様式Ｃ－１９

# 科学研究費助成事業（科学研究費補助金）研究成果報告書

平成　24　年　5　月　23　日現在

機関番号：１６１０１
研究種目：若手研究（Ｂ）
研究期間：2010　～　2011
課題番号：22700101
研究課題名（和文）　BASE 領域の削除によるダブル配列の辞書サイズ圧縮手法に関する研究

研究課題名（英文）　A compression method of double Array by deleting BASE array.

研究代表者
泓田　正雄（FUKETA MASAO）
徳島大学・大学院ソシオテクノサイエンス研究部・准教授
研究者番号：10304552

研究成果の概要（和文）：ダブル配列は，トライを実現する高速なデータ構造であるが，LOUDS に比べて，サイズが大きい．ダブル配列は BASE と CHECK の二つの配列から構成されるが，トライを深さごとに分割することにより，BASE 配列を削除し，ダブル配列のサイズの圧縮をおこなった．固定長の文字列を用いた実験を行った結果，ダブル配列の高速性を維持したまま，密なトライでは，LOUDS よりサイズを小さくすることができた．

研究成果の概要（英文）：Double-array is a data structure to implement a trie and can retrieves keywords very fast. But the size of LOUDS is smaller than that of double-array. Double array uses two one-dimensional arrays, named BASE and CHECK. This research proposed a compression data structure of the double array by dividing trie into each depth and removing the BASE array from that double array. From experimental results, the retrieval speed was almost the same as double array, and the size of the presented method was more compact than LOUDS for a large set of keywords with fixed length.

交付決定額

（金額単位：円）

| | 直接経費 | 間接経費 | 合　計 |
|---|---|---|---|
| ２０１０年度 | 900,000 円 | 270,000 円 | 1,170,000 円 |
| ２０１１年度 | 700,000 円 | 210,000 円 | 910,000 円 |
| 年度 | | | |
| 年度 | | | |
| 年度 | | | |
| 総　計 | 1,600,000 円 | 480,000 円 | 2,080,000 円 |

研究分野：総合領域
科研費の分科・細目：情報学，メディア情報学・データベース
キーワード：キー検索，トライ，ダブル配列，辞書圧縮

## １．研究開始当初の背景

コンピュータ上のデータを扱う最も基本的な操作である「検索」には，高速化や省メモリ化が必須の条件となっている．検索のためのデータ構造に，木構造の頂点を遷移することにより検索を行うトライがあり，トライの実装として，ダブル配列という高速な手法が提案されている．しかし，このダブル配列は，辞書のサイズが大きいという問題点がある．ダブル配列は，BASE，CHECK の２つの配列を使い，トライの１つの状態を表すのに，BASE に４バイト，CHECK に ４バイト必要となる．トライを分割する事により，約 50%のサイズとする手法なども提案されたが，大規模なキー集合を登録できない．キー集合によっては 検索速度が大幅に低下す

る．などの問題がある．

## ２．研究の目的

ダブル配列は，現在の状態番号を s，遷移文字を c，遷移先の状態番号を t としたとき，
t=BASE[s]+CODE[c]
CHECK[t]=s
の２式を満たすように BASE と CHECK の配列の値を決定する．ここで，s=BASE[s]とすることができれば，上記の２式は，
t=s+CODE[c]
CHECK[t]=s
と変更することができ，BASE 配列を削除する事が可能となる．そこで，本研究の目的は，ダブル配列の高速性を損なう事なく，BASE 配列を削除する事により，ダブル配列の辞書圧縮する事とする．

## ３．研究の方法

s=BASE[s]とすることにより，ダブル配列の基本２式は，
t=s+CODE[c]
CHECK[t]=s
とできる．ダブル配列では，CODE[c]の値は，使用頻度の高い文字の値を小さくしたり，文字の文字コード値をそのまま使用していた．これを辞書作成のときに，動的に決定する事により，辞書を作成する．しかし，単語 ab,ba を登録するとき，トライの先頭から順に登録し，状態番号を決定していくので，a の CODE 値を決めないと b の CODE 値を決定できない．また，b の CODE 値を決めないと a の CODE 値を決定できないということになり，すべての文字の CODE 値を同時に決める必要がある．大規模な単語集合を登録するとき，ダブル配列の２式を満たすように全ての CODE 値を同時に決める事は非常に難しいので，CODE をトライの深さごとに定義知る手法を提案する．つまり，トライの深さを k としたとき，ダブル配列の２式を
t=s+CODE[k][c]
CHECK[t]=s
とする．CODE 値は，４バイトで表現できるので，２次元配列としても，CHECK に比べサイズは非常に小さい．また，この方法では，深さ k において文字 c で同じ状態に遷移する事がないので，
t=s+CODE[k][c]
CHECK[t]=c
とすることができるので，CHECK の各要素は１バイトで表現することができる．この新しい手法はダブル拝謁に対して，シングル配列と呼ぶ．
シングル配列の構築は，トライの深さの浅い方から順に，使用している状態が重ならないように，全ての文字について CODE 値を決定していくことにより行なうことができる．
検索は，ダブル配列と同様に２式の計算だけで行なえるので，同速度で検索できる．

## ４．研究成果

トライの配列表現（Matrix）と，オリジナルのダブル配列（ODA），CHECK を１バイトで表すダブル配列（CDA），LOUDS（LOUDS）と，提案手法のシングル配列（SAMC）の比較を行った．
表１に検索の最悪時間計算量を示す．ここで，k は検索単語の長さ，｜Σ｜は，登録した単語に出現する文字種の数を表す．

表１．時間計算量

| 実装方法 | 時間計算量 |
|---|---|
| Matrix | k |
| ODA | k |
| CDA | k |
| LOUDS | k ｜Σ｜ |
| SAMC | k |

表１より，従来のダブル配列法と同じ時間計算量であることが分かる．また LOUDS より高速に検索できる．
次に表２に，辞書のサイズを示す．LOUDS はオープンソースのライブラリ tx を使用してサイズを計測した．ここで，|D|は，トライの状態数，m は単語の最大長を表す．ダブル配列は CODE 配列のサイズも含めてある．

表２．辞書のサイズ

| 実装方法 | 辞書サイズ |
|---|---|
| Matrix | 4｜Σ｜|D| |
| ODA | 8|D|+｜Σ｜ |
| CDA | 5|D|+｜Σ｜ |
| LOUDS | 1.34|D|(≒11/32|D|+|D|) |
| SAMC | |D|+4m｜Σ｜ |

4m｜Σ｜は，|D|と比較して非常に小さいので，提案手法のサイズが最も小さくなる．
Intel Xeon 2.4GHz(L2 cache : 256K-Byte)．において，検索速度と辞書のサイズを計測した．0000 から 9999 までの４桁の固定長の数字を使用した時の辞書サイズ（キロバイト）を図１に示す．横軸は 10,000 個の単語からランダムに抽出した単語数を示している．配列構造は非常にサイズが大きくなっている．語数が多いときには，簡潔構造の LOUDS よりもサイズが小さい結果となった．理論評価の示すように，常に提案手法のサイズが小さい結果とならなかった理由は，ダブル配列を構築するときに空き領域ができてしまい，最大の状態番号が LOUDS に比べ大きくなってしまったからである．図２に検索速度を示す．

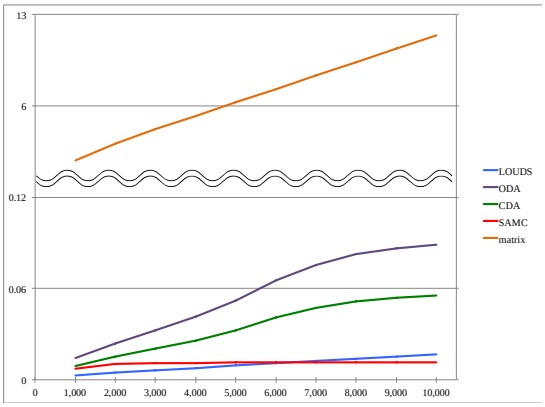


図１．辞書のサイズ（４桁の数字）

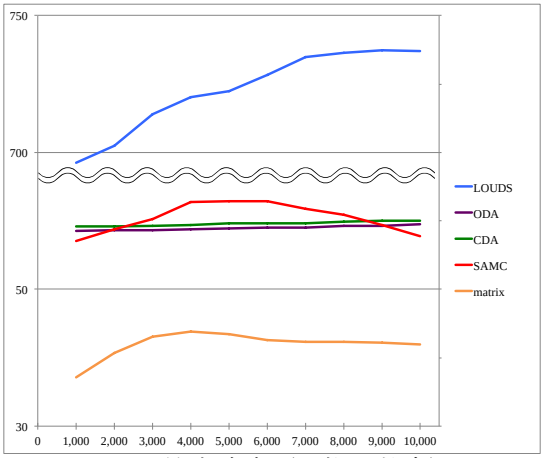


図２．検索速度（４桁の数字）

検索は，登録した全てのキーをランダムに検索し，1000 個当たりの検索速度(ミリ秒)を示している．LOUDS の検索速度は非常に遅く，提案手法の検索速度は，他のダブル配列とほぼ同様となった．図３，図４には，00000 から 99999 までの５桁の固定長の数字を使用した時の辞書サイズと検索速度を示す．結果は，図１，図２とほぼ同様の結果となった．配列構造の検索速度が遅くなっているのは，2 次キャッシュの影響だと考えられる．

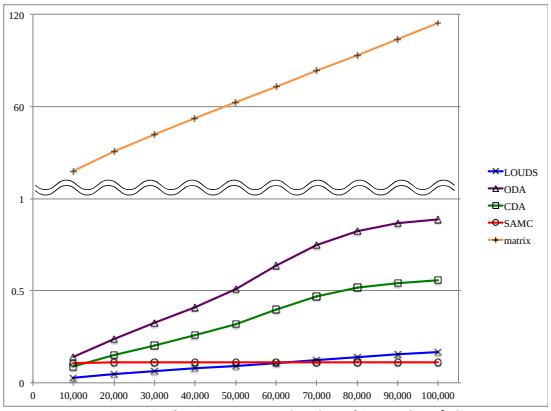


図３．辞書のサイズ（５桁の数字）

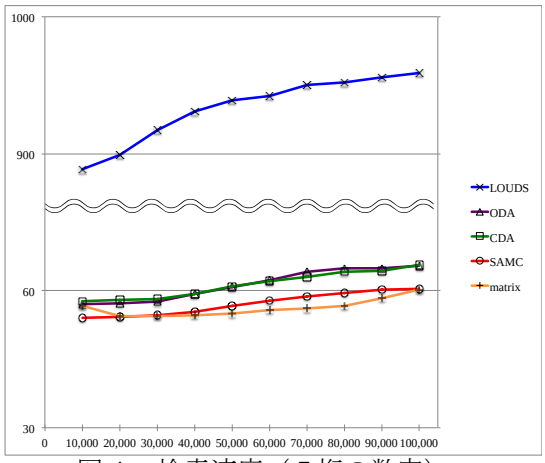


図４．検索速度（５桁の数字）

同様に，aaaa から zzzz の４桁の小文字のアルファベットについても実験を行った．結果を図５，図６に示す．

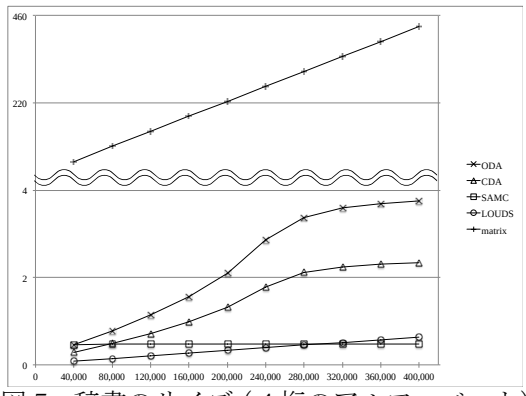


図５．辞書のサイズ（４桁のアルファベット）

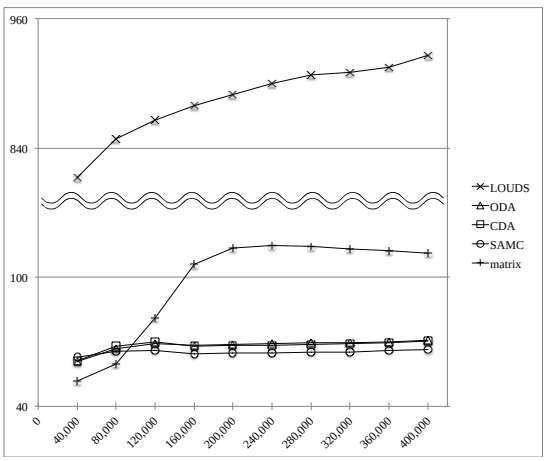


図６．検索速度（４桁のアルファベット）

この結果も，これまでの実験と同様の結果となった．配列構造の検索速度が急に遅くなっている部分があるが，これは，2 次キャッシュのサイズは 128k であり，ちょうどこの辺りで辞書のサイズが 128k を超えたのが原因だと考えられる．

以上の結果より，文字種の数を増やしても，文字数を増やしても，提案手法の速度は 2 次キャッシュに影響されず，他のダブル配列手法と常に同等の高速な検索を行なうことができた．辞書のサイズについては，トライが密な場合には，LOUDS よりも小さくなるという良好な結果を得られたが，キー数が少ない場合，サイズの縮小があまり行なわれなかった．これは，未使用領域が増えているためであり，この特徴は，可変長の文字列を扱った場合には，さらに顕著に現れる．提案手法は，密で固定長のデータについて，最も効果を得られる手法となった．

５．主な発表論文等
（研究代表者、研究分担者及び連携研究者には下線）

〔学会発表〕（計１件）
① 泓田正雄, Compression of Double Array Structures for Fixed Length Keywords, 2011.11.26, 2nd International Conference on Networking and Information Technology（香港）

６．研究組織
(1)研究代表者
泓田　正雄（FUKETA MASAO）
徳島大学・大学院ｿｼｵﾃｸﾉｻｲｴﾝｽ研究部・准教授
研究者番号：10304552

(2)研究分担者
（　　）

研究者番号：

(3)連携研究者
（　　）

研究者番号：