# 日本語文書に対する新しい索引検索方式
## 4F－2　— 索引作成と検索の原理 —

倉知 一晃＊　野口 直彦　菅野 祐司　稲葉 光昭

松下電器産業(株) マルチメディアシステム研究所

## 1 はじめに

近年，実用化が進んでいる大規模な全文検索システムにおいては，単純な文字列検索では高速化に限界があるため，文字成分表 [2][4][5] などのシグニチャファイル方式，あるいは n-gram による転置ファイル方式 [3] などの高速化手法が提案されている．更に高速化を図るためには，n-gram ではなく，検索文字列として意味のある通常の単語による転置ファイルを用いることが考えられるが，膠着語である日本語文書に対してその手法を適用しようとすると，形態素解析などの単語切り出しを行なわねばならず，その単語切り出しが完全に行なわれない限り，検索もれが避けられない．

筆者らは，上の問題を解決する，検索もれのない単語索引検索方式を考案した．本方式では，単語による索引ファイルを構成するため，n-gram による転置ファイルに比べて，原理的に高速化が可能であり，更に索引ファイルの容量も低減できる．

本稿では，その単語索引作成方法と索引検索方法の原理について報告する．

## 2 単語索引作成方法

通常の辞書を用いて，形態素解析によって単語索引を構成し全文検索に利用する場合，次のような問題がある．

- 辞書の登録単語以外では検索ができない．
- 切りだし方に曖昧さがあるので検索もれがでる．
- 文書中に辞書に未登録な語 (未知語) が存在するため，検索もれがでる．

これらの問題は，次のように単語索引を構成することで解決することができる．

- 全ての文字を一文字語相当として辞書に登録する．
- 切りだし可能な単語を全て抽出し，登録する．

全ての文字を辞書に登録しておくことで，見かけ上文書中には未知語が存在しないことになり，かつ切り出し可能な単語の出現位置を全て登録するので，非決定的な切り出し方に起因する検索もれは生じない．また，任意の検索文字列は，辞書中の単語の並びで表現されることになるので，単語以外の検索文字列に対してももれのない検索が保証できる．(このような構成法に従った単語索引を，完全索引と呼ぶことにする.) しかし，完全索引では，同一の文字位置が重複して登録されることになるので索引量が膨大になることが予想される．このような完全索引に対し，文書中で他の単語の真部分語になっている語の出現位置情報は，それが含まれる語の出現位置情報から原理的には復元可能であるから，索引から省くことができる．そのような基本的アイデアに従って構成した単語索引を，**完全延長極大索引**と呼ぶこととする．以下に**完全延長極大索引**の形式的な定義を与える．

【定義】 (延長語，部分語)

２つの文字列 $\alpha,\beta$ について，$\alpha$ が $\beta$ の延長文字列であるとは，$\alpha$ の一部分が $\beta$ 全体と一致し，$|\alpha| \geq |\beta|$ であることをいう．$\alpha$ が $\beta$ の延長文字列であるとき，$\beta$ は $\alpha$ の部分文字列であるという．$\alpha$ が $\beta$ の延長文字列であって，$\alpha$ の頭部 (終端) と $\beta$ 全体が一致する場合，$\alpha$ が $\beta$ の接頭 (接尾) 延長文字列であるという．この時 $\beta$ は $\alpha$ の接頭 (接尾) 部分文字列であるという．

さらに，$\alpha,\beta$ が辞書中の単語である場合はそれぞれ延長語，部分語，接頭 (接尾) 延長語，接頭 (接尾) 部分語であるという．

【定義】 (索引)

長さ $l$ の文書 $doc$ の，辞書 $dict$ による索引とは，次の条件を満たす二つ組 $(w,n)$ の集合である．

1. $n \in N$ かつ $1 \leq n \leq l$
2. $w \in dict$ かつ $w = doc[n : n+|w|-1]$

ただし，$doc[i:j]$ は文書 $doc$ の $i$ 文字目から $j$ 文字目までの部分文字列を表す．

索引の要素 $(w,n)$ を**索引要素**と呼ぶ．

**完全索引**とは，辞書 $dict$ を用いて切りだすことのできる索引要素 $(w,n)$ をすべて含む索引のことである．

次に，索引要素間の関係として，延長索引要素を以下のように定義する．

【定義】 (延長索引要素)

文書 $doc$ の，辞書 $dict$ による索引要素

$$a = (w_1, n_1),\ b = (w_2, n_2)$$

について $a$ が $b$ の延長索引要素であるとは，

$$n_1 \leq n_2 \text{ かつ } n_1 + |w_1| \geq n_2 + |w_2|$$

であることをいう．

延長索引要素という関係は索引要素集合上の半順序関係となる．従って，文書の辞書 $dict$ による任意の索引に対して，その中での延長極大索引要素が自然に定義される．

---

New indices for Japanese text
-The principle of making index and searching index-
Kazuaki Kurachi, Naohiko Noguchi,
Yuji Kanno, Mitsuaki Inaba
Matsushita Electric Industrial, Co., Ltd.
5-15, 4-chome, Higashi-Shinagawa,
Shinagawa-ku, Tokyo 140 Japan
*e-mail:kurachi@trl.mei.co.jp

【定義】 (延長極大索引要素)
文書 $doc$ の, 辞書 $dict$ による任意の索引 $ind$ を構成する索引要素 $a$ について、$ind$ 中に $a$ の延長索引要素が自身以外に存在しない時、$a$ は索引 $ind$ の延長極大索引要素であるという.

この関係を用いて, 完全延長極大索引は以下のように定義される.

【定義】 (完全延長極大索引)
文書 $doc$ の, 辞書 $dict$ による完全延長極大索引とは, 辞書 $dict$ に関する完全索引中の延長極大要素すべてからなる索引である.

完全延長極大索引では、辞書中の単語の出現位置は全て, 当該単語そのものか当該単語の延長語の出現位置情報中に必ず登録してあることが保証されるので, 任意の文字列でもれのない検索が可能となる. ただし, 完全索引を構成する時に行なったように, 全ての文字を辞書に登録しておく必要はなく, 辞書引きが失敗した時にのみ, 当該文字を辞書に追加登録すればよい. これは, 当該文字が辞書に追加される以前に, 文書中に存在したとしても, その文字は必ず他の切りだし単語の部分語になっているはずであるから, それ以前の単語切りだしには影響を与えないからである.

このような完全延長極大索引を作成するアルゴリズムを以下に示す.

完全延長極大索引作成アルゴリズム

```
procedure make_index(doc, dict, index)
begin
  tail := 1; l := | doc |;
  for head = 1 to l do begin
    W := {w | wはdoc[head : l]の接頭部分語 }
    if W = φ then begin
      dict に一文字 doc[head : head] を追加;
      W := {doc[head : head]};
    end;
    l_prefix:=W の中で文字長が最長の単語;
    if head+ | l_prefix |> tail then begin
      index に (l_prefix, head) を登録;
      tail := head+ | l_prefix |;
    end;
  end;
end { of make_index };
```

新聞のデータについて上のアルゴリズムを適用して索引作成の実験をしたところ, 索引容量は原文書以下に抑えることができた [1].

## 3 索引検索方法

次に, 完全延長極大索引を使用して任意の条件文字列でもれのない検索をする方法を示す. 完全延長極大索引に辞書 $dict$ に関する延長極大索引要素のみが登録されていることを利用し, 条件文字列が文書中に出現する可能性がある場所を効率良く検索しつくす方法である.

索引検索方法
いま,

$$L(w) = \{wt \mid w\text{の接尾延長語}wt\}$$
$$R(w) = \{wt \mid w\text{の接頭延長語}wt\}$$
$$E(w) = \{wt \mid w\text{の延長語}wt\}$$
$$p(w) = (w\text{の接頭部分語のうち最長の単語})$$
$$s(w) = (w\text{の接尾部分語のうち最長の単語})$$
$$P(W) = \{n \mid n\text{は}wt\text{の出現位置} \cap wt \in W\}$$

とし, $i =\mid p(w) \mid, j =\mid s(w) \mid, n =\mid w \mid, m = n - j + 1$

$$P_0 = P(E(w))$$
$$P_1 = \cup_{a,b} PE(w, a, b)$$

ただし, $i \le a < n, 1 < b \le m, a \ge b - 1$
としたとき,

$$P = P_0 \cup P_1$$

が文字列 $w$ の出現位置となる. ここで,

$$PE(w, a, b) = PL(w, a) \cap PR(w, b)$$
$$PL(w, k) =$$
$$\begin{cases} P(L(w[1:k])) & k = i \\ P(L(w[1:k]))\cup (P(s(w[1:k]))\cap pl(\max(i, k- \mid s(w[1:k]) \mid), k)) & i < k < n \end{cases}$$
$$PR(w, k) =$$
$$\begin{cases} P(R(w[k:n])) & k = m \\ P(R(w[k:n]))\cup (P(p(w[k:n]))\cap pr(k, \min(m, k+ \mid p(w[k:n]) \mid))) & 1 < k < m \end{cases}$$
$$pl(s, e) = \cup_{s \le k < e} PL(w, k)$$
$$pr(s, e) = \cup_{s < k \le e} PR(w, k)$$

この方法で新聞のデータについて検索実験を行ない, もれのない検索が実現できること, 単語あるいは複合語による検索が十分高速に行なえることを確認した [1].

## 4 おわりに

本稿では, もれのない全文検索を実現する単語索引検索方式を提案した. 今回試作した実験システム [1] では, 上記の方法をそのまま実装したが, 実用化に際しては, 条件文字列の先頭から単純に照合範囲を延長していくのではなく, 文書中の単語の出現頻度を考慮して, 処理ループの早い段階で検索対象範囲の絞り込みを行なうなどの工夫を行なうことによって, 更なる高速化が可能である. また, 延長語が非常に多い語を検索条件として与えた際に, 使用者に注意を促すような検索システムに本索引方式を応用する予定である.

## 参考文献


[1] 稲葉: 日本語文書に対する新しい索引検索方式 - 試作・実験および評価 -, 本大会予稿集,4F-3(1995).

[2] 福島 他: 全文検索用文字成分表の一圧縮方式, 第 47 回情報処理学会全国大会 (4), pp.83-84 (1993).

[3] 菊池 : 日本語文書検索用高速全文検索の一手法, 電子情報通信学会論文誌, Vol.J75-D-I,No.9, pp.836-846 (1992).

[4] 畠山 他: ソフトウエアによるテキストサーチマシンの実現, 情報処理学会研究会報告, Vol.FI25, pp.19-26 (1992).

[5] 宮原 : 文字連接を用いたフルテキスト検索の高速化, 第 40 回情報処理学会全国大会, pp.880 (1990).