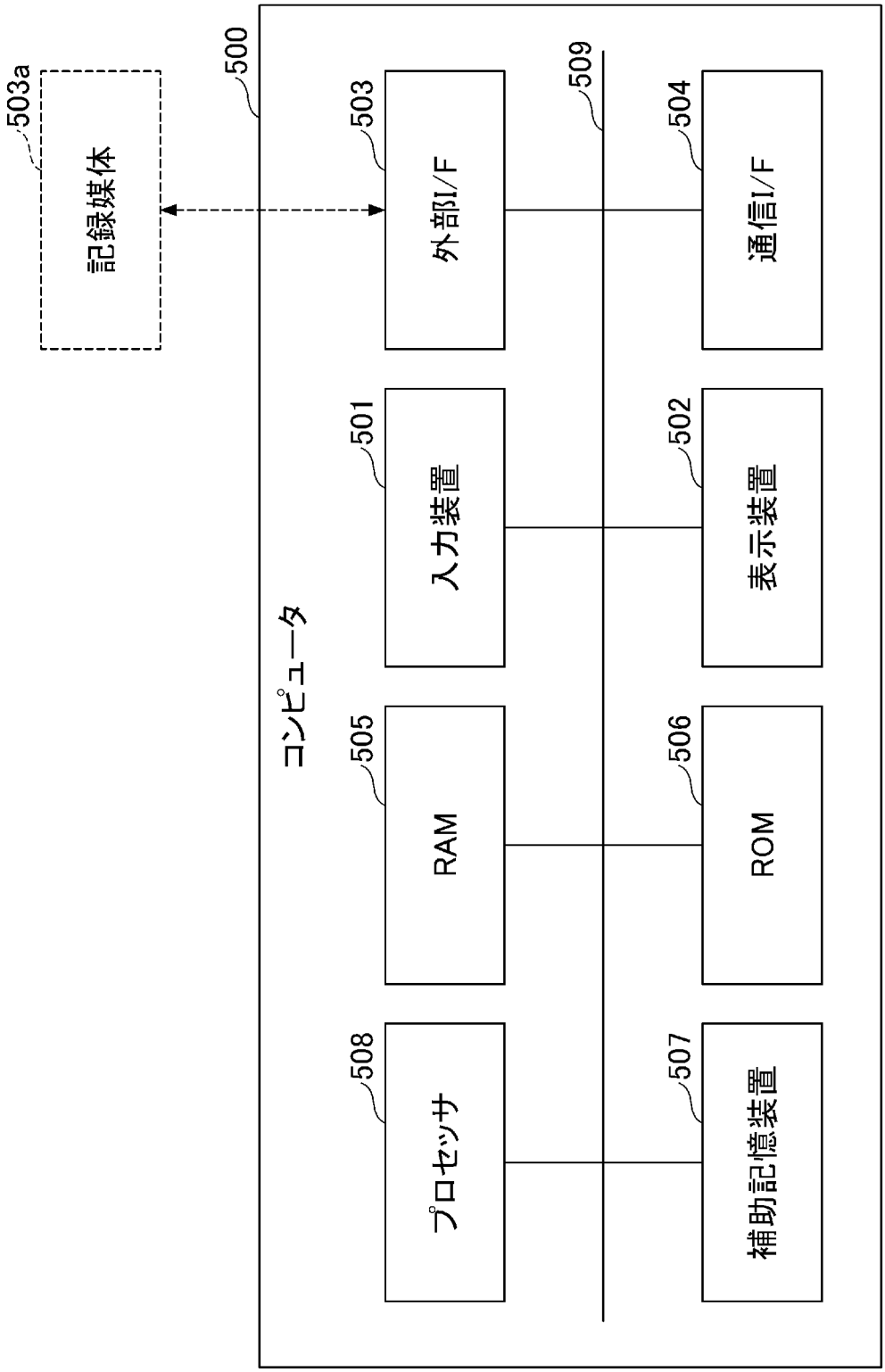
[図6]

Algorithm 5 Reshuffle($[[\tilde{a}]]$)

1: サーバは全ての $\ell = 0, \ldots, p-1$ において $[[A_\ell]] \leftarrow \mathsf{Extract}([[T_\ell]], [[\pi_\ell]])$ を並列に実行し，$[[A_p]] = [[A_0]] \,\|\, \cdots \,\|\, [[A_{p-1}]] \,\|\, ([[\tilde{a}]])$ とする．この処理により，各 $[[T_\ell]]$ は未構築状態へ戻る．

2: サーバは $[[T_p]], R_p, [[s_p]], [[\pi_p]], \mathsf{ctr} \leftarrow \mathsf{Build}([[A_p]])$ によりレベル p を構築する．

[図7]


500
503a
記録媒体
503
外部I/F
509
504
通信I/F
501
入力装置
502
表示装置
コンピュータ
505
RAM
506
ROM
508
プロセッサ
507
補助記憶装置

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2023/006016

A. CLASSIFICATION OF SUBJECT MATTER

G09C 1/00(2006.01)i
FI: G09C1/00 650Z; G09C1/00 660D

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G09C1/00

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Published examined utility model applications of Japan 1922-1996
Published unexamined utility model applications of Japan 1971-2023
Registered utility model specifications of Japan 1996-2023
Published registered utility model applications of Japan 1994-2023

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 2021/144905 A1 (NIPPON TELEGRAPH AND TELEPHONE CORPORATION) 22 July 2021 (2021-07-22) paragraphs [0005], [0008]	1-7
A	WO 2022/123744 A1 (NIPPON TELEGRAPH AND TELEPHONE CORPORATION) 16 June 2022 (2022-06-16) paragraph [0014]	1-7

☐ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
30 March 2023	**11 April 2023**
Name and mailing address of the ISA/JP **Japan Patent Office (ISA/JP)** **3-4-3 Kasumigaseki, Chiyoda-ku, Tokyo 100-8915** **Japan**	Authorized officer Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/JP2023/006016

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
WO 2021/144905 A1	22 July 2021	(Family: none)	
WO 2022/123744 A1	16 June 2022	(Family: none)	

国際調査報告

国際出願番号
PCT/JP2023/006016

A.　発明の属する分野の分類（国際特許分類（ＩＰＣ））

G09C 1/00(2006.01)i
FI: G09C1/00 650Z; G09C1/00 660D

B.　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

G09C1/00

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	１９２２－１９９６年
日本国公開実用新案公報	１９７１－２０２３年
日本国実用新案登録公報	１９９６－２０２３年
日本国登録実用新案公報	１９９４－２０２３年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

C.　関連すると認められる文献

引用文献のカテゴリー*	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求項の番号
A	WO 2021/144905 A1 (日本電信電話株式会社) 22.07.2021 (2021-07-22) 段落0005, 0008	1-7
A	WO 2022/123744 A1 (日本電信電話株式会社) 16.06.2022 (2022-06-16) 段落0014	1-7

☐ Ｃ欄の続きにも文献が列挙されている。

☑ パテントファミリーに関する別紙を参照。

*　引用文献のカテゴリー
“A” 特に関連のある文献ではなく、一般的技術水準を示すもの
“E” 国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
“L” 優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
“O” 口頭による開示、使用、展示等に言及する文献
“P” 国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
“T” 国際出願日又は優先日後に公表された文献であって出願と抵触するものではなく、発明の原理又は理論の理解のために引用するもの
“X” 特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
“Y” 特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
“&” 同一パテントファミリー文献

国際調査を完了した日	国際調査報告の発送日
30.03.2023	11.04.2023

名称及びあて先

日本国特許庁(ISA/JP)
〒100-8915
日本国
東京都千代田区霞が関三丁目４番３号

権限のある職員（特許庁審査官）

行田　悦資 5S 6304

電話番号 03-3581-1101 内線 3546

国際調査報告
パテントファミリーに関する情報

国際出願番号
PCT/JP2023/006016

引用文献	公表日	パテントファミリー文献	公表日
WO 2021/144905 A1	22.07.2021	（ファミリーなし）	
WO 2022/123744 A1	16.06.2022	（ファミリーなし）	