2L-04

節点の遷移集合に着目した自由領域管理による高速なダブル配列構築法の提案

村山　智也†　　望月　久稔†
†大阪教育大学

1　はじめに

トライ木を実現する高速な検索手法のデータ構造としてダブル配列があり，トライ木の検索性能を維持しつつ領域を削減する [1]．ダブル配列は節点から遷移可能な遷移種の集合を 1 次元配列上で組み合わせた構造をとる．そのため，構築の際にこの遷移集合を組み合わせる位置，すなわち基底値の計算が必要となり，リアルタイムな情報を追加する上で高速な基底検索法が必要となる．そこで本稿は，ダブル配列構築の高速化を目的として，ダブル配列上の未使用要素を周囲の未使用状況に応じて分類することで，遷移集合が当てはまる基底値の検索に必要な比較回数の削減をはかる．

2　ダブル配列の構築と基底検索

ダブル配列はトライ木における節点の遷移を 2 本の 1 次元配列 Base と Check により表現する．遷移種 a による節点 r から節点 t への遷移が存在するとき，ダブル配列は式 (1),(2) を満たす．Base 配列は節点 r の遷移集合を組み合わせる際の基底値 Base[r] を表し，Check 配列は遷移元の節点 t を表す．以降，配列の添字が共通する Base と Check の組をダブル配列における要素とし，トライ木の節点を表す要素を使用要素，それ以外を未使用要素とする．

$$Base[r] + a = t \quad (1)$$

$$Check[t] = r \quad (2)$$

ダブル配列を構築する際，遷移集合がダブル配列上で当てはまる基底値を検索する必要がある．遷移種の内部表現値を $\{a = 1, b = 2, c = 3, \cdots\}$ として，遷移集合 $s = \{a, c, d\}$ から，基底値を検索する例を示す．$a = 1$ を未使用要素 r に当てはめると，残りの遷移種 $c = 3, d = 4$ はそれぞれ要素 $r+2, r+3$ に対応する．そこで，$r+2, r+3$ が未使用要素となるような基底値を決定する．

基底検索の先行研究として，未使用要素リストを単方向で管理する森田らの手法 [2]，双方向で管理する大野ら [3]，矢田ら [4]，中村らの手法 [5] などがある．

3　使用状況による自由領域管理

従来手法は，全ての未使用要素を基底値の検索候補とするため，検索する遷移集合が当てはまらない未使用要素を比較する可能性がある．そこで提案手法は，未使用要素を右方向に隣接する要素の未使用状況に基づいて分類することで，基底値の検索における比較回数の削減をはかる．以降，遷移集合が当てはまる基底値の検索を XCheck とよぶ [1]．本章では，隣接未使用状況に基づく未使用要素の分類法を説明する．

XCheck で検索する遷移集合 s に含まれる遷移種のうち，内部表現値が最小の遷移種を s_{min} とする．提案手法の XCheck は，検索候補である未使用要素に s_{min} を当てはめた際，あらかじめ分類された未使用要素リストから，s_{min} 以外の遷移種がそれぞれ未使用要素に当てはまるような検索候補を決定する．これにより，未使用要素を比較する回数の抑制が期待できる．例として，2 章における遷移集合 $s = \{a, c, d\}$ の場合，$s_{min} = a = 1$ を検索候補の未使用要素 r に当てはめ，s_{min} 以外の遷移種が当てはまる未使用状況を選択する．

未使用要素とその右方向に存在する要素の未使用状況に基づく分類法を説明する．配列上で未使用要素の右に隣接する要素 m 個について，使用要素を 0，未使用要素を 1 とするビット列で未使用状況を数値化する．数値化した未使用状況は 2^m 通りが考えられる．そこで，未使用要素リストのヘッダとして要素数 2^m の配列を用意し，配列の添字と数値化した未使用状況とを対応させる．

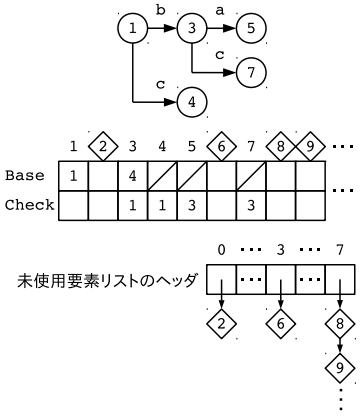

図 1: キー集合 K を登録した自由領域管理

Proposal to Fast Construction Method of Double-Array by Free-Space Management Focused on Node's Transitions
†Tomoya Murayama　†Hisatoshi Mochizuki
†Osaka Kyoiku University

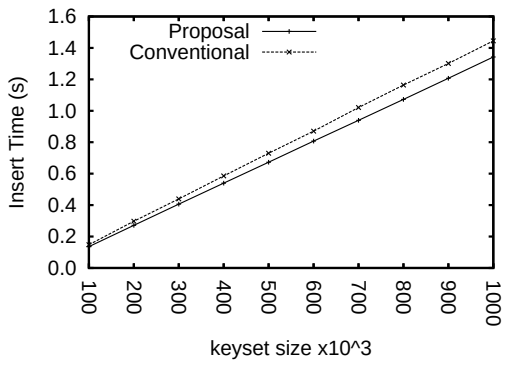

図 2: 構築時間

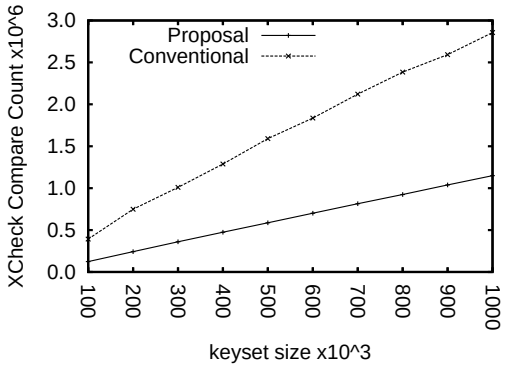

図 3: 比較回数

具体例として，キー集合 K = {"*ba*","*bc*","*c*"} を登録したトライ木とダブル配列，未使用要素リストを図 1 に示す．ここで $m = 3$ とする．図中，未使用要素 6 の右に隣接する 3 要素について，要素 7 は使用要素，要素 8，9 は未使用要素である．したがって，未使用要素 6 は，隣接未使用状況が $(011)_2 = (3)_{10}$ となり，未使用要素リストの 3 番へッダにつながる．

4 基底検索における比較回数の評価

提案手法を実験から評価する．実験環境として Intel(R) Core(TM) i7 920 @ 2.67GHz，CentOS6.6 32bit，キーセットとして英語版 Wikipedia[6] のタイトル集合を使用する．文字コードは UTF-8 とする．従来手法として，未使用要素を双方向リンクリストで管理する中村ら [5] の手法を使用する．提案手法において未使用状況を見る長さ m は 3 とする．

タイトル集合からランダムに抽出した 10 万〜100 万件を昇順に登録した際の構築時間を図 2 に示す．提案手法は構築時間を 8〜9 %短縮した．実験したどの件数でも提案手法の方が高速となった．

同じ実験で計測した XCheck の比較回数を図 3 に示す．提案手法は比較回数を 50〜60 %削減した．未使用要素を分類することで，余分な未使用要素での比較を削減したことが分かる．また構築時間と同様，実験したどの件数でも提案手法の方が少ない比較回数で基底値を検索した．

比較回数の変化率と構築時間の変化率との間の相関係数は 0.77 となり，正の相関を示した．すなわち，提案手法は比較回数を削減することで構築時間を短縮したことが分かった．

5 おわりに

本稿では，ダブル配列構築の高速化を目的として，未使用要素を右方向に隣接する未使用状況に基づいて管理することで遷移集合が当てはまる基底値の検索を高速化する手法を提案した．実験から，基底検索に要する比較回数を削減し，構築を高速化した．

今後の課題として，未使用状況を見る長さと性能との関係の分析，未使用状況の包含関係を考慮した分類方法が挙げられる．

参考文献

[1] Jun-ichi Aoe：An Efficient Digital Search Algorithm by Using a Double-Array Structure，IEEE Transactions on Software Engineering，Vol.15，No.9，pp.1066-1077，1989.

[2] 森田和宏，泓田正雄，大野将樹，青江順一：ダブル配列における動的更新の効率化アルゴリズム，情報処理学会論文誌，Vol42，No9，pp.2229-2238，2001.

[3] 大野将樹，森田和宏，泓田正雄， 青江順一：ダブル配列による自然言語辞書の高速更新法，言語処理学会第 11 回年次大会発表論文集，pp.745-748，2005.

[4] 矢田晋，大野将樹，森田和宏，泓田正雄，吉成友子，青江順一：接頭辞ダブル配列における空間効率を低下させないキー削除法，情報処理学会論文誌，Vol.47，No.6，pp.1894-1902，2006.

[5] 中村康正，望月久稔：圧縮ディジタル探索木における辞書情報更新の高速化手法，情報処理学会:データベース，Vol.47，No.SIG 13 (TOD 31)，pp.16-27，2006.

[6] Wikipedia：https://www.wikipedia.org/，2016/1/6 参照.