(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7631308号
(P7631308)

(45)発行日　令和7年2月18日(2025.2.18)

(24)登録日　令和7年2月7日(2025.2.7)

(51)国際特許分類
Ｈ０３Ｍ　7/30　(2006.01)

ＦＩ
Ｈ０３Ｍ　7/30　Ｚ

請求項の数　16　（全19頁）

(21)出願番号　特願2022-508996(P2022-508996)
(86)(22)出願日　令和2年8月12日(2020.8.12)
(65)公表番号　特表2022-545644(P2022-545644A)
(43)公表日　令和4年10月28日(2022.10.28)
(86)国際出願番号　PCT/US2020/045903
(87)国際公開番号　WO2021/034565
(87)国際公開日　令和3年2月25日(2021.2.25)
審査請求日　令和5年8月8日(2023.8.8)
(31)優先権主張番号　16/542,872
(32)優先日　令和1年8月16日(2019.8.16)
(33)優先権主張国・地域又は機関
米国(US)

(73)特許権者　591016172
アドバンスト・マイクロ・ディバイシズ・インコーポレイテッド
ＡＤＶＡＮＣＥＤ ＭＩＣＲＯ ＤＥＶＩＣＥＳ ＩＮＣＯＲＰＯＲＡＴＥＤ
アメリカ合衆国 ９５０５４ カリフォルニア州、 サンタ クララ、オーガスティン ドライブ ２４８５
(74)代理人　100108833
弁理士　早川 裕司
(74)代理人　100111615
弁理士　佐野 良太
(74)代理人　100162156
弁理士　村雨 圭介
(72)発明者　アレクサンダー ディー． ブレスロウ
最終頁に続く

(54)【発明の名称】　エンコーディング及びデコーディングテーブルを用いたセミソーティング圧縮

(57)【特許請求の範囲】
【請求項１】
データ処理プラットフォームであって、
メモリと、
前記メモリに結合されたプロセッサと、を備え、
前記プロセッサは、
データ項目のセットを受け取ることと、
前記データ項目の個々のデータ項目毎に、サフィックスデータ及びプレフィックスを前記個々のデータ項目のデータ内容に基づいて選択することと、
前記プレフィックスに基づいて前記データ項目のセットをソートすることと、
複数のエンコーディングテーブルに問い合わせ、前記複数のエンコーディングテーブルから得られた複数の整数を合計することによって前記プレフィックスをエンコードして、前記データ項目のセットに対する全てのプレフィックスの値を表す圧縮情報を含むコードワードσを生成することと、
前記データ項目毎の前記サフィックスデータと前記コードワードσとを前記メモリに記憶することと、
前記コードワードσを復元して前記プレフィックスを再生することと、
前記再生されたプレフィックスを各々のサフィックスデータと組にすることと、
を行うように動作可能である、
データ処理プラットフォーム。

【請求項２】
前記複数のエンコーディングテーブルの各々は、いくつかのプレフィックス位置を含み、
前記複数のエンコーディングテーブルに問い合わせることによって、前記プレフィックスの各々のプレフィックス位置を表す前記複数の整数が得られる、
請求項１のデータ処理プラットフォーム。
【請求項３】
前記エンコーディングテーブルは、プレフィックス毎に、前記プレフィックス位置を表す個々のプレフィックスインデックスが、順序付け特性によって順序付けられたプレフィックス値の複数の順序付き集合に基づく数値を提供するように設けられており、
前記数値は、前記順序付き集合内の前記プレフィックスの相対位置に基づいている、
請求項２のデータ処理プラットフォーム。
【請求項４】
個々のプレフィックスインデックス毎の前記数値は、プレフィックス値の順序付き集合の数を示し、各々の順序付き集合のサイズは、前記順序付き集合内の前記プレフィックス値の相対位置に適用される関数の出力であり、前記順序付き集合内の全てのプレフィックス値は、前記順序付け特性によって計算された前記個々のプレフィックスの値よりも小さい、
請求項３のデータ処理プラットフォーム。
【請求項５】
前記プロセッサは、前記コードワードσを復元する要求に応じて、前記複数のエンコーディングテーブルを少なくとも部分的に計算するように動作可能である、
請求項１のデータ処理プラットフォーム。
【請求項６】
前記複数のエンコーディングテーブルは、数ｋのエンコーディングテーブルl_0～l_{k-1}を含み、前記エンコーディングテーブルは、ｋ個のプレフィックスの各々のプレフィックスを部分的にエンコードする、
請求項１のデータ処理プラットフォーム。
【請求項７】
前記プロセッサは、
前記コードワードσを生成する場合に、前記複数のエンコーディングテーブルにアクセスして、以下の式の個々の部分値を取得するように動作可能であり、
【数１】

$$\sigma = \sum_{i=0}^{k-1} \binom{p_i + i}{i+1}$$

σは、コードワードであり、p_iは、前記プレフィックスに基づいて前記データ項目のセットをソートすることによって得られた順序におけるｉ番目のプレフィックスの値である、
請求項６のデータ処理プラットフォーム。
【請求項８】
複数のエンコーディングテーブルは、前記プレフィックス値と、前記プレフィックス値の順序で配列されていないプレフィックスインデックスと、を含む、
請求項７のデータ処理プラットフォーム。
【請求項９】
前記コードワードσを復元することは、複数の反復を行うことを含み、
前記複数の反復は、
第１の反復において、ｉをｋ－１に初期化し、検索値σ’を前記コードワードσに等しい値に初期化することと、

エンコーディングテーブルｌ$_{i}$を検索して、前記検索値σ’以下の最大値を検索することと、
前記プレフィックスｐ$_{i}$を、前記検索で見つかった前記最大値のテーブルインデックスに等しい値に設定することと、
後続の反復において、ｉをデクリメントして、前記検索値σ’を、以前の検索値σ’から以前の検索反復で見つかった最大値を引いたものに等しい値に設定することと、
エンコーディングテーブルｌ$_{i}$を検索して、前記検索値σ’以下の最大値を検索することと、
前記プレフィックスｐ$_{i}$を、前記検索で見つかった前記最大値のテーブルインデックスに等しい値に設定することと、を含む、
請求項８のデータ処理プラットフォーム。
【請求項１０】
前記プロセッサは、前記プレフィックスのみを用いて前記データ項目をソートする、
請求項１のデータ処理プラットフォーム。
【請求項１１】
データを圧縮して記憶する方法であって、
データ項目のセットを受け取ることと、
前記データ項目のセット内の個々のデータ項目毎に、サフィックスデータ及びプレフィックスを前記個々のデータ項目のデータ内容に基づいて選択することと、
前記プレフィックスに基づいて前記データ項目のセットをソートすることと、
複数のエンコーディングテーブルに問い合わせ、前記複数のエンコーディングテーブルから得られた複数の整数を合計することによって前記プレフィックスをエンコードして、前記データ項目のセットに対する全てのプレフィックスの値を表す圧縮情報を含むコードワードσを生成することと、
前記データ項目毎の前記サフィックスデータと前記コードワードσとを記憶することと、
前記コードワードを復元して前記プレフィックスを再生することと、
前記再生されたプレフィックスを各々のサフィックスデータと組にすることと、を含む、
方法。
【請求項１２】
前記複数のエンコーディングテーブルの各々は、いくつかのプレフィックス位置を含み、
前記複数のエンコーディングテーブルに問い合わせることによって、前記プレフィックスの各々のプレフィックス位置を表す前記複数の整数が得られる、
請求項１１の方法。
【請求項１３】
前記エンコーディングテーブルは、プレフィックス毎に、前記プレフィックス位置を表す個々のプレフィックスインデックスが、順序付け特性によって順序付けられたプレフィックス値の複数の順序付き集合に基づく数値を提供するように設けられており、
前記数値は、前記順序付き集合内の前記プレフィックスの相対位置に基づいている、
請求項１２の方法。
【請求項１４】
個々のプレフィックスインデックス毎の前記数値は、プレフィックス値の順序付き集合の数を示し、各々の順序付き集合のサイズは、前記順序付き集合内の前記プレフィックス値の相対位置に適用される関数の出力であり、前記順序付き集合内の全てのプレフィックス値は、前記順序付け特性によって計算された前記個々のプレフィックスの値よりも小さい、
請求項１３の方法。
【請求項１５】
前記複数のエンコーディングテーブルは、数ｋのエンコーディングテーブルｌ$_{0}$～ｌ$_{k-1}$を含み、前記エンコーディングテーブルは、ｋ個のプレフィックスの各々のプレフィックスを部分的にエンコードする、

請求項１１の方法。
【請求項１６】
前記コードワードσを生成する場合に、前記複数のエンコーディングテーブルにアクセスして、以下の式の個々の部分値を取得することをさらに含み、
【数２】

$$\sigma=\sum_{i=0}^{k-1}\binom{p_i+i}{i+1}$$

σは、コードワードであり、p_iは、前記プレフィックスに基づいて前記データ項目のセットをソートすることによって得られた順序におけるｉ番目のプレフィックスの値である、
請求項１５の方法。
【発明の詳細な説明】
【背景技術】
【０００１】
セミソーティング（semi-sorting）は、カッコウフィルタ(cuckoo filters)及びｄ－ｌｅｆｔカウンティングブルームフィルタ(d-left counting Bloom filters)においてフィンガープリントを圧縮する際に使用されることについて説明されている。これらは、他のアプリケーション（例えば、データベースシステム、ゲノムシーケンシング及びファイルシステム）の中でも、ネットワーキングハードウェア及びソフトウェアにおいて普及している２つの重要な近似セットメンバーシップデータ構造（ＡＳＭＤＳ）である。フィンガープリントは、ＡＳＭＤＳが近似的に表すセット内の項目（item）の存在をエンコードする短いハッシュである。典型的に、フィンガープリントは、ＡＳＭＤＳ内のバケット（キャッシュセットに似ている）に記憶される。バケット内のフィンガープリントの位置は、その意味を変えない。
【０００２】
従来の研究では、これらのフィンガープリントをそれらのプレフィックスによって順序付けて、プレフィックスをコードワードと取り替えている。フィンガープリントをエンコード及びデコードするために、従来の研究では、一組のエンコーディング及びデコーディングテーブルを用いている。しかし実際には、これらのテーブルのサイズの、連想度に対するこれらのテーブルのサイズの増加率は、連想度が４の場合にルックアップテーブルのサイズが数キロバイトになるため、バケットの連想度を４以下に制限している。したがって、これらのテーブルのサイズを小さくして、ハードウェアキャッシュや他のメモリへのそれらの記憶を改良するソリューションが望まれている。
【図面の簡単な説明】
【０００３】
【図１】いくつかの実施形態による、圧縮を含むデータ処理プラットフォームのブロック図である。
【図２】いくつかの実施形態による、データを圧縮及び復元するためのプロセスのフロー図である。
【図３】いくつかの実施形態による、圧縮データを含むコードワードを形成するためのより詳細なプロセスのフロー図である。
【図４】いくつかの実施形態による、エンコーディングテーブルを用いて圧縮データを含むコードワードを形成するためのプロセスを示すテーブルである。
【図５】いくつかの実施形態による、セミソーティングプロセスを示す図である。
【図６】いくつかの実施形態による、エンコーディングテーブルを用いてコードワードを形成する例を示す図である。
【図７】いくつかの実施形態による、コードワードを復元するためのプロセスのフロー図

である。
【図８】いくつかの実施形態による、エンコーディングテーブルを用いてコードワードを復元するプロセスの一部を示す図である。
【図９】図８のプロセスのさらなる部分を示す図である。
【図１０】いくつかの典型的な実施形態による、取得された圧縮性能の例を示すチャートである。
【発明を実施するための形態】
【０００４】
以下の説明において、異なる図面での同じ符号の使用は、類似又は同一のアイテムを示す。特に断らない限り、「結合された」という用語及びその対応付けられる動詞の形態には、当該技術分野で周知の手段による直接接続及び間接的な電気接続の両方が含まれ、特に断らない限り、直接接続の任意の説明には、間接的な電気接続の適切な形態を用いた代替的な実施形態の意味も含む。
【０００５】
データ処理プラットフォームは、メモリと、メモリに結合されたプロセッサと、を含み、プロセッサは、データ項目のセットの圧縮及び復元を行うことができる。プロセッサは、データ項目のセットを受け取り、データ項目のセット内の各々の個々のデータ項目に対してサフィックスデータ及びプレフィックスを、個々のデータ項目のデータ内容に基づいて選択する。データ項目のセットをプレフィックスに基づいてソートする。プレフィックスを、複数のエンコーディングテーブルに問い合わせる（querying）ことによってエンコードして、データ項目のセットに対する全てのプレフィックスの値を表す圧縮情報を含むコードワードを形成する。データ項目の各々及びコードワードに対するサフィックスデータを、メモリに記憶する。コードワードを復元してプレフィックスを再生し、再生したプレフィックスをその個々のサフィックスデータと組にする。
【０００６】
方法は、データ項目を圧縮及び記憶する。データ項目のセットを受け取り、データ項目のセット内の各々の個々のデータ項目に対してサフィックスデータ及びプレフィックスを、個々のデータ項目のデータ内容に基づいて選択する。データ項目のセットをプレフィックスに基づいてソートする。プレフィックスを、複数のエンコーディングテーブルに問い合わせることによってエンコードして、データ項目のセットに対する全てのプレフィックスの値を表す圧縮情報を含むコードワードを形成する。データ項目の各々に対するコードワード及びサフィックスデータを記憶する。コードワードを復元してプレフィックスを再生する。再生したプレフィックスをその個々のサフィックスデータと組にする。
【０００７】
有形の非一時的なコンピュータ可読媒体が、少なくとも１つのプロセッサによってデータ項目を圧縮及び復元することが実行可能なプログラム製品を保持する。データ項目のセットを受け取り、データ項目のセット内の各々の個々のデータ項目に対してサフィックスデータ及びプレフィックスを、個々のデータ項目のデータ内容に基づいて選択する。データ項目のセットをプレフィックスに基づいてソートする。プレフィックスを、複数のエンコーディングテーブルに問い合わせることによってエンコードして、データ項目のセットに対する全てのプレフィックスの値を表す圧縮情報を含むコードワードを形成する。データ項目の各々に対するコードワード及びサフィックスデータを、メモリに記憶する。コードワードを復元してプレフィックスを再生する。再生したプレフィックスをその個々のサフィックスデータと組にする。
【０００８】
図１に、いくつかの実施形態による、圧縮を含むデータ処理プラットフォームのブロック図を示す。データ処理プラットフォーム１００は、任意の並べ直し得る（may be reordered）高いエントロピを伴うデータ項目のセットを圧縮する必要があるプラットフォームであり得る。例えば、本明細書における技術には、近似セットメンバーシップデータ構造（approximate set membership data structures）におけるソフトウェア及びハー

ドウェアキャッシュタグ及びフィンガープリントの圧縮のような多くの用途がある。全般的に、データ処理プラットフォーム１００は、固定長の項目のセットに対する圧縮及び復元を行う。本技術は、その順番を置換することができる一様なランダム項目（すなわち、情報エントロピが最大）のときに、最も有用である。用途は、ネットワーキング、ソフトウェアシステム及びハードウェアキャッシュから変化し得る。

【０００９】

データ処理プラットフォーム１００は、ネットワーク又はホストシステム１０と通信して、データ記憶及び取り出しに対する要求を満たすか、ホストシステムに対する暗号化鍵取り出し又はキャッシング等の動作を実行する。データ処理プラットフォーム１００には、この例では、ランダムアクセスメモリ（ＲＡＭ）１１２と通信するプロセッサ１１０が含まれる。ランダムアクセスメモリ（ＲＡＭ）１１２は、プロセッサ１１０の内部又は外部にあってもよい。ＲＡＭ１１２は、例えば、プロセッサ１１０の統合キャッシュの一部であってもよいし、プラットフォーム１００のメインメモリ内にあってもよい。また、プロセッサ１１０は、不揮発性メモリ１２０と通信する。不揮発性メモリ１２０は、圧縮及び復元プログラムコード１２２等のコンピュータプログラムコードを保持する有形の非一時的なコンピュータ可読媒体（例えば、フラッシュメモリ又はハードドライブ）である。

【００１０】

ＲＡＭ１１２は、以下で説明する圧縮技術で用いられるデータを保持する。例えば、コードワードσ及びサフィックス０～サフィックスｋ－１（数ｋのサフィックスが存在し、それぞれｋ個のデータ項目のうち何れかと対応付けられる）である。このようなデータも不揮発性メモリに記憶されてよい。プログラムコード１２２が不揮発性メモリ１２０からロードされると、ＲＡＭ１１２内に圧縮及び復元プログラムコードが存在する。これを、ロードされた圧縮エンジン１１４及びロードされた復元エンジン１１６によって示す。本明細書で説明する技術と共に用いる他のデータ（例えば、圧縮されるデータ項目及びエンコーディングテーブル）も、ＲＡＭ１１２又は不揮発性メモリ１２０に保持されてよい。

【００１１】

この実施形態では、データ処理プラットフォームには、プログラムコード命令を実行するプロセッサ１１０が含まれているが、他の実施形態では、圧縮機能の一部又は全部を、ネットワーキングＡＳＩＣ、ＣＰＵ、ＧＰＵ又はＦＰＧＡ内のプログラマブルロジックにおけるデジタルロジック等のハードウェアで実施してもよい。

【００１２】

図２に、いくつかの実施形態による、データ圧縮及び復元するためのプロセス２００のフロー図を示す。データ項目のセットを圧縮用に用意する（ブロック２０２）。データ項目は、典型的には、固定長の項目であり、前述したようにエントロピが高い場合が多い。データ項目内のビットのいくつかのサブシーケンスはエントロピが高い場合があり、一方で、他のサブシーケンスの方が、ランダム性が低い場合がある。例えば、例を挙げると、ＡＳＭＤＳからの種々のタイプのキャッシュライン、キャッシュタグ、フィンガープリント、ネットワークルーティングデータ、分岐予測器若しくはハードウェアプリフェッチャからのハードウェアテーブル、又は、データベースエントリである。

【００１３】

ブロック２０４において、プロセス２００は、セット内の各データ項目に対してプレフィックス及びサフィックスデータを、項目のデータ内容に基づいて選択する。いくつかの実施形態では、データ項目は、識別されたプレフィックス及びサフィックスを既に有していることがあり、その場合、ブロック２０４は必要ない。例えば、データ項目のセットは、キャッシュタグが既に与えられているキャッシュラインのセットであってもよく、一部又は完全タグがプレフィックスとして用いられる。プレフィックスを選択するとき、プロセス２００は、好ましくは、各項目からビットの確定論的なサブシーケンス（例えば、項目から最初のビットの数）を選択する。サブシーケンスは、データ項目の任意の所望の一部から選択してもよく、データ項目からの不連続ビット（例えば、データ項目からの不連続ビットサブシーケンスから計算されたフィンガープリント又はハッシュ）を含んでいて

もよい。代替的に、データ項目の所望の一部に対するハッシュ又は計算を特定の実施形態で用いて、プレフィックスを選択するか生成する。サフィックスデータは、典型的に、プレフィックスを除いたデータ項目内に残ったビットである。

【００１４】

次に、ブロック２０６において、データ項目をプレフィックスに基づいてソートする。ソートは、データ項目の全セットに対して行ってもよいし、セットをサブセットに分割してサブセットをソートしてもよい。いくつかの実施形態では、データ項目をプレフィックスの順序付けに従う順番で記憶し、これらの実施形態の場合、ブロック２０６を実行する必要はない。次に、ブロック２０８において、セット内の全てのプレフィックスの圧縮された値を表すコードワードを形成する。これについては、後述する。コードワードを、ソートしたセットに基づいて形成する。コードワードは、非損失性（non-lossy）であってもよく、すなわち、圧縮されたプレフィックスの全てのビットを再現することができてもよい。複数のサブセットをソートする場合、コードワードは、典型的に、各サブセットに対して形成する。

【００１５】

ブロック２１０において、サフィックスデータ及びコードワードをメモリに記憶する。用途に応じて、記憶用に用いるメモリは、短期（short term）ＲＡＭ（例えば、ＲＡＭ１１２（図１））又は記憶メモリ（例えば、不揮発性メモリ１２０（図１））であってもよい。また、サフィックスデータ及びコードワードを、圧縮及び復元を行うデータ処理プラットフォームの一部ではないデータベース又はネットワーク接続された記憶装置に記憶してもよい。

【００１６】

データセットからの情報が要求されると、記憶したサフィックスデータ及びコードワードを取り出して、プレフィックスを復元する（ブロック２１２）。１つの好適な復元プロセスの例について以下に説明する。次に、復元したプレフィックスをプレフィックスデータと組にして、完全なデータ項目を得る。場合によっては、後述するように、所望のデータ項目が復元プロセスの早くに抽出された場合には、完全なコードワードを復元する必要がない。

【００１７】

図３に、いくつかの実施形態による、圧縮データを含むコードワードを形成するためのより詳細なプロセス３００のフロー図を示す。圧縮プロセスのさらなる例を図４～図６に示す。これらのプロセスは、ソフトウェア圧縮エンジン（例えば、図１の圧縮エンジン１１４）によって、又は、ハードウェア及びソフトウェアの他の好適な組み合わせによって行われてよい。

【００１８】

プロセス３００は、ブロック３０２においてコードワードの形成を始める。ブロック３０４では、コードワードの形成に必要なエンコーディングテーブルを、それらをメモリから取り出すか、それらを計算するか、又は、それらを部分的に計算することによって設ける。これについては後述する。

【００１９】

各プレフィックスに対して、ソーティングによって与えられる順番で、ブロック３０６は、複数のエンコーディングテーブルに問い合わせることによってプレフィックスをエンコードして、データ項目のセットに対する全てのプレフィックスの値を表す圧縮情報を含むコードワードσを形成する。この実施形態では、ブロック３０６は、エンコーディングテーブルに問い合わせて、エンコーディングテーブル内の個々のプレフィックス位置を表す複数の整数を取得する。

【００２０】

データ項目がそのプレフィックス値を介して順序付けられる限り、図５に示すように、プロセスは、ソーティング結果の変化を処理してもよい。ブロック３０８において、複数の入力テーブルに問い合わせた結果を用いて、この実施形態では、図示するように、結果

として得られる整数を合計することによって、コードワードσを形成する。図５に、プレフィックス及びサフィックスが識別されたデータ項目のセットの未処理入力を示す。図示したデータ項目は、簡単にするために、少数の小さいデータ項目を伴う単なる一例であり、本明細書の技術によりデータ項目の他の多くの構成を記憶してもよい。これらのデータ項目をブロック２０６（図２）で述べたようにソートする。このソートは、データ項目に対するセミソート（semi-sort）を構成する。なぜならば、ソーティングにおいてサフィックスデータを用いておらず、プレフィックスデータのみ用いるからである。したがって、ソートの結果は、順番が変化し得る。これは、図５のソート出力Ａ及びＢによって示す通りであり、これらは両方とも、データ項目に対するプレフィックスベースのソートの正当な結果である。なお、同じプレフィックス（１０１）を共有する２つのデータ項目が、Ａ及びＢにおける交換位置に現れている。より全般的には、同じプレフィックスを伴うデータ項目が互いのソートの出力に任意の順番で現れ得るが、出力の正しさには影響しない。入力データの順番の変化（例えば、図５に示すもの）が起こり得るにもかかわらず、プロセス３００は効果的である。

【００２１】

図３を再び参照すると、ブロックは順次的であると示しているが、実際の順番は実施形態に依存する。プロセッサによって行われると、プロセスは各プレフィックスを反復して、各反復において累積した合計に整数値を加え得る。より並列な実施形態の場合、エンコーディングテーブルルックアップを並列に実行し得る。全ての整数が合計されると（ブロック３０８）、完了した合計によってコードワードσが得られる。

【００２２】

図４に、いくつかの実施形態による、エンコーディングテーブルを用いて圧縮データを含むコードワードを形成するためのプロセスのテーブルを示す。圧縮エンコーディングは、ｋ個のｒビットプレフィックスの順序付きリストを取得し、それらを表現するコードワードを出力する。整数ｋはセット内のデータ項目の数を表し、ｒは各プレフィックスにおけるビットの数を表す。このようなエンコーディング方式の一つを示す。これは、組合せ論的表現

【数１】

$$\binom{2^r+k-1}{k}$$

の変形を用いてエンコーディングを列挙している。この方式では、セット０，１，２，．．．，ｋ－２，ｋ－１においてデータ項目を列挙するイテレータｉを用いている。列挙する際、2^rの代わりにｉ番目のデータ項目に対するプレフィックスの値（p_iと言う）、ｋの代わりにｉ＋１（ゼロインデックスでなければｋの代わりにｉとなる）番目のデータ項目に対するプレフィックスの値に付け加えられる。そして、これらの部分表現にわたって合計して、正味の合計を生成する。この実施形態ではｋ個のプレフィックスをエンコードするコードワードσに対する数式が、式（１）によって与えられる。

【数２】

$$\sigma=\sum_{i=0}^{k-1}\binom{p_i+i}{i+1}$$

【００２３】

図示したテーブル４００には、ｋ＝４及び$r=\log_2(3)$に続くこの数式の適用の例を示す。この例では、プロセスは、４個のデータ項目プレフィックスを取り、それらの

記憶コストを４ｌｏｇ$_2$（３）～６．３４ビットからｌｏｇ$_2$（１５）～３．９１ビットに減らす。このエンコーディングは、１つ以上の事前に計算されたテーブルにアクセスすることによって、計算値を必要とする圧縮又は復元要求に応じて閉形式表現（closed form expressions）を計算することによって、又は、これらの組み合わせによって行うことができる。さらに別の代替案は、値又は部分値を必要時に計算するのみであるが、関連するデータ又はハードウェア構造において値が計算された後にそれをキャッシュして、後の使用に備えることである。計算に含まれる乗算計算を一連の等価な加算、シフト及びマスキング動作まで減らすことも可能である。

【００２４】

事前に計算されたエンコーディングルックアップテーブルを用いるバージョンでは、セット内の各データ項目に対して別個のルックアップテーブルを用いることが好ましい（エンコーディングが恒等関数である０番目のデータ項目を除く）。テーブル４００から、４つのエンコーディングテーブルｌ$_0$～ｌ$_3$が形成されて、４つの各列において示される組合せ論的表現に対する値を保持している。そしてこれらの同じテーブルも、デコーディングに対して用いることができる。これについては後述する。

【００２５】

この実施形態では、テーブルの数は、プレフィックスｋの数に等しい。他の実施形態では、より少ないテーブルを用いてもよい。例えば、暗黙的なエンコーディングを用いてテーブルｌ$_0$を実装することができる。これについては、図１０に関してさらに後述する。テーブルは、概して、テーブルｌ$_0$～ｌ$_{k-1}$と呼ばれる。各ｌ$_i$は、ｉ番目のデータ項目のプレフィックスｐ$_i$に対するエンコーディングテーブルである。各ｌ$_i$は、２^{r}個のエントリを保持し、特定のｐ$_i$（特定のエンコーディングを示す）によってインデックス付けされる。具体的には、

【数３】

$$l_i[p_i] = \binom{p_i + i}{i+1}$$

は、ｉ（０≦ｉ≦ｋ－１）の全ての整数値及びｐ$_i$（０≦ｐ$_i$≦２^r－１）の全ての整数値に対するものである。この実施形態では、テーブル内のプレフィックス位置を表すインデックス値を用いてプレフィックスをエンコードする。他の実施形態では、他の値を用いてもよい。例えば、インデックス付き値が、テーブル内で示される順番とは異なる順番で現れ得る。この場合、返されるインデックスは、必ずしもプレフィックス位置ではない。

【００２６】

エンコーディングテーブルを計算又は記憶する効率を上げるために種々の技術を用いてもよい。実行中にエンコーディングテーブルを部分的に計算してその記憶コストを減らすことができる。例えば、全ての２の乗算をシフト演算として実施してもよく、したがって、シフトされた値ではなくこのようなシフトのカウントを記憶する方が効率的である。エンコーディングテーブルｌ$_0$～ｌ$_{k-1}$を提供する際に、エンコーディングテーブル内の２つのエントリが同じ値を記憶する場合には、プロセスは、２つのエントリをマージして単一のエントリにしてもよい。例えば、ｒ及びｋに対する適切な値を選択することによって、プロセスは、

【数４】

$$\binom{p_i + i}{i+1} = \binom{p_i + i}{p_i - 1}$$

という特性又は他の暗号化方式に対する同様の特性を利用して、記憶する必要があるテー

ブル毎のエントリの数を減らしてもよい（例えば、ｐｉ＝５及びｉ＝２の場合、
【数５】

$$\binom{5+2}{2+1}=\binom{5+2}{5-1}$$

であり、何れも３５である）。その場合、プロセスは、個々のｌ$_i$を、タプルｐ$_i$及びｉによってインデックス付けされた単一の行列と取り替えてもよい。これは、ｉ＋１がｐ$_i$－１に等しい場合には重複値を記憶しない（例えば、プロセスは
【数６】

$$\binom{7}{3} \text{ 及び } \binom{7}{4}$$

の両方を記憶しない）。
【００２７】
いくつかの実施形態では、エンコーディングテーブルは、ｒ及びｋの共通の値に対してサイズがキロバイトを超えない場合がある。プロセスはただ、２^{r}ｋ個のエントリのオーダーで記憶すれば良いので、約４～８の範囲のｒの値を用いてもよく、一方で、依然としてスペース節約の大部分が得られ、エンコーディングテーブルに対する面積オーバーヘッドが小さい。この範囲のｒ値における典型的な実施形態に対して記憶要求が下がることを以下の図１０で見ることができる。
【００２８】
図５及び図６のシーケンスに、８ビットデータ項目を使用して３ビットプレフィックスを用いる圧縮プロセス例を示す。図５に、いくつかの実施形態による、セミソーティングプロセスの結果５００を示す。図示しているのは、プロセスに対する４つの未処理入力データ項目（それぞれ８ビット）である。３ビットプレフィックスと残りの５ビットサフィックスとが識別される。セミソートされた出力（例えば、図２のブロック２０６の出力）の２つの正当な変化を、示されたＡ及びＢとして示す。図から分かるように、セット内の２つのデータ項目のプレフィックス値は１０１である。その個々のプレフィックスの相対的な順序付けを受けるデータ項目がソートによって再配列されるため、ソートされた出力において、プレフィックス１０１である２つの図示したデータ項目は、正確さに影響することなく互いに置き換えられ得る。
【００２９】
図６に、いくつかの実施形態による、エンコーディングテーブルを用いてコードワードを形成する典型的なプロセス６００を示す。図示したシナリオでは、図５からのデータ項目のセット（出力Ａによりソートされている）を用いる。符号化プロセスは、概して、ｋ個のｒビットプレフィックスで開始され、プレフィックスをエンコードして、プレフィックスの圧縮表現であるコードワードσを生成する。このシナリオでは、ｋは４であり、プロセス６００は、４つのエンコーディングテーブルｌ$_3$～ｌ$_0$を用いる。エンコーディングテーブルは、図４に関して説明した技術により、又は、他の好適な技術を用いて生成してもよい。図に、各プレフィックス値からこのプレフィックスに対する個々のエンコーディングテーブル内のテーブルエントリまでの矢印を示す。プレフィックス値は、テーブル内でインデックスとして用いられることが示されている。関数「ｎＣｒ」は、組合せ論的表現「ｎ選択ｋ（ｎ ｃｈｏｏｓｅ ｋ）」（順番が問題とならない場合にｎ個の項目のグループからｋ個の項目を選択する方法の数）を示す。それぞれの個々のエントリにおいて整数値が結果として生じるのは、図３のブロック３０６に関して説明した通りである。テーブルエントリから加算器ブロックまでの矢印は、整数値が合計されてコードワードσ（この例で

は、２６１の値を有する）が生成されることを表す。このコードワード値を５ビットのサフィックス値と一緒にメモリに記憶する。

【００３０】

いくつかの実施形態では、エンコーディングテーブルは、各プレフィックスに対して、個々のプレフィックスインデックスが、順序付け特性によって順序付けられたプレフィックス値の複数の順序付き集合に基づいてカウントを与えるように設けられており、カウントは、順序付き集合内のプレフィックスの相対位置に基づいている。それぞれの個々のプレフィックスインデックスに対するカウントは、プレフィックス値の順序付き集合の数を示し、それぞれの順序付き集合のサイズは、順序付き集合内でのプレフィックス値の相対位置に適用される関数の出力であり、順序付き集合内の全てのプレフィックス値は、順序付け特性によって計算された個々のプレフィックスの値よりも小さい。例えば、図６の実施形態では、ｉ番目のプレフィックス（０≦ｉ＜ｋ）を試験する場合、ｉ番目のプレフィックスのエンコーディングを、l_i［プレフィックス］上でテーブルルックアップを行うことによって取り出す。返されるインデックス値は、ｉ番目のプレフィックスを含む第１のリストを数値的に先行するプレフィックスの長さｉ＋１のソートされたリストの数のカウントである。例えば、ｉが３でプレフィックスが２である場合、l_3［２］に、リスト２，０，０，０に先行する非増加の順番で順序付けられた長さ３＋１のリストの数のカウントが記憶される。これらは（１，１，１，１）、（１，１，１，０）、（１，１，０，０）、（１，０，０，０）及び（０，０，０，０）であり、全体で５つである。この実施形態では、順序付き集合とともに用いる順序付け特性は非増加であるが、規定された順番でプレフィックス値を配置する任意の好適な順序付け特性を用いてもよい。

【００３１】

図７に、いくつかの実施形態による、コードワードを復元するためのプロセス７００のフロー図を示す。図８に、いくつかの実施形態による、エンコーディングテーブルを用いてコードワードを復元するプロセス８００の一部を示す。図９は、図８のプロセスのさらなる部分９００を示す図である。以下の説明では、必要に応じて図７～図９を参照する。プロセス７００は、ソフトウェア復元エンジン１１６（図１）、又は、他の好適なハードウェア実施形態によって行われてもよい。図示した復号プロセスは、前述した所定の符号化プロセスを用い得る一例である。上記のエンコーディング又は他のエンコーディングに対して、他のデコーディング方式が可能である。

【００３２】

ブロック７０２において、データ項目のセットからデータを得る要求に応じて、プロセス７００は、データ項目のセットに対するコードワードσをメモリから取り出して、復元を始める。図８の一番上に沿って示すように、データ項目のセットの記憶したサフィックスも取り出す。デコーディングは、全般的に、ｋ個のｒビットプレフィックスの圧縮表現であるコードワードσで始まり、プレフィックスのデコードをｋ－１番目のプレフィックスから始めて０番目のプレフィックスまで行う。プロセス７００にはｋ個の反復（この例では４つ、プレフィックスあたり１つ）が含まれる。反復では、エンコーディングテーブルl_3～l_0（図８及び９に示すような）を、セミソーティングによって生成された順番で用いる。この実施形態では、使用するテーブルは、テーブルl_{k-1}からテーブルl_0のエンコーディングで用いたのと同じエンコーディングテーブルである。

【００３３】

第１の反復において、ブロック７０４におけるプロセス７００では、整数ｉをｋ１に、検索値σ’をコードワードσに等しい値に初期化する。次にブロック７０６において、エンコーディングテーブルl_iを検索して、検索値σ’以下の最大値を探す。これを図８に例示する。図８では、テーブルl_3を、２６１のσ’よりも小さい最高値の検索から強調された検索結果とともに示している。これは、最後のエントリで見られる値２１０である。

【００３４】

この検索結果をブロック７０８で用いる。ブロック７０８では、再生したプレフィックスp_iを、検索で見つかった最大値のテーブルインデックスに等しい値に設定する。図９

の例において、再生したプレフィックスを、テーブルl_3内の最後のエントリのインデックス１１１に設定する。再生したプレフィックス値の１１１を、記憶したサフィックス値１０１１１と、コードワードをエンコードするときに用いたソートによって得られた順番で結合し、テーブルl_3の下に示す完全なデータ項目が得られる。

【００３５】

次にブロック７１０において、整数ｉを減らして設定して、次の反復に進む。ブロック７１２において、検索値σ’を、以前の検索値σ’から、以前の検索値σ’以下であった以前の検索反復で見つかった最大値を引いたものに等しい値に設定する。図９の例では、この値を、２６１－２１０＝５１として計算された新しい検索値σ_2’によって示す。値の供給元を、加算ブロック９０２に送り込まれる矢印によって示す。

【００３６】

次にブロック７１４において、エンコーディングテーブルl_iを検索して、新しい検索値σ’以下の最大値を探す。これを、図８において、テーブルl_2内の強調された検索結果である３５（５１よりも小さい最大のテーブルエントリ）によって示す。ブロック７１６において、再生したプレフィックスp_iを、検索で見つかった最大値のテーブルインデックスに等しい値に設定する。例では、再生したプレフィックスp_2を、テーブルl_2に対して、強調されたインデックス値１０１に設定する。このプレフィックスを、記憶したサフィックスデータと結合して、プレフィックスp_2に対する完全なデータ項目が得られ、１０１１１１１１である。

【００３７】

ブロック７１０～７１６を、整数ｉがゼロにおいて最後の反復が完了するまで繰り返し、再生したプレフィックスのセットが完成する。図８～図９の例では、最後の２つの反復が示され、結果として得られる値を示している。テーブルl_1をσ_1’の値１６を用いて検索する。これは、以前の反復からの以前の検索値５１から、見つかった最大値３５を引いたものに等しい。再生したインデックスp_1を、１６を用いた検索で見つかったテーブルインデックス値（１０１であり、テーブルl_1からのインデックス１５）に設定する。次にテーブルl_0を、σ_0’の値１を用いて検索する。これは、以前の検索値１６から、これまでに見つかった最大値１５を引いたものに等しい。再生したインデックスp_0を、１を用いた検索で見つかったテーブルインデックス値（００１であり、テーブルl_0内で見られる値１のインデックス）に設定する。

【００３８】

なお、図示したプロセス７００では、プレフィックス再生又は復元が完全に生じると示しているが、これは限定ではなく、プロセスの早くに再生されたプレフィックスからのみ値が必要である場合には、プロセスは単に部分的に完了する場合がある。

【００３９】

ブロック７０６及び７１４の検索は、小さいテーブルに対して線形又は並列な方法で行ってもよいし、代替的に、より大きいテーブルに対して、修正された二分探索等の別の好適な検索アルゴリズムを用いて行ってもよい。前述したように、プロセスは、プログラムコードを実行するマイクロプロセッサを用いて、特定用途向けデジタルロジックを用いて、又は、プログラマブルロジックを用いて実施してもよい。

【００４０】

図１０に、いくつかの実施形態により得られた圧縮性能の例のチャートを示す。チャート１０００に、複数のデータセットに対する前述したプロセス例の性能を示し、データ項目あたりのビット減少（垂直軸）対ビットで表したプレフィックス長（ｒ）（項目あたりの圧縮ビットの数）（水平軸）を示す。チャート記号によってデータセットの異なる連想度を識別している。本明細書では、「連想度」という用語を、当該技術分野で通常使用されるように用いて、データのキャッシュセット又は同様のセットを説明する。すなわち、各メモリアドレス又は他のデータ項目識別子は、データ項目のセット又はグループ内の可能な数の位置にマッピングされる。可能な位置の数が連想度である。連想度が２の累乗で増える毎に、圧縮されたプレフィックスあたりほぼ１ビット節約されることに留意された

い（ｒに対する適切な値を想定する）。

【００４１】

ｋ項目セットがｒビットプレフィックスを伴うことを想定すると、これらのプレフィックスを記憶する最初のスペースはｋｒビットである。このようなエンコーディングによって、記憶プロセスは、複製が許されるｋ個のｒビット数のセットの全て$2^{r}{}^{k}$個の並べ換えをエンコードすることができる。しかし、データ項目あたり正確に2^{r}個の別個の値がある場合、これらの値が順序付けられていると、問題は、ｋｒビット数の全ての並べ換えを記憶する必要があることから、単に組み合わせを記憶する必要があることに発展する。

【数７】

$$\binom{2^{r}+k-1}{k}$$

個のこのような組み合わせが存在し、完全な$2^{r}{}^{k}$個の並べ換えよりもはるかに少ない。したがって、本明細書における技術は、

【数８】

$$\left\lceil \log_{2}\binom{2^{r}+k-1}{k} \right\rceil$$

ビットにおけるこれらの組み合わせを最も近い全体ビットに丸めたときに、エンコードすることができる。したがって、セットあたりの正味の節約は、ｋ個のｒビット数（ｋｒビット）の正味のサイズから新しいサイズ（

【数９】

$$\left\lceil \log_{2}\binom{2^{r}+k-1}{k} \right\rceil$$

ビット）を引いたものである。正味の節約をセットあたりの項目数（ｋ）で割ることで、圧縮されたプレフィックス（又は、その事に対する項目）あたりのビット節約が得られる。これは、図１０でいくつかの例に対してプロットされ、式（２）によって概略的に記述される。

【数１０】

$$\text{(2)}\quad \text{ｒビットプレフィックスあたりの節約されるビット} = \frac{r*k-\left\lceil \log_{2}\binom{2^{r}+k-1}{k} \right\rceil}{k}$$

【００４２】

チャート１０００から理解され得るように、連想度が高い構造の場合、プレフィックスの記憶コストは、セット連想度に応じて１～４ビットだけ下げることができる場合が多い。ＡＳＭＤＳでは、データ項目は、長さが単に８～数１０ビットであることが多く、項目あたり１～４ビットの節約は重要である。さらに、プレフィックスは、圧縮の利益の大部分を実現するために、長さが数ビットであればよい。この特性は重要である。なぜならば、本明細書のプロセス例における個々のエンコーディングルックアップテーブルは、それぞれ2^{r}個のエントリを有しているため、プレフィックスが長くなるとエンコーディングテーブルサイズの点でコストがかかるからである。

【００４３】
従来の研究では、１つのデコーディングテーブルと１つのエンコーディングテーブルとを用いている。圧縮の前にソーティングを行っていないと、完全な2^{rk}個の並べ換えは、それぞれエンコーディングテーブル内にエントリが必要となる。圧縮の前にプレフィックスによるソーティングを行っていれば、
【数１１】

$$\binom{2^r+k-1}{k}$$

個のエントリを伴うテーブルのみが必要となる。例えば、各エントリが４バイトのサイズである場合、ｒ＝４及びｋ＝４であっても、ルックアップテーブル内に３８６７エントリが存在する。項目あたり２バイトの場合、各テーブルは約８ＫＢ（両方に対して１６ＫＢ）となる。
【００４４】
対照的に、本明細書のいくつかの実施例ではｋ個のルックアップテーブルを使用し、それぞれ2^rエントリを伴っている。ｋ＝４及び４＝４の場合、このようなデザインでは最大で$k2^r$個のエントリを用いる。エントリあたり２バイトでは、全体的なスペースコストは２バイト＊４＊２＾４＝１２８バイトとなる。このアプローチの記憶コストは、ｒが固定されているときにｋに対してほぼ直線的にスケール変更するが、線形挙動から逸脱する。なぜならば、各ルックアップテーブルエントリは、
【数１２】

$$\log_2\binom{2^r+k-1}{k}$$

ビットのオーダーでエンコードするからである。実行中にルックアップテーブル値を計算することによってエンコーディング及びデコーディングを行うことで、記憶コストがさらに減る。前述したように、別の利用可能な最適化は、最後のルックアップテーブルを暗黙にすることである。なぜならば、それは常に
【数１３】

$$\binom{p_0}{0+1}$$

であり、単にp_0であるため、復元プロセスのｋ－１個の反復後に残っているものは何であれ、最後のプレフィックスだからである。同様に、エンコーディング中に、ゼロ番目のプレフィックスを整数合計に加えることによって、符号化プロセスの最後のプレフィックスをコードワードにエンコードしてもよい（図３の３０８）。
【００４５】
本明細書で説明した技術は、ハードウェア及びソフトウェアの様々な組み合わせを用いて実施することができる。例えば、ハードウェア回路は、デジタルロジック、有限状態機械、プログラマブルロジックアレイ（ＰＬＡ）等を含んでもよい。圧縮及び復元プロセスは、マイクロコントローラが、記憶されたプログラム命令を実行して、保留中のコマンドの相対的なタイミングの適格性を評価することで実施することができる。この場合、命令の一部を非一時的なコンピュータメモリ又はコンピュータ可読記憶媒体に記憶して、マイクロコントローラが実行するようにしてもよい。様々な実施形態では、非一時的なコンピ

ュータ可読記憶媒体は、磁気若しくは光ディスク記憶装置、フラッシュメモリ等の個体記憶装置、又は、他の不揮発性メモリ装置若しくはデバイスを含む。非一時的なコンピュータ可読記憶媒体に記憶されるコンピュータ可読命令は、ソースコード、アセンブリ言語コード、オブジェクトコード、又は、１つ以上のプロセッサによって解釈され及び／又は実行可能な他の命令フォーマットであってもよい。

【００４６】

図１のデータ処理プラットフォーム１００、その任意の部分（例えば、プロセッサ１１０等）、又は、他の実施形態（例えば、カスタマイズされたデジタルロジック実施形態等）は、データベース、データベースインデックスの形態のコンピュータアクセス可能なデータ構造、または他のデータ構造であって、集積回路を製造するためにプログラムによって読み出し、直接又は間接的に使用できるものによって、記述又は表現されてもよい。例えば、このデータ構造は、例えばＶｅｒｉｌｏｇ又はＶＨＤＬ等の高レベル設計言語（ＨＤＬ）におけるハードウェア機能の動作レベル記述又はレジスタ転送レベル（ＲＴＬ）記述であってもよい。この記述は、合成ライブラリからゲートのリストを含むネットリストを生成するためにこの記述を合成し得る合成ツールによって読み出されてもよい。ネットリストには、集積回路を含むハードウェアの機能を表すゲートのセットが含まれる。そして、マスクに適用される幾何学的形状を記述するデータセットを生成するために、ネットリストを配置及びルーティングしてもよい。そして、マスクを種々の半導体製造ステップで用いて、集積回路を製造してもよい。代替的に、コンピュータアクセス可能な記憶媒体上のデータベースは、ネットリスト（合成ライブラリ付き若しくは無し）又はデータセット、必要に応じて、グラフィックスデータシステム（ＧＤＳ）ＩＩデータであってもよい。

【００４７】

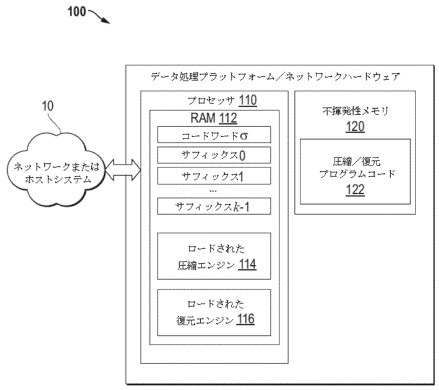
特定の実施形態について説明してきたが、これらの実施形態に対する種々の変更が当業者には明らかである。例えば、データセット内のデータのタイプは、異なる実施形態において異なっていてもよく、例えば、様々なタイプのキャッシュライン、キャッシュタグ、ＡＳＭＤＳからのフィンガープリント、ネットワークルーティングデータ、分岐予測器若しくはハードウェアプリフェッチャからのハードウェアテーブル、又は、データベースエントリ等である。エンコーディングテーブルを生成するために用いられる数学的アルゴリズムは、異なっていてもよい。複数のテーブルに基づくエンコーディング及びデコーディング用の特定のプロセスも異なっていてもよい。エンコーディングテーブルを設ける方法も異なっていてもよく、テーブル値の計算又は部分的な計算が含まれる。さらに、本明細書の例では単一のコードワードを用いているが、他の実施形態では、使用する圧縮アルゴリズムに応じて、特定のデータセットに対して複数のコードワードを生成してもよい。

【００４８】

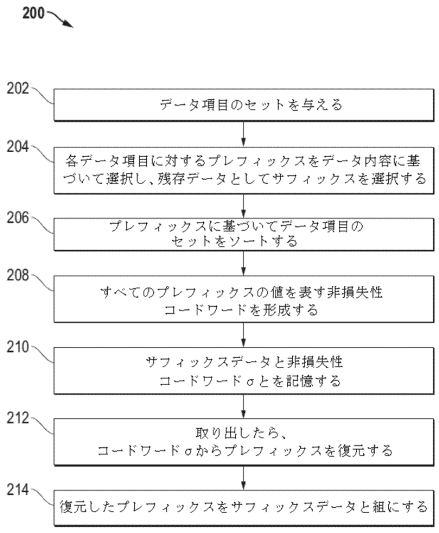
したがって、添付の特許請求の範囲によって、本開示の範囲に含まれる開示した実施形態の全ての変更をカバーすることが意図されている。

【図面】

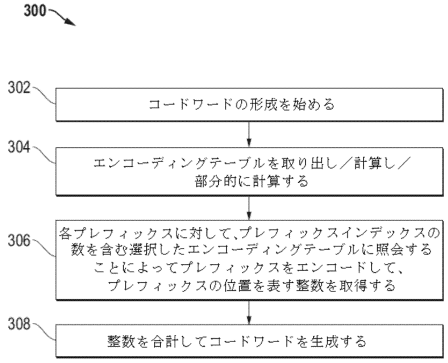
【図１】



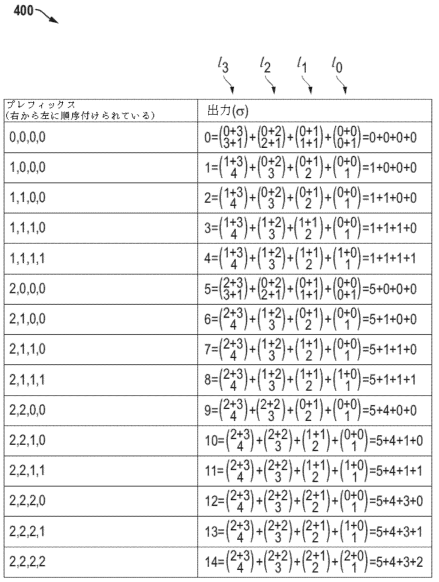
【図２】



【図３】

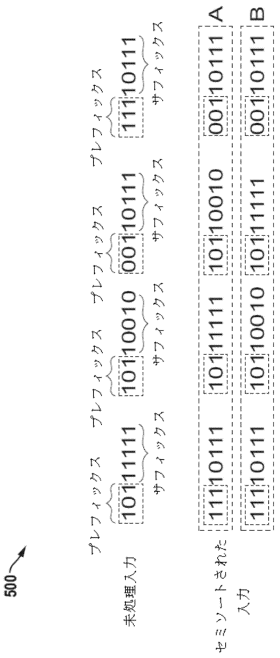


【図４】

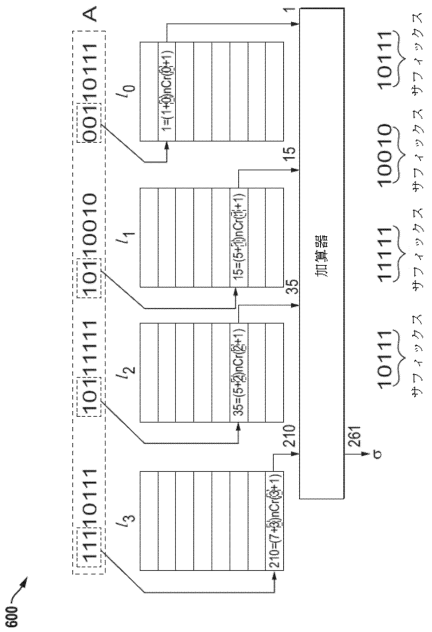


プレフィックス（右から左に順序付けられている）	出力(σ)
0,0,0,0	$0=\binom{0+3}{3+1}+\binom{0+2}{2+1}+\binom{0+1}{1+1}+\binom{0+0}{0+1}=0+0+0+0$
1,0,0,0	$1=\binom{1+3}{4}+\binom{0+2}{3}+\binom{0+1}{2}+\binom{0+0}{1}=1+0+0+0$
1,1,0,0	$2=\binom{1+3}{4}+\binom{0+2}{3}+\binom{0+1}{2}+\binom{0+0}{1}=1+1+0+0$
1,1,1,0	$3=\binom{1+3}{4}+\binom{1+2}{3}+\binom{1+1}{2}+\binom{0+0}{1}=1+1+1+0$
1,1,1,1	$4=\binom{1+3}{4}+\binom{1+2}{3}+\binom{1+1}{2}+\binom{1+0}{1}=1+1+1+1$
2,0,0,0	$5=\binom{2+3}{3+1}+\binom{0+2}{2+1}+\binom{0+1}{1+1}+\binom{0+0}{0+1}=5+0+0+0$
2,1,0,0	$6=\binom{2+3}{4}+\binom{1+2}{3}+\binom{0+1}{2}+\binom{0+0}{1}=5+1+0+0$
2,1,1,0	$7=\binom{2+3}{4}+\binom{1+2}{3}+\binom{1+1}{2}+\binom{0+0}{1}=5+1+1+0$
2,1,1,1	$8=\binom{2+3}{4}+\binom{1+2}{3}+\binom{1+1}{2}+\binom{1+0}{1}=5+1+1+1$
2,2,0,0	$9=\binom{2+3}{4}+\binom{2+2}{3}+\binom{0+1}{2}+\binom{0+0}{1}=5+4+0+0$
2,2,1,0	$10=\binom{2+3}{4}+\binom{2+2}{3}+\binom{1+1}{2}+\binom{0+0}{1}=5+4+1+0$
2,2,1,1	$11=\binom{2+3}{4}+\binom{2+2}{3}+\binom{1+1}{2}+\binom{1+0}{1}=5+4+1+1$
2,2,2,0	$12=\binom{2+3}{4}+\binom{2+2}{3}+\binom{2+1}{2}+\binom{0+0}{1}=5+4+3+0$
2,2,2,1	$13=\binom{2+3}{4}+\binom{2+2}{3}+\binom{2+1}{2}+\binom{1+0}{1}=5+4+3+1$
2,2,2,2	$14=\binom{2+3}{4}+\binom{2+2}{3}+\binom{2+1}{2}+\binom{2+0}{1}=5+4+3+2$

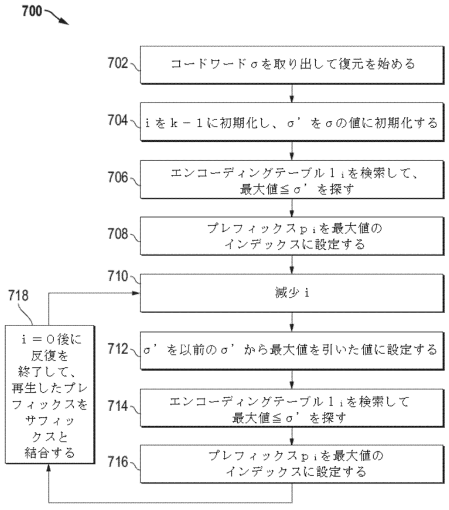
【図５】


500
プレフィックス
サフィックス
未処理入力
セミソートされた入力
A
B

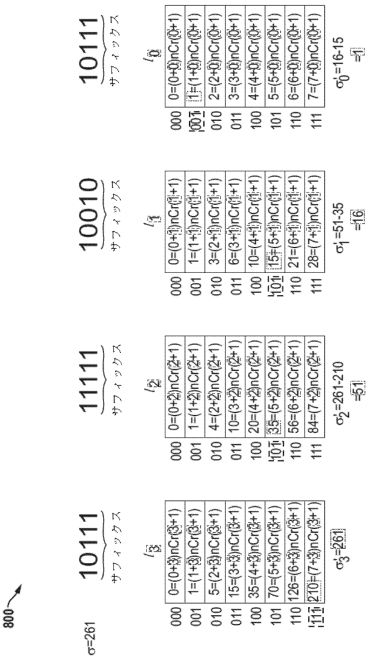
【図６】


600
A
加算器
σ
261
210
35
15
1

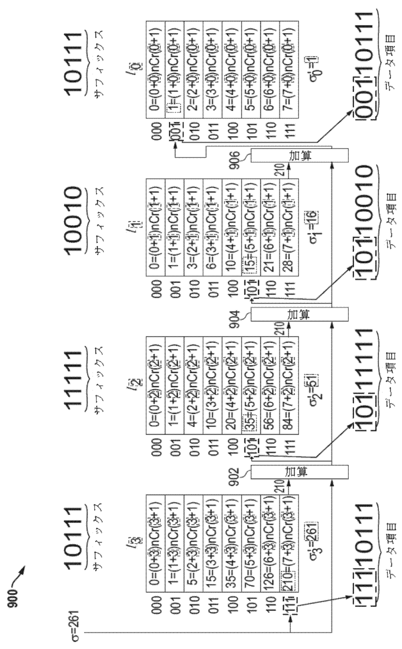
【図７】


700
コードワードσを取り出して復元を始める
iをk－1に初期化し、σ’をσの値に初期化する
エンコーディングテーブルl_iを検索して、最大値≦σ’を探す
プレフィックスp_iを最大値のインデックスに設定する
減少 i
σ’を以前のσ’から最大値を引いた値に設定する
エンコーディングテーブルl_iを検索して最大値≦σ’を探す
プレフィックスp_iを最大値のインデックスに設定する
i＝0後に反復を終了して、再生したプレフィックスをサフィックスと結合する

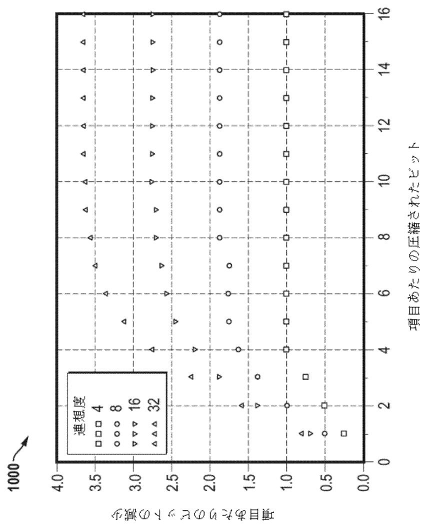
【図８】


800
σ=261
10111
11111
10010
サフィックス

【図９】


900
σ=261
サフィックス
データ項目
加算
902
904
906

【図１０】


1000
連想度
4
8
16
32
項目あたりの圧縮されたビット

フロントページの続き

アメリカ合衆国 ９５０５４ カリフォルニア州、サンタ クララ、オーガスティン ドライブ ２４８５、アドバンスト・マイクロ・ディバイシズ・インコーポレイテッド内
(72)発明者 ヌワン ジャヤセーナ
アメリカ合衆国 ９５０５４ カリフォルニア州、サンタ クララ、オーガスティン ドライブ ２４８５、アドバンスト・マイクロ・ディバイシズ・インコーポレイテッド内
(72)発明者 ジョン カラマティアノス
アメリカ合衆国 ０１７１９ マサチューセッツ州、ボックスボロー、セントラル ストリート ９０、１，２＆３ フロア、アドバンスト・マイクロ・ディバイシズ・インコーポレイテッド内

審査官 原田 聖子
(56)参考文献 特表２０１２－５０２５７３（ＪＰ，Ａ）
米国特許出願公開第２０１１／０１５８３２３（ＵＳ，Ａ１）
(58)調査した分野 (Int.Cl.，ＤＢ名)
Ｈ０３Ｍ ７／３０