(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
2009 年7 月30 日 (30.07.2009)

PCT

(10) 国際公開番号
WO 2009/093290 A1

(51) 国際特許分類:
G06F 17/30 (2006.01)

(21) 国際出願番号: PCT/JP2008/003266

(22) 国際出願日: 2008 年11 月11 日 (11.11.2008)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(30) 優先権データ:
特願2008-011148 2008 年1 月22 日 (22.01.2008) JP
特願2008-168003 2008 年6 月26 日 (26.06.2008) JP

(71) 出願人 (米国を除く全ての指定国について): 株式会社エスグランツ (S.GRANTS CO., LTD.) [JP/JP]; 〒2610004 千葉県千葉市美浜区高洲三丁目５番３棟１２１０号 Chiba (JP).

(72) 発明者; および

(75) 発明者/出願人 (米国についてのみ): 新庄敏男 (SHINJO, Toshio) [JP/JP]; 〒2610004 千葉県千葉市美浜区高洲三丁目５番３棟１２１０号株式会社エスグランツ内 Chiba (JP). 國分光裕 (KOKUBUN, Mitsuhiro) [JP/JP]; 〒2610004 千葉県千葉市美浜区高洲三丁目５番３棟１２１０号株式会社エスグランツ内 Chiba (JP).

(74) 代理人: ▲徳▼永民雄 (TOKUNAGA, Tamio); 〒2020012 東京都西東京市東町３丁目２番１９号 Tokyo (JP).

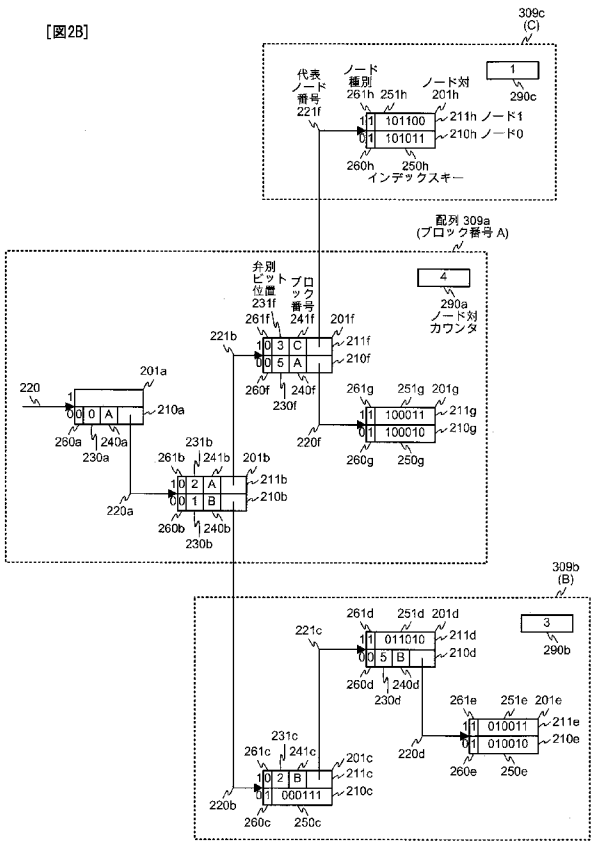
(81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG,

[続葉有]

(54) **Title:** BIT STRING RETRIEVAL DEVICE, RETRIEVAL METHOD AND PROGRAM

(54) **発明の名称:** ビット列検索装置、検索方法及びプログラム

[図2B]



221f REPRESENTATIVE NODE NUMBER
261h NODE CLASS
201h NODE PAIR
211h NODE 1
210h NODE 0
250h INDEX KEY
309a ARRANGEMENT (BLOCK NUMBER A)
231f DISCRIMINATION BIT POSITION
241f BLOCK NUMBER
290a NODE PAIR COUNTER

(57) Abstract: A coupled node tree is divided to be arranged in a plurality of storage areas. A branch node of the coupled node tree includes an identifier of a storage area where a pair of nodes of a link is stored and position information indicating a position of a representative node of a pair of nodes of a link in the storage area.When a leaf node including a new index key is inserted into the coupled node tree, a vacant storage area having a pair of vacant nodes is searched from the insertion position of the inserting leaf node and a pair of the vacant nodes is moved from the area to the insertion position, so that the leaf node is inserted.

(57) 要約: カップルドノードツリーを複数の格納区域に分割して配置する。カップルドノードツリーのブランチノードは、リンク先のノード対の格納された格納区域の識別子とその格納区域におけるリンク先のノード対の代表ノードの位置を示す位置情報を含む。カップルドノードツリーに新しくインデックスキーを含むリーフノードを挿入するときは、挿入するリーフノードの挿入位置から空ノード対を有する空格納区域を探索し、そこから空ノード対を挿入位置まで移動させてリーフノードを挿入する。

WO 2009/093290 A1

BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, 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, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), **ユーラシア** (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), **ヨーロッパ** (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

添付公開書類:

— **国際調査報告書**

明　細　書

ビット列検索装置、検索方法及びプログラム

技術分野

[0001] 本発明はビット列の集合から所望のビット列を検索する検索装置、検索方法及びプログラムに関するものであり、特にビット列を記憶するデータ構造に工夫をして、検索速度等の向上を図る技術分野のものである。

背景技術

[0002] 近年、社会の情報化が進展し、大規模なデータベースが各所で利用されるようになってきている。このような大規模なデータベースからレコードを検索するには、各レコードの記憶されたアドレスと対応づけられたレコード内の項目をインデックスキーとして検索をし、所望のレコードを探し出すことが通例である。また、全文検索における文字列も、文書のインデックスキーと見なすことができる。

[0003] そして、それらのインデックスキーはビット列で表現されることから、データベースの検索はビット列の検索に帰着されるということができる。

上記ビット列の検索を高速に行うために、ビット列を記憶するデータ構造を種々に工夫することが従来から行われている。このようなものの一つとして、パトリシアツリーという木構造が知られている。

[0004] 図１は、上述の従来の検索処理に用いられているパトリシアツリーの一例を示すものである。パトリシアツリーのノードは、インデックスキー、検索キーの検査ビット位置、左右のリンクポインタを含んで構成される。明示はされていないが、ノードにはインデックスキーに対応するレコードにアクセスするための情報が含まれていることは勿論である。

[0005] 図１の例では、インデックスキー“１０００１０”を保持するノード１７５０ａがルートノードとなっており、その検査ビット位置は０である。ノード１７５０ａの左リンク１７４０ａにはノード１７５０ｂが接続され、右リンク１７４１ａにはノード１７５０ｆが接続されている。

[0006] ノード１７５０ｂの保持するインデックスキーは“０１００１１”であり、検査ビット位置２０３０ｂは１である。ノード１７５０ｂの左リンク１７４０ｂにはノード１７５０ｃが、右リンク１７４１ｂにはノード１７５０ｄが接続されている。ノード１７５０ｃが保持するインデックスキーは“０００１１１”、検査ビット位置は３である。ノード１７５０ｄが保持するインデックスキーは“０１１０１０”、検査ビット位置は２である。

[0007] ノード１７５０ｃから実線で接続された部分はノード１７５０ｃの左右のリンクポインタを示すものであり、点線の接続されていない左ポインタ１７４０ｃは、その欄が空欄であることを示している。点線の接続された右ポインタ１７４１ｃの点線の接続先は、ポインタの示すアドレスを表しており、今の場合ノード１７５０ｃを右ポインタが指定していることを表している。

[0008] ノード１７５０ｄの右ポインタ１７４１ｄはノード１７５０ｄ自身を指しており、左リンク１７４０ｄにはノード１７５０ｅが接続されている。ノード１７５０ｅの保持するインデックスキーは“０１００１０”、検査ビット位置は５である。ノード１７５０ｅの左ポインタ１７４０ｅはノード１７５０ｂを、右ポインタ１７４１ｅはノード１７５０ｅを指している。

[0009] また、ノード１７５０ｆの保持するインデックスキーは“１０１０１１”であり、検査ビット位置１７３０ｆは２である。ノード１７５０ｆの左リンク１７４０ｆにはノード１７５０ｇが、右リンク１７４１ｆにはノード１７５０ｈが接続されている。

[0010] ノード１７５０ｇの保持するインデックスキーは“１０００１１”であり、検査ビット位置１７３０ｇは５である。ノード１７５０ｇの左ポインタ１７４０ｇはノード１７５０ａを、右ポインタ１７４１ｇはノード１７５０ｇを指している。

[0011] ノード１７５０ｈの保持するインデックスキーは“１０１１００”であり、検査ビット位置１７３０ｈは３である。ノード１７５０ｈの左ポインタ１７４０ｈはノード１７５０ｆを、右ポインタ１７４１ｈはノード１７５０ｈを指している。

[0012] 図１の例では、ルートノード１７５０ａからツリーを降りるにしたがって、各ノードの検査ビット位置が大きくなるように構成されている。

ある検索キーで検索を行うとき、ルートノードから順次各ノードに保持される検索キーの検査ビット位置を検査していき、検査ビット位置のビット値が１であるか０であるか判定を行い、１であれば右リンクをたどり、０であれば左リンクをたどる。そして、リンク先のノードの検査ビット位置がリンク元のノードの検査ビット位置より大きくなければ、すなわち、リンク先が下方でなく上方に戻れば（図１において点線で示されたこの逆戻りのリンクをバックリンクという）、リンク先のノードのインデックスキーと検索キーの比較を行う。比較の結果、等しければ検索成功であり、等しくなければ検索失敗であることが保証されている。

[0013] 上記のように、パトリシアツリーを用いた検索処理では、必要なビットの検査だけで検索できること、キー全体の比較は１回ですむことなどのメリットがあるが、各ノードからの２つのリンクが必ずあることにより記憶容量が増大することや、バックリンクの存在による判定処理の複雑化、バックリンクにより戻ることで初めてインデックスキーと比較することによる検索処理の遅延及び追加削除等データメンテナンスの困難性などの欠点がある。

[0014] これらのパトリシアツリーの欠点を解消しようとするものとして、例えば下記特許文献１に開示された技術がある。下記特許文献１に記載されたパトリシアツリーにおいては、下位の左右のノードは連続した領域に記憶することによりポインタの記憶容量を削減するとともに、次のリンクがバックリンクであるか否かを示すビットを各ノードに設けることにより、バックリンクの判定処理を軽減している。

[0015] しかしながら、下記特許文献１に開示されたものにおいても、１つのノードは必ずインデックスキーの領域とポインタの領域を占めること、下位の左右のノードを連続した領域に記憶するようにしてポインタを１つとしたため、例えば図１に示したパトリシアツリーの最下段の部分である左ポインタ１７４０ｃ、右ポインタ１７４１ｈ等の部分にもノードと同じ容量の記憶領域

を割り当てる必要があるなど、記憶容量の削減効果はあまり大きいものではない。また、バックリンクによる検索処理の遅延の問題や追加削除等の処理が困難であることも改善されていない。

特許文献1：特開２００１－３５７０７０号公報

発明の開示

[0016] 上述の従来の検索手法における問題点を解決するものとして、本出願人は、特願２００６－１８７８２７において、ルートノードと、隣接した記憶領域に配置されるブランチノードとリーフノードまたはブランチノード同士またはリーフノード同士のノード対からなるビット列検索に用いるツリーであって、ルートノードはツリーの始点を表すノードであって、該ツリーのノードが１つのときはリーフノード、ツリーのノードが２つ以上のときは前記ブランチノードであり、前記ブランチノードは、ビット列検索を行う検索キーの弁別ビット位置とリンク先のノード対の一方のノードである代表ノードの位置を示す位置情報を含み、前記リーフノードは検索対象のビット列からなるインデックスキーを含むカップルドノードツリーを用いたビット列検索を提案した。

[0017] ところで近年において、情報処理技術の進展とともに情報サービスへの要求がさらに拡大し、より厳しいものになってきている。情報サービスを提供する上で基本となるのはデータベースの構築とそのデータベースからの情報の取出しであるが、データベースに蓄積されるデータ量はますます増大し、極めて巨大なものとなりつつある。

本出願人の先に提案した検索手法は、上記巨大化しつつあるデータベースの高速検索を可能とするものであるが、データベースが巨大化するとともにそれに対応するカップルドノードツリーも、従来のツリー構造のものより記憶容量が少なくてすむとはいえ、大きなものとなる。一方、各種記憶手段の記憶容量にはそれぞれ制限があり、経済的なシステムを構築するためには各種アクセス速度、記憶容量の記憶手段を組み合わせて利用する必要がある。特に、キャッシュメモリを有効利用できることが望ましい。

[0018] そこで本発明の解決しようとする課題は、カップルドノードツリーを複数の格納区域に分割して配置することを可能とするカップルドノードツリーの構造とそれを用いた検索手法を提供することである。

[0019] 本発明のカップルドノードツリーのブランチノードは、リンク先のノード対の格納された格納区域の識別子とその格納区域におけるリンク先のノード対の一方のノードである代表ノードの位置を示す位置情報を含む。

[0020] 検索時には、カップルドノードツリーの任意のノードを検索開始ノードとしてブランチノードにおいて、該ブランチノードに含まれる弁別ビット位置の検索キーのビット値に応じて、該ブランチノードに含まれる格納区域の識別子の示す格納区域に格納されたリンク先のノード対の代表ノードかあるいはそれと隣接した記憶領域に配置されたノードにリンクすることを順次前記リーフノードに至るまで繰り返すことにより、前記リーフノードに格納されたインデックスキーを、前記検索開始ノードをルートノードとする前記ツリーの任意の部分木の前記検索キーによる検索結果である検索結果キーとする。

[0021] 本発明のカップルドノードツリーに新たにインデックスキーを挿入する場合は、まずそのインデックスキーを検索キーとして、ブランチノードにおいて該ブランチノードに含まれる弁別ビット位置の検索キーのビット値に応じてブランチノードに含まれる格納区域の識別子の示す格納区域に格納されたリンク先のノード対の代表ノードかあるいはそれと隣接した記憶領域に配置されたノードにリンクすることを、ルートノードから順次リーフノードに至るまでリンク経路を記憶しながら繰り返し、該リーフノードに格納されたインデックスキーを検索結果キーとして取得する。

[0022] 挿入するインデックスキーと検索の結果得られたインデックスキーの間でビット列比較を行い、ビット列比較で異なるビット値となる先頭のビット位置と、リンク経路上のブランチノードの弁別ビット位置との相対的位置関係により、挿入されるリーフノードともう一方のノードからなる挿入ノード対の挿入される格納区域の識別子と挿入ノード対の代表ノードの該格納区域にお

ける位置情報を含むブランチノードの位置情報を示す挿入位置を決定し、挿入ノード対を格納可能な空き領域を備えた空格納区域を探索し、空ノード対を取得し、空ノード対に空格納区域の上位の格納区域のノード対の内容を格納し、該上位の格納区域のノード対を解放して空ノード対を取得する操作を、該上位の格納区域が、挿入ノード対が挿入される格納区域となるまで繰り返し、検索キーと検索結果キーの大小関係に応じて、挿入されるリーフノードを挿入ノード対のどちらのノードとするかを決定し、挿入するインデックスキーをリーフノードに格納する。

[0023] 本発明のカップルドノードツリーからあるインデックスキーを削除するときは、削除するキーにより検索を行い、削除対象のインデックスキーを保持するノードと同一ノード対を構成するノードの内容を当該ノード対のリンク元のブランチノードに書き込み、当該ノード対を削除することにより行う。

[0024] 本発明によれば、ブランチノードがリンク先のノード対の格納された格納区域の識別子を含むので、カップルドノードツリーを複数の格納区域に分割して配置することが可能になる。

図面の簡単な説明

[0025] [図1]従来の検索で用いられるパトリシアツリーの一例を示す図である。

[図2A]配列に格納されたカップルドノードツリーの構成例を説明する図である。

[図2B]カップルドノードツリーのツリー構造を概念的に示す図である。

[図3]本発明を実施するためのハードウェア構成例を説明する図である。

[図4]本発明の一実施形態における検索処理を説明するフローチャートである。

[図5]本発明の一実施形態における挿入処理を概念的に説明する図である。

[図6]本発明の一実施形態における挿入処理全体の処理フローの概要を説明する図である。

[図7A]挿入キーを検索キーとして、ルートノードよりカップルドノードツリーを検索して検索結果キーを得る処理フローを説明する図である。

[図7B]挿入ノード対の挿入位置を求める処理フローを説明する図である。

[図7C]ノード対用の空配列要素を備えた空ブロックを探索し、空ノード対を取得する処理フローを説明する図である。

[図7D]挿入ノード対の挿入位置に空ノード対を移動する処理フローを説明する図である。

[図7E]空ノード対に挿入キー等を格納して挿入ノード対を完成させる処理フローを説明する図である。

[図8]本発明の一実施の形態におけるルートノードの挿入処理を含むノード挿入処理全体を説明する処理フロー図である。

[図9]本発明の一実施形態における削除処理の前段である検索処理の処理フローを説明する図である。

[図10]本発明の一実施形態における削除処理の後段の処理フローを説明する図である。

[図11A]ノードの削除前のカップルドノードツリーと削除処理において検索処理を実行した探索経路スタックの状態を説明する図である。

[図11B]削除処理を完了した後のカップルドノードツリー等の状態を説明する図である。

[図12A]挿入処理の具体例１において、ノードの挿入前のカップルドノードツリーと探索経路ブロックの状態と配列を新たに取得した状態を説明する図である。

[図12B]挿入処理の具体例１における挿入処理後のカップルドノードツリーを説明する図である。

[図12C]挿入処理の具体例２において、ノードの挿入前のカップルドノードツリーと探索経路ブロックの状態と探索経路スタック上のブロック番号の指す配列に空ノード対を取得した状態を説明する図である。

[図12D]挿入処理の具体例２において、空のノード対を移動して挿入ノード対を挿入位置に挿入可能とした状態を説明する図である。

[図12E]挿入処理の具体例２において、挿入位置に挿入ノード対を挿入すると

ともに、挿入位置のブランチノードの弁別ビット位置を更新して挿入処理を完成させた状態を説明する図である。

発明を実施するための最良の形態

[0026] 以下、本発明を実施するための最良の形態として、カップルドノードツリーを配列に格納する例について説明する。ブランチノードが保持するリンク先の代表ノードの位置を示すデータとして、記憶装置のアドレス情報とすることもできるが、ブランチノードあるいはリーフノードのうち占有する領域の記憶容量の大きい方を格納可能な配列要素からなる配列を用いることにより、ノードの位置を配列番号で表すことができ、代表ノードの位置を示す位置情報の情報量を削減することができる。

[0027] 図２Ａは、本発明の一実施形態における配列に格納されたカップルドノードツリーの構成例を説明する図である。

図２Ａを参照すると、ブロック番号Ａで識別される格納区域に配置された配列１００の配列番号１０の配列要素にノード１０１が記憶されている。ノード１０１はノード種別１０２、弁別ビット位置１０３、代表ノード番号１０４ａ及びブロック番号１０４ｂで構成されている。ノード種別１０２は０であり、ノード１０１がブランチノードであることを示している。弁別ビット位置１０３には１が格納されている。代表ノード番号１０４ａにはリンク先のノード対の代表ノードの配列番号２０が格納され、ブロック番号１０４ｂにはＡが格納されている。なお、以下では表記の簡略化のため、代表ノード番号に格納された配列番号を代表ノード番号ということもある。また、代表ノード番号に格納された配列番号をそのノードに付した符号あるいはノード対に付した符号で表すこともある。さらに、弁別ビット位置に格納された値を、単に弁別ビット位置ということもある。

[0028] 配列番号２０の配列要素には、ノード対１１１の代表ノードであるノード［０］１１２が格納されている。そして隣接する次の配列要素（配列番号２０＋１）に代表ノードと対になるノード［１］１１３が格納されている。ノード［０］１１２はノード１０１と同様にブランチノードである。ノード［

０］１１２のノード種別１１４には０が、弁別ビット位置１１５には３が、代表ノード番号１１６ａには３０が格納され、ブロック番号１１６ｂにはＢが格納されている。またノード［１］１１３は、ノード種別１１７とインデックスキー１１８ａで構成されている。ノード種別１１７には１が格納されており、ノード［１］１１３がリーフノードであることを示している。インデックスキー１１８には、“０００１”が格納されている。以下では表記の簡略化のため、インデックスキーに格納されたデータのこともインデックスキーということがある。

[0029] 配列１００には、さらにノード対カウンタ１１９が格納されており、そこには配列１００に格納されているノード対の個数である２が記憶されている。

パトリシアツリーについて先に述べたと同様に、インデックスキーと対応するレコードにアクセスするためのアクセス先情報も当然必要である。インデックスキーとアクセス先情報との対応づけは、例えば、インデックスキーを記憶している記憶領域に隣接する記憶領域に、当該インデックスキーに対応するアクセス先情報を記憶することによって行ってもよい。以下ではアクセス先情報については省略して説明する。

[0030] なお、代表ノードをノード［０］で表し、それと対になるノードをノード［１］で表すことがある。また、ある配列番号の配列要素に格納されたノードを、その配列番号のノードということがあり、ノードの格納された配列要素の配列番号を、ノードの配列番号ということもある。

[0031] 上述のノード［０］１１２のブロック番号１１６ｂの値Ｂで識別される配列１２０には、ノード対カウンタ１２９と、配列番号３０及び３１の配列要素に格納されたノード１２２とノード１２３からなるノード対１２１が格納されている。

[0032] ノード対カウンタ１２９には、配列１２０に格納されているノード対の個数である１が記憶されている。ノード１２２とノード１２３からなるノード対１２１の内容は省略されている。

[0033] ノード［０］１１２、ノード［１］１１３、ノード１２２、及びノード１

２３の格納された配列要素にそれぞれ付された０あるいは１は、検索キーで検索を行う場合にノード対のどちらのノードにリンクするかを示すものである。検索キーの、前段のブランチノードの弁別ビット位置にあるビット値である０か１を代表ノード番号に加えた配列番号のノードにリンクする。

[0034] したがって、前段のブランチノードの代表ノード番号に、検索キーの弁別ビット位置にあるビット値を加えることにより、リンク先のノードが格納されたブロック番号で識別される配列の配列要素の配列番号を求めることができる。

[0035] なお、上記の例では代表ノード番号をノード対の配置された配列番号のうち小さい方を採用しているが、大きいほうを採用することも可能であることは明らかである。

図２Ｂは、カップルドノードツリーのツリー構造を概念的に示す図である。図示の６ビットのインデックスキーは、図１に例示されたパトリシアツリーのものと同じである。

[0036] 符号２１０ａで示すのがルートノードである。図示の例では、ルートノード２１０ａはブロック番号Ａの配列３０９ａの配列番号２２０に配置されたノード対２０１ａの代表ノードとしている。

[0037] ツリー構造としては、ルートノード２１０ａの下にノート対２０１ｂが、その下層にノード対２０１ｃとノード対２０１ｆが配置され、ノード対２０１ｆの下層にはノード対２０１ｈとノード対２０１ｇが配置されている。ノード対２０１ｃの下にはノード対２０１ｄが、さらにその下にはノード対２０１ｅが配置されている。

[0038] 図に示すように、ノード対２０１ａ、２０１ｂ、２０１ｆ及び２０１ｇはブロック番号Ａの配列３０９ａに格納され、ノード対カウンタ２９０ａには４が記憶されている。ノード対２０１ｃとそれより下層のノード対２０１ｄ、２０１ｅは、ブロック番号Ｂの配列３０９ｂに格納され、ノード対カウンタ２９０ｂには３が記憶されている。ノード対２０１ｈはブロック番号Ｃの配列３０９ｃに格納され、ノード対カウンタ２９０ｃには１が記憶されてい

る。

[0039] 各ノードの前に付された０あるいは１の符号は、図１において説明した配列要素の前に付された符号と同じである。検索キーの弁別ビット位置のビット値に応じてツリーをたどり、検索対象のリーフノードを見つけることになる。

[0040] 図２Ｂに示す例では、ルートノード２１０ａのノード種別２６０ａは０でブランチノードであることを示し、弁別ビット位置２３０ａは０を示している。ブロック番号２４０ａはＡであり、リンク先のノード対２０１ｂがブロック番号Ａの配列３０９ａに配置されていることを示している。代表ノード番号は２２０ａであり、それはノード対２０１ｂの代表ノード２１０ｂの格納された配列３０９ａの配列要素の配列番号である。

[0041] ノード対２０１ｂはノード２１０ｂと２１１ｂで構成され、それらのノード種別２６０ｂ、２６１ｂはともに０であり、ブランチノードであることを示している。ノード２１０ｂの弁別ビット位置２３０ｂには１が格納され、ブロック番号２４０ｂには配列３０９ｂのブロック番号Ｂが格納されている。リンク先の代表ノード番号にはノード対２０１ｃの代表ノード２１０ｃの格納された配列３０９ｂの配列要素の配列番号２２０ｂが格納されている。

[0042] ノード２１０ｃのノード種別２６０ｃには１が格納されているので、このノードはリーフノードであり、したがって、インデックスキーを含んでいる。インデックスキー２５０ｃには“０００１１１”が格納されている。一方ノード２１１ｃのノード種別２６１ｃは０であり、ブランチノードであることを示している。弁別ビット位置２３１ｃは２、ブロック番号２４１ｃはＢであり、代表ノード番号にはノード対２０１ｄの代表ノード２１０ｄの格納された配列３０９ｂの配列要素の配列番号２２１ｃが格納されている。

[0043] ノード２１０ｄのノード種別２６０ｄは０であり、ブランチノードであることを示している。弁別ビット位置２３０ｄは５、ブロック番号２４０ｄはＢであり、代表ノード番号にはノード対２０１ｅの代表ノード２１０ｅの格納された配列３０９ｂの配列要素の配列番号２２０ｄが格納されている。ノ

ードＡ１０ｄと対になるノードＡ１１ｄのノード種別Ａ６１ｄは１であり、インデックスキーＡ５１ｄには“０１１０１０”が格納されている。

[0044] ノード対Ａ０１ｅのノードＡ１０ｅ、Ａ１１ｅのノード種別Ａ６０ｅ、Ａ６１ｅはともに１であり双方ともリーフノードであることを示し、それぞれのインデックスキーＡ５０ｅ、Ａ５１ｅにはインデックスキーとして“０１００１０”と“０１００１１”が格納されている。

[0045] ノード対Ａ０１ｂのもう一方のノードであるノードＡ１１ｂの弁別ビット位置Ａ３１ｂには２が格納され、ブロック番号にはＡが格納されている。リンク先の代表ノード番号にはノード対Ａ０１ｆの代表ノードＡ１０ｆの格納された配列Ａ３０９ａの配列要素の配列番号Ａ２２１ｂが格納されている。

[0046] ノード対Ａ０１ｆのノードＡ１０ｆ、Ａ１１ｆのノード種別Ａ６０ｆ、Ａ６１ｆはともに０であり双方ともブランチノードである。それぞれの弁別ビット位置Ａ３０ｆ、Ａ３１ｆには５、３が格納されている。また、ノードＡ１０ｆのブロック番号Ａ２４０ｆにはＡ、ブロック番号Ａ２４１ｆにはＣが格納されている。ノードＡ１０ｆの代表ノード番号にはノード対Ａ０１ｇの代表ノードＡ１０ｇの格納された配列Ａ３０９ａの配列要素の配列番号Ａ２２０ｆが格納され、ノードＡ１１ｆの代表ノード番号にはノード対Ａ０１ｈの代表ノードであるノード［０］Ａ１０ｈの格納された配列Ａ３０９ｃの配列要素の配列番号Ａ２２１ｆが格納されている。

[0047] ノード対Ａ０１ｇのノードＡ１０ｇ、Ａ１１ｇのノード種別Ａ６０ｇ、Ａ６１ｇはともに１であり双方ともリーフノードであることを示し、それぞれのインデックスキーＡ５０ｇ、Ａ５１ｇには“１０００１０”と“１０００１１”が格納されている。

[0048] また同じくノード対Ａ０１ｈの代表ノードであるノード［０］Ａ１０ｈとそれと対をなすノード［１］Ａ１１ｈのノード種別Ａ６０ｈ、Ａ６１ｈはともに１であり双方ともリーフノードであることを示し、それぞれのインデックスキーＡ５０ｈ、Ａ５１ｈには“１０１０１１”と“１０１１００“が格納されている。

[0049] 以下、上述のツリーからインデックスキー“１０００１０”を検索する処理の流れを簡単に説明する。弁別ビット位置は、左から０、１、２、・・・とする。

まず、ビット列“１０００１０”を検索キーとしてルートノード２１０ａから処理をスタートする。ルートノード２１０ａの弁別ビット位置２３０ａは０であるので、検索キー“１０００１０”の弁別ビット位置が０のビット値をみると１である。また、ブロック番号２４０ａはＡであり、リンク先のノード対は配列３０９ａに存在することが示されている。そこで代表ノード番号の格納された配列番号２２０ａに１を加えた配列番号の配列３０９ａの配列要素に格納されたノード２１１ｂにリンクする。

[0050] ノード２１１ｂの弁別ビット位置２３１ｂには２が格納されているので、検索キー“１０００１０”の弁別ビット位置が２のビット値をみると０である。ブロック番号２４０ｂはＡであり、リンク先のノード対は配列３０９ａに存在することが示されている。そこで代表ノード番号の格納された配列番号２２１ｂに０を加えた配列番号の配列３０９ａの配列要素に格納されたノード２１０ｆにリンクする。

[0051] ノード２１０ｆの弁別ビット位置２３０ｆには５が格納されているので、検索キー“１０００１０”の弁別ビット位置が５のビット値をみると０である。ブロック番号２４０ｆはＡであり、リンク先のノード対は配列３０９ａに存在することが示されている。そこで代表ノード番号の格納された配列番号２２０ｆに０を加えた配列番号の配列３０９ａの配列要素に格納されたノード２１０ｇにリンクする。

[0052] ノード２１０ｇのノード種別２６０ｇは１でありリーフノードであることを示しているので、インデックスキー２５０ｇを読み出して検索キーと比較すると両方とも“１０００１０”であって一致している。このようにしてカップルドノードツリーを用いた検索が行われる。

[0053] 図２Ｂに示すように、カップルドノードツリーのノード対は、いわばツリーの深さ方向を優先して分割して格納領域に配置されている。したがって、

上述の検索キー“１０００１０”によるルートノード２１０ａを検索開始ノードとした検索は、配列３０９ａに対するアクセスだけで実行することができる。

[0054] 次に、図２Ｂを参照してカップルドノードツリーの構成（論理的構成）の意味について説明する。

カップルドノードツリーの構成はインデックスキーの集合により規定される。図２Ｂの例で、ルートノード２１０ａの弁別ビット位置が０であるのは、図２Ｂに例示されたインデックスキーに０ビット目が０のものと１のものがあるからである。０ビット目が０のインデックスキーのグループはノード２１０ｂの下に分類され、０ビット目が１のインデックスキーのグループはノード２１１ｂの下に分類されている。

[0055] ノード２１１ｂの弁別ビット位置が２であるのは、ノード２１１ｈ、２１０ｈ、２１１ｇ、２１０ｇに格納された０ビット目が１のインデックスキーの１ビット目がすべて０で等しく、２ビット目で初めて異なるものがあるという、インデックスキーの集合の性質を反映している。

[0056] 以下０ビット目の場合と同様に、２ビット目が１であるものはノード２１１ｆ側に分類され、２ビット目が０であるものはノード２１０ｆ側に分類される。

そして２ビット目が１であるインデックスキーは３ビット目の異なるものがあるのでノード２１１ｆの弁別ビット位置には３が格納され、２ビット目が０であるインデックスキーでは３ビット目も４ビット目も等しく５ビット目で異なるのでノード２１０ｆの弁別ビット位置には５が格納される。

[0057] ノード２１１ｆのリンク先においては、３ビット目が１のものと０のものがそれぞれ１つしかないことから、ノード２１０ｈ、２１１ｈはリーフノードとなり、それぞれインデックスキー２５０ｈと２５１ｈに“１０１０１１”と“１０１１００”が格納されている。

[0058] 仮にインデックスキーの集合に“１０１１００”の代わりに“１０１１０１”か“１０１１１０”が含まれていたとしても、３ビット目までは“１０

１１００”と等しいので、ノード２１１ｈに格納されるインデックスキーが変わるだけで、ツリーの論理構造自体は変わることはない。しかし、“１０１１００”に加えて“１０１１０１”が含まれていると、ノード２１１ｈはブランチノードとなり、その弁別ビット位置は５になる。追加されるインデックスキーが“１０１１１０”であれば、弁別ビット位置は４となる。それらの場合、ブランチノード２１１ｈのリンク先に例えば配列３０９ｃに格納されたノード対があり、リーフノード２１１ｈに格納されていたインデックスキー“１０１１００”とインデックスキー“１０１１０１”あるいは“１０１１１０”の弁別ビット位置でのビット値に応じて、ビット値が０のインデックスキーはノード［０］に、ビット値が１のインデックスキーはノード［１］に格納される。ノード対カウンタ２９０ｃの値は２となる。

[0059] 以上説明したように、カップルドノードツリーの論理構造は、インデックスキーの集合に含まれる各インデックスキーの各ビット位置のビット値により決定される。

そしてさらにいえば、異なるビット値となるビット位置ごとにビット値が“１”のノードとビット値が“０”のノードとに分岐していることから、ノード［１］側とツリーの深さ方向を優先させてリーフノードをたどると、それらに格納されたインデックスキーは、ノード２１１ｈのインデックスキー２５１ｈの“１０１１００”、ノード２１０ｈのインデックスキー２５０ｈの“１０１０１１”、・・・、ノード２１０ｃのインデックスキー２５０ｃの“０００１１１”となり降順にソートされている。

[0060] すなわち、カップルドノードツリーにおいては、インデックスキーはソートされてツリー上に配置されている。

検索キーで検索するときはインデックスキーがカップルドノードツリー上に配置されたルートをたどることになり、例えば検索キーが“１０１１００”であればノード２１１ｈに到達することができる。また、上記説明からも想像がつくように、“１０１１０１”か“１０１１１０”を検索キーとした場合でもノード２１１ｈにたどり着き、インデックスキー２５１ｈ“１０１

１００”が検索結果キーとして得られる。

[0061] また、例えば“１００１００”で検索した場合でも、ノード２１０ａ、２１１ｂ、２１０ｆのリンク経路では検索キーの３ビット目と４ビット目は使われることがなく、“１００１００”の５ビット目が０なので、“１０００１０”で検索した場合と同様にノード２１０ｇに到達することになる。このように、カップルドノードツリーに格納されたインデックスキーのビット構成に応じた弁別ビット位置を用いて分岐が行われる。

[0062] 図３は、本発明を実施するためのハードウェア構成例を説明する図である。

本発明の検索装置による検索処理及びデータメンテナンスは中央処理装置３０２及びキャッシュメモリ３０３を少なくとも備えたデータ処理装置３０１によりデータ格納装置３０８を用いて実施される。カップルドノードツリーが配置される配列３０９ａと検索中にたどるノードが格納された配列の配列要素の配列番号と配列のブロック番号を記憶する探索経路スタック３１０を有するデータ格納装置３０８は、主記憶装置３０５または外部記憶装置３０６で実現することができ、あるいは通信装置３０７を介して接続された遠方に配置された装置を用いることも可能である。

[0063] 図３の例示では、主記憶装置３０５、外部記憶装置３０６及び通信装置３０７が一本のバス３０４によりデータ処理装置３０１に接続されているが、接続方法はこれに限るものではない。また、主記憶装置３０５をデータ処理装置３０１内のものとすることもできるし、探索経路スタック３１０を中央処理装置３０２内のハードウェアとして実現することも可能である。あるいは、配列３０９ａは外部記憶装置３０６に、探索経路スタック３１０を主記憶装置３０５に持つなど、使用可能なハードウェア環境、インデックスキー集合の大きさ等に応じて適宜ハードウェア構成を選択できることは明らかである。

[0064] また、特に図示されてはいないが、処理の途中で得られた各種の値を後の処理で用いるためにそれぞれの処理に応じた主記憶装置３０５の一時記憶領

域が用いられることは当然である。そして、以下の説明においては、一次記憶領域に格納されるあるいは設定される値を一時記憶領域の名前で呼ぶことがある。

[0065] 次に、本発明の一実施態様に係るカップルドノードツリーを用いた基本的な操作である検索処理について説明する

図４は、一実施形態におけるビット列の検索処理を示すフローチャートである。

[0066] まず、ステップＳ４０１ａで、検索開始ノードを格納した配列のブロック番号と配列要素の配列番号を取得する。検索開始ノードの指定は、検索処理を利用する各種アプリケーションや、利用者によって行われる。

[0067] 取得された検索開始ノードのブロック番号と配列番号は、図示しない検索開始ノード設定エリアに設定されるが、この検索開始ノード設定エリアは、先に述べた「処理の途中で得られた各種の値を後の処理で用いるためにそれぞれの処理に応じた一時記憶領域」の一つである。以下の説明では、「図示しない検索開始ノード設定エリアに設定する」のような表現に変えて、「検索開始ノードの配列番号を得る。」、「検索開始ノードとして設定する」あるいは単に「検索開始ノードに設定する」のように記述することもある。検索開始ノード以外についても同様である。

[0068] 次に、ステップＳ４０２ｂで、探索経路スタック３１０に取得されたブロック番号と配列番号を格納し、ステップＳ４０３ａで、そのブロック番号と配列番号に対応する配列要素を参照すべきノードとして読み出す。そして、ステップＳ４０４で、読み出したノードから、ノード種別を取り出し、ステップＳ４０５で、ノード種別がブランチノードであるか否かを判定する。

[0069] ステップＳ４０５の判定において、読み出したノードがブランチノードである場合は、ステップＳ４０６に進み、ノードから弁別ビット位置についての情報を取り出し、更に、ステップＳ４０７で、取り出した弁別ビット位置に対応するビット値を検索キーから取り出す。そして、ステップＳ４０８ａで、ノードからブロック番号と代表ノード番号を取り出して、ステップＳ４

０９で、検索キーから取り出したビット値と代表ノード番号とを加算し、新たな配列番号として、ステップＳ４０２ｂに戻る。

[0070] 以降、ステップＳ４０５の判定においてリーフノードと判定されてステップＳ４１０に進むまで、ステップＳ４０２からステップＳ４０９までの処理を繰り返す。ステップＳ４１０で、リーフノードからインデックスキーを取り出して、処理を終了する。

[0071] 次に、図５～図８によりカップルドノードツリーにおけるノード挿入処理を説明する。図５～図７Ｅが通常の挿入処理を説明するものであり、図８はルートノードの挿入処理を説明するものである。ルートノードの挿入処理と通常の挿入処理により、カップルドノードツリーが生成されることから、ノード挿入処理の説明はカップルドノードツリーの生成処理の説明でもある。

[0072] 図５は、本発明の一実施形態における挿入処理、特に空ノード対を挿入位置に取得する処理を概念的に説明する図である。図５に例示するカップルドノードツリーは、後に図１２Ｃ～図１２Ｅに示すカップルドノードツリーの部分木である。なお、図５においては、一部において引用符号の表記を省略しているが、図１２Ｃにはすべて表記されている。

[0073] 図５の（１）に示すものは、挿入キーにより検索を行い、挿入キーの挿入位置を求め、空の配列要素の存在する配列である空ブロックを探索した状態である。挿入キー３２７０の値は“０１０００”であり、ビット位置１のみが値“１”であり、他のビット位置の値は“０”である。カップルドノードツリーはルートノード３２１０ｂ、とその下位に直列に連なるノード対３２０１ｃ、ノード対３２０１ｄ、３２０１ｅから構成されている。

[0074] ルートノード３２１０ｂは、ブロック番号がＡである配列３３０９ａの配列番号３２２０ａの配列要素に配置されている。その弁別ビット位置は２、ブロック番号はＢ、代表ノード番号は３２２０ｂである。配列３３０９ａのノード対カウンタ３３９０ａには“２”が格納されている。図の例では配列３３０９ａには部分木のルートノード３２１０ｂしか示されていないが、先に述べたように、図示しない上位のノード対により、配列３３０９ａの配列

要素が使われているものとしている。

[0075] ノード対３２０１ｃの代表ノードはブランチノードであり、ブロック番号がＢである配列３３０９ｂの配列番号３２２０ｂの配列要素に配置されている。その弁別ビット位置は３、ブロック番号はＢ、代表ノード番号は３２２０ｃである。代表ノードと対をなすノードはリーフノードであり、インデックスキーには“００１００”が格納されている。

[0076] ノード対３２０１ｄの代表ノードはブランチノードであり、ブロック番号がＢである配列３３０９ｂの配列番号３２２０ｃの配列要素に配置されている。その弁別ビット位置は４、ブロック番号はＣ、代表ノード番号は３２２０ｄである。代表ノードと対をなすノードはリーフノードであり、インデックスキーには“００００１０”が格納されている。配列３３０９ｂのノード対カウンタ３３９０ｂには“１”が格納されている。

[0077] ノード対３２０１ｅの代表ノードはリーフノードであり、ブロック番号がＣである配列３３０９ｃの配列番号３２２０ｃの配列要素に配置されている。そのインデックスキーには“００００００”が格納されている。また、代表ノードと対をなすノードはリーフノードであり、インデックスキーには“０００００１”が格納されている。配列３３０９ｃのノード対カウンタ３３９０ｃには“２”が格納されている。

[0078] 上述のカップルノードツリーを、検索キーを挿入キー３２７０、検索開始ノードをルートノード３２１０ｂとして検索すると、太枠で示すリーフノードに格納されたインデックスキー“００００００”が検索結果キーとして得られる。そして、検索結果キーと挿入キーの上位からのビット列比較により、最初に異なるビット値となる差分ビット位置を求めると、値１が得られる。差分ビット位置と各弁別ビット位置との相対的な大小関係から、挿入位置として、太枠で示すルートノード３２１０ｂが得られる。

[0079] ここで、挿入位置がルートノード３２１０ｂであるとは、ブランチノードであるルートノードのブロック番号Ｂの配列の代表ノード番号３２２０ｂの指す配列要素に、挿入キーをインデックスキーとして含むリーフノードとそ

れと対をなすノードからなる挿入ノードの代表ノードが配置されるということである。つまり、挿入ノード対の実際に挿入される配列要素は、挿入位置のブランチノードのリンク先である。

[0080] 図に示す例では、各配列のノード対の最大格納数は２としている。そこで、ブロック番号Ｂの配列３３０９ｂのノード対カウンタ３３９０ｂの値は２で最大格納数を示していることから下位のノードの配置された空ブロックを探索し、ブロック番号Ｃの配列３３０９ｃ（以下、ブロックＣということがある。他の配列についても同様である。）から空ノード対３２０１ｆを取得した状態が、図５の（１）に示すものである。

[0081] 図５の（２）に示すものは、空のノード対を含むブロックＣのノードにリンクするブロックＢのノードを含むノード対をブロックＣに移動してブロックＢに空ノード対を確保し、挿入ノード対を挿入位置に挿入可能とした状態である。

図５の（１）に示すブロックＢのノード対３２０１ｄの内容がブロックＣの空ノード対３２０１ｆに書き込まれ、ノード対３２０１ｄの配置されていた配列要素は空になっている。すなわち、挿入ノード対が挿入されるブロックＢに空ノード対が確保されている。したがって、ノード対カウンタ３３９０ｂの値は１つ減って１になっている。また、ノード対カウンタ３３９０ｃの値は１つ増えて２になっている。

[0082] 図５の（３）に示すものは、挿入位置に挿入ノード対を挿入するとともに、挿入位置のブランチノードの弁別ビット位置を更新して挿入処理を完成させた状態である。

挿入キー“０１０００”が検索結果キー“００００００”より大きいことから、挿入キーをインデックスキーとして含むリーフノードはノード対３２０１ｄのノード［１］に配置され、挿入位置のブランチノードであるルートノード３２１０ｂの内容がノード対３２０１ｄのノード［０］に書き込まれている。そして、挿入位置のブランチノードであるルートノード３２１０ｂの弁別ビット位置は差分ビット位置の値１に更新されている。ノード対カウン

タ３３９０ｂの値は２に更新されている。

[0083] 図６は、本発明の一実施形態における挿入処理全体の処理フローの概要を説明する図である。

まず、ステップＳ６００として示すように、挿入キーを検索キーとして、ルートノードよりカップルドノードツリーを検索して検索結果キーを得る。ステップＳ６００の詳細フローは図７Ａを参照して説明する。

次にステップＳ６１０として示すように、挿入ノード対の挿入位置を求める。ステップＳ６１０の詳細フローは図７Ｂを参照して説明する。

次にステップＳ６３０として示すように、ノード対用の空き領域を備えた空格納区域を探索し、空ノード対を取得する。ステップＳ６３０の詳細フローは図７Ｃを参照して説明する。

上述のステップＳ６００、ステップＳ６１０及びステップＳ６３０の処理により、図５の（１）に示す状態が得られる。

[0084] 次にステップＳ６４０として示すように、挿入ノード対の挿入位置に空ノード対を移動する。この処理は、図５の（２）に示す状態に対応する。ステップＳ６４０の詳細フローは図７Ｄを参照して説明する。

[0085] 最後にステップＳ６６０として示すように、空ノード対に挿入キー等を格納して挿入ノード対を完成させる。この処理は、図５の（３）に示す状態に対応する。ステップＳ６６０の詳細フローは図７Ｅを参照して説明する。

[0086] 以下、図７Ａ～図７Ｅを参照して、図６に示す概略フローチャートの各ステップの詳細フローチャートを説明する。この詳細フローチャートにおいては、カップルドノードツリーは配列に格納されているものとして説明する。

[0087] 図７Ａは、挿入キーを検索キーとして、ルートノードよりカップルドノードツリーを検索して検索結果キーを得る図６に示すステップＳ６００の詳細な処理フローを説明する図である。挿入キーは検索キーとして設定されているものとする。

[0088] ステップＳ７０１において、検索開始ノードに、ルートノードの配置された配列のブロック番号とその配列の配列要素の配列番号を設定する。図５に

示す例では、検索開始ノードにブロック番号Ａと配列番号３２２０ａが設定される。

次にステップＳ７０３において、図４に示す検索処理により、検索結果キーとしてのインデックスキーを得る。図５の例示では、インデックスキー“０００００”が得られる。

[0089] ステップＳ７０５において、挿入キー（検索キー）とインデックスキー（検索結果キー）が等しいか判定し、等しければ挿入失敗として処理を終了する。等しくなければ図７Ｂに示すステップＳ７１１に進む。

[0090] 図７Ｂは、挿入ノード対の挿入位置を求める図６に示すステップＳ６１０の詳細な処理フローを説明する図である。

ステップＳ７１１で、挿入キーとステップＳ７０３で得たインデックスキーのビット列比較を例えば排他的論理和で行い、差分ビット列を得る。

ステップＳ７１２に進み、ステップＳ７１１で得た差分ビット列から、上位０ビット目から見た最初の不一致ビットのビット位置を、差分ビット位置に設定する。この処理は、例えばプライオリティエンコーダを有するＣＰＵではそこに差分ビット列を入力し、不一致のビット位置を得ることができる。また、ソフト的にプライオリティエンコーダと同等の処理を行い最初の不一致ビットのビット位置（差分ビット位置）を得ることも可能である。図５に示す例では、差分ビット位置には１が設定される。

[0091] 次にステップＳ７１３で、リーフ位置ポインタに、探索経路スタックのスタックポインタの値を設定する。図５に示す例では、リーフ位置ポインタに設定される値は、探索経路スタックのブロック番号Ｃと配列番号３２２０ｄを指すスタックポインタの値である。探索経路スタックの一番下のスタックエリアを指すポインタ値を０とすると、リーフ位置ポインタに設定される値は３になる。

[0092] 次にステップＳ７１４に進み、探索経路スタックのスタックポインタがルートノードの配列番号を指しているか判定する。指していればステップＳ７２０に移行し、指していなければステップＳ７１５に進む。

ステップＳ７１５において、探索経路スタックのスタックポインタを１つ戻してそこにスタックされているブロック番号と配列番号を取り出す。

ステップＳ７１６に進み、ステップＳ７１５で取り出したブロック番号と配列番号の指す配列要素を配列からノードとして読み出す。

ステップＳ７１７に進み、ステップＳ７１６で読み出したノードから、弁別ビット位置を取り出す。

次にステップＳ７１８に進み、ステップＳ７１７で取り出した弁別ビット位置がステップＳ７１２で設定した差分ビット位置より上位の位置関係か判定する。ここで上位の位置関係とは、ビット列のより左側の位置、すなわちビット位置の値が小さい位置であることとする。

[0093] ステップＳ７１８の判定結果が否定であれば、ステップＳ７１４に戻り、ステップＳ７１８での判定が肯定になるかステップＳ７１４での判定が肯定になるまで繰り返す。ステップＳ７１８での判定が肯定になると、ステップＳ７１９で経路探索スタックのスタックポインタを1つ進め、ステップＳ７２０に移行する。

[0094] ステップＳ７２０では、探索経路スタックから、スタックポインタの指すブロック番号と配列番号を取り出し、挿入位置のブロック番号と配列番号に設定する。次にステップＳ７２１において、挿入位置ポインタに、探索経路スタックポインタの値を設定して、図７Ｃに示すステップＳ７３１に進む。図５に示す例示では、ルートノード３２１０ｂの弁別ビット位置の値が２であり、差分ビット位置より下位なので、ステップＳ７１４での判定が肯定になってステップＳ７１４からステップＳ７１８のループ処理から抜け出す。したがって、挿入位置のブロック番号と配列番号には、ブロック番号Ａと配列番号３２１０ｂが設定される。また、挿入位置ポインタの値には、上述のポインタ値の表記（以下、この表記を断りなく用いることがある。）によれば、ルートノードの配列番号を指すポインタの値である０が設定される。

[0095] 図７Ｃは、ノード対に用いる空配列要素を備えた空ブロックを探索し、空ノード対を取得する図６に示すステップＳ６３０の詳細な処理フローを説明す

る図である。

まずステップＳ７３１で、ブロック番号に、ステップＳ７２０で設定した挿入位置のブロック番号を設定する。図５の例示では、ここでブロック番号に設定されるのはブロック番号Ａである。

[0096] 次にステップＳ７３２で、ブロック番号の指す配列のノード対カウンタの値は上限値か判定する。上限値でなければステップＳ７３６に分岐し、上限値であればステップＳ７３３に進み、探索経路スタックのスタックポインタの値はリーフ位置ポインタの値と等しいか判定する。それらの値が等しくなければ、ステップＳ７３４に分岐し、探索経路スタックのスタックポインタを１つ進めて、スタックポインタの指すブロック番号を取り出し、ステップＳ７３２に戻る。

[0097] 探索経路スタックのスタックポインタの値とリーフ位置ポインタの値が等しければ、ステップＳ７３５において、ブロック管理から、空状態の配列のブロック番号を取得し、ステップＳ７３６に進む。なお、本発明は、配列の空塞がりの管理を行うブロック管理の存在を前提としており、ブロック管理から、空状態の配列のブロック番号を取得するものとしている。また、配列に格納可能なノード対の上限も、ブロック管理から与えられ、ノード対カウンタの値により配列に空が存在するか判断するものとする。

[0098] ステップＳ７３６では、ノード対の移動先のブロック番号に、取得したブロック番号を設定し、図７Ｄに示すステップＳ７４１に進む。ここで設定される取得したブロック番号は、ステップＳ７３１で設定した挿入位置のブロック番号、ステップＳ７３４で取り出したブロック番号、あるいはステップＳ７３５で取得したブロック番号である。

[0099] 図５の例示では、ブロック番号Ａの配列３３０９ａのノード対カウンタ３３９０ａ及びブロック番号Ｂの配列３３０９ｂのノード対カウンタ３３９０ｂの値は上限値であり、ブロック番号Ｃの配列３３０９ｃのノード対カウンタ３３９０ｃの値は上限値でないので、探索経路スタックからブロック番号Ｃが取り出されて、ノード対の移動先のブロック番号に設定される。このとき

、探索経路スタックのポインタ値は、ブロック番号Ｃと代表ノード番号３２２０ｄを指す３である。

[0100] 図７Ｄは、挿入ノード対の挿入位置に空ノード対を移動する図６に示すステップＳ６４０の詳細な処理フローを説明する図である。挿入ノード対の挿入位置に空ノード対を確保するために、空ノード対を取得したブロック番号の配列（移動先）に配置されたノードから、ステップＳ７０３で実行した検索処理のリンク経路を遡り、空ノード対を取得したブロック番号の配列とは異なるブロック番号の配列に配置された上位のノードを含むノード対（移動元）を移動先に移動し、移動元を空ノード対とすることを、挿入位置に空ノード対を確保するまで繰り返す。

[0101] まず、ステップＳ７４１で、探索経路スタックのスタックポインタの値は挿入位置ポインタの値と等しいか判定する。等しければ図７Ｄに示すステップＳ７６１に分岐し、等しくなければステップＳ７４２に進む。ステップＳ７４２では、探索経路スタックのスタックポインタの指すブロック番号と配列番号を取り出し、スタックポインタを１つ戻す。

次にステップＳ７４３で、取り出したブロック番号とステップＳ７３６で設定した移動先のブロック番号は一致するか判定する。一致すればステップＳ７４１に戻り、一致しなければステップＳ７４４に進む。図５に示す例示では、図７Ｄに示す処理の最初の処理のときには探索経路スタックのスタックポインタの値は３であり、移動先のブロック番号はＣである。したがって、ステップＳ７４１～ステップＳ７４３の処理ループを一周してからステップＳ７４４に進む。このとき、ステップＳ７４２で取り出したブロック番号はＢであり、探索経路スタックのスタックポインタの値は１になっている。

[0102] ステップＳ７４４では、ノード対の移動元のブロック番号にステップＳ７４２で取り出したブロック番号を、ノード対の移動元の代表ノード番号にステップＳ７４２で取り出した配列番号から得た代表ノード番号を設定する。ここで配列番号から代表ノード番号を得ることは、ノード［０］とノード［１］の配列番号は１つ違いであり、その大小とどちらが偶数であるか決めら

れていることにより可能であることは明らかである。

図５の例示では、ノード対の移動元のブロック番号にはＢが、代表ノード番号には３２２０ｃが設定される。

[0103] 次に、ステップＳ７４５で、ステップＳ７３６で設定したノード対の移動先のブロック番号の配列から空のノード対の代表ノードの配列番号を取得するとともに、ノード対カウンタを１つ加算し、ステップＳ７４６で、ノード対の移動先の代表ノード番号に、ステップＳ７４５で取得した空のノード対の代表ノードの配列番号を設定する。

次に、ステップＳ７４７において、ステップＳ７４４で設定したノード対の移動元のブロック番号と代表ノード番号の指す配列要素の組の内容をノード対として読み出し、ステップＳ７４８で、該読み出したノード対の内容を、ステップＳ７４６で設定したノード対の移動先のブロック番号と代表ノード番号の指す配列要素の組に書き込む。

[0104] 図５の例示では、図５の（１）に示すように、ノード対の移動先のブロック番号がＣである配列３３０９ｃから空のノード対３２０１ｆが取得され、その代表ノードの配列番号は、図５の（２）に示すように３２２０ｃ’である。そして、ノード対カウンタ３３９０ｃは１つ加算され１から２になっており、ノード対の移動元のブロック番号Ｂと代表ノード番号３２２０ｃの指す配列要素の組（ノード対３２０１ｄ）の内容がノード対の移動先のブロック番号Ｃと代表ノード番号３２２０ｃ’の指す配列要素の組（ノード対３２０１ｆ）に書き込まれている。ノード対３２０１ｆと代表ノード番号３２２０ｆに付された括弧書き（３２０１ｄ）と（３２２０ｄ）は、それぞれの値が図５の（１）で示すものの内容と等しいことを示している。

[0105] 次に、ステップＳ７４９において、探索経路スタックのスタックポインタの指すブロック番号と配列番号を取り出し、ステップＳ７５０で、該取り出したブロック番号と配列番号の指す配列要素の、ブロック番号にノード対の移動先のブロック番号を、代表ノード番号にノード対の移動先の代表ノード番号を書き込む。

次に、ステップＳ７５１において、ノード対の移動元のブロック番号と代表ノード番号の指すノード対を解放するとともに、ノード対カウンタを１つ減算する。そして、ステップＳ７５２において、ノード対の移動先のブロック番号に、ノード対の移動元のブロック番号を設定してステップＳ７４１に戻る。

[0106] 図５に示す例では、探索経路スタックのスタックポインタの値は１となっており、ブロック番号Ｂと配列番号３２２０ｂが取り出される。ブロック番号Ｂと配列番号３２２０ｂの指すノード３２１０ｃのブロック番号にノード対の移動先のブロック番号Ｃを、代表ノード番号にノード対の移動先の代表ノード番号３２２０ｃ’を、書き込んだ状態が図５の（２）に示されている。また、ノード対の移動元のブロック番号Ｂと代表ノード番号３２２０ｃの指すノード対３２０１ｄを解放するとともに、ノード対カウンタ３３９０ｂを１つ減算して１としている。また、ノード対の移動先のブロック番号には、ノード対の移動元のブロック番号Ｂが設定される。

[0107] 図５に示す例においては、ステップＳ７４１に最初に戻ったとき、探索経路スタックのスタックポインタの値は１であるので、ステップＳ７４２を経て探索経路スタックのスタックポインタの値を０としてからステップＳ７４３に至り、比較対象のブロック番号が共にＢであるので、再度ステップＳ７４１に戻る。このときは、探索経路スタックのスタックポインタの値は０であって、挿入位置ポインタの値と等しいので、図７Ｅに示すステップＳ７６１に進む。

[0108] 図７Ｅは、空ノード対に挿入キー等を格納して挿入ノード対を完成させる図６に示すステップＳ６６０の詳細な処理フローを説明する図である。

まず、ステップＳ７６１において、空のノード対のブロック番号に、ノード対の移動先のブロック番号を設定し、次にステップＳ７６２に進み、空のノード対のブロック番号の配列から、空のノード対の代表ノードの配列番号を取得するとともに、ノード対カウンタを１つ加算する。

[0109] 次にステップＳ７６３において、挿入キーとステップＳ７０３で得たインデ

ックスキーの大小を比較し、挿入キーが大きいときは値１を小さいときは値０のブール値を得る。

次にステップＳ７６４で、ステップＳ７６２で設定した代表ノード番号にステップＳ７６３で得たブール値を加えて、リーフノードの配列番号に設定する。また、ステップＳ７６５において、ステップＳ７６２で設定した代表ノードの配列番号にステップＳ７６３で得たブール値の論理否定値を加算した配列番号を対ノードの配列番号に設定する。ステップＳ７６４で設定した配列番号は、挿入キーをインデックスキーとして含むリーフノードが配置される配列要素の配列番号であり、ステップＳ７６５で設定した配列番号は、そのリーフノードとノード対を成すノードが配置される配列要素のものである。

[0110] 次にステップＳ７６６において、空ノード対のブロック番号とステップＳ７６４で設定したリーフノードの配列番号の指す配列要素の、ノード種別にリーフを、インデックスキーに挿入キーを書き込む。

次にステップＳ７６７において、挿入位置のブロック番号と配列番号の指す配列要素からその内容を読み出し、ステップＳ７６８で、該読み出した内容を空ノード対のブロック番号とステップＳ７６５で設定した配列番号の指す配列要素に書き込む。

最後にステップＳ７６９において、挿入位置のブロック番号と配列番号の指す配列要素の、ノード種別にブランチを、弁別ビット位置に差分ビットビット位置を、代表ノード番号に挿入するノード対の代表ノード番号を書き込み、処理を終了する。

[0111] 図８は、本発明の一実施の形態におけるルートノードの挿入処理を含むインデックスキーを挿入する場合のノード挿入処理全体を説明する処理フロー図である。

ステップＳ８０１において、取得することを求められたカップルドノードツリーのルートノードのブロック番号と配列番号が登録済みであるか判定される。登録済みであれば、図６～図７Ｅを用いて説明した通常の挿入処理が

行われる。

[0112] ステップ８０１での判定が登録済みでなければ、まったく新しいカップルドノードツリーの登録、生成が始まることになる。

まず、ステップＳ８０２において、ブロック管理から、空状態の配列のブロック番号を取得する。次にステップＳ８０３において、該取得したブロック番号の指す配列から空きのノード対の代表ノードとなるべき配列要素の配列番号を取得するとともに、ノード対カウンタを１つ加算する。

[0113] 次にステップＳ８０４において、ステップＳ８０３で得た配列番号に０を加えた配列番号を求める（実際には、ステップＳ８０３で取得した配列番号に等しい。）。さらにステップＳ８０５において、ブロック番号とステップＳ８０４で得た配列番号の指す配列要素の、ノード種別にリーフを、インデックスキーに挿入キーを書き込む。

最後にステップＳ８０６で、ステップＳ８０２とステップＳ８０４でそれぞれ取得したルートノードのブロック番号と配列番号を登録して処理を終了する。

[0114] 先にも述べたように、インデックスキーの集合があるとき、そこから順次インデックスキーを取り出し、図８及び図６～図７Ｅの処理を繰り返すことにより、インデックスキーの集合に対応した本発明のカップルドノードツリーを構築することができることは明らかである。

[0115] 次に図９、図１０を参照して、本発明の一実施の形態におけるカップルドノードツリーに係るインデックスキーの集合から、特定のインデックスキーを削除する処理フローを説明する。

[0116] 図９は、削除処理の前段である検索処理の処理フローを説明する図である。削除対象のインデックスキーは削除キーとして指定されているものとする。

まず、ステップＳ９０１で、検索開始ノードに、ルートノードの配置された配列のブロック番号とルートノードの配置された配列要素の配列番号を設定する。

[0117] 次にステップＳ９０２において、図４に示す検索処理により、削除キーを検索キーとして検索開始ノードより配列を検索し、検索結果キーとしてのインデックスキーを得る。

次にステップ９０３において、削除キーとインデックスキーを比較し、等しくなければ削除するインデックスキーはカップルドノードツリーに存在しないのであるから、削除は失敗となり、処理を終了する。等しければ次の処理、図１０のステップＳ９０４以下の処理に進む。

[0118] 図１０は、削除処理の後段の処理フローを説明する図である。

まず、ステップＳ９０４で探索経路スタックに２つ以上の配列番号が格納されているか判定する。２つ以上の配列番号が格納されていないということは、言い換えれば１つだけで、その配列番号はルートノードの格納された配列要素のものである。その場合はステップＳ９１２に移行し、ステップＳ９０１で得たブロック番号の指す配列を解放する。次にステップＳ９１３に進み、ルートノードのブロック番号と配列番号の登録を抹消して処理を終了する。

[0119] ステップＳ９０４において探索経路スタックに２つ以上の配列番号が格納されていると判定されたときはステップＳ９０５に進み、ステップＳ９０２において実行した図４に示す検索処理のステップＳ４０８ａで得た代表ノード番号にステップＳ４０７で得たビット値を反転した値を加算した配列番号を得る。この処理は、削除対象のインデックスキーが格納されたリーフノードと対をなすノードの配置された配列番号を求めるものである。

[0120] 次にステップＳ９０６において、ステップＳ９０５で得た配列番号の配列要素の内容を読み出し、さらにステップＳ９０７において、探索経路スタックのスタックポインタを１つ戻してブロック番号と配列番号を取り出す。

次にステップＳ９０８に進み、ステップＳ９０７で読み出した配列要素の内容をステップＳ９０７で得たブロック番号と配列番号の指す配列要素に書き込む。この処理は、削除対象のインデックスキーが格納されたリーフノードへのリンク元であるブランチノードを上記リーフノードと対をなすノード

に置き換えるものである。

[0121] 次にステップＳ９０９において、上述の図４に示すステップＳ４０８ａで得たブロック番号と代表ノード番号の指すノード対を削除するとともに、ノード対カウンタの値を１つ減算し、ステップＳ９１０に進む。

ステップＳ９１０では、ノード対カウンタの値は０か判定し、０でなければ処理を終了し、０であれば、ステップＳ９１１において、上述の図４に示すステップＳ４０８ａで得たブロック番号の指す配列を解放して処理を終了する。

[0122] 次に、具体例により、本発明の一実施形態に係る削除処理と挿入処理を説明する。

図１１Ａは、ノードの削除前のカップルドノードツリーと削除処理において検索処理を実行した探索経路スタックの状態を説明する図である。

[0123] 図１１Ａに例示するカップルドノードツリーは、ノード対１２０１ａ、１２０１ｂ、１２０１ｃで構成されている。ノード対１２０１ａとノード対１２０１ｂはブロック番号Ａの配列１３０９ａに配置され、ノード対１２０１ｃはブロック番号Ｂの配列１３０９ｂに配置されている。したがって、配列１３０９ａのノード対カウンタ１３９０ａの値は２であり、配列１３０９ｂのノード対カウンタ１３９０ｂの値は１である。

[0124] ノード対１２０１ａの代表ノードであるルートノード１２１０ａは配列１３０９ａの配列番号１２２０の配列要素に配置されており、ノード種別１２６０ａは０であってブランチノードである。弁別ビット位置１２３０ａには０が保持され、ブロック番号１２４０ａにはＡが保持されている。また、代表ノード番号には、ノード対１２０１ｂの代表ノード１２１０ｂの配置された配列番号である１２２０ａが保持されている。

[0125] ノード対１２０１ｂの代表ノード１２１０ｂはノード種別１２６０ｂが０であってブランチノードであり、弁別ビット位置１２３０ｂには２が保持され、ブロック番号にはＢが保持されている。また、代表ノード番号には、ノード対１２０１ｃの代表ノード１２１０ｃの配置された配列番号である１２

２０ｂが保持されている。

代表ノード１２１０ｂと対をなすノード１２１１ｂはノード種別１２６１ｂが１であってリーフノードであり、インデックスキー１２５１ｂには“１００００”が格納されている。

[0126] ノード対１２０１ｃの代表ノード１２１０ｃはノード種別１２６０ｃが１であってリーフノードであり、インデックスキー１２５０ｃには“０００００”が格納されている。

代表ノード１２１０ｃと対をなすノード１２１１ｃはノード種別１２６１ｃが１であってリーフノードであり、インデックスキー１２５１ｃには“００１００”が格納されている。

[0127] 削除キー１２７０には、“００１００”が設定されている。この削除キーを検索キーとしてルートノード１２１０ａより検索をした経路が、探索経路スタック３１０にスタックされている。探索経路スタック３１０には、配列番号とブロック番号の組が、（１２２０、Ａ）、（１２２０ａ、Ａ）、（１２２０ｂ＋１、Ｂ）とスタックされている。これに対応してリンク経路が、カップルドノードツリーの太枠で囲んだノードと太線の矢印で示されている。検索結果としてリーフノード１２１１ｃのインデックスキー“００１００”が得られ、リーフノード１２１１ｃが削除対象の削除ノードとなることが図１１Ａに示されている。また、太線の矢印で示す探索経路スタック３１０のスタックポインタは、配列番号１２２０ｂ＋１とブロック番号Ｂを指している。

[0128] 図１１Ｂは、削除処理を完了した後のカップルドノードツリー等の状態を説明する図である。図１１Ａに示す削除ノード１２１１ｃと対をなすノード１２１０ｃの内容が、探索経路スタック３１０のスタックポインタを１つ戻したところにスタックされているブロック番号Ａと配列番号１２２０ａの指すノード１２１０ｂに書き込まれた状態が図１１Ｂに示されている。ノード種別１２６０ｂの括弧書き（１２６０ｃ）は、ノード種別１２６０ｂにはノード種別１２６０ｃの値が書き込まれたことを示している。インデックスキー

１２５０ｂとその括弧書き（１２５０ｃ）についても同様である。

[0129] 削除ノード１２１１ｃとそれと対をなし、その内容が直近上位のブランチノード１２１０ｂに書き込まれたノード１２１０ｃからなるノード対１２０１ｃは削除され、ノード対カウンタ１３９０ｂの値は１つ減って０となっている。したがって、ブロック番号Ｂの配列１３０９ｂは解放される。

[0130] 次に、図１２Ａ及び図１２Ｂを参照して挿入処理の具体例１を説明する。具体例１は、探索経路スタック上のブロック番号の指す配列に空のノード対を確保する配列要素が存在しない場合のものである。

[0131] 図１２Ａは、ノードの挿入前のカップルドノードツリーと探索経路ブロックの状態とブロック番号Ｂの配列２３０９ｂを新たに取得した状態を説明する図である。

図１２Ａに示すように、ノード対２２０１ａとノード対２２０１ｂからなるカップルドノードツリーがブロック番号Ａの配列２３０９ａに配置されている。したがって、配列２３０９ａのノード対カウンタ２３９０ａの値は２である。

[0132] ノード対２２０１ａの代表ノードであるルートノード２２１０ａは、ブロック番号Ａの配列２３０９ａの配列番号２２２０の配列要素に配置されている。ルートノード２２１０ａのノード種別は０であってブランチノードである。弁別ビット位置２２３０ａには０が保持され、ブロック番号２２４０ａにはＡが保持されている。また、代表ノード番号には、ノード対２２０１ｂの代表ノード２２１０ｂの配置された配列番号である２２２０ａが保持されている。

ノード対２２０１ｂの代表ノード２２１０ｂはノード種別２２６０ｂが１であってリーフノードであり、インデックスキー２２５０ｂには“００００１”が格納されている。代表ノード２２１０ｂと対をなすノード２２１１ｂはノード種別２２６１ｂが１であってリーフノードであり、インデックスキー２２５１ｂには“１００００”が格納されている。

[0133] 挿入キー２２７０には、“００００００”が設定されている。この挿入キー

を検索キーとしてルートノード２２１０ａより検索をしたリンク経路が、カップルドノードツリーの太枠で囲んだノードと太線の矢印で示され、さらに検索結果としてリーフノード２２１０ｂのインデックスキー“００００１”が得られることが示されている。

[0134] 探索経路スタック３１０には、配列番号とブロック番号の組が、（２２２０、Ａ）、（２２２０ａ、Ａ）とスタックされている。挿入キーと検索結果キーの差分ビット位置は４であるから、検索処理後に１つ戻された探索経路スタックのスタックポインタは図７Ｂに示すステップＳ７１９の処理により１つ進められ、図１２Ａに太線の矢印で示すように、配列番号２２２０ａとブロック番号Ａを指している。

したがって、挿入位置のブロック番号はＡ、配列番号は２２２０ａであるから、リーフノード２２１０ｂが挿入位置となることが図１２Ａに示されている。

[0135] 図１２Ａの例示では、ノード対カウンタ２３９０ａが上限値となっていること、探索経路スタックのスタックポインタはリーフ位置ポインタであることから、ノード対カウンタ２３０９ｂの値が０の空のブロック番号Ｂの配列２３０９ｂを新たに取得する。さらに配列２３０９ｂから、代表ノード２２１０ｃとそれと対をなすノード２２１１ｃからなる空ノード対２２０１ｃを取得した状態が図１２Ａに示されている。

[0136] 図１２Ｂは、挿入処理後のカップルドノードツリーを説明する図である。挿入キー“００００００”と検索結果のインデックスキー“００００１”の大小関係により、確保した空ノード対２２０１ｃのノード［０］を、挿入キー“００００００”をインデックスキー２２５０ｃとして含むリーフノード２２１０ｃとし、ノード［１］を、検索結果キー“００００１”

をインデックスキー２２５１ｃとして含むリーフノード２２１１ｃとしている。また、ノード対カウンタ２３９０ｃの値は１つ増えて１となっている。

挿入位置のノード２２１０ｂはブランチノードとなり、ノード種別２２６０ｂは０に更新され、その弁別ビット位置には挿入キーと検索結果キーの差

分ビット位置である４が格納され、ブロック番号２２４０ｂにはＢが格納されている。また、代表ノード番号２２２０ｂは挿入されたノード対２２０１ｃの代表ノード２２１０ｃが配置された配列要素の配列番号が格納されている。

[0137] 次に図１２Ｃ、図１２Ｄ及び図１２Ｅを参照して、挿入処理の具体例２を説明する。具体例２は、探索経路スタック上のブロック番号の指す配列に空のノード対を確保する配列要素が存在する場合のものである。図１２Ｃ、図１２Ｄ及び図１２Ｅに例示するカップルドノードツリーは、図５にその部分木を示したものである。図５には記載されていないノード対３２０１ａと挿入位置のノード３２１０ｂと対をなすノード３２１１ｂが示されている。その他のツリー構造は、図５に示すものと同じであるので、説明を省略する。

[0138] 図１２Ｃは、図５の（１）に示すものに対応し、ノードの挿入前のカップルドノードツリーと探索経路ブロックの状態と探索経路スタック上のブロック番号の指す配列に空ノード対を取得した状態を説明する図である。

ノード対３２０１ａの代表ノードであるルートノード３２１０ａは、ブロック番号Ａの配列３３０９ａの配列番号３２２０の配列要素に配置されている。ルートノード３２１０ａのノード種別３２６０ａは０であってブランチノードである。弁別ビット位置３２３０ａには０が保持され、ブロック番号３２４０ａにはＡが保持されている。また、代表ノード番号には、ノード対３２０１ｂの代表ノード３２１０ｂの配置された配列番号である３２２０ａが保持されている。

代表ノード３２１０ｂと対をなすノード３２１１ｂのノード種別３２６１ｂはリーフノードであることを示す１であり、インデックスキー３２５１ｂには“１００００”が格納されている。

[0139] 図５に示すものと同じく、挿入キー３２７０には、“０１０００”が設定されている。この挿入キーを検索キーとしてルートノード３２１０ａより検索をしたリンク経路が、カップルドノードツリーの太枠で囲んだノードと太線の矢印で示され、さらに検索結果としてリーフノード３２１０ｅのインデッ

クスキー“０００００”が得られることが示されていることも、ブランチノード３２１０ｂ以降は図５の（１）に示すものと同じである。

探索経路スタック３１０には、配列番号とブロック番号の組が、（３２２０、Ａ）、（３２２０ａ、Ｂ）、（３２２０ｂ、Ｂ）、（３２２０ｃ、Ｂ）、（３２２０ｄ、Ｃ）とスタックされている。探索経路スタック３１０のスタックポインタは、図１２Ｃに太線の矢印で示すように、最初に見つかった空ノード対の存在する配列３３０９ｃのブロック番号Ｃと配列番号３２２０ｄを指している。

[0140] また、図５の（１）に示すものと同様に、ブロック番号Ｃの配列３３０９ｃに空ノード対３２０１ｆを取得した状態が示されている。

図１２Ｃの例示では、検索開始ノードはノード３２１０ａであり、弁別ビット位置３２３０ａが０、ブロック番号３２４０がＡ、代表ノード番号３２２０ａがノード対３２０１ｂの代表ノードを指しており、挿入キーの０ビット目のビット値が０である。したがって、挿入処理の検索処理において、ルートノード３２１０ａからノード３２１０ｂにリンクする。それ以降の処理は図５の（１）を参照して説明したものとほぼ同じである。図５の（１）に示すものでは、検索開始ノードがノード３２１０ｂであって、その弁別ビット位置が“２”で差分ビット位置の値“１”より下位であるから、図７Ｂに示すステップＳ７２０にはステップＳ７１４から直接分岐するものであるが、図１２Ｃに示すものでは、検索開始ノードがノード３２１０ａであって、その弁別ビット位置が“０”で差分ビット位置の値“１”より上位であるから、ステップＳ７１９を介してステップＳ７２０に進む。したがって、どちらの場合も、ノード３２１０ｂが挿入位置となり、同じになるので、説明を省略する。

[0141] 図１２Ｄは、図５の（２）に示すものに対応する図であり、空のノード対を含むブロックＣのノードにリンクするブロックＢのノードを含むノード対をブロックＣに移動してブロックＢに空ノード対を確保し、挿入ノード対を挿入位置に挿入可能とした状態を示している。各ブロック番号の配列の状態

については図５の（２）について説明したとおりである。

探索経路スタック３１０のスタックポインタは、図１２Ｃに示す最初に見つかった空ノード対の存在する配列３３０９ｃのブロック番号Ｃと配列番号３２２０ｄを指すものから、空ノード対に移動してブロック番号Ｂの配列３３０９ｂに空ノード対を確保するノード対３２０１ｄの直近上位のノード３２１０ｃのブロック番号Ｂと配列番号３２２０ｃを指すポインタ値から１つ戻した、ブロック番号Ｂと配列番号３２２０ｂを指すものとなっている。このポインタ値を１つ戻す操作は、図７Ｄに示すステップＳ７４２で実行される。

[0142] 図１２Ｅは、図５の（３）に示すものに対応する図であり、挿入位置に挿入ノード対を挿入するとともに、挿入位置のブランチノードの弁別ビット位置を更新して挿入処理を完成させた状態を示している。各ブロック番号の配列の状態については図５の（３）について説明したとおりである。

探索経路スタック３１０のスタックポインタは、図７Ｂに示すステップＳ７２１で設定された挿入位置ポインタと等しく、ブロック番号Ａと配列番号３２２０ａを指すものとなっている。

[0143] 以上本発明を実施するための最良の形態について詳細に説明したが、本発明の実施の形態はそれに限ることなく種々の変形が可能であることは当業者に明らかである。例えばリーフノードが、インデックスキー自体を含むことに代えてインデックスキーを格納した記憶領域の位置を示す情報を含むようにすることが可能であることは、当業者に自明である。

[0144] また、本発明のビット列検索方法を実行する装置が、カップルドノードツリーを格納する記憶手段と図４に例示した処理をコンピュータに実行させるプログラムによりコンピュータ上に構築可能なことは明らかである。そして、そのプログラムにより、ブランチノードとリーフノードの識別手段、ブランチノードの弁別ビット位置に応じてリンク先のノード対のどちらかにリンクする手段等がコンピュータ上に実現される。

[0145] また、本発明のインデックスキー挿入方法あるいはインデックスキー削除

方法を実行する装置が、図６あるいは図９及び図１０に例示した処理をコンピュータに実行させるプログラムによりコンピュータ上に構築可能なことも明らかである。

したがって、上記プログラム、及びプログラムを記憶したコンピュータ読み取り可能な記憶媒体は、本発明の実施の形態に含まれる。さらに、本発明のカップルドノードツリーのデータ構造も、本発明の実施の形態に含まれる。

請求の範囲

[1] ルートノードと、隣接した記憶領域に配置されるブランチノードとリーフノードまたはブランチノード同士またはリーフノード同士のノード対、からなるビット列検索に用いるツリーであって、

前記ルートノードは、ツリーの始点を表すノードであって、該ツリーのノードが１つのときは前記リーフノード、ツリーのノードが２つ以上のときは前記ブランチノードであり、

前記ブランチノードは、ビット列検索を行う検索キーの弁別ビット位置とリンク先のノード対の格納区域の識別子と該ノード対の一方のノードである代表ノードの前記格納区域における位置を示す位置情報を含み、前記リーフノードは、検索対象のビット列からなるインデックスキーを含むカップルドノードツリーを用いたビット列検索装置において、

前記ツリーの任意のノードを検索開始ノードとして、前記ブランチノードにおいて、該ブランチノードに含まれる弁別ビット位置の検索キーのビット値に応じて、該ブランチノードに含まれる前記格納区域の識別子の示す格納区域に格納されたリンク先のノード対の代表ノードかあるいはそれと隣接した記憶領域に配置されたノードにリンクすることを順次前記リーフノードに至るまで繰り返すことにより、前記リーフノードに格納されたインデックスキーを、前記検索開始ノードをルートノードとする前記ツリーの任意の部分木の前記検索キーによる検索結果である検索結果キーとすることを特徴とするビット列検索装置。

[2] 前記格納区域は配列であって、前記カップルドノードツリーは複数個の配列に分割して記憶され、前記位置情報は、該位置情報に対応する前記代表ノードが格納された前記配列の配列要素の配列番号であり、

スタックを備え、前記検索開始ノードの格納された配列の識別子と該配列の配列要素の配列番号、及び前記検索開始ノードから前記リーフノードに至るリンク先のノードの格納された配列の識別子と該配列の配列要素の配列番号が、順次前記スタックに保持されていくことを特徴とする請求項１記載の

ビット列検索装置。

[3] ルートノードと、隣接した記憶領域に配置されるブランチノードとリーフノードまたはブランチノード同士またはリーフノード同士のノード対、からなるビット列検索に用いるツリーであって、

前記ルートノードは、ツリーの始点を表すノードであって、該ツリーのノードが１つのときは前記リーフノード、ツリーのノードが２つ以上のときは前記ブランチノードであり、

前記ブランチノードは、ビット列検索を行う検索キーの弁別ビット位置とリンク先のノード対の格納区域の識別子と該ノード対の一方のノードである代表ノードの前記格納区域における位置を示す位置情報を含み、前記リーフノードは、検索対象のビット列からなるインデックスキーを含むカップルドノードツリーを用いたビット列検索方法において、

前記ツリーの任意のノードを検索開始ノードとし、

前記ブランチノードにおいて、該ブランチノードに含まれる弁別ビット位置の検索キーのビット値に応じて、該ブランチノードに含まれる前記格納区域の識別子の示す格納区域に格納されたリンク先のノード対の代表ノードかあるいはそれと隣接した記憶領域に配置されたノードにリンクすることを順次前記リーフノードに至るまで繰り返し、

前記リーフノードに格納されたインデックスキーを、前記検索開始ノードをルートノードとする前記ツリーの任意の部分木の前記検索キーによる検索結果である検索結果キーとする、

ことを特徴とするビット列検索方法。

[4] 前記格納区域は配列であって、前記カップルドノードツリーは複数個の配列に分割して記憶され、前記位置情報は、該位置情報に対応する前記代表ノードが格納された前記配列の配列要素の配列番号であり、

スタックを備え、前記検索開始ノードの格納された配列の識別子と該配列の配列要素の配列番号、及び前記検索開始ノードから前記リーフノードに至るリンク先のノードの格納された配列の識別子と該配列の配列要素の配列番

号が、順次前記スタックに保持されていくことを特徴とする請求項３記載のビット列検索方法。

[5] 請求項３または４記載のビット列検索方法をコンピュータに実行させるためのプログラム。

[6] ビット列検索に用いるツリー状のデータ構造であって、

ルートノードと、隣接した記憶領域に配置されるブランチノードとリーフノードまたはブランチノード同士またはリーフノード同士のノード対、からなり、

前記ルートノードは、ツリーの始点を表すノードであって、該ツリーのノードが１つのときは前記リーフノード、ツリーのノードが２つ以上のときは前記ブランチノードであり、

前記ブランチノードは、ビット列検索を行う検索キーの弁別ビット位置とリンク先のノード対の格納区域の識別子と該ノード対の一方のノードである代表ノードの前記格納区域における位置を示す位置情報を含み、

前記リーフノードは、検索対象のビット列からなるインデックスキーを含み、

前記ツリーの任意のノードを検索開始ノードとして、前記ブランチノードにおいて、該ブランチノードに含まれる弁別ビット位置の検索キーのビット値に応じて、該ブランチノードに含まれる前記格納区域の識別子の示す格納区域に格納されたリンク先のノード対の代表ノードかあるいはそれと隣接した記憶領域に配置されたノードにリンクすることを順次前記リーフノードに至るまで繰り返すことにより、前記リーフノードに格納されたインデックスキーを、前記検索開始ノードをルートノードとする前記ツリーの任意の部分木の前記検索キーによる検索の実行を可能とすることを特徴とするデータ構造。

[7] ルートノードと、隣接した記憶領域に配置されるブランチノードとリーフノードまたはブランチノード同士またはリーフノード同士のノード対、からなるビット列検索に用いるツリーであって、前記ルートノードは、ツリーの

始点を表すノードであって、該ツリーのノードが１つのときは前記リーフノード、ツリーのノードが２つ以上のときは前記ブランチノードであり、前記ブランチノードは、ビット列検索を行う検索キーの弁別ビット位置とリンク先のノード対の格納区域の識別子と該ノード対の一方のノードである代表ノードの前記格納区域における位置を示す位置情報を含み、前記リーフノードは、検索対象のビット列からなるインデックスキーを含むカップルドノードツリーに、挿入キーとして指定されたビット列からなるインデックスキーを含むリーフノードを挿入するインデックスキー挿入方法において、

前記挿入キーを検索キーとして、前記ブランチノードにおいて該ブランチノードに含まれる弁別ビット位置の検索キーのビット値に応じて該ブランチノードに含まれる前記格納区域の識別子の示す格納区域に格納されたリンク先のノード対の代表ノードかあるいはそれと隣接した記憶領域に配置されたノードにリンクすることを、前記ルートノードから順次前記リーフノードに至るまでリンク経路を記憶しながら繰り返し、前記リーフノードに格納されたインデックスキーを、前記検索キーによる検索結果である検索結果キーとして取得する検索ステップと、

前記検索キーと前記検索結果キーの間でビット列比較を行い、ビット列比較で異なるビット値となる先頭のビット位置と、前記リンク経路上のブランチノードの弁別ビット位置との相対的位置関係により、挿入される前記リーフノードともう一方のノードからなる挿入ノード対の挿入される格納区域の識別子と挿入ノード対の代表ノードの該格納区域における位置情報を含むブランチノードの位置情報を示す挿入位置を決定する挿入位置決定ステップと、

前記挿入ノード対を格納可能な空き領域を備えた空格納区域を探索し、空ノード対を取得する空格納区域探索ステップと、

前記空ノード対に前記空格納区域の直近上位の格納区域のノード対の内容を格納し、該直近上位の格納区域のノード対を解放して空ノード対を取得する操作を、該直近上位の格納区域が、前記挿入ノード対が挿入される格納区域となるまで繰り返す空ノード対移動ステップと、

前記検索キーと前記検索結果キーの大小関係に応じて、挿入される前記リーフノードを前記挿入ノード対のどちらのノードとするかを決定し、前記指定されたインデックスキーを前記リーフノードに格納するリーフノード挿入ステップと、

を備えたことを特徴とするインデックスキー挿入方法。

[8] 前記格納区域は配列であって、前記カップルドノードツリーは複数個の配列に分割して記憶され、前記位置情報は、該位置情報に対応する前記代表ノードが格納された前記配列の配列要素の配列番号であり、

スタックを備え、前記ルートノードの格納された配列の識別子と該配列の配列要素の配列番号、及び前記ルートノードから前記リーフノードに至るリンク先のノードの格納された配列の識別子と該配列の配列要素の配列番号が、順次前記スタックに保持されていくことを特徴とする請求項７記載のインデックスキー挿入方法。

[9] ルートノードと、隣接した記憶領域に配置されるブランチノードとリーフノードまたはブランチノード同士またはリーフノード同士のノード対、からなるビット列検索に用いるツリーであって、前記ルートノードは、ツリーの始点を表すノードであって、該ツリーのノードが１つのときは前記リーフノード、ツリーのノードが２つ以上のときは前記ブランチノードであり、前記ブランチノードは、ビット列検索を行う検索キーの弁別ビット位置とリンク先のノード対の格納区域の識別子と該ノード対の一方のノードである代表ノードの前記格納区域における位置を示す位置情報を含み、前記リーフノードは、検索対象のビット列からなるインデックスキーを含むカップルドノードツリーから、削除キーとして指定されたビット列からなるインデックスキーを含むリーフノードを削除するインデックスキー削除方法において、

前記削除キーを検索キーとして、前記ブランチノードにおいて該ブランチノードに含まれる弁別ビット位置の検索キーのビット値に応じて該ブランチノードに含まれる前記格納区域の識別子の示す格納区域に格納されたリンク先のノード対の代表ノードかあるいはそれと隣接した記憶領域に配置された

ノードにリンクすることを、前記ルートノードから順次前記削除キーとして指定されたビット列からなるインデックスキーを含むリーフノードに至るまでリンク経路を記憶しながら繰り返し、

前記リーフノードと対をなすもう一方のノードの内容を、当該ノード対のリンク元のブランチノードに書き込み、

当該ノード対を削除する、

ことを特徴とするインデックスキー削除方法。

[10] 前記格納区域は配列であって、前記カップルドノードツリーは複数個の配列に分割して記憶され、前記位置情報は、該位置情報に対応する前記代表ノードが格納された前記配列の配列要素の配列番号であり、

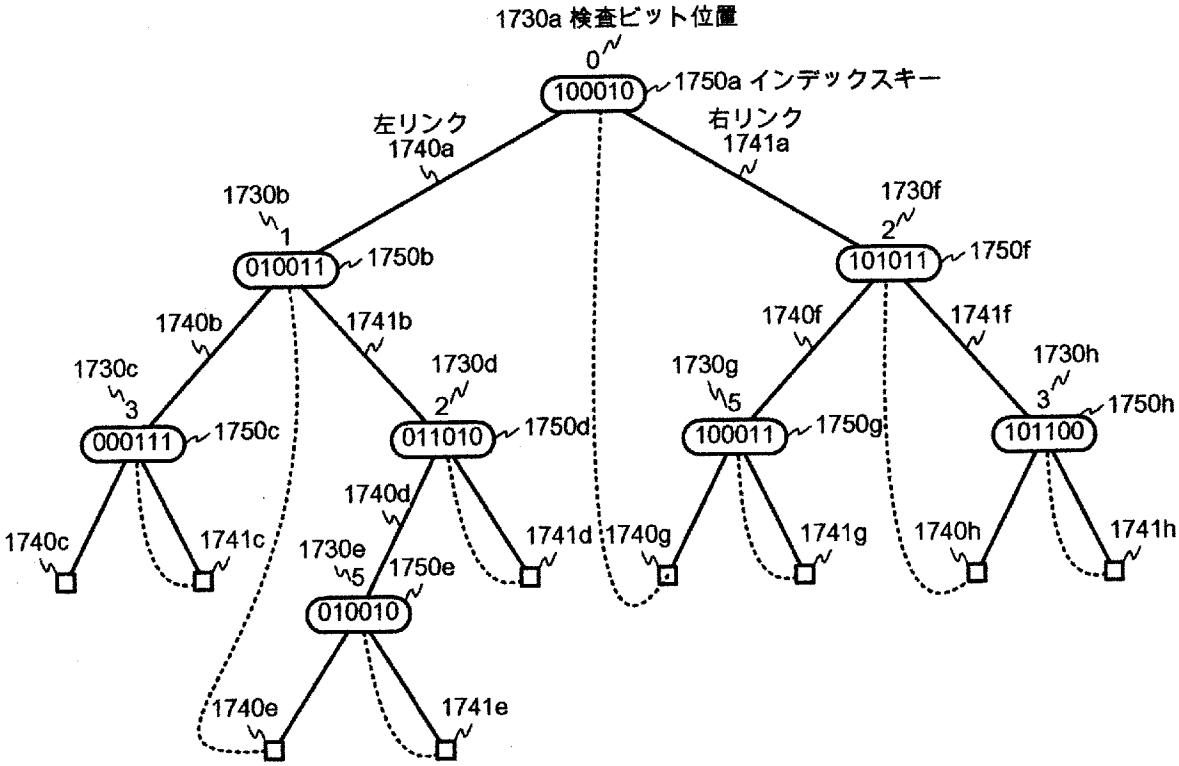
スタックを備え、前記ルートノードの格納された配列の識別子と該配列の配列要素の配列番号、及び前記ルートノードから前記リーフノードに至るリンク先のノードの格納された配列の識別子と該配列の配列要素の配列番号が、順次前記スタックに保持されていくことを特徴とする請求項９記載のインデックスキー削除方法。

[11] 請求項７または８記載のインデックスキー挿入方法をコンピュータに実行させるためのプログラム。

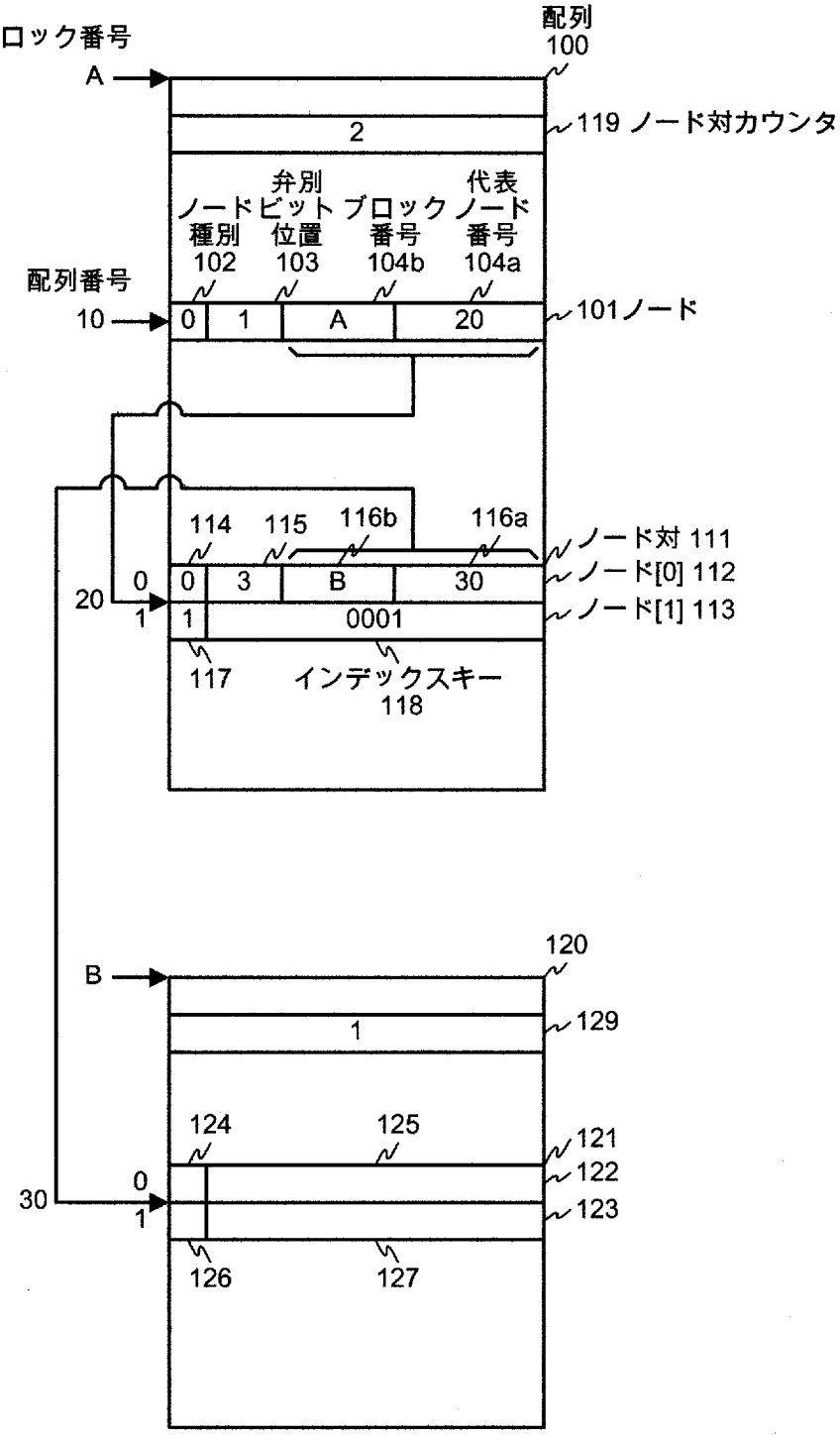
[12] 請求項９または１０記載のインデックスキー削除方法をコンピュータに実行させるためのプログラム。

[13] 請求項５、請求項１１又は請求項１２いずれか１項記載のプログラムを記録したコンピュータ読み取り可能な記録媒体。

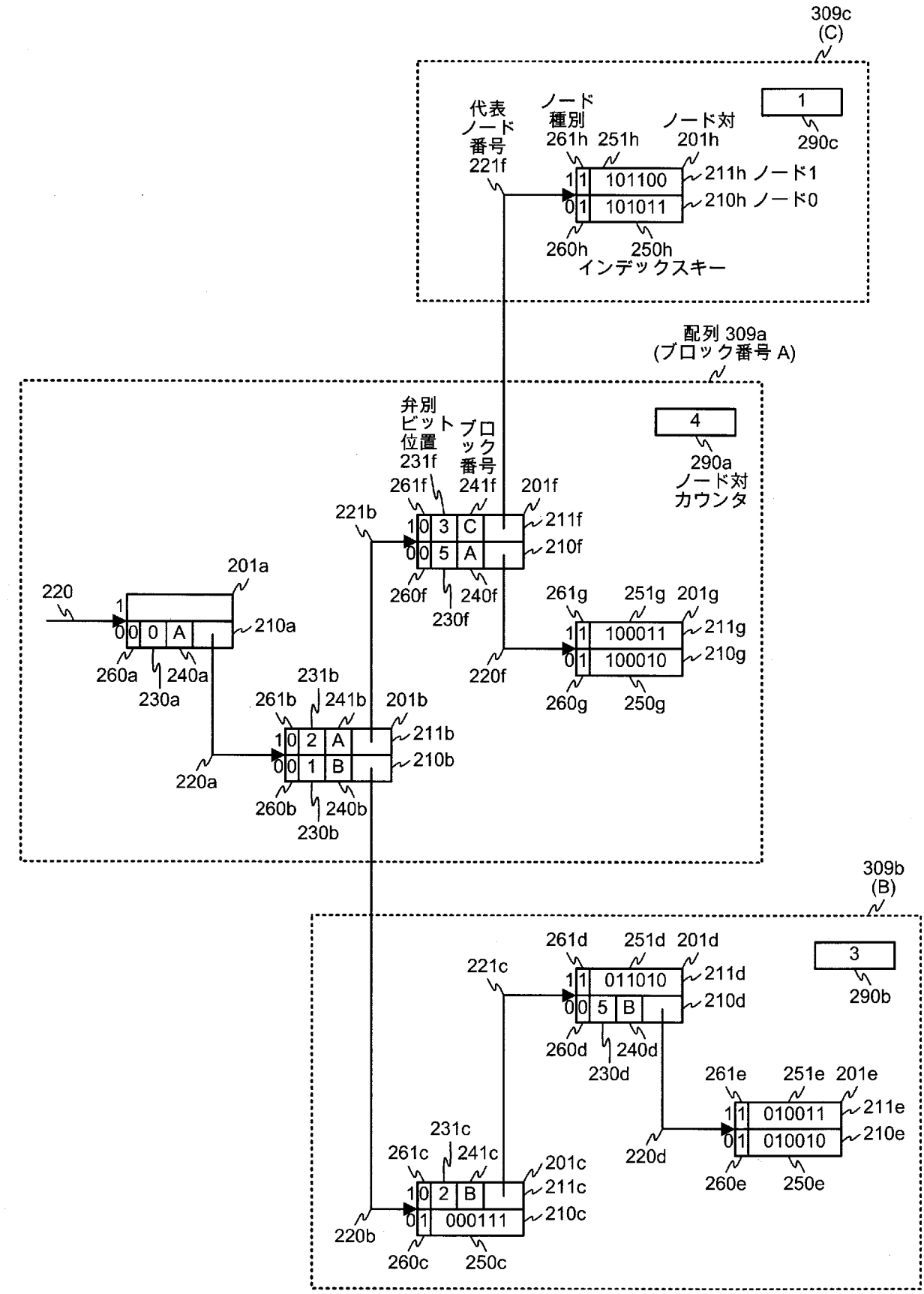
[図1]


1730a 検査ビット位置
0
100010
1750a インデックスキー
左リンク
1740a
右リンク
1741a
1730b
1
010011
1750b
1730f
2
101011
1750f
1740b
1741b
1740f
1741f
1730c
3
000111
1750c
1730d
2
011010
1750d
1730g
5
100011
1750g
1730h
3
101100
1750h
1740d
1740c
1741c
1730e
5
1750e
1741d
1740g
1741g
1740h
1741h
010010
1740e
1741e

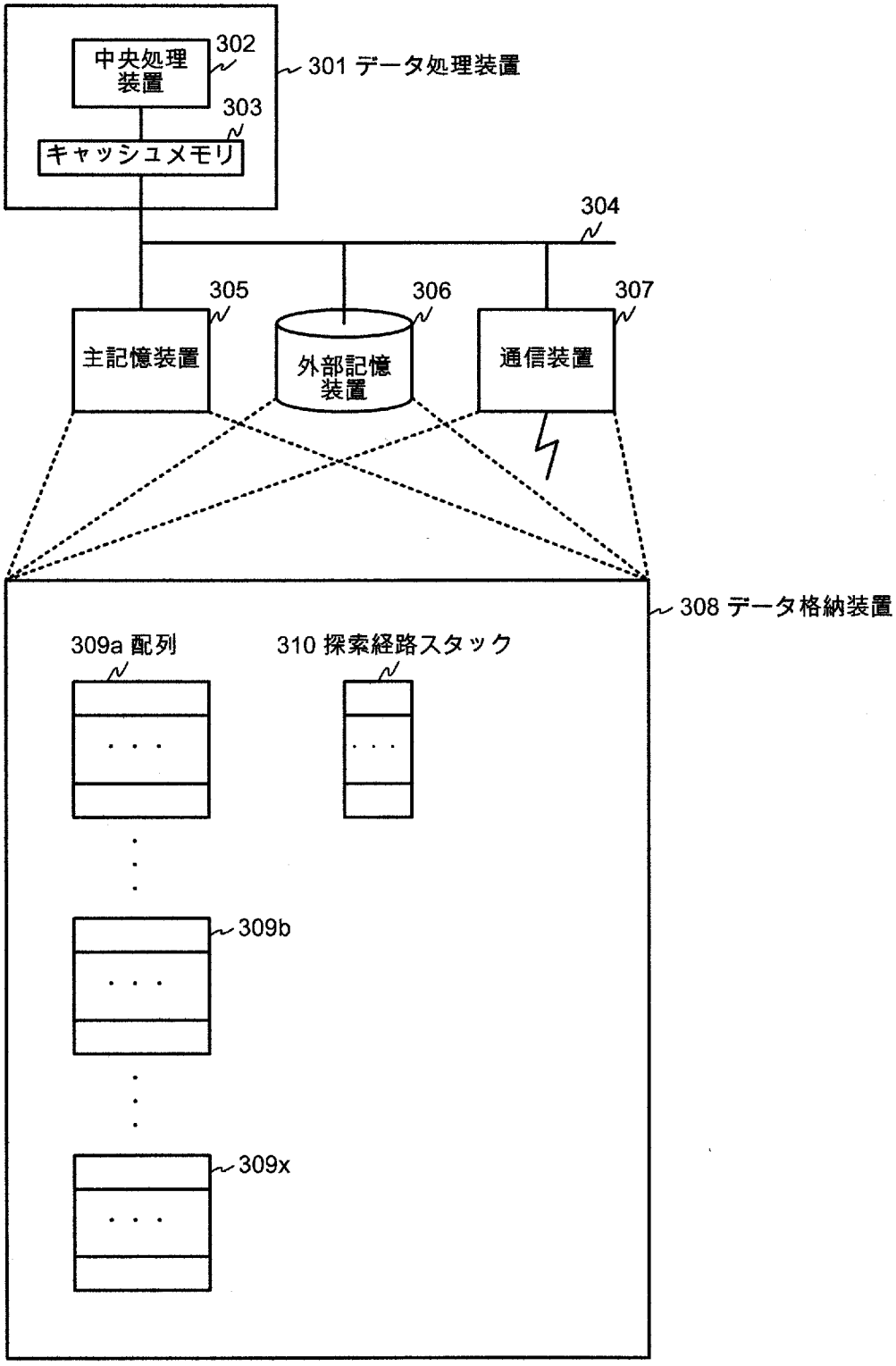
[図2A]


ブロック番号
配列
100
A
2
119 ノード対カウンタ
弁別
ノードビット
ブロック
代表
ノード
種別
位置
番号
番号
102
103
104b
104a
配列番号
10
0
1
A
20
101ノード
114
115
116b
116a
ノード対 111
20
0
0
3
B
30
ノード[0] 112
1
1
0001
ノード[1] 113
117
インデックスキー
118
120
B
1
129
124
125
121
122
30
0
1
123
126
127

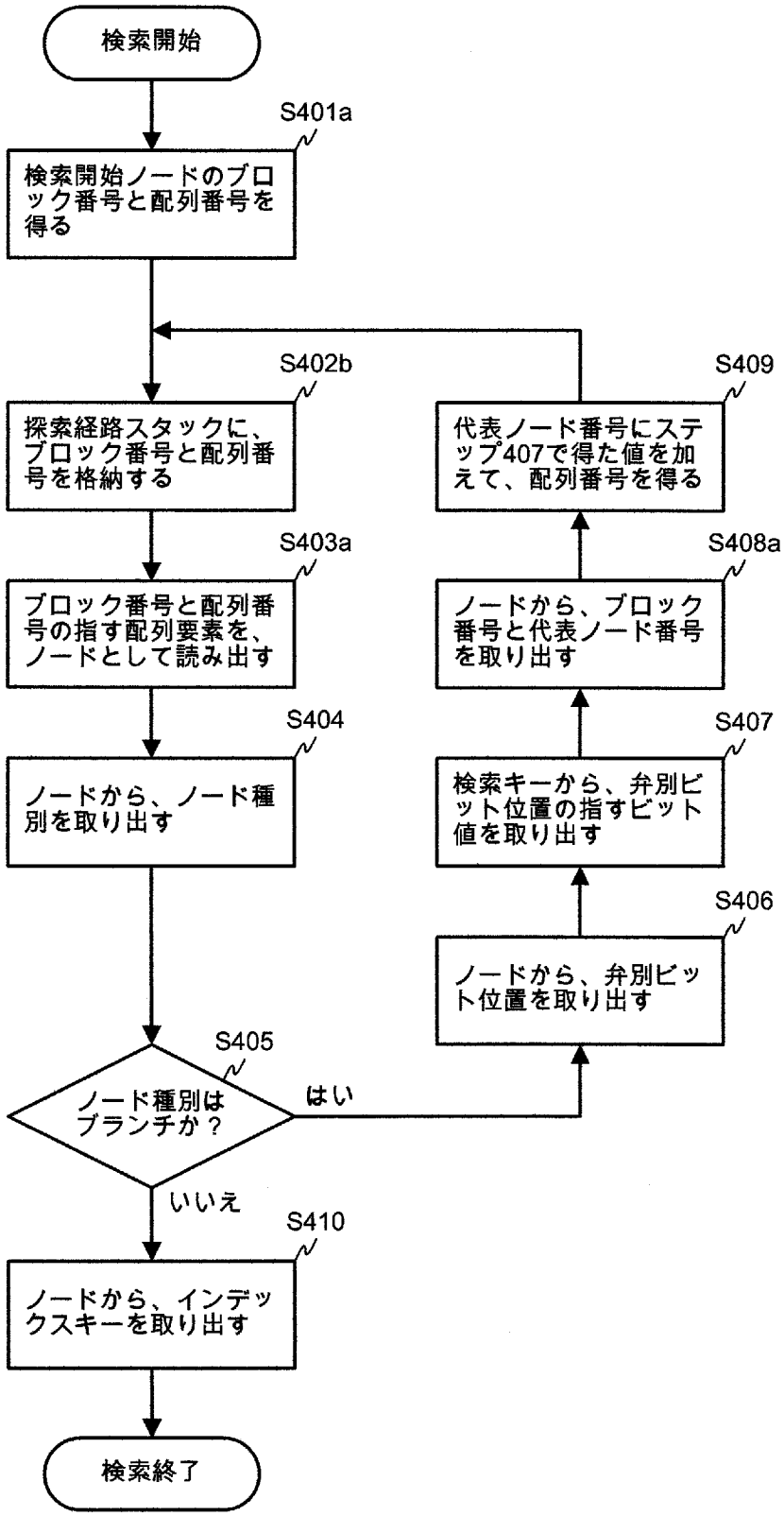
[図2B]


309c
(C)
1
290c
代表
ノード
番号
221f
ノード
種別
261h
251h
ノード対
201h
1 1 101100
211h ノード1
0 1 101011
210h ノード0
260h
250h
インデックスキー
配列 309a
(ブロック番号 A)
4
290a
ノード対
カウンタ
弁別
ビット
位置
231f
ブロック
番号
241f
261f
201f
221b
1 0 3 C
211f
0 0 5 A
210f
260f
240f
230f
220
201a
1
0 0 0 A
210a
260a
240a
230a
261g
251g
201g
1 1 100011
211g
0 1 100010
210g
220f
260g
250g
231b
261b
241b
201b
1 0 2 A
211b
0 0 1 B
210b
220a
260b
240b
230b
309b
(B)
261d
251d
201d
221c
1 1 011010
211d
0 0 5 B
210d
3
290b
260d
240d
230d
261e
251e
201e
1 1 010011
211e
0 1 010010
210e
220d
260e
250e
231c
261c
241c
201c
1 0 2 B
211c
0 1 000111
210c
220b
260c
250c

[図3]

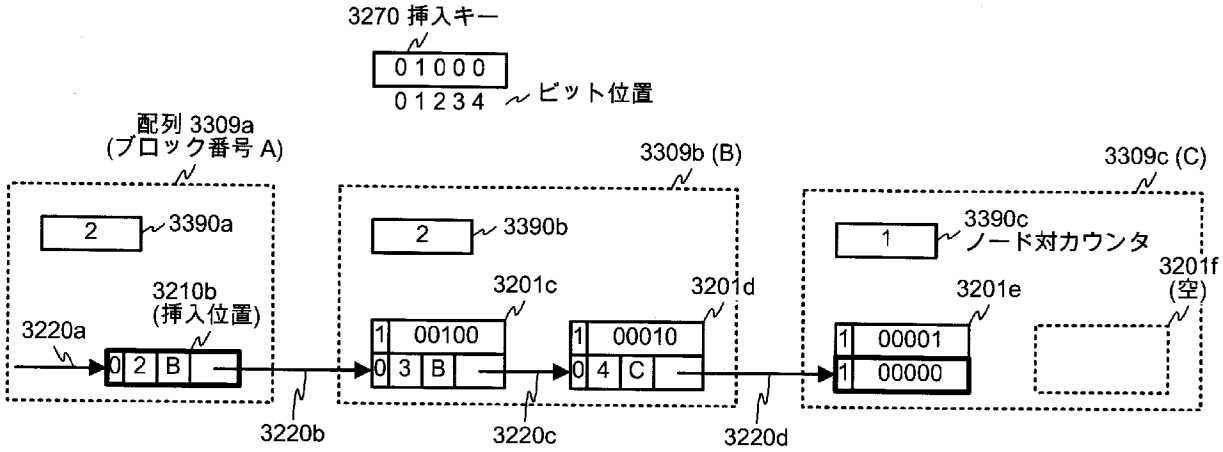

中央処理装置
302
301 データ処理装置
303
キャッシュメモリ
304
305
306
307
主記憶装置
外部記憶装置
通信装置
308 データ格納装置
309a 配列
310 探索経路スタック
309b
309x

[図4]

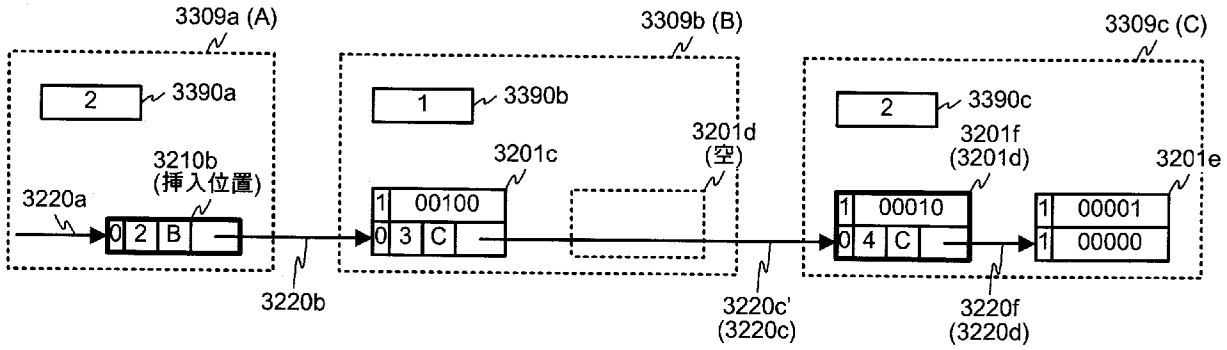

検索開始
S401a
検索開始ノードのブロック番号と配列番号を得る
S402b
探索経路スタックに、ブロック番号と配列番号を格納する
S403a
ブロック番号と配列番号の指す配列要素を、ノードとして読み出す
S404
ノードから、ノード種別を取り出す
S405
ノード種別はブランチか？
はい
いいえ
S406
ノードから、弁別ビット位置を取り出す
S407
検索キーから、弁別ビット位置の指すビット値を取り出す
S408a
ノードから、ブロック番号と代表ノード番号を取り出す
S409
代表ノード番号にステップ407で得た値を加えて、配列番号を得る
S410
ノードから、インデックスキーを取り出す
検索終了

[図5]

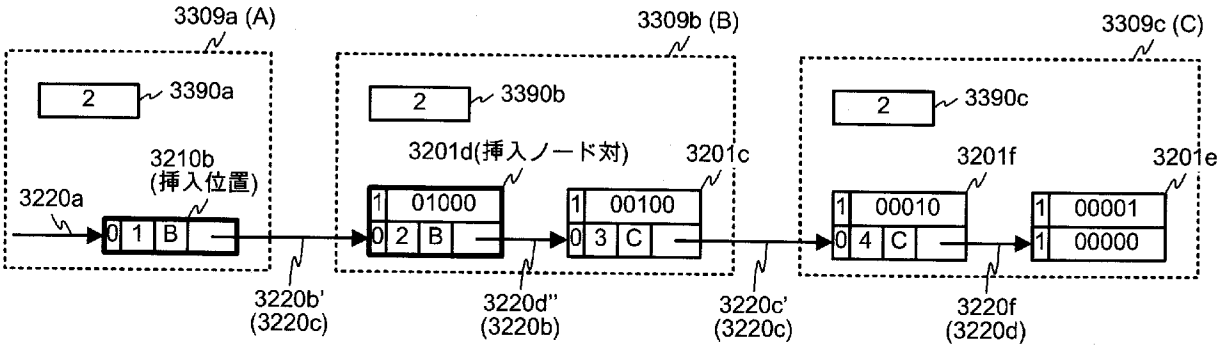
(1)


3270 挿入キー
01000
01234
ビット位置
配列 3309a
(ブロック番号 A)
3309b (B)
3309c (C)
2
3390a
2
3390b
1
3390c
ノード対カウンタ
3210b
(挿入位置)
3201c
3201d
3201e
3201f
(空)
3220a
1 00100
0 3 B
1 00010
0 4 C
1 00001
1 00000
0 2 B
3220b
3220c
3220d

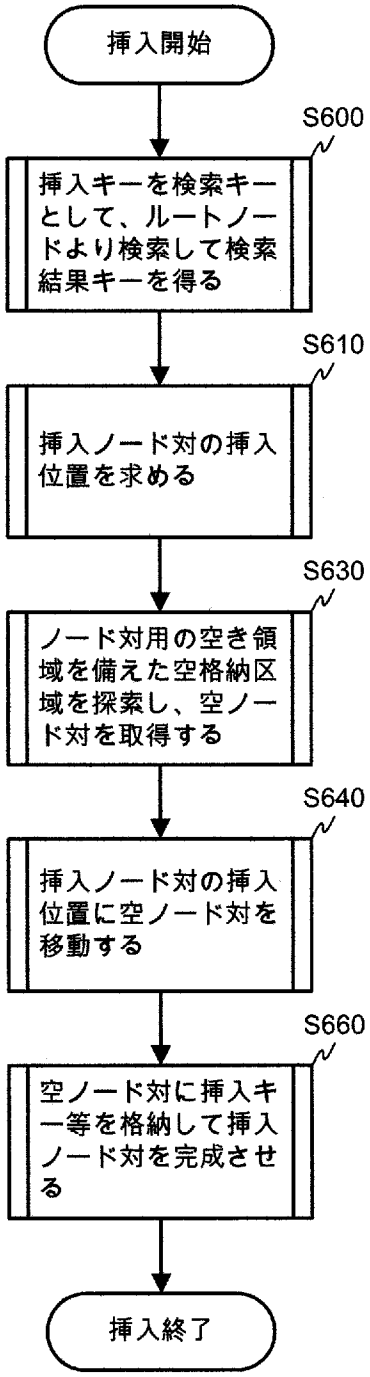
(2)


3309a (A)
3309b (B)
3309c (C)
2
3390a
1
3390b
2
3390c
3210b
(挿入位置)
3201c
3201d
(空)
3201f
(3201d)
3201e
3220a
1 00100
0 3 C
1 00010
0 4 C
1 00001
1 00000
0 2 B
3220b
3220c'
(3220c)
3220f
(3220d)

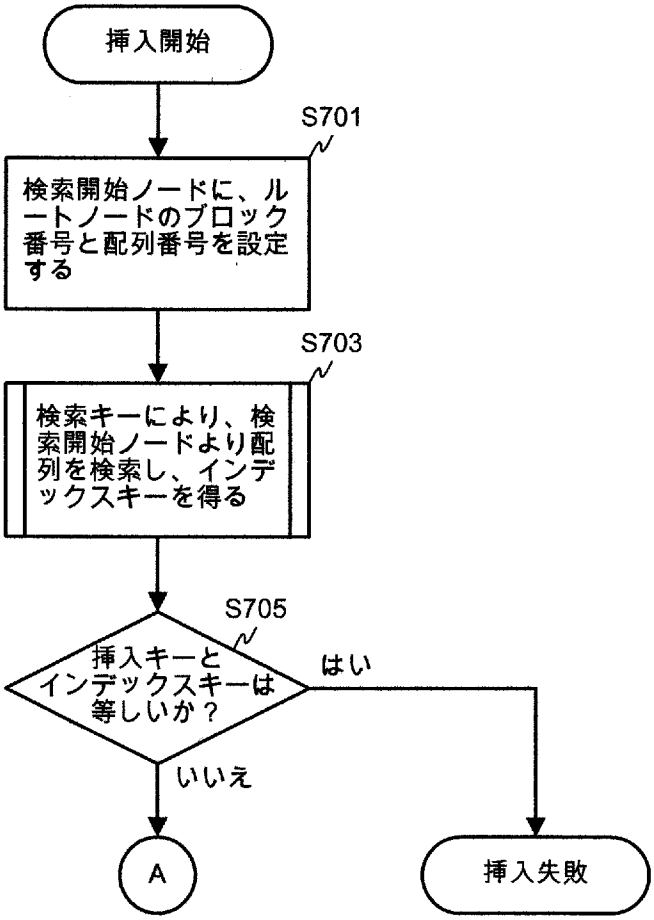
(3)


3309a (A)
3309b (B)
3309c (C)
2
3390a
2
3390b
2
3390c
3210b
(挿入位置)
3201d(挿入ノード対)
3201c
3201f
3201e
3220a
1 01000
0 2 B
1 00100
0 3 C
1 00010
0 4 C
1 00001
1 00000
0 1 B
3220b'
(3220c)
3220d''
(3220b)
3220c'
(3220c)
3220f
(3220d)

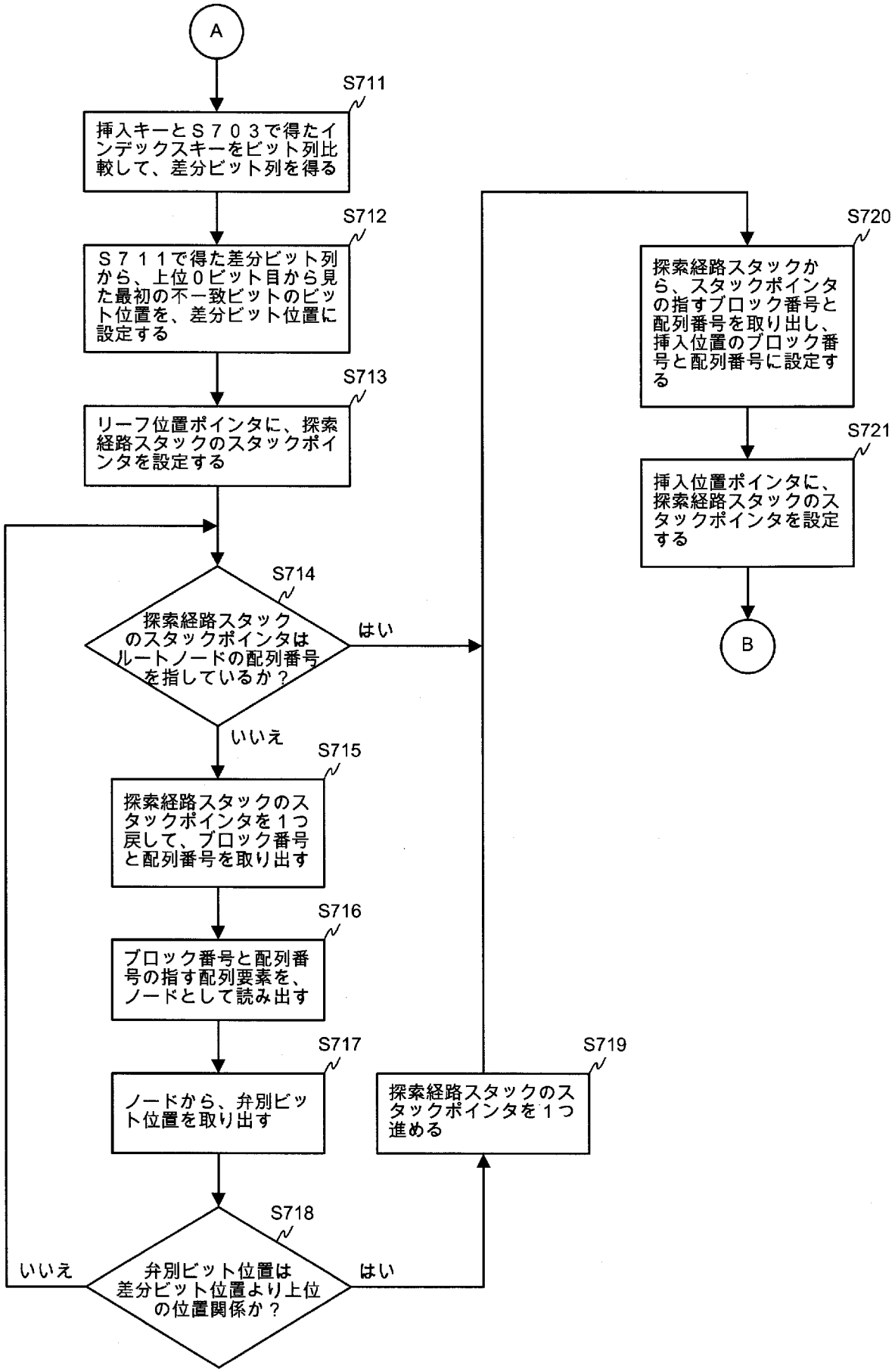
[図6]


挿入開始
S600
挿入キーを検索キーとして、ルートノードより検索して検索結果キーを得る
S610
挿入ノード対の挿入位置を求める
S630
ノード対用の空き領域を備えた空格納区域を探索し、空ノード対を取得する
S640
挿入ノード対の挿入位置に空ノード対を移動する
S660
空ノード対に挿入キー等を格納して挿入ノード対を完成させる
挿入終了

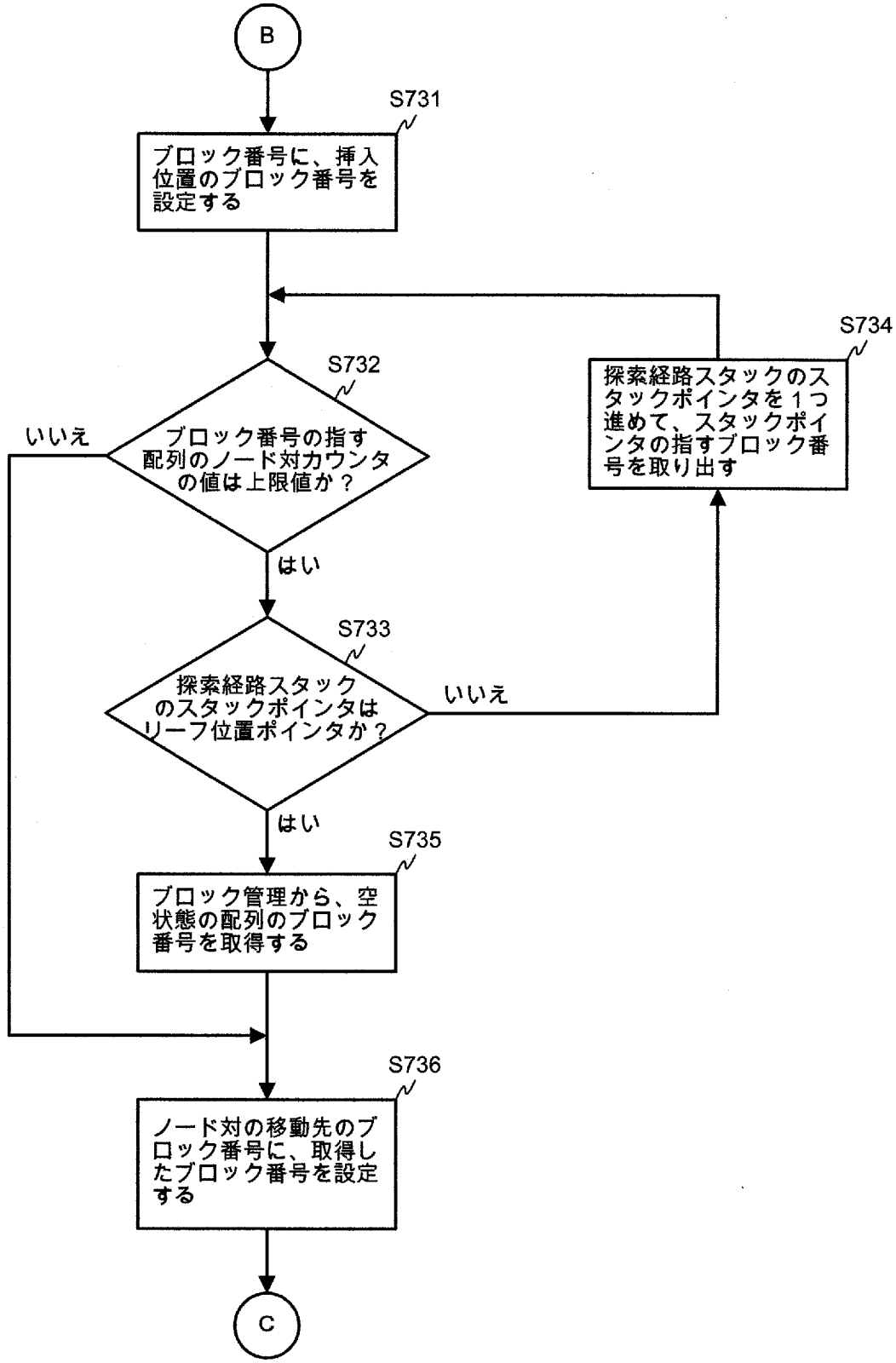
[図7A]


挿入開始
S701
検索開始ノードに、ルートノードのブロック番号と配列番号を設定する
S703
検索キーにより、検索開始ノードより配列を検索し、インデックスキーを得る
S705
挿入キーとインデックスキーは等しいか？
はい
いいえ
A
挿入失敗

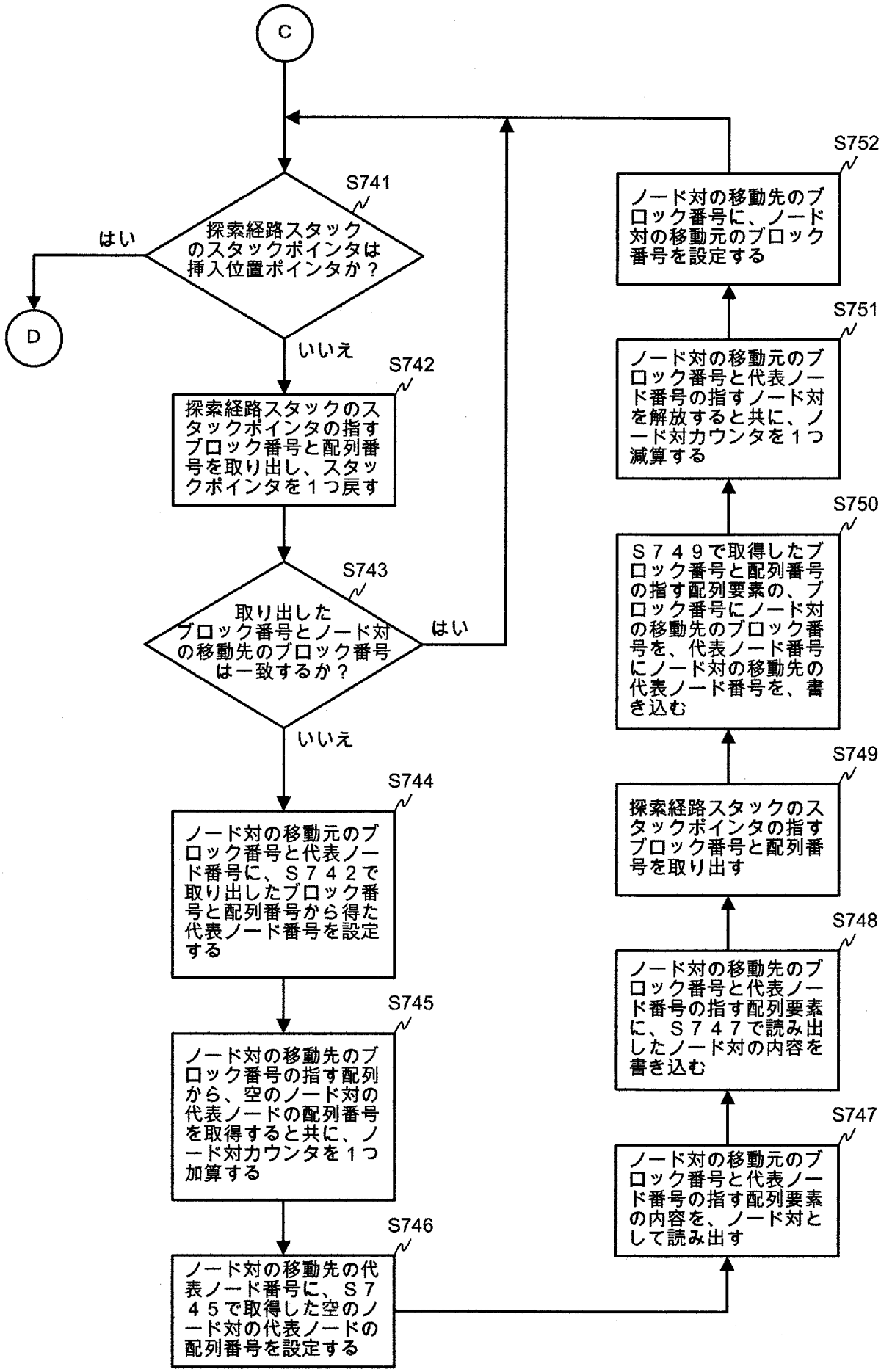
[図7B]


A
S711
挿入キーとＳ７０３で得たインデックスキーをビット列比較して、差分ビット列を得る
S712
Ｓ７１１で得た差分ビット列から、上位0ビット目から見た最初の不一致ビットのビット位置を、差分ビット位置に設定する
S713
リーフ位置ポインタに、探索経路スタックのスタックポインタを設定する
S714
探索経路スタックのスタックポインタはルートノードの配列番号を指しているか？
はい
いいえ
S715
探索経路スタックのスタックポインタを１つ戻して、ブロック番号と配列番号を取り出す
S716
ブロック番号と配列番号の指す配列要素を、ノードとして読み出す
S717
ノードから、弁別ビット位置を取り出す
S718
弁別ビット位置は差分ビット位置より上位の位置関係か？
いいえ
はい
S719
探索経路スタックのスタックポインタを１つ進める
S720
探索経路スタックから、スタックポインタの指すブロック番号と配列番号を取り出し、挿入位置のブロック番号と配列番号に設定する
S721
挿入位置ポインタに、探索経路スタックのスタックポインタを設定する
B

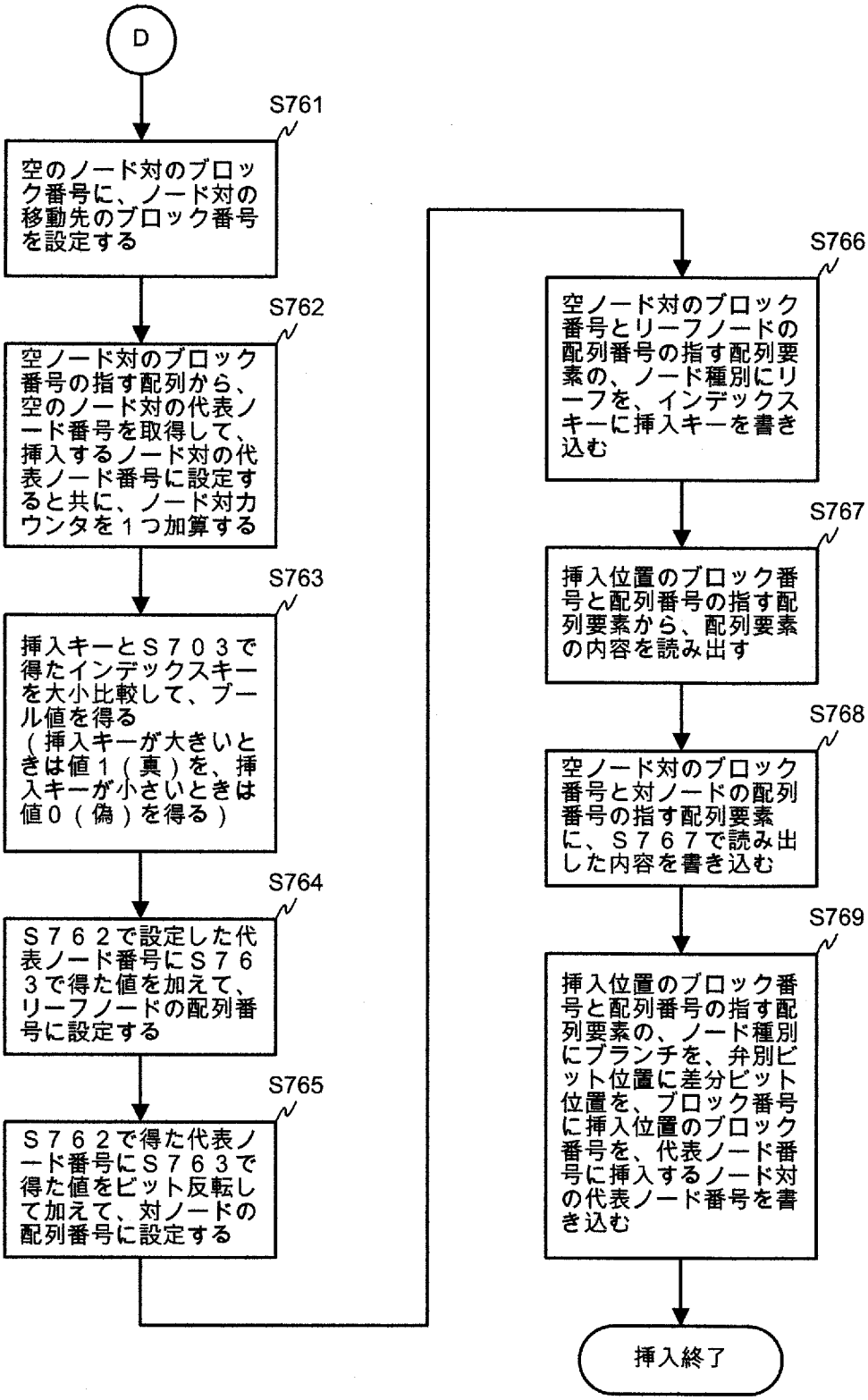
[図7C]


B
S731
ブロック番号に、挿入位置のブロック番号を設定する
S732
ブロック番号の指す配列のノード対カウンタの値は上限値か？
いいえ
はい
S733
探索経路スタックのスタックポインタはリーフ位置ポインタか？
いいえ
はい
S734
探索経路スタックのスタックポインタを１つ進めて、スタックポインタの指すブロック番号を取り出す
S735
ブロック管理から、空状態の配列のブロック番号を取得する
S736
ノード対の移動先のブロック番号に、取得したブロック番号を設定する
C

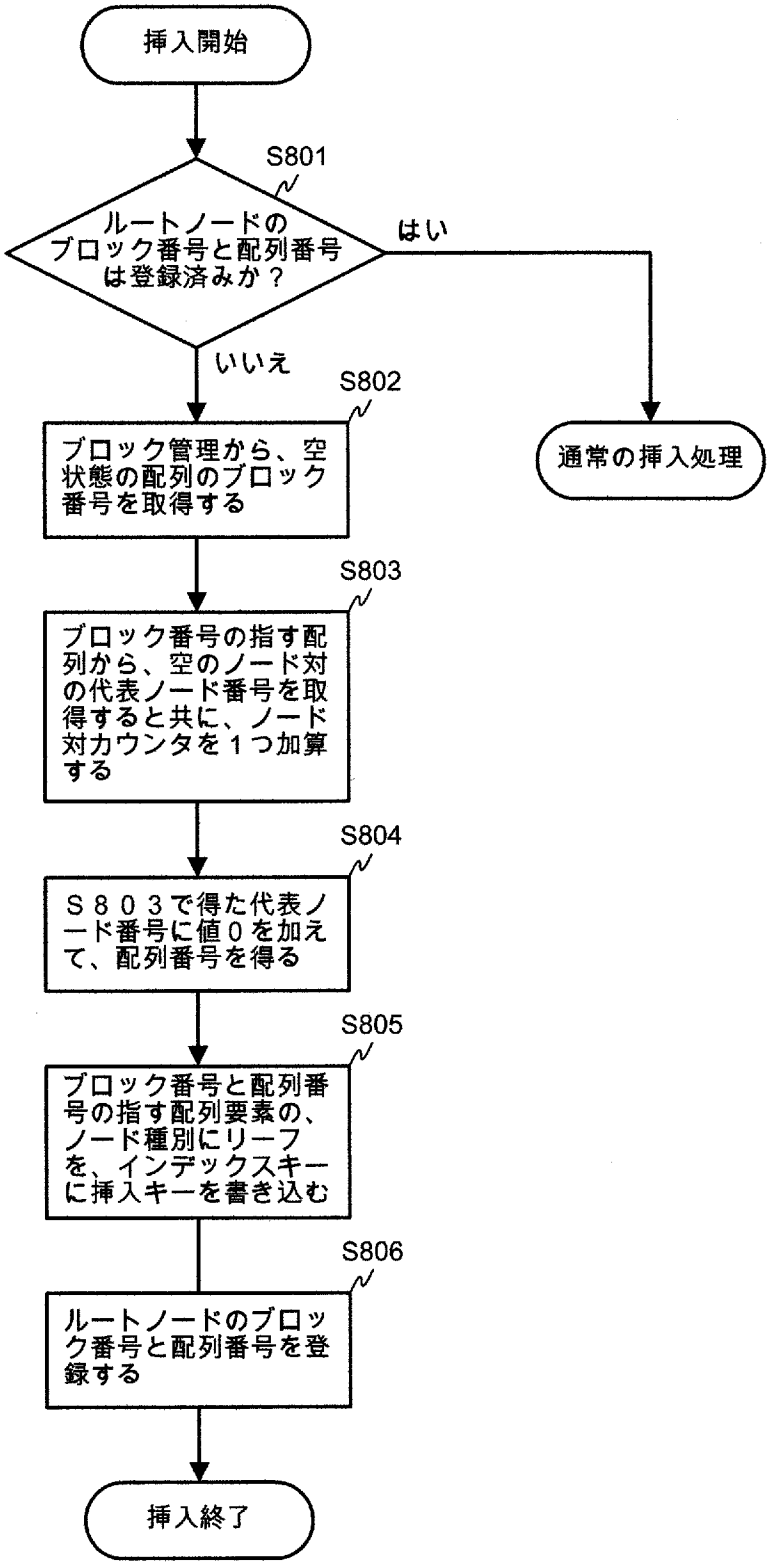
[図7D]


C
S741
探索経路スタックのスタックポインタは挿入位置ポインタか？
はい
D
いいえ
S742
探索経路スタックのスタックポインタの指すブロック番号と配列番号を取り出し、スタックポインタを１つ戻す
S743
取り出したブロック番号とノード対の移動先のブロック番号は一致するか？
はい
いいえ
S744
ノード対の移動元のブロック番号と代表ノード番号に、Ｓ７４２で取り出したブロック番号と配列番号から得た代表ノード番号を設定する
S745
ノード対の移動先のブロック番号の指す配列から、空のノード対の代表ノードの配列番号を取得すると共に、ノード対カウンタを１つ加算する
S746
ノード対の移動先の代表ノード番号に、Ｓ７４５で取得した空のノード対の代表ノードの配列番号を設定する
S747
ノード対の移動元のブロック番号と代表ノード番号の指す配列要素の内容を、ノード対として読み出す
S748
ノード対の移動先のブロック番号と代表ノード番号の指す配列要素に、Ｓ７４７で読み出したノード対の内容を書き込む
S749
探索経路スタックのスタックポインタの指すブロック番号と配列番号を取り出す
S750
Ｓ７４９で取得したブロック番号と配列番号の指す配列要素の、ブロック番号にノード対の移動先のブロック番号を、代表ノード番号にノード対の移動先の代表ノード番号を、書き込む
S751
ノード対の移動元のブロック番号と代表ノード番号の指すノード対を解放すると共に、ノード対カウンタを１つ減算する
S752
ノード対の移動先のブロック番号に、ノード対の移動元のブロック番号を設定する

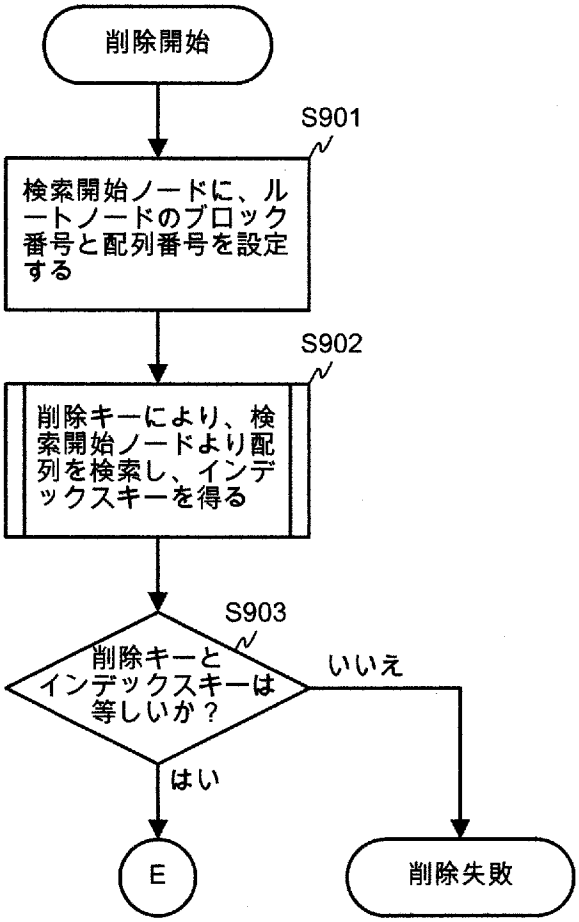
[図7E]


D
S761
空のノード対のブロック番号に、ノード対の移動先のブロック番号を設定する
S762
空ノード対のブロック番号の指す配列から、空のノード対の代表ノード番号を取得して、挿入するノード対の代表ノード番号に設定すると共に、ノード対カウンタを１つ加算する
S763
挿入キーとＳ７０３で得たインデックスキーを大小比較して、ブール値を得る（挿入キーが大きいときは値１（真）を、挿入キーが小さいときは値０（偽）を得る）
S764
Ｓ７６２で設定した代表ノード番号にＳ７６３で得た値を加えて、リーフノードの配列番号に設定する
S765
Ｓ７６２で得た代表ノード番号にＳ７６３で得た値をビット反転して加えて、対ノードの配列番号に設定する
S766
空ノード対のブロック番号とリーフノードの配列番号の指す配列要素の、ノード種別にリーフを、インデックスキーに挿入キーを書き込む
S767
挿入位置のブロック番号と配列番号の指す配列要素から、配列要素の内容を読み出す
S768
空ノード対のブロック番号と対ノードの配列番号の指す配列要素に、Ｓ７６７で読み出した内容を書き込む
S769
挿入位置のブロック番号と配列番号の指す配列要素の、ノード種別にブランチを、弁別ビット位置に差分ビット位置を、ブロック番号に挿入位置のブロック番号を、代表ノード番号に挿入するノード対の代表ノード番号を書き込む
挿入終了

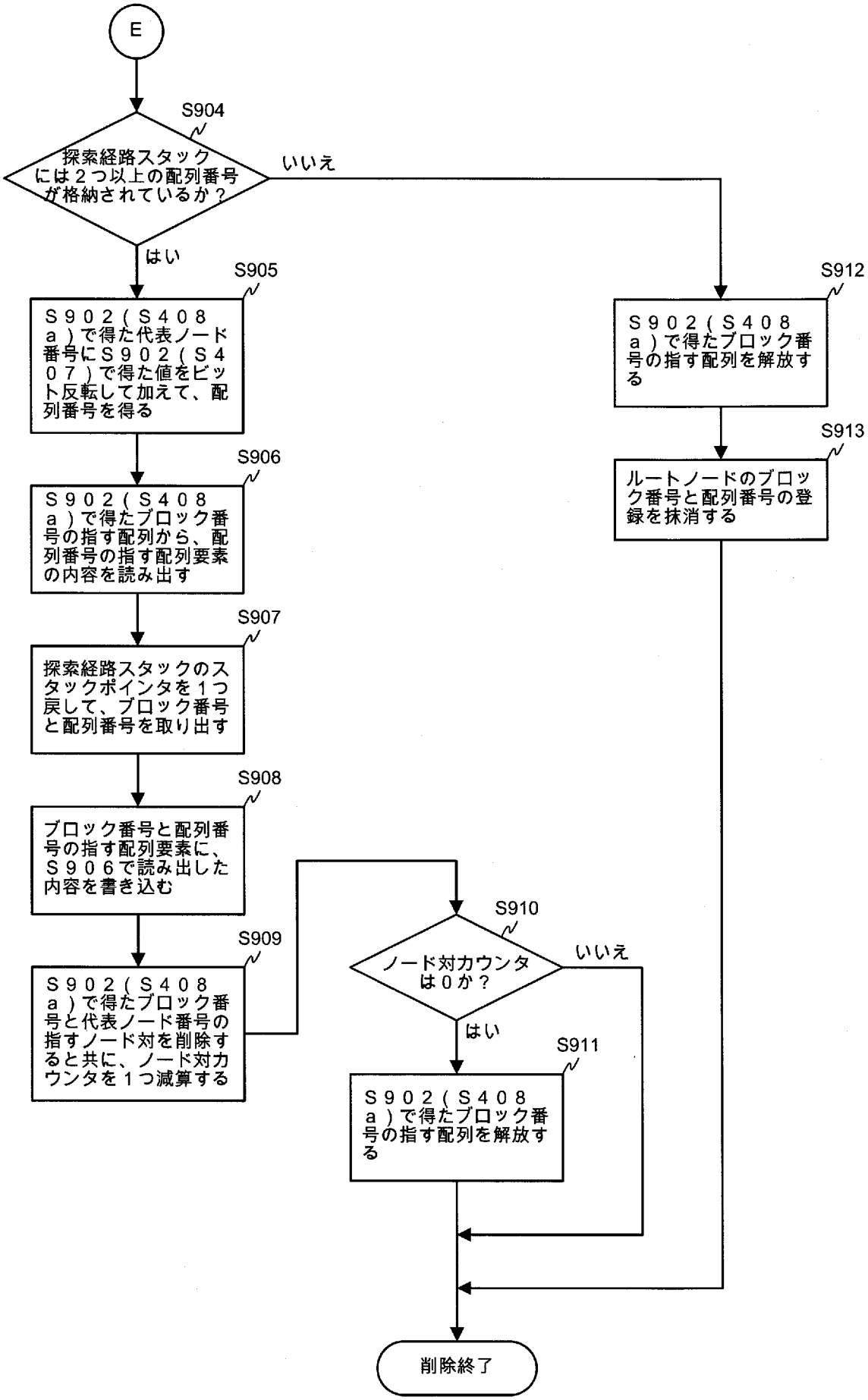
[図8]


挿入開始
S801
ルートノードの
ブロック番号と配列番号
は登録済みか？
はい
いいえ
通常の挿入処理
S802
ブロック管理から、空
状態の配列のブロック
番号を取得する
S803
ブロック番号の指す配
列から、空のノード対
の代表ノード番号を取
得すると共に、ノード
対カウンタを１つ加算
する
S804
Ｓ８０３で得た代表ノ
ード番号に値０を加え
て、配列番号を得る
S805
ブロック番号と配列番
号の指す配列要素の、
ノード種別にリーフ
を、インデックスキー
に挿入キーを書き込む
S806
ルートノードのブロッ
ク番号と配列番号を登
録する
挿入終了

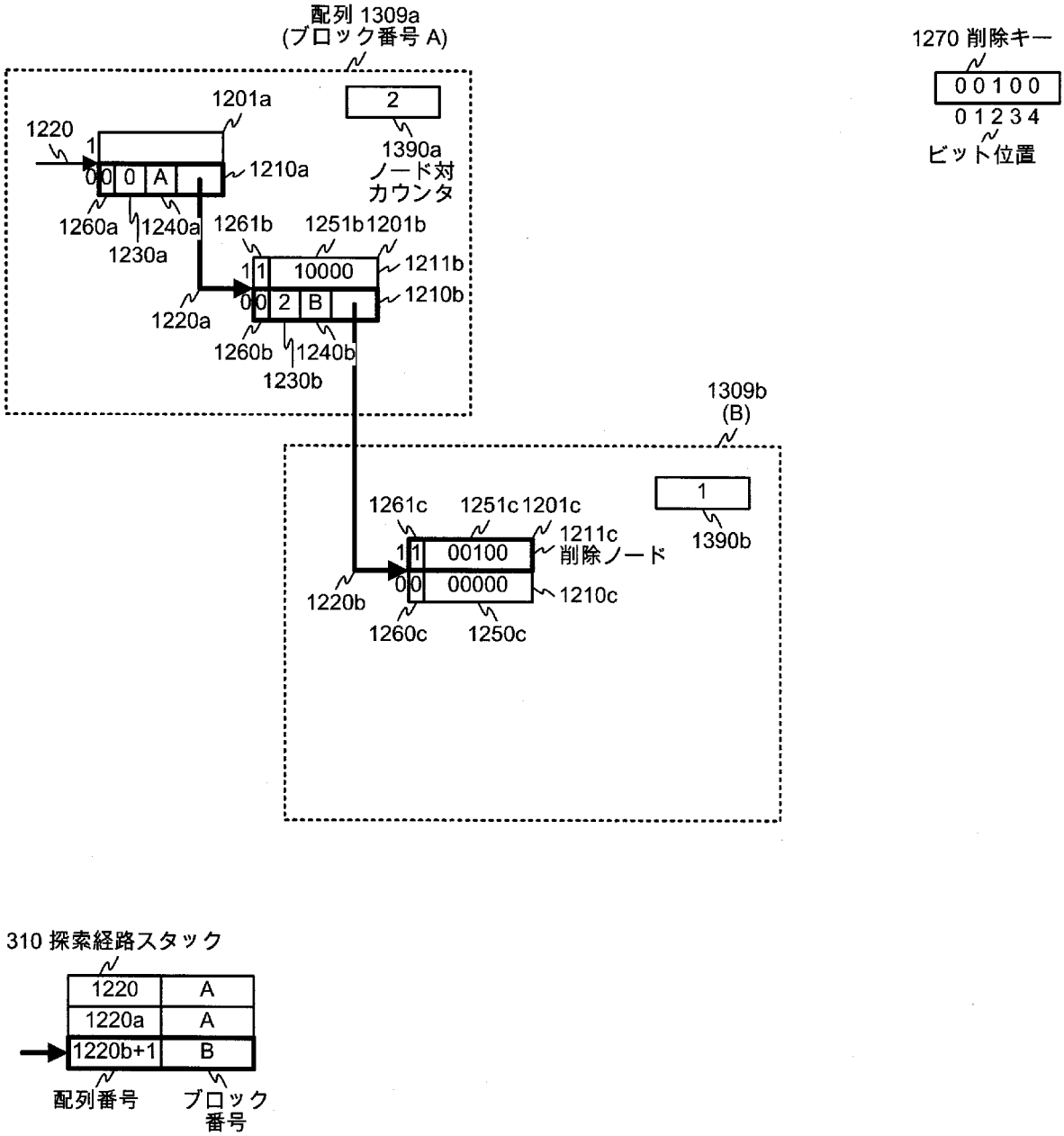
[図9]


削除開始
S901
検索開始ノードに、ルートノードのブロック番号と配列番号を設定する
S902
削除キーにより、検索開始ノードより配列を検索し、インデックスキーを得る
S903
削除キーとインデックスキーは等しいか？
いいえ
はい
E
削除失敗

[図10]

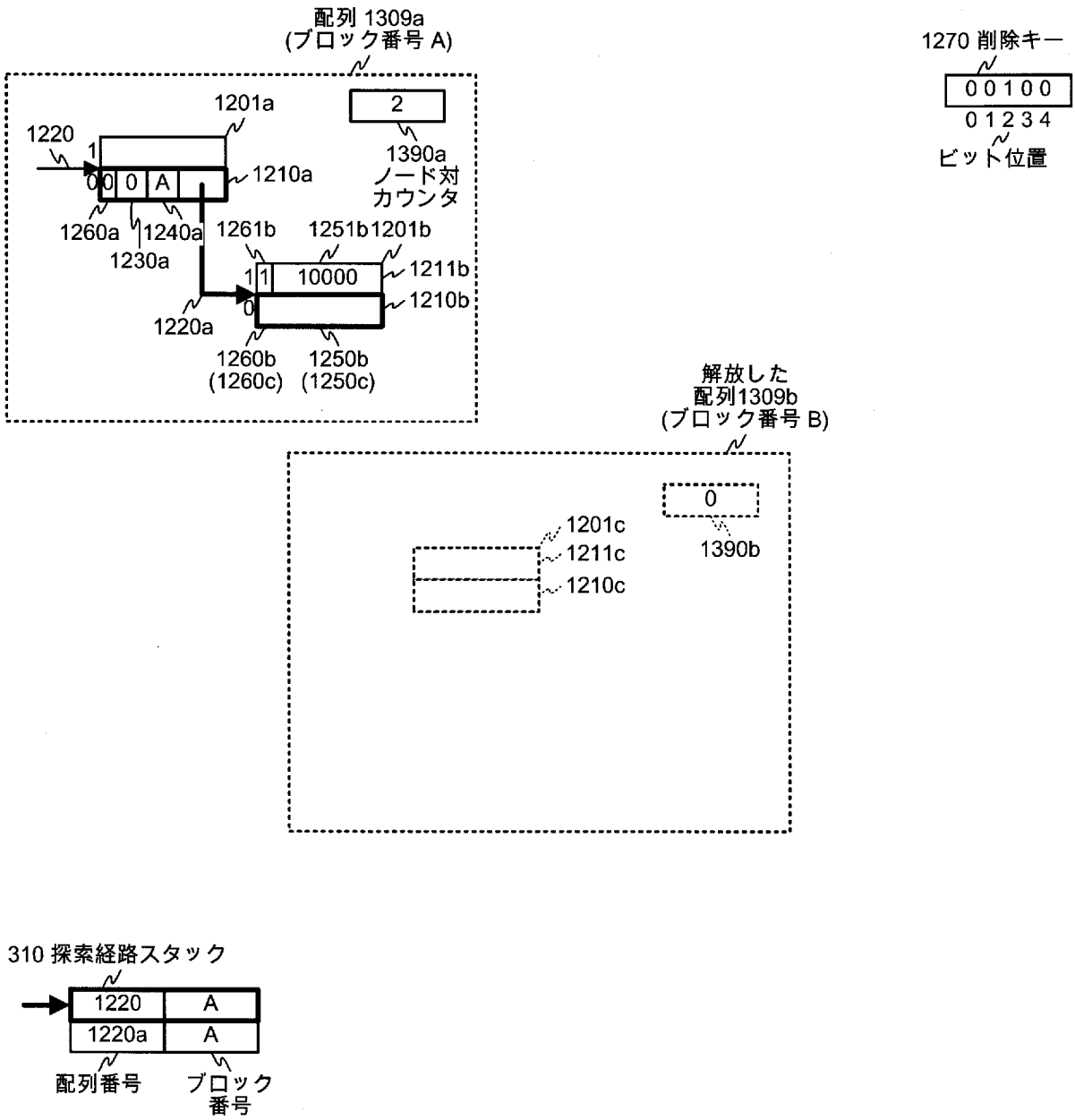

E
S904
探索経路スタックには２つ以上の配列番号が格納されているか？
いいえ
はい
S905
Ｓ９０２（Ｓ４０８ａ）で得た代表ノード番号にＳ９０２（Ｓ４０７）で得た値をビット反転して加えて、配列番号を得る
S906
Ｓ９０２（Ｓ４０８ａ）で得たブロック番号の指す配列から、配列番号の指す配列要素の内容を読み出す
S907
探索経路スタックのスタックポインタを１つ戻して、ブロック番号と配列番号を取り出す
S908
ブロック番号と配列番号の指す配列要素に、Ｓ９０６で読み出した内容を書き込む
S909
Ｓ９０２（Ｓ４０８ａ）で得たブロック番号と代表ノード番号の指すノード対を削除すると共に、ノード対カウンタを１つ減算する
S910
ノード対カウンタは０か？
いいえ
はい
S911
Ｓ９０２（Ｓ４０８ａ）で得たブロック番号の指す配列を解放する
S912
Ｓ９０２（Ｓ４０８ａ）で得たブロック番号の指す配列を解放する
S913
ルートノードのブロック番号と配列番号の登録を抹消する
削除終了

[図11A]

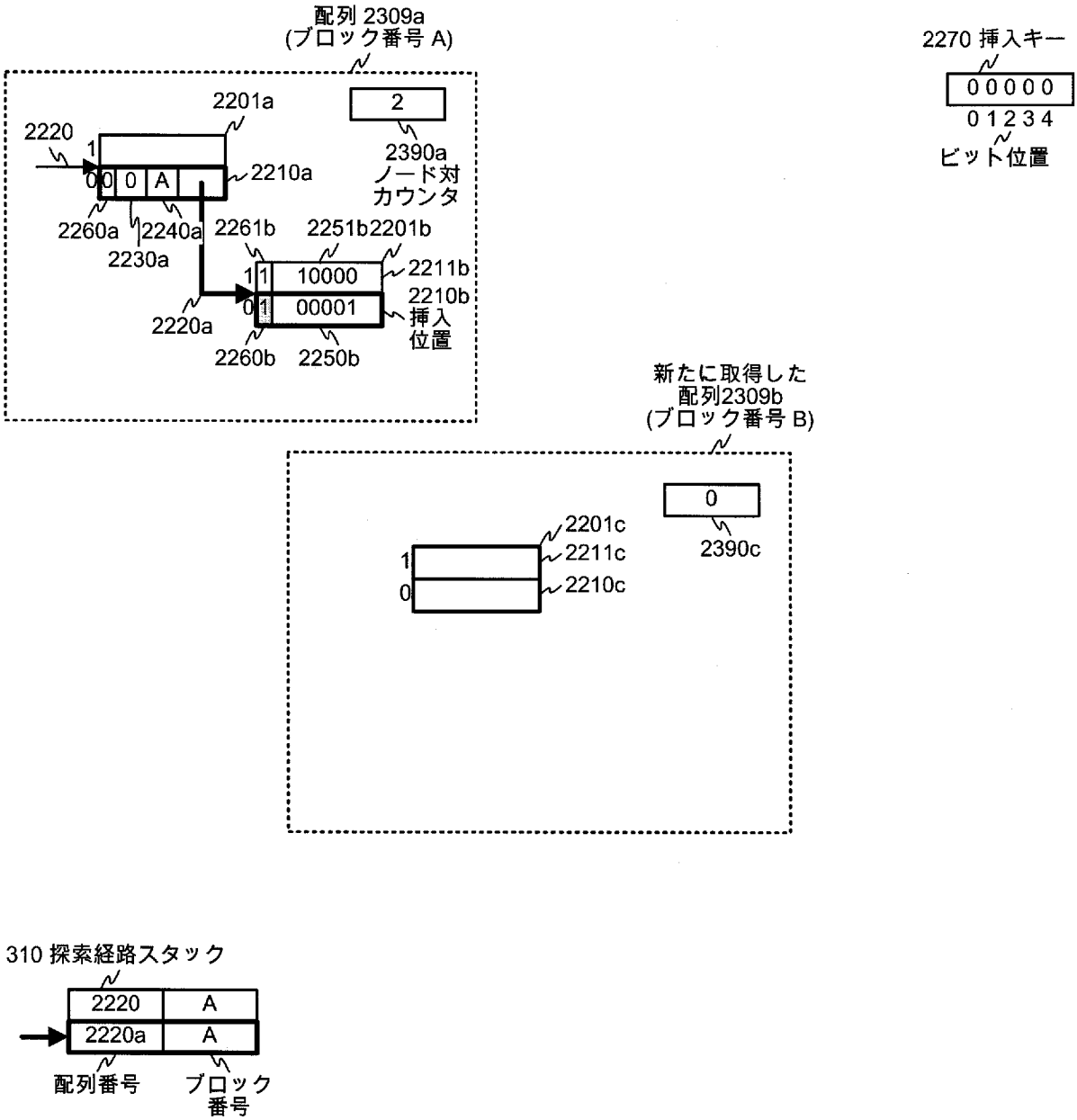

配列 1309a
(ブロック番号 A)
1201a
2
1220
1
1210a
0 0 0 A
1390a
ノード対
カウンタ
1260a
1240a
1230a
1261b
1251b
1201b
1 1 10000
1211b
0 0 2 B
1210b
1220a
1260b
1240b
1230b
1270 削除キー
0 0 1 0 0
0 1 2 3 4
ビット位置
1309b
(B)
1
1261c
1251c
1201c
1211c
1 1 00100
削除ノード
0 0 00000
1210c
1220b
1260c
1250c
1390b

310 探索経路スタック
1220 A
1220a A
1220b+1 B
配列番号
ブロック
番号

[図11B]

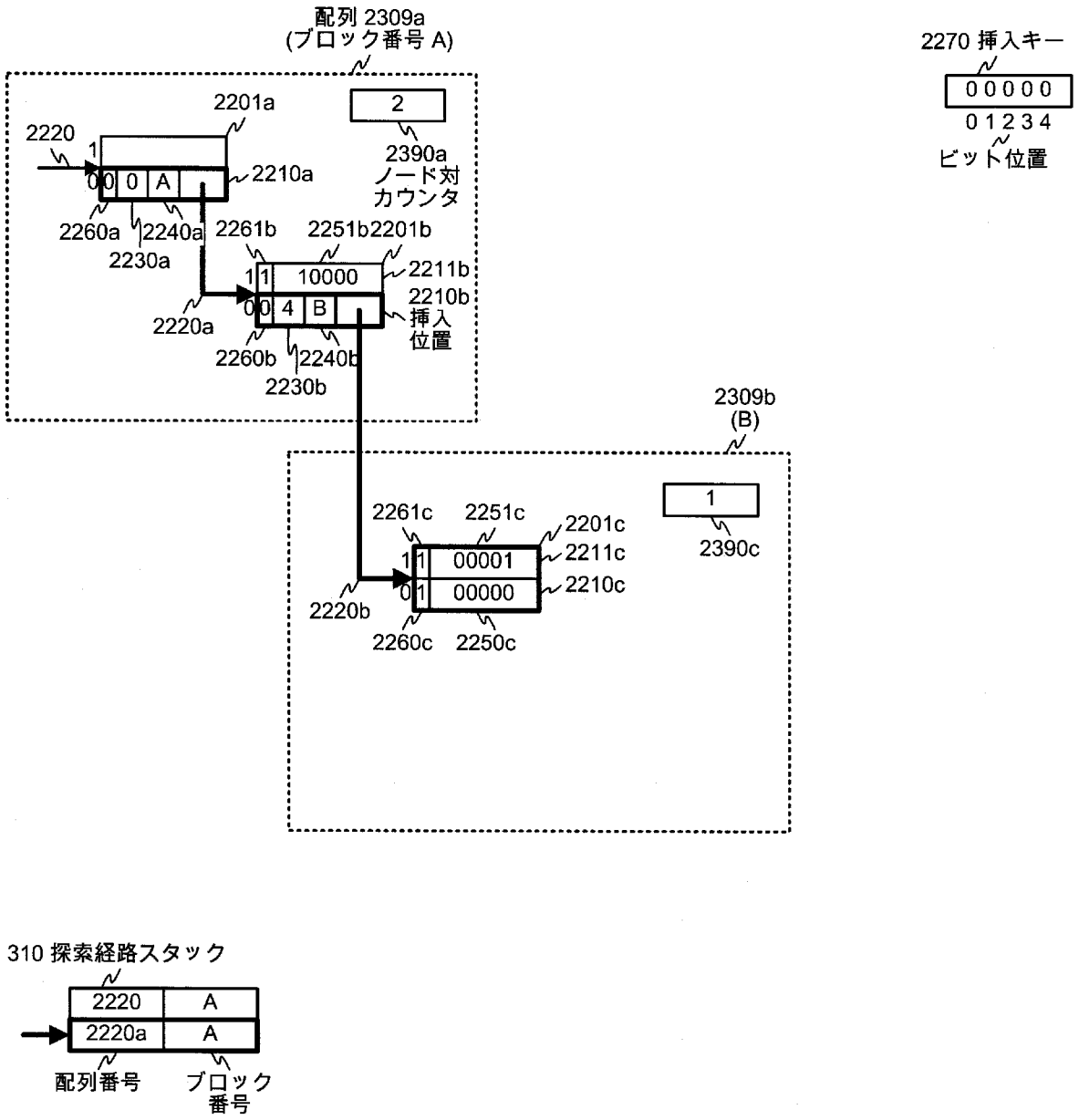

配列 1309a
(ブロック番号 A)
1201a
2
1220
1
1390a
ノード対
カウンタ
1210a
0 0 0 A
1260a
1240a
1230a
1261b
1251b
1201b
1 1 10000
1211b
1210b
0
1220a
1260b
(1260c)
1250b
(1250c)
1270 削除キー
0 0 1 0 0
0 1 2 3 4
ビット位置
解放した
配列1309b
(ブロック番号 B)
0
1390b
1201c
1211c
1210c
310 探索経路スタック
1220 A
1220a A
配列番号
ブロック
番号

[図12A]


配列 2309a
(ブロック番号 A)
2201a
2
2220
1
0 0 0 A
2210a
2390a
ノード対
カウンタ
2260a
2240a
2230a
2261b
2251b
2201b
1 1 10000
2211b
0 1 00001
2210b
挿入
位置
2220a
2260b
2250b
2270 挿入キー
0 0 0 0 0
0 1 2 3 4
ビット位置
新たに取得した
配列2309b
(ブロック番号 B)
0
2201c
2211c
2210c
2390c
1
0

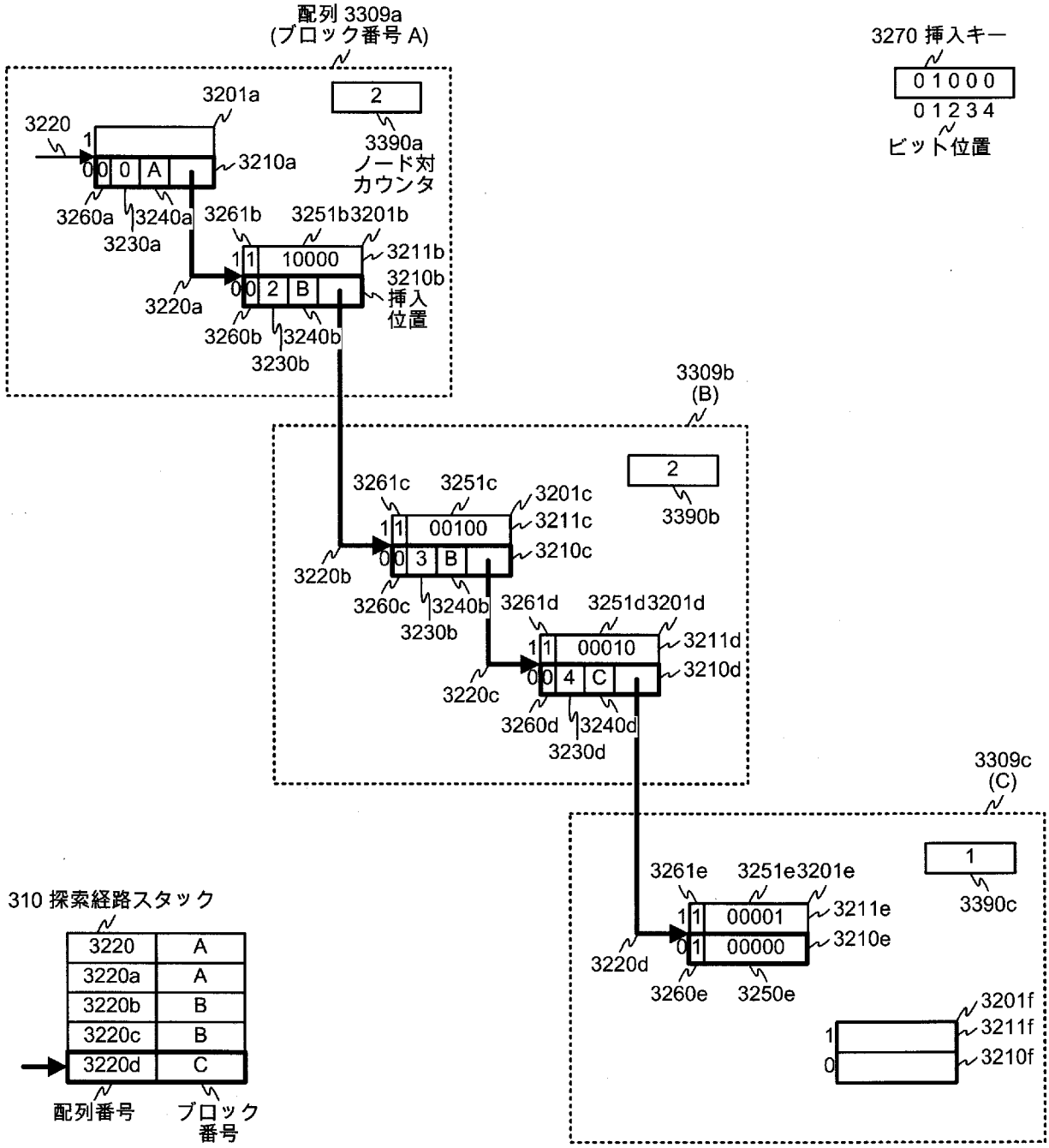
310 探索経路スタック
2220 A
2220a A
配列番号
ブロック
番号

[図12B]

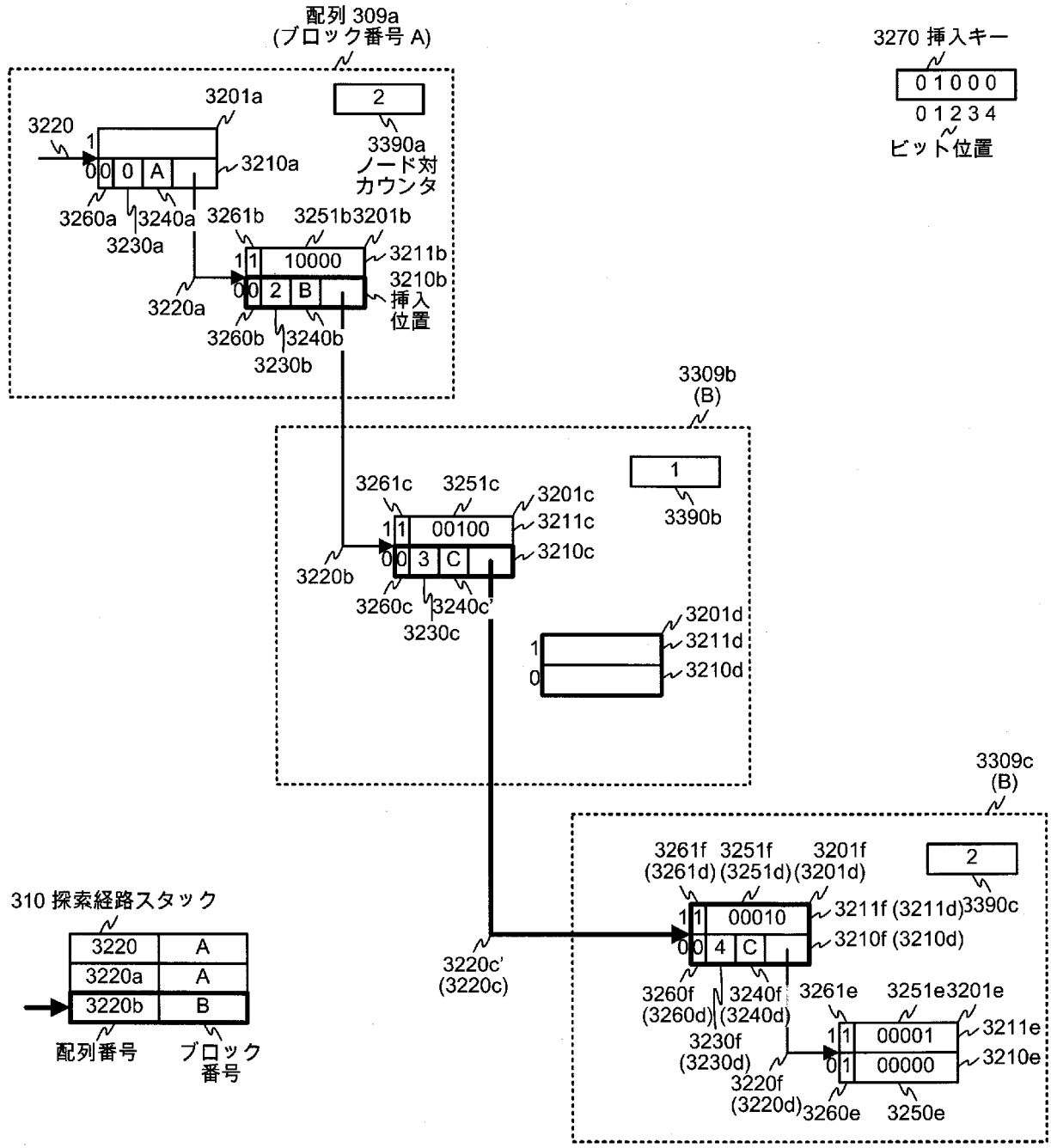

配列 2309a
(ブロック番号 A)
2201a
2
2220
1
00 0 A
2210a
2390a
ノード対
カウンタ
2260a
2240a
2230a
2261b
2251b
2201b
1 1 10000
2211b
00 4 B
2210b
挿入
位置
2220a
2260b
2240b
2230b
2270 挿入キー
00000
0 1 2 3 4
ビット位置
2309b
(B)
1
2390c
2261c
2251c
2201c
1 1 00001
2211c
0 1 00000
2210c
2220b
2260c
2250c

310 探索経路スタック
2220 A
2220a A
配列番号
ブロック
番号

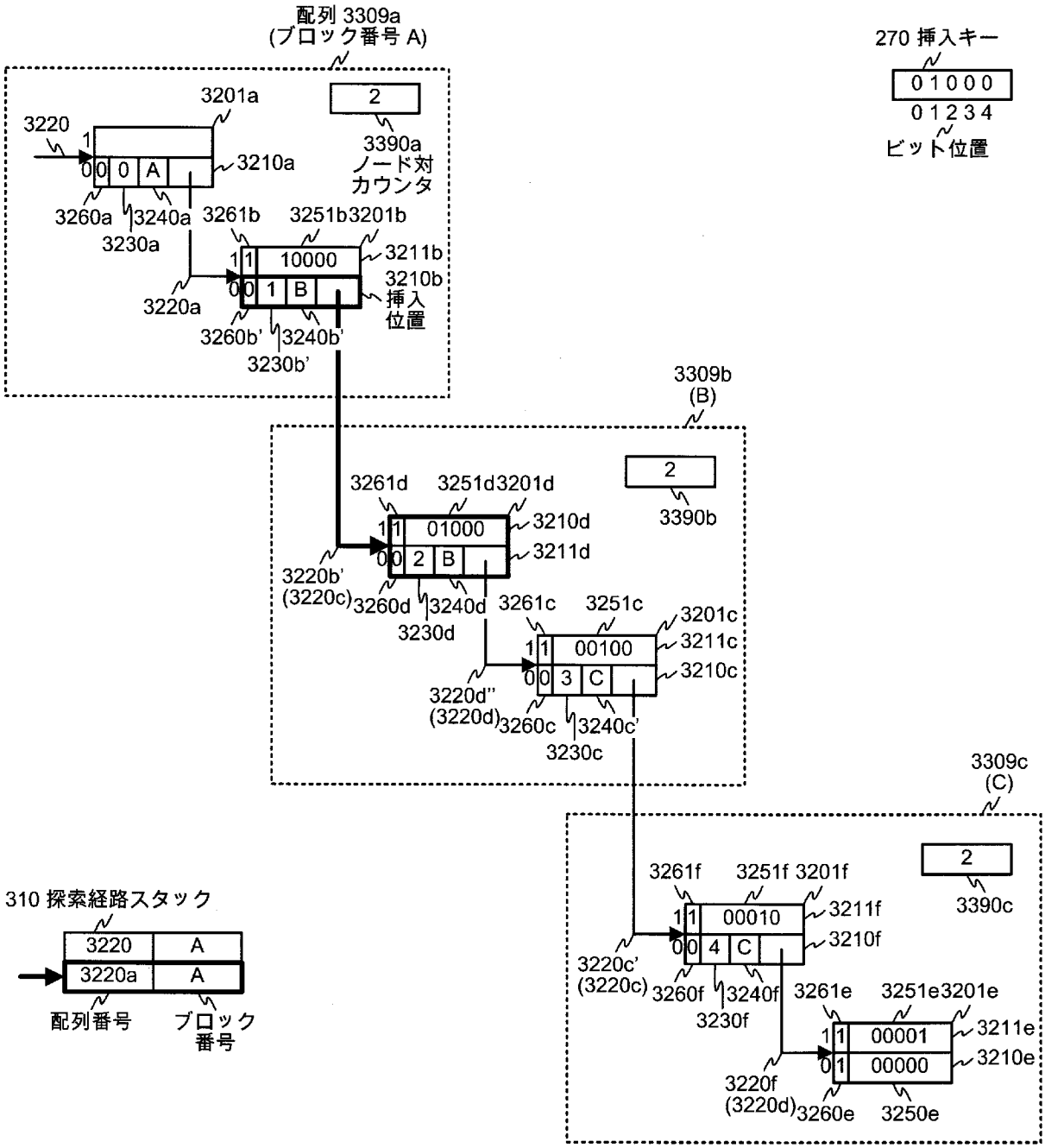
[図12C]


配列 3309a
(ブロック番号 A)
3201a
2
3220
3390a
ノード対
カウンタ
3210a
3260a
3240a
3230a
3261b
3251b
3201b
3211b
3210b
挿入
位置
3220a
3260b
3240b
3230b
3270 挿入キー
01000
01234
ビット位置
3309b
(B)
2
3390b
3261c
3251c
3201c
3211c
3210c
3220b
3260c
3240b
3230b
3261d
3251d
3201d
3211d
3210d
3220c
3260d
3240d
3230d
3309c
(C)
1
3390c
3261e
3251e
3201e
3211e
3210e
3220d
3260e
3250e
3201f
3211f
3210f
310 探索経路スタック
3220 A
3220a A
3220b B
3220c B
3220d C
配列番号
ブロック
番号

[図12D]


配列 309a
(ブロック番号 A)
3201a
2
3220
1
00 0 A
3210a
3390a
ノード対
カウンタ
3260a
3240a
3230a
3261b
3251b
3201b
1 1 10000
3211b
00 2 B
3210b
挿入
位置
3220a
3260b
3240b
3230b
3270 挿入キー
0 1 0 0 0
0 1 2 3 4
ビット位置
3309b
(B)
1
3390b
3261c
3251c
3201c
1 1 00100
3211c
00 3 C
3210c
3220b
3260c
3240c'
3230c
3201d
3211d
1
0
3210d
3309c
(B)
3261f
(3261d)
3251f
(3251d)
3201f
(3201d)
2
3390c
1 1 00010
3211f (3211d)
00 4 C
3210f (3210d)
3220c'
(3220c)
3260f
(3260d)
3240f
(3240d)
3230f
(3230d)
3220f
(3220d)
3261e
3251e
3201e
1 1 00001
3211e
0 1 00000
3210e
3260e
3250e
310 探索経路スタック
3220 A
3220a A
3220b B
配列番号
ブロック
番号

[図12E]


配列 3309a
(ブロック番号 A)
3201a
2
3220
3390a
ノード対
カウンタ
3210a
3260a
3240a
3230a
3261b
3251b
3201b
3211b
3210b
挿入
位置
3220a
3260b'
3240b'
3230b'
270 挿入キー
01000
0 1 2 3 4
ビット位置
3309b
(B)
3261d
3251d
3201d
3210d
3211d
3390b
3220b'
(3220c)
3260d
3240d
3230d
3261c
3251c
3201c
3211c
3210c
3220d"
(3220d)
3260c
3240c'
3230c
3309c
(C)
3261f
3251f
3201f
3211f
3210f
3390c
3220c'
(3220c)
3260f
3240f
3230f
3261e
3251e
3201e
3211e
3210e
3220f
(3220d)
3260e
3250e
310 探索経路スタック
3220
A
3220a
A
配列番号
ブロック
番号

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2008/003266

A. CLASSIFICATION OF SUBJECT MATTER
G06F17/30(2006.01)i

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)
G06F17/30

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Jitsuyo Shinan Koho 1922-1996 Jitsuyo Shinan Toroku Koho 1996-2009
Kokai Jitsuyo Shinan Koho 1971-2009 Toroku Jitsuyo Shinan Koho 1994-2009

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.
Y	WO 2008/004335 A1 (S. Grants Co., Ltd.), 10 January, 2008 (10.01.08), Full text; all drawings & JP 2008-15872 A	1-13
Y	JP 5-265821 A (Hitachi, Ltd.), 15 October, 1993 (15.10.93), Par. No. [0017] (Family: none)	1-13
A	JP 7-210569 A (Oki Electric Industry Co., Ltd.), 11 August, 1995 (11.08.95), Par. No. [0019] (Family: none)	1-13

☐ 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
28 January, 2009 (28.01.09)	10 February, 2009 (10.02.09)
Name and mailing address of the ISA/ Japanese Patent Office	Authorized officer
Facsimile No.	Telephone No.

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２００８／００３２６６

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　G06F17/30(2006.01)i

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））

Int.Cl.　G06F17/30

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	１９７１－２００９年
日本国実用新案登録公報	１９９６－２００９年
日本国登録実用新案公報	１９９４－２００９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	関連する請求の範囲の番号
Y	WO 2008/004335 A1（株式会社エスグランツ）2008.01.10, 全文・全図 & JP 2008-15872 A	1-13
Y	JP 5-265821 A（株式会社日立製作所）1993.10.15, 【００１７】段落（ファミリーなし）	1-13
A	JP 7-210569 A（沖電気工業株式会社）1995.08.11, 【００１９】段落（ファミリーなし）	1-13

☐　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矛盾するものではなく、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 ２８．０１．２００９	国際調査報告の発送日 １０．０２．２００９
国際調査機関の名称及びあて先 日本国特許庁（ＩＳＡ／ＪＰ） 郵便番号１００－８９１５ 東京都千代田区霞が関三丁目４番３号	特許庁審査官（権限のある職員）　５Ｍ　９４７５ 池田　聡史 電話番号　０３－３５８１－１１０１　内線　３５９９