4Q-02　転置ファイルの圧縮・復号アルゴリズムとその評価

二神常爾
愛知学院大学

1. はじめに

キーワードを用いた文献検索において転置ファイルを用いる方法が知られている。転置ファイルを用いると、高速な検索を行なえる利点がある一方で転置ファイルのサイズが大きくなる不利な点がある。そこで、記憶装置のメモリ量を節約するために転置ファイルの圧縮を行なう。転置ファイルの圧縮については多くの研究が行なわれてきた。転置ファイルの圧縮のためにハフマン符号やエライアス符号、算術符号等を用いる[1,2]。また、多段階で圧縮を行なうアルゴリズムがChouekaらにより提案され、ファイルの圧縮率が議論されてきた[3]。この研究では二段階の場合についてこのアルゴリズムを改良し、メモリ量を低減できることを示す。またAND検索時の復号計算量を低減できることを示す。これまで、復号の計算量は理論的に評価されてこなかったが、この研究ではメモリ量と復号計算量を定式化して数値計算により評価した。また、本研究の提案アルゴリズムは符号理論のシンドローム情報源符号化の考え方を利用しているのが特徴である。

2. 準備

2.1 数学モデル

本研究では転置ファイルの数学的モデルとして確率pのベルヌイモデルを用いる。検索システムに収録されている文献総数をN_0、検索キーワード総数をMとして転置ファイルを$M\times N_0$の行列で表現する。行列の各位置にシンボル1が出現する確率はpである。各キーワードが含まれる文献のリストは行ベクトルに対応するので、これをキーワードベクトルと呼ぶ。

2.2 従来の圧縮・復号方式

Chouekaらの報告した従来の二段階の圧縮・復号のアルゴリズムは、長さN_0のキーワードベクトルを長さNのサブブロックに分割しビットマップ写像を行なう。すなわち、サブブロックが全0ならばシンボル0を、非全0ならば1を割り当てて長さN_0/Nのビットマップをつくる。これをビットマップベクトルと呼ぶ。さらに非全0のサブブロックの重みをkとして$k\lceil\log_2 N_0\rceil<N$が満たされるならば、シンボル1の位置を$k\lceil\log_2 N_0\rceil$ビットで表して記憶する。上式が満たされないならば、サブブロックそのものを記憶する。

3. 提案方式のアルゴリズムとその評価

3.1 圧縮アルゴリズム

提案方式は非全0のサブブロックをその重みに応じてシンドローム情報源符号化を行う点が従来方式と異なる。すなわち、パラメータ(N,K,D)、訂正シンボル数$T=\lfloor(D-1)/2\rfloor$の線形符号を用いて長さNのサブブロックの重みwが

(i) $1\le w\le T$ならば長さ$N-K$のシンドロームに圧縮する、

(ii) $w\ge T+1$ならば圧縮せずにそのまま記憶する。

なお、(i)と(ii)のサブブロックを区別するために圧縮または非圧縮のサブブロックの先頭に1シンボルの0または1のフラグを付ける。サブブロックの全個数N_0/Nのうち、全0のサブブロックの個数をN_0q_0/N、(i)の条件を満たすサブブロックの個数をN_0q_1/N、(ii)の条件を満たすサブブロックの個数をN_0q_2/Nとすると、次式が成立つ。

$$q_0=(1-p)^N$$

$$q_1=\sum_{i=1}^{T}{}_NC_i p^i(1-p)^{N-i}$$

$$q_2=1-q_0-q_1$$

メモリ量R_2は次式で与えられる。

$$R_2=M\{N_0/N+N_0q_1(N-K)/N$$
$$+N_0q_2+N_0(q_1+q_2)/N\}+N2^{N-K}$$

第一項はビットマップの長さ、第二項はシンドローム圧縮される非全0のサブブロックの長さ、第三項は圧縮されないサブブロックの長さ、第四項はフラグの長さ、第五項は復号木の大きさである。

3.2 復号・検索アルゴリズム

M_q個のキーワードを指定し、これらのキーワード全てを含む文献を求めるAND検索を考える。

①M_q個のキーワードのビットマップベクトルを$\vec{a}_{i_k}(k=1,2,\cdots,M_q)$としてこれらのAND積をとる。

Compression and decoding of an inverted file
Tsuneji Futagami
Aichi Gakuin University, Araike 12, Iwasakicho, Nisshin-shi, Aichi Prefecture 470-0195, Japan

$$\vec{a}_{prod} = \vec{a}_{i_1} \cap \vec{a}_{i_2} \cap \cdots \cap \vec{a}_{i_{M_q}}$$

ベクトル $\vec{a}_{prod}$ の第 $j(1 \le j \le N_0/N)$ 成分を $a_{prod,j}$ とする。

(i) $a_{prod,j} = 0$ ならば、いずれかの k に対して $a_{i_k,j} = 0$ が成立つ。k 番目のキーワードは j 番目のサブブロック内の文献に含まれないので、以後の復号を行なう必要がない。

(ii) $a_{prod,j} = 1$ ならば、②の復号を行なう。

②サブブロックがシンドローム情報源符号化されているならば、復号木を用いて元の長さ N の系列を復号する。圧縮されていないならば、長さ N のまま出力する。k 番目のキーワードに対する j 番目のサブブロック $\vec{B}_{i_k,j}$ に対して次の演算を行なう。

$$\vec{U}_j = \vec{B}_{i_1,j} \cap \vec{B}_{i_2,j} \cap \cdots \cap \vec{B}_{i_{M_q},j}$$

ベクトル $\vec{U}_j$ 内でシンボル 1 をもつ位置を k とすればもとのキーワードベクトルの第 $(j-1)N+k$ 成分に対応する文献が求める文献となる。復号・検索の計算量は次式で与えられる。

$$C_2 = \alpha M_q N_0 \{1/NM_q + (q_1+q_2)/N$$
$$+ q_1(q_1+q_2)^{M_q-1}(N-K)/N + (q_1+q_2)^{M_q}/N\}$$

第一項はビットマップベクトルベクトルの復号の計算量、第二項はフラグを読み込む計算量、第三項は圧縮ベクトルを復号木と照合する計算量、第四項はベクトル $\vec{U}_j$ を読み込む計算量になる。

4. 数値計算条件

本研究では実際の文献検索システムで用いられている値を考慮して、つぎのパラメータの $3\times3\times3\times2 = 54$ 通りの組み合わせに対してメモリ量と復号計算量の関係を評価した。

$$p = 10^{-4}, 10^{-3}, 10^{-2}$$
$$M_q = 2, 4, 6$$
$$N_0 = 10^4, 10^5, \infty$$
$$M = 10^3, 10^4$$

符号理論によればパラメータ (N,K,D) が次の VG 限界式を満足するならば線形符号が存在することが知られている。

$$2^{N-K} > \sum_{j=0}^{D-2} {}_{N-1}C_j$$

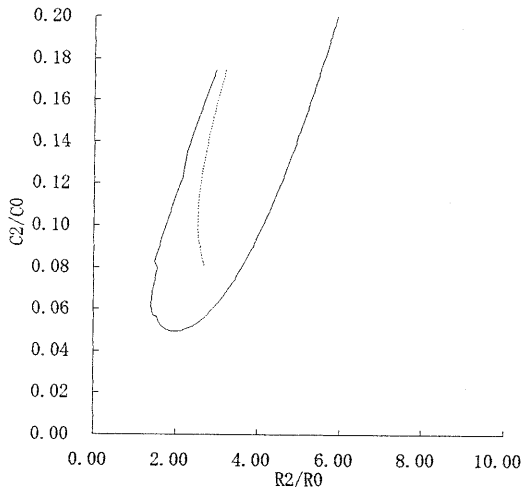
そこで、上式を満足するパラメータ (N,K,D) に対してメモリ量 R_2 と計算量 C_2 の関係を評価した。

5. 評価結果及び考察

一例を図に示す。計算条件は $M_q = 6, N_0 = 10^5, p = 10^{-2}, M = 10^4$ である。実線と点線がそれぞれ提案アルゴリズムと従来アルゴリズムに対応する。横軸はメモリ量を理論的限界値で正規化し、縦軸は計算量を一段階アルゴリズムの理論的限界値で正規化した。原点に近い領域で、メモリ量と計算量はトレード・オフ関係を示す。すなわち、メモリ量と計算量のうち、一方が増加するともう一方は減少する。この領域で、提案アルゴリズムは従来アルゴリズムよりメモリ量、計算量ともに優れている。計算した全ての場合についてメモリ量と計算量はトレード・オフ関係を示す。また、提案方式はメモリ量の増加を低く抑えて計算量を従来よりも著しく低減できる特徴をもつ。

6. まとめ

本研究ではシンドローム情報源符号化を利用した圧縮・復号方式を提案、評価した。その結果、提案方式の計算量は著しく低減され、従来方式よりも優れていることが明らかになった。



参考文献

[1]A. Moffat and J. Zobel: Parameterized compression for sparse bitmaps, 15th Ann Int'l SIGIR'92, pp. 274-285(1992).

[2]A. Bookstein and S. T. Klein: Flexible compression for bitmap sets., Proceedings of the IEEE Data Compression Conference, pp. 402-410(1991).

[3]Y. Choueka, A. S. Fraenkel, S. T. Klein, and E. Segal: Improved hierarchical bit-vector compression in document retrieval systems, Proc. 9'th ACM-SIGIR Conference on Information Retrieval, pp. 88-97 (1986).