1Q-04 ダブル配列における追加の高速化手法

森田 和宏　住友 徹　泓田 正雄　青江 順一
徳島大学工学部知能情報工学科

1 はじめに

トライ構造はキーの表記文字単位に構成された木構造を用いて検索するキー検索技法の一つであり，登録キーの総数に依存せず高速な検索ができること，検索失敗位置の特定が容易であること，検索文字列中の接頭辞の検出が容易であることなどの理由により，形態素辞書，かな漢字変換辞書などの自然言語辞書を中心として広く用いられている．

このトライ構造を実現するデータ構造として一般的な手法には，配列を用いた手法とリストを用いた手法とがある．しかし，配列を用いた手法では配列がスパースになるため記憶量に無駄が生じ，リストを用いた手法では高速な検索ができなくなる．

これらを解決する手法に，青江 [1, 2] の提案したダブル配列法がある．ダブル配列法は検索の高速性とコンパクト性を併せ持つ優れたデータ構造である．しかし，ダブル配列法は更新時間に問題があり，特に追加時間の問題は，その応用分野を限定させている．

本論文ではダブル配列の追加時間を高速化する手法を提案し，実験により本手法の有効性を実証する．

2 ダブル配列と問題点

2.1 ダブル配列法

ダブル配列はトライ構造でのノード間の遷移を，2つの配列 BASE と CHECK で実現する．トライ構造において，ノード s からノード t へラベル a による遷移が定義されている場合，ダブル配列では次式を満足する．

$$t = \mathrm{BASE}[s] + \mathrm{N}(a), \quad \mathrm{CHECK}[t] = s$$

即ち，ノード t は，ノード s に対する BASE[s] とアークラベル a の内部コード値 (numerical code) $N(a)$ の和で計算され，CHECK[t] にはノード s から引かれたことを定義する s を格納する．また，トライの葉ノードに対する BASE 値には負数を格納し，その他のノードと区別する．

2.2 ダブル配列の問題点

ダブル配列における問題点は，追加アルゴリズムが複雑となるため追加時間に影響を及ぼすことである．

A Fast Insertion Method of a Double-Array Structure
Kazuhiro Morita, Toru Sumitomo,
Masao Fuketa and Jun-ichi Aoe
Department of Information Science & Intelligent Systems,
University of Tokushima

ダブル配列へのキーの追加は，ダブル配列の状態によって次の3つに分けることができる．

A. ダブル配列にキーが登録されていない場合．
B. キー登録時に一度も衝突が起きなかった場合．
C. キー登録時に衝突が起こった場合．

特に問題となるのが C. の場合であり，衝突を回避するためダブル配列中のノードを移動しなければならいが，移動先の未使用ノードを検索するにあたり，ダブル配列のノードをシーケンシャルに走査する必要がある．このため，登録キー数が増加するとダブル配列のノード数も増加し，さらにキー登録時の衝突回数も増加するため，ダブル配列の走査時間が大きく影響し，追加時間の低下を招いている．

3 空ノードリンク法

ダブル配列での追加の問題を解決するために，空ノードリンク法を提案する．空ノードリンク法では空ノード（未使用ノード）同士をリンクで結び，追加時の未使用ノードを検索する際の走査対象を最小限に抑えることで追加時間の高速化を図る．まず始めに，空ノードをリンクで結ぶために以下を定義する．

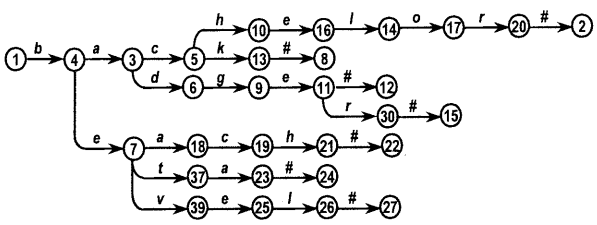
ダブル配列の空ノード（未使用ノード）の番号を，昇順に $r_1, r_2, \ldots, r_m$ とするとき，

$$\mathrm{CHECK}[r_i] = -r_{i+1} \quad (1 \le i \le m-1)$$
$$\mathrm{CHECK}[r_m] = -(DA\_SIZE + 1)$$

なるリンクを作成する．但し，r_1 は E_HEAD なる変数に格納される．これらを空ノードリンクと呼ぶ．空ノードの CHECK 値を負数にするのは，通常のノードと区別するためである．また，変数 DA_SIZE は，ダブル配列で使用されている最大のノード番号を格納する．

キーの追加において，C. の場合のように衝突が起こった場合，衝突したノードの移動先を検索するために，従来は CHEKC[1] から CHECK[DA_SIZE] までの DA_SIZE 回ノードを調査する必要があったが，空ノードリンク法では，CHECK[E_HEAD] から m 回の調査ですみ，m は非常に小さくなるため追加時間が高速化される．

空ノードリンク法を用いたダブル配列とトライの例を図1に示す．



(a)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
BASE	1	-1	1	1	1	1	16	-2	5	10	11	-3	7	1	-4	1	1	15	12	1
CHECK	1	20	4	1	3	3	4	13	6	5	9	11	5	16	30	10	14	7	18	17

	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
BASE	21	-5	23	-6	13	26	-7	0	0	14	0	0	0	0	0	0	21	0	19
CHECK	19	21	37	23	39	25	26	-29	-31	11	-32	-33	-34	-35	-36	-38	7	-40	7

(b)

図 1: 空ノードリンク法の例

4 評価

4.1 理論的評価

ダブル配列の検索の最悪時間計算量は，検索キーの長さを k とすると $O(k)$ となる．

ダブル配列への追加の最悪時間計算量は，衝突したノードの移動先を検索する時間に依存する．更に，空ノードリンク法の場合はCHECKの変更による空ノードリンクの更新時間にも依存する．トライの入力記号の最大数を e，ダブル配列の使用，未使用インデックス番号の数をそれぞれ n, m とし，m が非常に小さいことを考慮すると，従来のダブル配列の追加時間は，$O(e \cdot (n+e))$，空ノードリンク法の場合は，$O(e \cdot (1+e))$ となり，ダブル配列のサイズに依存しなくなるので非常に高速となる．

削除の最悪時間計算量は，葉ノードを削除するだけであるので，従来のダブル配列は $O(1)$，空ノードリンク法は $O(m)$ となる．

4.2 実験による評価

本手法の構成システムは約 1,000 行の C++ 言語で記述されており，DELL Precision 410 (OS:WindowsNT4.0, CPU:pentiumII[400MHz]) 上で稼動している．入力記号の総数 e は，1バイトで表現できる数 256 に端記号を加えた 257 である．また，実験に用いたデータは，EDR 電子化辞書 [3] の英語単語辞書である．

図 2 に登録キー数を変化させた場合の追加時間の実験結果を示す．図中の追加時間は全キーの登録時間をキー数で割った，1件あたりの時間を示している．図 2 の結果から，空ノードリンク法は 9～320 倍高速に追加できることがわかる．また，追加時間がダブル配列のインデックス数に依存する従来の手法は，追加キー

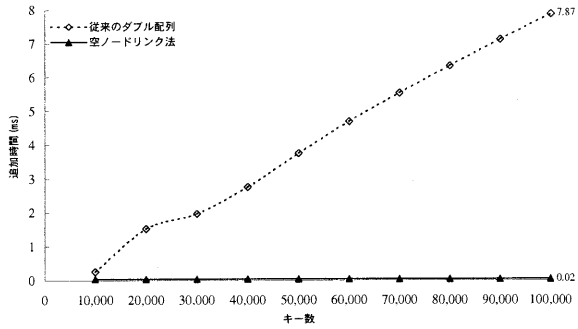


図 2: 追加時間の実験結果

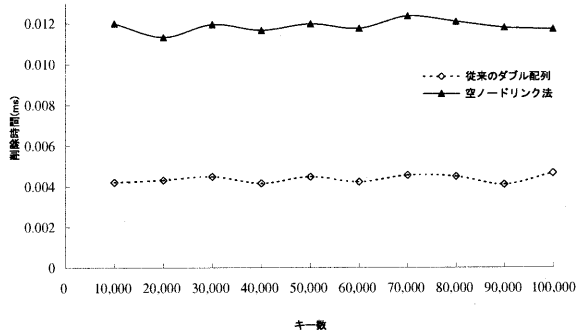


図 3: 削除時間の実験結果

数が増加するにつれて追加時間も増加しているのに対し，ダブル配列のインデックス数に依存しない空ノードリンク法はほぼ一定の値を示していることもわかる．

図 3 に登録キー数を変化させた場合のキー登録後，100 件のキーを削除した場合の削除時間の実験結果を示す．図中の削除時間は 1 件あたりの時間である．図 3 の結果から，登録キー数に関係なく一定の値を示していることがわかる．また，空ノードリンク法は削除時間が空ノードの数に依存するので従来手法に比べ削除時間が遅くなっているが，それでも 1 件あたり 0.012ms であり非常に高速であることがわかる．

5 おわりに

ダブル配列の追加時間の問題を解決するため，ダブル配列のサイズに依存しない高速な追加手法を提案した．また，実験により本手法の有効性を実証した．

参考文献

[1] 青江順一. ダブル配列による高速ディジタル検索アルゴリズム. 電子情報通信学会論文誌, Vol. J71–D, No. 4, pp. 1592–1600, 1988.

[2] Aoe J. An efficient digital search algorithm by using a double-array structure. *IEEE Transactions on Software Engineering*, Vol. SE-15, No. 9, pp. 1066–1077, 1989.

[3] 日本電子化辞書研究所. EDR 電子化辞書, 1996.