拡張ハッシュ法を用いた類似キー検索ファイル

6W－6　田中栄一　川出慎二
神戸大学　工学部

1．はじめに

拡張ハッシュ法[1]と類名表記[2]を用いて，次の条件を満たすファイルを構成する方法を提案する．
(1) 誤りが小さい場合は誤ったキーでも検索できる．
(2) 入力キーと類似しているキーを全て列挙する．
(3) キーの挿入が容易で，またキーの削除によっても記憶利用率が低下しない．類名表記を用いたファイル[2]は条件 (1) と (2) を満たすが (3) を満たさない．条件 (1)～(3) を満たす方法には類名表記とB^+木を用いたもの[3]がある．本文では，これより操作が単純で能力もほぼ等しいファイルを提案する．

2．類名表記と拡張ハッシュ法

いまアルファベットを次のような2つの類に分類する．

$$A=\{a,b,c,d,e,f,g,h,i,j,k,l,m,n\},$$
$$B=\{o,p,q,r,s,t,u,v,w,x,y,z\}.$$

A,B を類名と呼び，キーを類名で表したものを類名表記という．“apple” を類名で書くと ABBAA となり，これを “apple ” の類名表記という．類名表記を用いるとキー集合を部分集合に分割できる．

類名表記をディレクトリに用い，拡張ハッシュ法によるファイルの例を図2に示す．ページ部の1つのブロックに5つのキーまで記憶できるものとする．ここではキーの長さは4であるが，ディレクトリ部の類名表記の長さは2である．

キーの数が増加してページが溢れると，ディレクトリ部の類名表記を長くして，ページ部のキーを分割する．またキーの削除が増え，ページ部の空き領域が多くなると，ディレクトリ部の類名表記の長さを短くし，ページを合併して，記憶利用率の低下を防ぐ．

AAAA → cake, face
AAAB → baby, lady
ABAA → cold, home
BBAB → star, stay

図1．類名表記によるキー分類の例

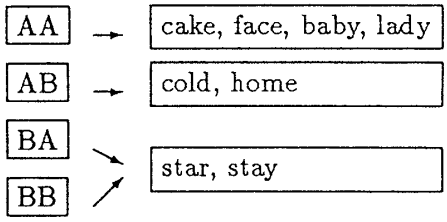

図2．拡張ハッシュ法によるファイルの例

3．類似キーの検索法

入力キーを x とし，その類名表記を $E(x)$ とする．2つの文字列を α,β とするとき，$D(\alpha,\beta)$ を α から β へのレーベンシュタイン距離とする．ディレクトリにある類名表記の長さを m，$E(x)$ の最初の m 桁を $(E(x))_m$，距離のしきい値を d_t とし，$S(x)$ を x との距離が d_t 以内のキーの集合とする．$S(x)$ は最初空にしておく．x と類似したキーの列挙は次のように行う．

(1) $D(E(x),E')\le d_t$ となる類名表記 E' を全て発生させ，(2) を行う．
(2) $(E')_m$ をディレクトリで探し，あれば $(E')_m$ が指すブロック B を呼び，(3) を行う．
(3) ブロック B にある全てのキー y に対して $D(x,y)$ を計算し，$D(x,y)\le d_t$ なら y を $S(x)$ に記憶する．

x と最も類似したキーを出力したいときは，(3)

A File for Similar Key Search Based on Extendible Hashing
Eiichi Tanaka and Shinji Kawade
Kobe University

の代わりに (3)′ を行う.

(3)′ ブロックBにある全てのキー y に対して $D(x,y)$ を計算し，距離が d_t 以下のもののうち，$D(x,y) \leq D(x,z)$ $(z \in B)$ となる y を出力する. (このような y は複数あることもある).

(1) で E' を発生させる代わりに，E_k をディレクトリ部の類名表記とし，$D(E(x),E_k)$ を計算して，その値が d_t 以下となる E_k を求めて，(2) の操作をすることも考えられるが，キーの数が数千以上で，$d_t = 1$ あるいは $= 2$ の時は，(1) の方が速い.

4 . 実験結果

約 23 万語の英語の辞書を用いて実験を行った. $d_t = 1$，すなわち，1つのキーの文字の置換誤り，挿入誤り，脱落誤りは高々1とした. 正しいキー x' を誤らせて x とし，(1)〜 (3)′ を用いて最も類似したキーを検索し，(3)′ の出力 y が1つで，かつ $x' = y$ となるとき，すなわち，誤り訂正と見たときの長さ8の場合の正答率を図3に示す. これは誤ったキーによる検索の成功率でもある. このときのページの読み込み回数の平均値を図4に示す. 図5に記憶利用率を示す. 類数4のときは類数2のときに比べページの読み込み回数は約2倍であった. ランダムに選んだキーを挿入して記憶利用率を測定した. 記憶利用率は約 0.675 で B木のそれに近い.

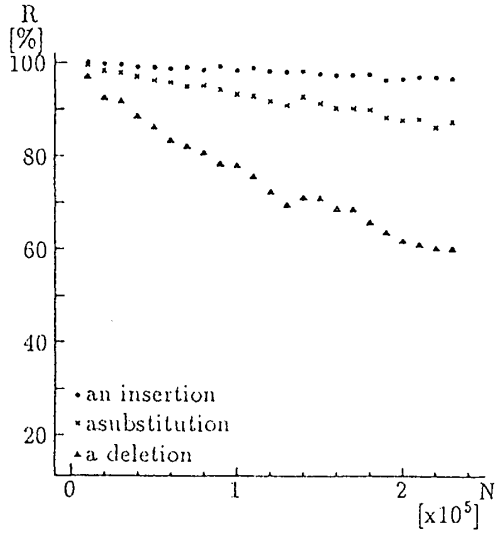

図3. 誤ったキーによる検索の成功率 R

5 . あとがき

類似キーの検索結果は文字類の作り方には影響されない. 記憶利用率を向上させることは今後の問題である.

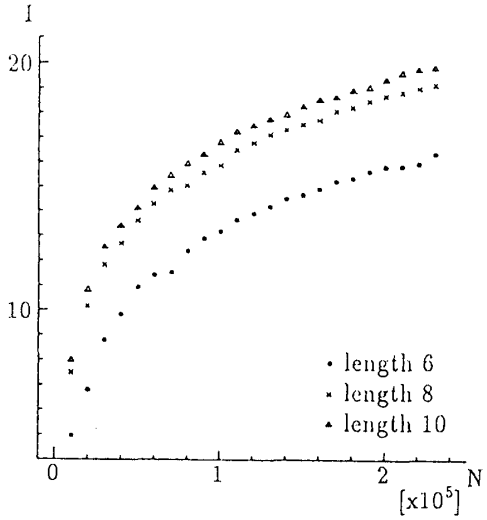

図4. 誤ったキーによる検索時のページの読み込み回数の平均値 I

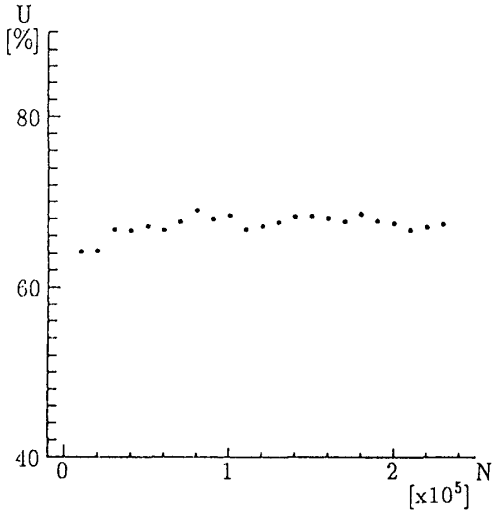

図5. 記憶利用率 U

参考文献

[1] D. Fagin et al. : Extendible hashing − a fast access method for dynamic files, ACM Trans. Database Systems, vol.4, no.3, pp.315-344 (1979).

[2] E. Tanaka et al. : A high speed string correction method using a hierarchical file, IEEE Trans. PAMI, vol.9, no.6, pp.806-815 (1987).

[3] 平出基一，田中栄一：誤ったキーでも検索できるファイル構成法，情報処理学会データベースシステム研究会研究報告，no.96，pp.65-74 (1993).