検索効率を考慮した転置ファイルの圧縮

二 神 常 爾[†,☆]

キーワードを用いた文献検索システムでは，検索を高速化するために転置ファイルを利用する．本論文ではAND検索を行う際の転置ファイルの新たな圧縮・復号方式を提案する．提案方式の特徴はキーワード検索を効率良く実行するために，シンドローム情報源符号化を用いて2段階で圧縮・復号を行うことである．ここで，圧縮は第1段階の符号化とシンドローム情報源符号化による第2段階の符号化からなる．まず，転置ファイルをベルヌイモデルで表現し，圧縮ファイルおよび復号木を格納するのに要するメモリ量と復号・検索に要する計算量によりシステムを評価する．次に，シンドローム情報源符号化の性能を示す符号の限界式（VG限界式）を満足するパラメータに対してメモリ量と計算量を評価し，いくつかの重要な性質とともに提案する方式が優れた性能を持つことを示す．その結果，① 2つの評価基準はトレード・オフの関係にあり，わずかなメモリ量の増加に対して計算量が大きく低減すること，② 提案アルゴリズムの計算量は従来の2段階アルゴリズムより著しく低減すること，③ 検索キーワード数が小さい場合には高頻度で出現するキーワードに対するほど提案アルゴリズムの性能が良くなること，④ 転置ファイルの大きさとともに検索システムの性能が良くなることが明らかになった．

Compression of an Inverted File, Considering Retrieving Efficiencies

Tsuneji Futagami[†,☆]

In document retrieval system by keywords, an inverted file is often used. We propose a new method for compressing and decoding it at two stages. In the second stage, it is compressed by syndrome source coding. We represent an inverted file by a Bernoulli model and evaluate the system by two quantities. One is size of memory where an inverted file and a decoding tree are stored. Another is time complexity for decoding it and retrieving documents including keywords given as a query. The numerical calculation shows the followings: ① The two quantities show trade-off relation. Slight increase in memory size gives drastic reduction in time complexity. ② Time complexity for the proposed algorithm decreases strikingly compared with for the previous two-stage algorithm. ③ In case the number of query keywords is small, performance of the system is more improved for the more frequent keyword. ④ The performance of the system is improved as the size of an inverted file becomes large.

1. はじめに

文献検索では複数のキーワードを指定した際に，これらのキーワードを含む文献の文献情報が出力される．キーワードによる文献検索は日本科学技術情報センター（JICST）などでの従来のオンライン検索システム[1)]のほか，インターネットを利用した検索システム[2)]上でも行われている．文献検索を高速に行うためにしばしば文献とキーワードの二次元の対応表を利用する．通常転置ファイルはキーワードを含む文書番号のリストを指すが[3)]，ここではキーワードと文献の対応表を広義の意味で転置ファイルと呼ぶことにする．転置ファイルを用いる利点は検索を高速に行うことができる点である．逆に，不利な点は転置ファイルを格納するのに大容量の記憶装置が必要なことである[4)]．転置ファイルはサイズが大きいので，しばしば圧縮して二次記憶装置に格納する[5)]．たとえば，キーワード数10,000，文献数100万件としても転置ファイルの大きさは1ギガバイト以上にもなる．記憶装置の容量を節約するうえで，転置ファイルの圧縮方法の研究は重要である．

転置ファイルの圧縮については多くの研究が行われている．転置ファイルの圧縮には，ハフマン符号やエライアス符号[3)]，順位符号[6)]，シャノン–ファノ符号，漸近的にエントロピーレートを達成する情報源符号化である算術符号[7)]などが用いられる．いずれの研究で

† 愛知学院大学大学院客員研究員
Aichi Gakuin University
☆ 現在，関東学園大学
Presently with Kanto Gakuen University

も，転置ファイルの圧縮例が報告され圧縮率が議論されている．文献 7) では文書番号のリスト全体を符号化しているが，文献 3)，6) では文書番号の差分値を符号化している．文書番号を符号化する方法と文書番号の差分値をランレングス符号により符号化する方法を併用する方式[8]も提案・評価されている．また，多段階で転置ファイルを圧縮，復号するアルゴリズムが提案・評価されている[9)~11)]．

一方，符号理論（誤り訂正符号）の視点を用いた文献ファイルの圧縮・復号方法が研究されているが比較的少ない．これは二元ベクトルを圧縮するのに符号のシンドロームを利用する考え方に基づいている[12)~14)]．Chien らの論文[12)]では文献検索システムを符号理論の枠組みでとらえて，圧縮した文献ファイルの復号過程を論じている．この際に，検索効率を考慮しているが，転置ファイルが対象ではない．Ancheta, Jr. [13)]は二元無記憶情報源から発生する二元系列をシンドローム情報源符号化方式により圧縮する際の圧縮率の理論的限界性能を述べ，圧縮率がエントロピーに到達することを示した．また，BCH 符号のシンドロームを利用して圧縮する際の圧縮率を数値計算により求めた．

本論文では符号理論の考え方を生かして，多段階での圧縮・復号法[9)~11)]をさらに発展させた転置ファイルの新しい圧縮，復号・検索のアルゴリズムを提案，評価する．本論文では，文献検索システムの性能を 2 つの評価基準により評価する．第 1 に，圧縮した転置ファイルおよび復号木を格納するのに要する記憶装置の容量（メモリ量）を評価する．これはファイルの圧縮率と関わりの深い量である．一方，文献検索における転置ファイルの圧縮問題は単に圧縮効率では議論できない．検索キーワードを与えたとき，該当する部分を効率良く見い出しこれを復号しなければならない．そこで，メモリ量と同時に第 2 に検索の効率を評価する．本論文では，検索効率の指標となる量として圧縮ファイルの復号・検索に要する計算量を理論的に評価した．ハフマン符号化した圧縮ファイルの復号の効率は 1 秒あたり圧縮ファイルの大きさにして 0.1 メガビットのオーダになる[15)]．したがって，復号効率の良いアルゴリズムの研究は転置ファイルに収録される文献数が非常に大きいときに重要である．また，多段階で復号を行うことにより，検索時間が短くなることが報告されている[11)]．本論文では AND 検索時のメモリ量と計算量の関係を理論的に評価した．AND を含む検索時には 2 段階で圧縮・復号することにより，復号・検索の計算量が従来のアルゴリズムに比べて大幅に減少することを，数値計算および理論的解析によって示す．

2. 準備と従来の研究

2.1 文献検索における転置ファイルのモデル

転置ファイルを行列 A により表現する．キーワードと文献の総数はそれぞれ M と N_0 とし，

$$A = [A_{ij}], \quad A_{ij} \in \{0,1\}, \quad 1 \le i \le M,\ 1 \le j \le N_0, \tag{1}$$

とおく．i $(i = 1,2,\ldots,M)$ 番目のキーワードが j $(j = 1,2,\ldots,N_0)$ 番目の文献に含まれている場合は $A_{ij} = 1$，含まれていない場合は $A_{ij} = 0$ によって表現する．また，行列は長さ N_0 の行ベクトル $\vec{A}_i$ の集合により表すことができるので，行列 A を次のように書き直す．

$$A = \begin{bmatrix} \vec{A}_1 \\ \vec{A}_2 \\ \vdots \\ \vec{A}_M \end{bmatrix}. \tag{2}$$

ここで，長さ N_0 の行ベクトル $\vec{A}_i$ を i 番目のキーワードに対するキーワードベクトルと呼ぶ．すなわち，キーワードベクトル $\vec{A}_i \in \{0,1\}^{N_0}$ は

$$\vec{A}_i = (A_{i,1}, A_{i,2}, \ldots, A_{i,N_0}), \tag{3}$$

で与えられ，これから i 番目のキーワードが含まれている文献を見い出すことができる．i $(1 \le i \le M)$ 番目のキーワードを含む文献の数を n_i とする．すなわち，n_i はキーワードベクトル $\vec{A}_i$ のキーワード出現数（キーワードベクトル内の値 1 の個数）に等しい．提案アルゴリズムでは後述するように長さ N_0 のキーワードベクトルを長さ N $(N < N_0)$ のサブブロックに分割してサブブロック内のキーワード出現数（サブブロック内の値 1 の個数）に応じてサブブロックを単位に圧縮する．本論文では N_0/N が十分に大きいと仮定してサブブロックのキーワード出現数の分布は確率 p_i $(= n_i/N_0)$ のベルヌイ分布に従うと仮定する．すなわち，キーワード出現数 k $(1 \le k \le N)$ のサブブロック数はサブブロックの総数 n $(= \lceil N_0/N \rceil)$ を用いて

$$n \binom{N}{k} p_i^k \{1 - p_i\}^{N-k}, \tag{4}$$

と表せると仮定する．$\lceil x \rceil$ は x 以上の最小の整数を表す．ベルヌイモデルは転置ファイルの数学的モデルとして用いられているが[5)]，この仮定の正当性は 4 章で議論する．

2.2 従来の圧縮・復号アルゴリズム

2.2.1 1段階圧縮・復号アルゴリズム

キーワードベクトルをサブブロックに分割しないで圧縮・復号を行う方式を1段階の圧縮，復号・検索のアルゴリズムと呼ぶ．キーワード出現数 n_i のキーワードベクトルの圧縮率 ρ_i について次式が成り立つ．

$$\begin{aligned}\rho_i &\geq H(p_i)\\ &= -p_i \log_2 p_i - (1-p_i)\log_2(1-p_i). \end{aligned} \quad (5)$$

ここで，$p_i = n_i/N_0$ であり，$H(p_i)$ は確率 p_i に対するエントロピーである．任意の符号化により圧縮した転置ファイルの大きさ R については次の関係が成り立つ．

$$R = N_0 \sum_{i=1}^{M} \rho_i \geq N_0 \sum_{i=1}^{M} H(p_i) \equiv R_0. \quad (6)$$

したがって，R_0 は転置ファイルの大きさの理論的下限値である．ここですべてのキーワードベクトルが等確率 p のベルヌイ分布に従うと仮定すれば

$$R_0 = N_0 M H(p), \quad (7)$$

が成り立つ．復号・検索は M_q $(M_q \leq M)$ 個のキーワードを指定し，これらすべてのキーワードを含む文献を求める AND 検索を仮定する．復号手順は次のとおりである．

① 検索語として指定した M_q 個のキーワードの圧縮したキーワード・ベクトル $\vec{A}_{i_k}$ $(k = 1, 2, \ldots, M_q)$ を選択し別々に復号する．

② 次に，M_q 個の復号した長さ N_0 のキーワード・ベクトルの要素ごとの AND をとり長さ N_0 のベクトル $\vec{U}$ を得る．

$$\vec{U} = \vec{A}_{i_1} \cap \vec{A}_{i_2} \ldots \cap \vec{A}_{i_{M_q}}. \quad (8)$$

③ 長さ N_0 のベクトル $\vec{U}$ 内で値1を持つ要素が求める文献に対応する．

それぞれの過程における復号計算量を評価する．①の計算量についてはハフマン符号のような木を用いて復号する場合，復号の計算量は圧縮ファイルの大きさに比例する[15]．この場合には検索キーワード数 M_q 個のキーワードベクトルを復号する必要がある．すべてのキーワードベクトルが等確率 p のベルヌイ分布に従うと仮定すれば，復号の計算量 $C_{0,a}$ について次式が成り立つ．

$$C_{0,a} \geq \alpha N_0 M_q H(p). \quad (9)$$

② で AND をとる計算量 $C_{AND,0}$ はキーワードベクトル長 N_0 とキーワード数 M_q に比例する．したがって，

$$C_{AND,0} = \beta N_0 M_q, \quad (10)$$

が成り立つ．③ ではベクトル $\vec{U}$ の各要素が0か1かを調べる．この計算量はベクトル $\vec{U}$ の長さ N_0 に比例するので

$$C_{0,b} = \gamma N_0, \quad (11)$$

と書ける．α，β，γ は1要素あたりの計算量を表す．① と ③ は各要素が0か1かを判定する計算量であるので，① と ③ を一緒にして評価する．比例係数は等しい（$\alpha = \gamma$）と仮定すると次式が成り立つ．

$$C_{0,a+b} \geq \alpha N_0 \{1 + M_q H(p)\} \equiv C_0. \quad (12)$$

2.2.2 2段階圧縮・復号方式

提案アルゴリズムが2段階なので従来のアルゴリズム[10]を2段階の場合に限定する．まず，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する．サブブロックが全0ならば値0を，非全0ならば値1を割り当てて長さ N_0/N のベクトルをつくる．圧縮ベクトルから元のベクトルを一意に復号するために長さ N の非全0のサブブロックの集合を記憶する．さらに，非全0のサブブロックのキーワード出現数に応じてサブブロックを圧縮する．

$$k_0 = \lfloor N/\lceil \log_2 N_0 \rceil \rfloor, \quad (13)$$

とおいて（$\lfloor x \rfloor$ は x 以下の最大の整数）

(i) サブブロックのキーワード出現数 k が $k \leq k_0$ を満たすならば値1を持つ要素の位置を $k\lceil \log_2 N_0 \rceil$ ビットに圧縮する．

(ii) $k > k_0$ ならば長さ N のままで圧縮しない．

文献10)では(i)の場合をさらに場合分けして圧縮率を改善することを提案しているが，ここでは簡単のためにこの場合分けは行わない．また，非全0のサブブロック各々をさらにサブブロックに分割して上の操作を繰り返し1段目と2段目のベクトルと非全0のサブブロックを記憶すれば，3段階の圧縮アルゴリズムとなる．この過程を繰り返して，さらに多段階の圧縮アルゴリズムを考えることもできる．

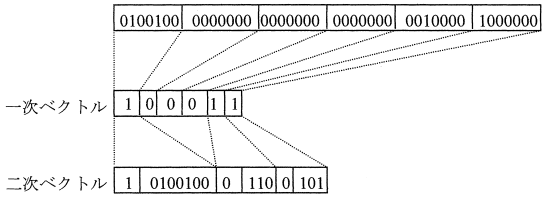
3. 提案アルゴリズムと定式化

3.1 提案アルゴリズム

3.1.1 圧縮アルゴリズム

（キーワードベクトルの分割）

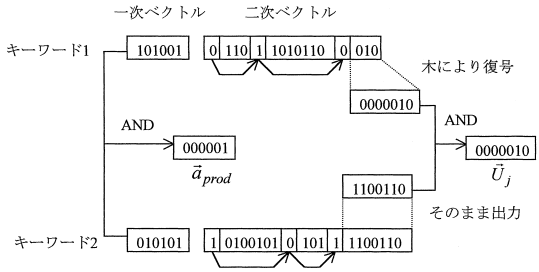
文献数 N_0 のキーワードベクトルを長さ N のサブブロックに分割する．N_0 がサブブロック長 N で割り切れない場合には，キーワードベクトルの長さを N で割り切れる最小の整数 $\acute{N}_0$ まで増加し，等長 N のサブブロックに分割する．その場合には，長さ $\acute{N}_0$ のキーワードベクトルの N_0+1 番目から $\acute{N}_0$ 番目までの要素はすべて0であり，実際の文献に対応しない．5章ではこのことを考慮して数値計算を行う．サブブ



(7,4,3) ハミング符号のシンドロームに圧縮。

圧縮率： 0.52

図 1 圧縮過程の例（$N_0 = 42,\ N = 7,\ n = 6$）
Fig. 1 An example of compression ($N_0 = 42$, $N = 7$, $n = 6$).



7·5+6＝41番目の要素が検索条件を満たす文献に対応する。

図 2 復号・検索過程の例（$M_q = 2,\ N_0 = 42,\ N = 7,\ n = 6$）
Fig. 2 An example of decoding and retrieval ($M_q = 2$, $N_0 = 42$, $N = 7$, $n = 6$).

ロックの数を n とする．i 番目のキーワードベクトルの j $(j = 1, 2, \ldots, n)$ 番目のサブブロックをベクトル $\vec{B}_{ij}$ で表す．キーワードベクトルを 2 段階で圧縮するが，第 1 段階および第 2 段階で圧縮されるベクトルを各々一次ベクトル，二次ベクトルと呼ぶ．**図 1** に長さ $N_0 = 42$ のキーワードベクトルを (7,4,3) ハミング符号のシンドロームを利用して圧縮する例を示した．

（第 1 段階の符号化）

サブブロックのすべての要素が 0 ならば 0，そうでなければ 1 としてキーワードベクトルから長さ n の一次ベクトル $\vec{a}_i$ をつくる．

$$\vec{a}_i = (a_{i1}, a_{i2}, \ldots, a_{in}) \in \{0,1\}^n. \qquad (14)$$

（第 2 段階の符号化）

サブブロック $\vec{B}_{ij}$ 内でのキーワード出現数 k が 0 でない場合のみ，サブブロックを符号化する．

① $1 \le k \le T$ のとき：シンドローム情報源符号化法により圧縮する．

② $k \ge T+1$ のとき：長さ N のまま記憶する（非圧縮）．

なお，非全 0 のサブブロックが圧縮されているか（①）非圧縮のままか（②）を区別するためにフラグを用いる．圧縮したサブブロックまたは非圧縮のサブブロックの先頭にそれぞれ値 0 または 1 を付ける．一次ベクトル内で $a_{ij} = 1$ を満足する j について j の増加する順番に，フラグを含めて圧縮または非圧縮のサブブロックをつないだものが二次ベクトルである．ここで，シンドローム情報源符号化とは長さ N のサブブロックをパラメータ（N，K，D）の誤り訂正符号の長さ $N-K$ のシンドロームに圧縮することを指す．ここで，N は符号長，K は情報記号数，D は最小距離である．シンドロームは，通信路で生じる誤り語を推定するために用いる[16]．誤り語の値 1 の個数が符号の誤り訂正能力 $T = \lfloor (D-1)/2 \rfloor$ 以下ならば，シンドロームと誤り語は 1 対 1 に対応するので，正しく推定できる．この性質を利用して長さ N のサブブロックのキーワード出現数が符号の誤り訂正能力 T 以下ならば，長さ $N-K$ のシンドロームに圧縮する．サブブロックのキーワード出現数が $T+1$ 以上ならば圧縮しない．ベルヌイ・モデルによれば，キーワード出現数が大きくなるにつれてキーワード数が少なくなるので，提案方式により効率的に圧縮できると予想される．非圧縮と圧縮のサブブロックを合わせた平均の圧縮率が最適になる符号の最小距離 D が存在すると考えられる．シンドローム情報源符号化の際には長さ N の二元系列に $(N-K) \times N$ のパリティ検査行列 H の転置行列 H^T をかければよいだけなので（$\vec{s} = \vec{B}_{ij} H^T$，$\vec{B}_{ij}$：$i$ 番目のキーワードの j 番目のサブブロック，$\vec{s}$：シンドローム情報源符号化したベクトル），転置ファイルの圧縮の手間が小さい利点を持つ．ここで，$\vec{B}_{ij}$ と $\vec{s}$ は圧縮前と圧縮後のサブブロックに対応する．

3.1.2 復号・検索アルゴリズム

全キーワード数 M の中から M_q 個のキーワードが検索語として与えられたときに，これらのキーワードに対する一次ベクトルおよび二次ベクトルを復号する．**図 2** に検索キーワード数が 2 の場合の復号・検索の例を示した．

（第 1 段の復号化）

M_q 個のキーワードの一次ベクトル $\vec{a}_{i_k}$ $(k = 1, 2, \ldots, M_q)$ の AND をとる．これを $\vec{a}_{prod} \in \{0,1\}^n$ によって表す．

（第 2 段の復号化）

① ベクトル $\vec{a}_{prod}$ の第 j 番目の要素を $a_{prod,j}$ として，$a_{prod,j} = 0$ ならば j 番目のサブブロックについての第 2 段階の復号を行わない．

② $a_{prod,j} = 1$ が成り立つならば，M_q 個のキーワードについて j 番目のサブブロックを二次ベクトルより

復号する．j 番目のサブブロックの二次ベクトル内での順番を $c_{i_k,j}$ $(k=1,2,\ldots,M_q,\ j=1,2,\ldots,n)$ とする．$c_{i_k,j}$ は j と必ずしも一致しない．よって，M_q 個の一次ベクトルに対する数値 $c_{i_k,j}$ の $M_q \times n$ の表をあらかじめつくって復号時に利用する．二次ベクトルの先頭の要素から次の過程を $c_{i_k,j}-1$ 回繰り返すことにより，復号対象のサブブロックまで各サブブロックの先頭に付けられたフラグをたどる．

(i) 値が 0 ならば $N-K+1$ ビット先へ飛ぶ．

(ii) 値が 1 ならば $N+1$ ビット先へ飛ぶ．

たどりついた要素は復号対象のサブブロックに付けられたフラグを表す．フラグが値 1 であるならば，サブブロックはシンドローム情報源符号化法により圧縮されているので，木を用いて復号する．値 0 であるならば，長さ N のまま記憶されているので続く N ビットをそのまま出力する．こうして復号したベクトル $\vec{B}_{i_k,j}$ を M_q 個のキーワードに対して AND をとり，ベクトル $\vec{U}_j$ を得る：

$$\vec{U}_j = \vec{B}_{i_1,j} \cap \vec{B}_{i_2,j} \ldots \cap \vec{B}_{i_{M_q},j}. \tag{15}$$

ベクトル U_j 内で l 番目の要素が値 1 を持つならば，元のキーワードベクトルの $N(j-1)+l$ 番目の要素が求める文献に対応する．以上の過程を $a_{prod,j}=1$ を満足するすべての要素 j に対して行う．AND 検索を行う場合には，一次ベクトルの AND をとったベクトル $\vec{a}_{prod}$ の値 1 を持つ要素についてのみ二次ベクトルの復号を行う．すべての二次ベクトルを復号する必要はないので，1 段階で圧縮・復号を行う場合より復号の計算量が低減する．一方で，OR 検索，NOT 検索を行う場合にはすべてのサブブロックを復号しなければならないので 1 段階の場合に比べて計算量は低減しない．したがって，提案アルゴリズムは AND を含む検索式に対してのみ有効になる．

3.2 定 式 化

3.2.1 メモリ量

提案アルゴリズム

この節では，簡単のために全文献数 N_0 がサブブロック長 N で割り切れる場合を取り扱う．メモリ量は転置ファイルの大きさと復号木の大きさの和である．転置ファイルの大きさは一次ベクトルと二次ベクトルの大きさの和になる．一次ベクトルの長さはサブブロック数 n である．二次ベクトルは圧縮したサブブロックまたは非圧縮のサブブロックを先頭にフラグを付けてつないだものである．圧縮および非圧縮のサブブロックの個数は，1 つのキーワードベクトルあたり各々 nq_1 と nq_2 により与えられる．ここで n は長さ N のサブブロックの総数であり，q_1 と q_2 はサブブロックのキーワード出現数が 1 以上 T 以下である確率，および $T+1$ 以上である確率を表す．T は符号の誤り訂正能力である．q_1 と q_2 は以下で与えられる：

$$q_1 = \sum_{i=1}^{T} \binom{N}{i} p^i \{1-p\}^{N-i}. \tag{16}$$

$$q_2 = \sum_{i=T+1}^{N} \binom{N}{i} p^i \{1-p\}^{N-i}. \tag{17}$$

圧縮および非圧縮のサブブロックの長さはそれぞれ $N-K$ と N であるので，両者の和は $nq_1(N-K)+nq_2N$ で与えられる．フラグの長さの和は $n(q_1+q_2)$ により表せる．以上より二次ベクトルの長さは以下で与えられる．

$$nq_1(N-K)+nq_2N+n(q_1+q_2). \tag{18}$$

式 (18) に一次ベクトルの長さ n を加えて全キーワード数 M を乗じて圧縮した転置ファイルの大きさを得る．一方で，復号木の大きさは葉ノードの数と葉に割り当てられた非圧縮のサブブロックの長さの積で表される[15]．葉ノードの総数は 2^{N-K}，非圧縮のサブブロック長は N なので復号木の大きさは $N2^{N-K}$ で与えられる．圧縮ファイルと木を合わせたメモリ量 R_2 は次式で与えられる．

$$R_2 = M\{n+nq_1(N-K)+nq_2N + n(q_1+q_2)\}+N2^{N-K}. \tag{19}$$

従来アルゴリズム

提案アルゴリズムとの比較を容易にするために提案アルゴリズムと同様の枠組みに従って従来の 2 段階アルゴリズムを評価する．従来アルゴリズムでは復号木を用いないのでメモリ量は圧縮した転置ファイルの大きさに等しい．圧縮した転置ファイルの大きさは一次ベクトルと二次ベクトルの和になる．サブブロックのキーワード出現数 k によって 2.2.2 項に示すように記憶の方式が異なるので，提案アルゴリズムと同様に両者を区別するために非全 0 のサブブロックの先頭に 1 ビットのフラグを付けることを考える．一次ベクトルとフラグの評価式は提案アルゴリズムと同じである．二次ベクトルの長さは 2.2.2 項 (i)，(ii) の場合に分けて評価する．ベルヌイモデルによればキーワード出現数が k であるサブブロック数は式 (4) で与えられる．キーワード出現数 k $(1 \le k \le k_0)$ のサブブロック内の値 1 を持つ要素の位置を記憶するには $k\lceil \log_2 N_0 \rceil$ ビットが必要なので，2.2.2 項 (i) の方式で圧縮されるサブブロックの長さの和は

$$n\lceil\log_2 N_0\rceil\sum_{k=1}^{k_0}\binom{N}{k}kp^k\{1-p\}^{N-k}, \quad (20)$$

となる．一方で，2.2.2 項 (ii) の方式で記憶するサブブロックの長さの和は nq_2N となることを考慮して，メモリ量 R_1 として次式を得る．

$$R_1 = M\left\{n + n\lceil\log_2 N_0\rceil\sum_{k=1}^{k_0}\binom{N}{k}kp^k\{1-p\}^{N-k} + nq_2N + n(q_1+q_2)\right\}. \quad (21)$$

ここで，q_1，q_2 は次式で表せる．

$$q_1 = \sum_{k=1}^{k_0}\binom{N}{k}p^k\{1-p\}^{N-k}. \quad (22)$$

$$q_2 = \sum_{k=k_0+1}^{N}\binom{N}{k}p^k\{1-p\}^{N-k}. \quad (23)$$

3.2.2 計 算 量

提案アルゴリズム

提案アルゴリズムは 2 段階過程であり，かつ検索過程を含むので複雑である．検索の効率を表す量として検索時間を考える．文献 17) を参考にして，検索時間 t_0 を次のように表す．

$$t_0 = t_1 + t_2. \quad (24)$$

ここで時間 t_1 はディスクアクセス時間である．時間 t_1 は圧縮ファイルの大きさに比例するので，圧縮を行った場合のディスクアクセス時間 t_1 は圧縮を行わない場合より短い．よって，検索時間 t_0 も圧縮を行わない場合より短くなりうる．時間 t_2 は圧縮ファイルの復号・検索に要する時間である．本項では，時間 t_2 の指標として提案アルゴリズムの復号・検索の計算量を ① 圧縮ベクトルの各要素が 0 か 1 かを判定するのに要する計算量，② 圧縮ベクトルの AND，あるいは要素の値の和算に要する計算量に分けて評価する．

① の計算量は次の 4 つよりなる．

(i) ベクトル $\vec{a}_{prod}$ の各要素の値を判定する計算量．これはベクトル長 n に比例する．比例係数を α として計算量 C_{2a} は

$$C_{2a} = \alpha n, \quad (25)$$

と表せる．

(ii) 二次ベクトル内のサブブロックの先頭に付けられたフラグの値を判定しながら，復号対象のサブブロックまでフラグをたどる計算量．この計算量 C_{2b} は，二次ベクトル内のフラグの個数の平均値 n_{flag} と検索キーワード数 M_q に比例するので，以下のように表せる．

$$C_{2b} = \alpha M_q n_{flag}. \quad (26)$$

(iii) 圧縮したサブブロックと復号木を照合する計算量．計算量は圧縮ベクトルの長さに比例する．二次ベクトル内でシンドローム長 $N-K$ に圧縮したサブブロックの数を n_1 とすれば，計算量 C_{2c} は以下のように書ける．

$$C_{2c} = \alpha M_q n_1(N-K). \quad (27)$$

(iv) サブブロックの AND をとったベクトル $\vec{U}_j$（式 (15)）の各要素の値を判定する計算量．計算量 C_{2d} は次のように表せる：

$$C_{2d} = \alpha nN\{q_1+q_2\}^{M_q}. \quad (28)$$

ここで，$\{q_1+q_2\}^{M_q}$ は AND をとったベクトル $\vec{a}_{prod}$ の要素 j $(1\le j\le n)$ が値 1 を持つ（$a_{prod,j}=1$）確率を表す．(ii)，(iii) の n_{flag} および n_1 は次のように書ける：

$$n_{flag} = n(q_1+q_2). \quad (29)$$

$$n_1 = nq_1\{q_1+q_2\}^{M_q-1}. \quad (30)$$

式 (29)，(30) を考慮に入れて復号の計算量 C_{2a}〜C_{2d}（式 (25)〜(28)）の和をとり，合計の復号・検索の計算量 C_2 を得る．

$$\begin{aligned}C_2 &= C_{2a}+C_{2b}+C_{2c}+C_{2d}\\ &= \alpha M_q\{n/M_q + n(q_1+q_2)\\ &\quad + nq_1\{q_1+q_2\}^{M_q-1}(N-K)\\ &\quad + nN\{q_1+q_2\}^{M_q}/M_q\}.\end{aligned} \quad (31)$$

次に，ベクトルの AND および要素の値の和算の計算量は以下のように評価する．M_q 個の一次ベクトルの AND をとりベクトル $\vec{a}_{prod}$ をつくる．この計算量は比例係数を β として

$$C_{AND,2a} = \beta M_q n, \quad (32)$$

と表せる．式 (15) の演算は $n\{q_1+q_2\}^{M_q}$ 回行う．式 (15) のベクトル $\vec{U}_j$ を得る計算量は次式で与えられる．

$$C_{AND,2b} = \beta M_q nN\{q_1+q_2\}^{M_q}. \quad (33)$$

また，$c_{i_k,j}$ $(1\le k\le M_q, 1\le j\le n)$ の $M_q\times n$ の表をつくる計算量は比例係数を $\acute{\beta}$ として

$$C_{AND,2c} = \acute{\beta} M_q n, \quad (34)$$

と書ける．式 (32)〜(34) より $\beta=\acute{\beta}$ を仮定すれば次式を得る．

$$\begin{aligned}&C_{AND,2}\\ &\quad = C_{AND,2a}+C_{AND,2b}+C_{AND,2c}\\ &\quad = \beta M_q\{2n + nN\{q_1+q_2\}^{M_q}\}.\end{aligned} \quad (35)$$

従来アルゴリズム

提案アルゴリズムと同じく，① 圧縮ベクトルの各要素が 0 か 1 かを判定するのに要する計算量と ② ベクトルの AND・要素の値の和算の計算量に分けて評価する．提案アルゴリズムと同じく，① の計算量は前

出の (i)〜(iv) の 4 つに分けて評価する．(i)〜(iv) の計算量を各々 C_{1a}，C_{1b}，C_{1c}，C_{1d} と表すと

$$C_{1a} = C_{2a}.$$
$$C_{1b} = C_{2b}.$$
$$C_{1d} = C_{2d}. \quad (36)$$

が成り立つ．

従来アルゴリズムではサブブロックのキーワード出現数 k が $k \leq k_0$ を満たす場合，値 1 を持つ要素の位置を表すベクトルを復号するが，途中からの復号はできない．このベクトルの長さは式 (20) で与えられるので，復号の計算量 C_{1c} は次式で与えられる．

$$C_{1c} = \alpha M_q n \lceil \log_2 N_0 \rceil \sum_{k=1}^{k_0} \binom{N}{k} k p^k \{1-p\}^{N-k}. \quad (37)$$

以上のことを考慮して合計の復号の計算量 C_1 を得る．

$$\begin{aligned} C_1 &= C_{1a} + C_{1b} + C_{1c} + C_{1d} \\ &= \alpha M_q \{n/M_q + n(q_1+q_2) \\ &\quad + n\lceil \log_2 N_0 \rceil \sum_{k=1}^{k_0} \binom{N}{k} k p^k \{1-p\}^{N-k} \\ &\quad + nN\{q_1+q_2\}^{M_q}/M_q\}. \end{aligned} \quad (38)$$

次に，ベクトルの AND および要素の値の和算の計算量は提案アルゴリズムと同じ式で与えられる．

$$C_{AND,1} = \beta M_q \{2n + nN\{q_1+q_2\}^{M_q}\}. \quad (39)$$

4. 転置ファイルの分析

本論文でベルヌイ・モデルを用いて解析することの正当性を確かめるために，**表 1** に示す条件で転置ファイルを作成してその特性を調べた．各文献に出現するすべてのキーワードに対してキーワードを含む文献番号のリスト（キーワードベクトルに相当する）を作成した．文献には筆頭著者のアルファベット順に番号を割り当てた（全文献数 $N_0 = 1793$）．同じ著者による文献が近い主題を持つと考えると近い番号の文献が近い主題を持つことが予想される．キーワード出現数が 1 であるキーワードは 3344 個であり，異なりキーワード数（4,460），延べキーワード数（8,047）に対する割合は各々0.75 および 0.42 である．キーワード出現数が 1 であるキーワードが以下の分析に与える影響が大きくなることを避けるために，最初の単語が同じキーワード，および最初の単語の語尾は異なるが同じ意味を持つキーワードを 1 つにまとめた．すると，キーワード出現数が 1 であるキーワードの数は 607 に減少し，異なりキーワード数（1,360）および延べキーワード数（7,789）に占める割合はそれぞれ 0.45 と 0.08 に減少した．表 1 の括弧内の数字はキーワードをまとめる前に対応し，括弧の前の数字はキーワードをまとめた後に対応する．こうして作成した転置ファイルの 1）キーワードの頻度分布，2）リスト内の文献番号の差分値の分布を調べた．

1）キーワードの頻度分布

i 番目のキーワードの頻度 $f_i = n_i/N_0$ を計算する．ここで，n_i は i 番目のキーワードのキーワード出現数である．頻度 f_i の大きな順に i 番目のキーワードに順位 r_i $(1 \leq r_i \leq N_0)$ を付ける．同じ f_i 値を持つキーワードが複数ある場合には異なる順位を付ける．**図 3** には順位 r_i と頻度 f_i の関係を示した．なお，参考のために Zipf の法則として知られる順位と頻度の関係を示した．Zipf の法則は文書中での単語の出現順位 r と出現頻度 $f(r)$ の間に成り立つ次の関係式である[18]：

$$f(r) = C/r. \quad (40)$$

高頻度出現語に対して $C \sim 0.1$ が成り立つ[6]．式 (40)

表 1 転置ファイル作成条件
Table 1 Condition of an inverted file.

雑誌名：IEEE Transactions on information theory	
年度：	1993 年 No. 1－2000 年 No. 1
論文数：1793	
異なりキーワード数：1360(4460)	
延べキーワード数：7789(8047)	
1 論文当たりの平均キーワード数：4. 3(4. 5)	
1 キーワードが文献に含まれる割合：	0. 0032(0. 0010)

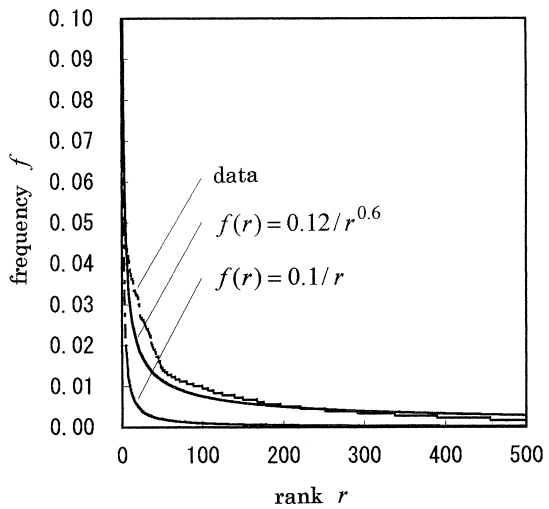


図 3 キーワードの出現順位 r と出現頻度 $f(r)$ の関係（最適曲線：$f(r) = 0.12/r^{0.6}$，Zipf's law：$f(r) = 0.1/r$）
Fig. 3 Relation between rank r and frequency $f(r)$ of a keyword.

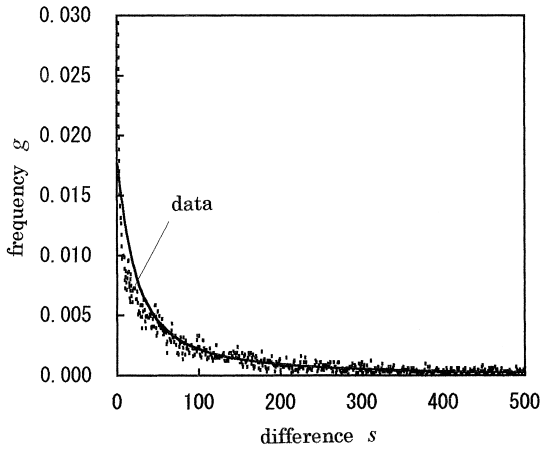


図 4 転置ファイル内の値 1 の差分値 s と頻度 $g(s)$ の関係
Fig. 4 Relation between difference s of the binary value 1 and frequency $g(s)$.

は索引語の分布についてもあてはまることが報告されているが[8]，本例にはあまり適合しない．そこで，

$$f(r,s) = \acute{C}/r^s, \tag{41}$$

$$t = \sum_{i=1}^{N_0} \{f(r,s) - f_i\}^2, \tag{42}$$

とおき，t を最小にするパラメータ $\acute{C}$，s を求めたところ

$$s = 0.6 \quad \acute{C} = 0.12. \tag{43}$$

を得た．この曲線を図 3 に示したが，データの示す傾向をおおまかに説明する．

2）差分値の分布

図 4 にはキーワード リスト内の文献番号の差分値 s $(1 \le s \le N_0)$ の頻度分布 $g(s)$ を示した．これをキーワードの頻度分布を考慮したベルヌイ分布 $H(s)$ と比較する．i $(1 \le i \le M)$ 番目のキーワード ベクトル内での値 1 が生起する確率が上の 1）で求めた確率 f_i のベルヌイ分布に従うと仮定する．すると，差分値 s $(1 \le s \le N_0)$ の出現頻度 $h_i(s)$ は次式で与えられる：

$$h_i(s) = \alpha_i f_i \{1 - f_i\}^{s-1}. \tag{44}$$

ここで $\sum_{s=1}^{N_0} h_i(s) = 1$ となるように正規化すると

$$\alpha_i = 1/\{1 - \{1 - f_i\}^{N_0}\}, \tag{45}$$

が成り立つ．分布 $h_i(s)$ をすべてのキーワードについて和をとって（$1 \le i \le M$）転置ファイル全体の差分値 s の頻度分布 $H(s)$ を得る．

$$H(s) = (1/M) \sum_{i=1}^{M} h_i(s). \tag{46}$$

式 (46) によって得られる曲線を図 4 に示した．曲線は差分値の小さいところでデータから少しはずれるがデータの示す傾向をおおまかに説明する．個々のキーワード ベクトル内では値 1 がベルヌイモデルに従って生起するとしてかまわない．すなわち，個々のキーワード ベクトル内ではベルヌイ・モデルの仮定は正当化される．

5. 結　　果

5.1 数値計算の条件

本論文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，次の $3 \times 3 \times 5 = 45$ 通りのパラメータに対してメモリ量と復号の計算量の関係を評価した．

$$p = 0.01, 0.001, 0.0001.$$
$$M_q = 2, 4, 6.$$
$$(N_0, M) = (10^4, 10^3), (10^4, 10^4), (10^5, 10^3), (10^5, 10^4), (\infty, 10^3). \tag{47}$$

JICST 文献検索ファイルでは 1 記事あたりに統制語キーワードが $M_a = 8$〜9 個与えられていて，統制語キーワードの総数は $M \cong 34,000$ である[19]．転置ファイル内での値 1 の平均生起確率は $\bar{p} = M_a/M = 2.5 \times 10^{-4}$ となる．また，転置ファイルの全文献数 N_0 は収録文献の分野により異なり $N_0 = 10^3$〜10^7 の範囲である[20]．転置ファイルの大きさは $N_0 M = 10^7$〜10^{11} ビットの範囲になる．また，Moffat らは 3 種類の文書の各章に出現する単語を列挙して，各単語がどの章に出現するのかを示す二次元の表を作成して，この表の圧縮について論じた[3]．この表を転置ファイルと見なし，N_0 を章の総数，M を全単語数，$\bar{p}$ を値 1 の平均出現頻度とすると，これらのパラメータの値は次のように与えられる：

$$2 \cdot 10^3 \le N_0 \le 3 \cdot 10^5, \quad 2 \cdot 10^4 \le M \le 7 \cdot 10^4,$$
$$5 \cdot 10^{-4} \le p \le 5 \cdot 10^{-3}. \tag{48}$$

式 (47) のベルヌイ確率 p，全文献数 N_0，全キーワード数 M，検索キーワード数 M_q の値は，JICST 文献検索ファイル，Moffat らの実験で用いられた値，および本論文の表 1 のデータを参考にして設定した．なお，全文献数 N_0 または全キーワード数 M が大きくなるにつれて復号木の大きさがメモリ量に占める割合が減少するので，従来アルゴリズムより提案アルゴリズムが有利になる．そこで，数値計算は全文献数 N_0 と全キーワード数 M が比較的小さい場合（$N_0 = 10^4, 10^5$，$M = 10^3, 10^4$）と漸近的場合（$N_0 = \infty$）について行った．検索キーワード数 M_q は通常小さいと考えられるので，$M_q = 2, 4, 6$ に設定した．サブブロック長 N の値は 1 刻みで変化させた．ちなみに，従来の

多段階の圧縮実験では計算を容易にするためにサブブロック長 N をバイト単位にした（$N = 8, 16, 32$ [10]，$N = 32$ [11]）．

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符号長 N，情報記号数 K，最小距離 D の線形符号が存在することが知られている（存在定理[16]）．

$$2^{N-K} > \sum_{j=0}^{D-2} \binom{N-1}{j}. \tag{49}$$

そこで，数値計算ではサブブロック長を符号長にとり，式 (49) を満足する符号のパラメータに対して 2.2.1 項と 3.2 節で導いた

R_2/R_0, R_1/R_0, C_2/C_0, C_1/C_0, $C_{AND,2}/C_{AND,0}$

を求めメモリ量比 R_2/R_0 と復号計算量比 C_2/C_0 の関係の特性を調べ，提案アルゴリズムの有効性を示す．

5.2 評価結果

5.2.1 トレード・オフ関係

図 5 にメモリ量と計算量の関係の例を示す．グラフの横軸は提案アルゴリズムのメモリ量 R_2（式 (19)）を 1 段階のアルゴリズムのメモリ量 R_0（式 (7)）で正規化し，グラフの縦軸は提案アルゴリズムの復号・検索の計算量 C_2（式 (31)）を 1 段階のアルゴリズムの計算量 C_0（式 (12)）で正規化している．2.2.1 項で述べたように，R_0 は圧縮ファイルの大きさの理論的下限値なので $R_2/R_0 > 1$ が成り立つ．C_0 は 1 段階で復号する際の復号・検索の計算量の下限値である．

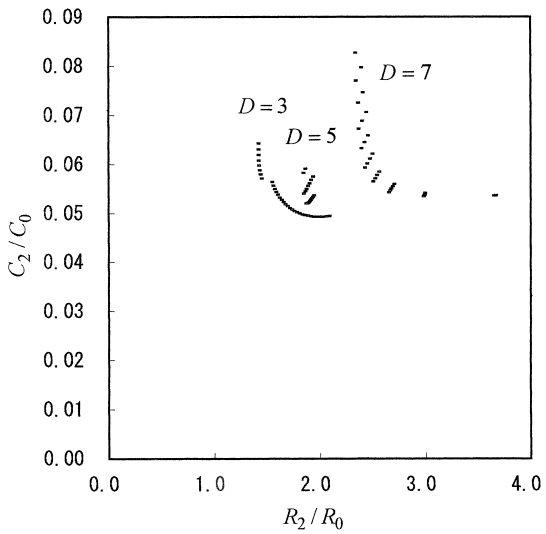


図 5 メモリ量比 R_2/R_0 と計算量比 C_2/C_0 のトレード・オフ関係（$p = 10^{-2}$，$M_q = 6$，$N_0 = 10^5$，$M = 10^4$）
Fig. 5 Trade-off relation between R_2/R_0 and C_2/C_0 ($p = 10^{-2}$, $M_q = 6$, $N_0 = 10^5$, $M = 10^4$).

図 5 のグラフ上で原点方向に向かうにつれて提案アルゴリズムの性能が良くなる．グラフの左下方向に沿って一番外側にある点をつないだ曲線はトレード・オフ関係を示すので，これをトレード・オフ曲線と呼ぶ．図 5 には符号の最小距離 D が 3，5，7 の場合の結果を示したが $D = 3$ の場合が最も良い曲線を与える．トレード・オフ曲線はグラフ上で左上から右下方向に延びているので，メモリ量比と計算量比の上限のうち一方を改善すればもう一方が悪化することを示す．数値計算された 45 通りのすべての場合についてメモリ量比 R_2/R_0 と復号・検索の計算量比 C_2/C_0 の間にトレード・オフの関係をみることができる．トレード・オフ曲線の特徴として以下の点をあげることができる：

(i) 計算したすべての場合についてトレード・オフ曲線上の点は $10^{-3} \le C_2/C_0 \le 10^{-1}$ および $10^{-3} \le C_{AND,2}/C_{AND,0} \le 0.15$ の範囲にある．要素の値を判定する計算量，AND および和算の計算量ともに 1 段階のアルゴリズムより著しく低減される．また，計算量比最小の端点は $1 < R_2/R_0 < 1000$ の範囲にあるが，メモリ量比最小の点は $1 < R_2/R_0 < 10$ の範囲にある．すなわち，トレード・オフ曲線のメモリ量比最小の点の近傍ではメモリ量を理論的限界値の 10 倍以内におさえて，1 段階アルゴリズムに比べて計算量を著しく低減できる特徴を持つ．また，従来の 2 段階アルゴリズムのメモリ量比 R_1/R_0 と計算量比 C_1/C_0 もトレード・オフ関係を示す．

(ii) 左上端の点から右下端の点にかけてサブブロック長 N の値が単調に変化する．計算した 45 通りのうち 12 通りの場合は，曲線の右下端から左上端にかけてサブブロック長 N が単調に増加するが，他の 33 通りの場合は逆に左上端から右下端にかけてサブブロック長 N が増す．また，トレード・オフ曲線の両端点の距離は検索キーワード数 M_q が大きい（$M_q = 6$）ときに大きくなる．

(iii) トレード・オフ曲線は下に凸である．とくに，メモリ量比最小の端点の近傍ではメモリ量のわずかな増加に対して復号・検索の計算量の低減量が大きく実用上有効である．

(iv) トレード・オフ曲線を与えるサブブロック長 N は計算したすべての場合について $(0.01 \sim 0.5)/p$ の範囲にある．符号の最小距離 D は全文献数 $N_0 = \infty$ の場合に 3 または 5 であるが，転置ファイルの大きさ（$N_0 M$）が有限であるすべての場合には 3 となる．すなわち，符号長はそのうちに値 1 がせいぜい 1 つ程度含まれる程度の長さにとり，最小距離 D が 3 ないし 5 の符号を用いるときに最適になる．転置ファイルの大

きさが有限のときには復号木の大きさを無視できない．復号木の大きさは 2^{N-K} に比例するが，VG 限界式（式 (49)）を満たす 2^{N-K} の値は符号の D 値とともに急増する．すなわち，小さいシンドローム長 $N-K$ を与える最小距離 $D=3$ の符号が有効である．一方で，転置ファイルの大きさが無限のとき（$N_0M=\infty$）には，木の大きさは転置ファイルの大きさに比べて無視できるので，転置ファイルを効率良く圧縮できる最小距離 $D=3$ または $D=5$ を満たす符号が有効になる．

(v) 図 5 の最小距離 $D=3,5,7$ の曲線がそれぞれ途切れるところで，符号のシンドローム長 $N-K$ が変化する．シンドローム情報源符号化の効率（圧縮率）$(N-K)/N$ が不連続に変化するために横軸のメモリ量比 R_2/R_0 が不連続に変化することが曲線が途切れる原因である．

5.2.2 トレード・オフ曲線の転置ファイルの大きさに対する依存性

図 6 にはトレード・オフ曲線の両端点のメモリ量比 R_2/R_0 と計算量比 C_2/C_0 を全文献数 N_0 に対して示した（ベルヌイ確率 $p=10^{-3}$，検索キーワード数 $M_q=2$，全キーワード数 $M=10^3$）．全文献数 N_0 が 10^4 程度（$N_0M=10^7$）より小さい場合には，メモリ量比 R_2/R_0 比と計算量比 C_2/C_0 比ともに全文献数 N_0 とともに減少する．これはアルゴリズムの性能が全文献数 N_0 とともに良くなることを意味する．全文献数 N_0 が 10^4 程度（$N_0M=10^7$）の値を超えると両端点のメモリ量比 R_2/R_0 比と計算量比 C_2/C_0 比はほぼ一定になる．この領域では 6.2 節で述べるように提案アルゴリズムの従来アルゴリズムに対する優位性が保証される．同様の傾向は他の多くの計算例に見られた．

6. 考　　察

6.1 漸近的場合のトレード・オフ曲線の特性

付録に最小距離 $D=3$，全文献数 $N_0=\infty$ の場合についてメモリ量比が最小になる点の近傍でのトレード・オフ曲線の解析を示している．

① 付録より圧縮率の上界式 $\rho_{2,upper}$ はサブブロック長 $N\approx N_{min}=p^{-2/3}$ で最小値をとる．式 (58) は一次ベクトルと二次ベクトルの圧縮率の釣り合いによって全体の圧縮率が最小になる点が決まることを表す．$N<N_{min}$ の場合には一次ベクトルの圧縮率が全体のメモリ量を支配する．$N>N_{min}$ の場合には二次ベクトルの圧縮率が全体のメモリ量を支配する．$N<N_{min}$ と $N>N_{min}$ のどちらの場合も一次ベクトルの復号効率が全体の復号・検索の計算量を支配する．

② 付録の式 (59)，(60) よりサブブロック長 $N=N_{min}$ でメモリ量比 R_2/R_0 と計算量比 C_2/C_0 は以下のようになる．

$$R_2/R_0\approx -3/2p^{1/3}\log_2 p.$$
$$C_2/C_0=-1/\{M_qp^{1/3}\log_2 p-p^{-2/3}\}.$$

図 7 には検索キーワード数 $M_q=2$ の場合について厳密な計算結果と式 (59)，(60) に基づく近似計算

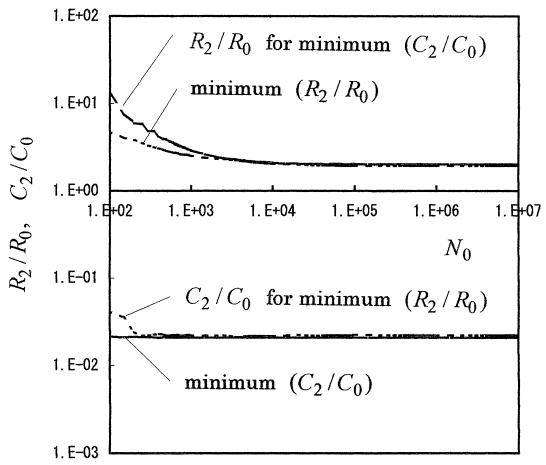


図 6 トレード・オフ曲線の両端点でのメモリ量比 R_2/R_0 と計算量比 C_2/C_0 の全文献数 N_0 依存性（$p=10^{-3}$，$M_q=2$，$M=10^3$）

Fig. 6 N_0 *v.s.* R_2/R_0 *or* C_2/C_0 on end points of the trade-off curve ($p=10^{-3}$, $M_q=2$, $M=10^3$).

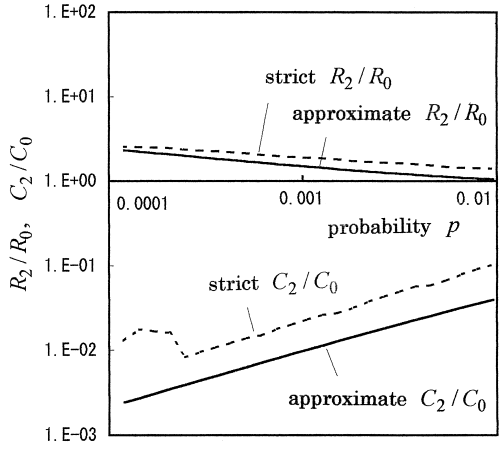


図 7 R_2/R_0 比最小の端点でのメモリ量比 R_2/R_0 と計算量比 C_2/C_0 のベルヌイ確率 p 依存性（$M_q=2$，$N_0=\infty$，$M=10^3$）

Fig. 7 p *v.s.* R_2/R_0 *or* C_2/C_0 for the minimum R_2/R_0 value ($M_q=2$, $N_0=\infty$, $M=10^3$).

結果を示した．近似計算結果はメモリ量については厳密な計算結果の良い近似になっているが，計算量についてはやや差がある．厳密計算および近似計算の結果によれば，メモリ量比はベルヌイ確率 p の減少関数となるが計算量比はベルヌイ確率 p の増加関数である．$p = 10^{-4}$ に近い場合に，厳密計算した計算量がベルヌイ確率 p の単調増加関数となっていないのは，この場合には最適な符号の最小距離が $D = 5$ であり，他の場合は $D = 3$ であるためである．検索キーワード数 $M_q = 4, 6$ の場合についても図 7 と同様の結果が得られる．ただし，近似計算した計算量比と厳密計算した計算量比の差は検索キーワード数 $M_q = 4, 6$ の場合の方が小さい．

③ メモリ量比が最小になる点の近傍では，メモリ量比 R_2/R_0—計算量比 C_2/C_0 の関係は，検索キーワード数 M_q が $M_q \ll p^{-1/3}$ を満たすならば，傾き $-\log_e 2/M_q p^{1/3}\{1 - 1/M_q p \log_2 p\}$ の直線になる（式 (66)）．

6.2 提案アルゴリズムの有効性

メモリ量がほぼ同じになる条件のもとで提案アルゴリズムと従来の 2 段階アルゴリズムの計算量を比較してコンピュータの CPU の進歩のデータと比較する．提案アルゴリズムと従来の 2 段階アルゴリズムのトレード・オフ曲線上で横軸の値がほぼ同じになる点での縦軸の値を比較した．提案アルゴリズムのトレード・オフ曲線上のメモリ量比が最小になる端点でのメモリ量比と計算量比を $(R_2/R_0)_{min}$ と $(C_2/C_0)_{min}$ とおく．従来の 2 段階アルゴリズムのトレード・オフ曲線上の点を $(R_1/R_0, C_1/C_0)$ とおき，

$$|R_1/R_0 - (R_2/R_0)_{min}| < 0.1(R_2/R_0)_{min}, \tag{50}$$

を満足する点のなかで C_1/C_0 比最小となる点を見い出す．$(R_2/R_0)_{min}$，R_1/R_0，$(C_2/C_0)_{min}$，C_1/C_0 の値から提案アルゴリズムと従来の 2 段階アルゴリズムのメモリ量比 R_2/R_1 と計算量比 C_2/C_1 を求めた．これらの比を検索キーワード数 $M_q = 2$，全文献数 $N_0 = 10^4$，全キーワード数 $M = 10^3$ の場合についてベルヌイ確率 p の関数として示す（**図 8**）．計算量比 C_2/C_1 比は 1 より小さく提案アルゴリズムの計算量は従来の 2 段階アルゴリズムの計算量より低減することが分かる．計算量比 C_2/C_1 比はベルヌイ確率 p とともに小さくなるので，ベルヌイ確率 p が大きいときのほうが提案アルゴリズムの優位性は明らかである．なお，図 8 でベルヌイ確率 p が 10^{-2} に近づくと式 (50) を満たす従来の 2 段階アルゴリズムと 1 段階のアルゴリズムの計算量比は $C_1/C_0 > 1$ となるので，

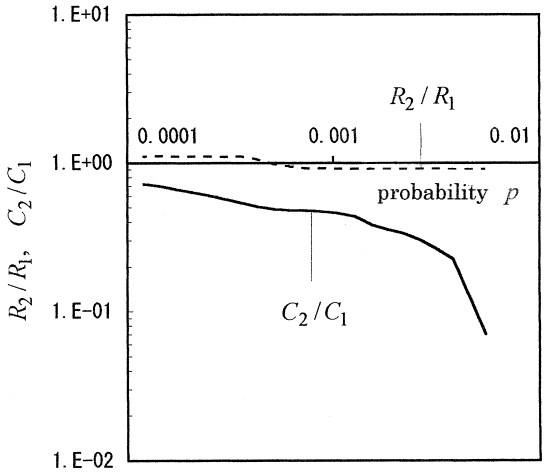


図 8 R_2/R_0 比最小の端点でのメモリ量比 R_2/R_1 と計算量比 C_2/C_1 のベルヌイ確率 p 依存性（$M_q = 2$, $N_0 = 10^4$, $M = 10^3$）

Fig. 8 p *v.s.* R_2/R_1 *or* C_2/C_1 for the minimum R_2/R_0 value ($M_q = 2$, $N_0 = 10^4$, $M = 10^3$).

図 8 から省いた．3 通りの検索キーワード数 M_q，2 通りの全キーワード数 M，2 通りの全文献数 N_0 の計 12 通りすべての場合についてベルヌイ確率 p のほぼ全範囲で計算量比 $C_2/C_1 < 1$ が成り立ち，計算量比 C_2/C_1 はベルヌイ確率 p とともに減少する．また，計算したすべての場合について計算量比 C_2/C_1 は 0.05〜1 の範囲にある．コンピュータの処理速度の目安としてマイクロプロセッサの動作周波数を考えると，高性能プロセッサでは 3 年に 2 倍，組み込み用では 2 年に 2 倍のペースで動作周波数が向上している[21]．計算速度がマイクロプロセッサの動作周波数に比例すると仮定すると，計算量の 0.05〜1 倍の低減量は最大十数年かけての向上に相当する．したがって，提案アルゴリズムの計算量の低減量は十分に有効である．

7. むすび

（1）メモリ量と復号・検索の計算量はトレード・オフ関係を示す．

（2）提案アルゴリズムの計算量は 1 段階の圧縮，復号・検索のアルゴリズムの計算量の 10 倍から 1000 倍低減する．

（3）ベルヌイ確率 p が大きくなるにつれて提案アルゴリズムの従来の 2 段階アルゴリズムに対する優位性は明らかになる．

（4）1 文献あたりのメモリ量，計算量は転置ファイルのサイズとともに減少する．すなわち，文献検索システムが大規模になるほど有利になる．

（5）提案アルゴリズムの計算量の 2 段階の従来アル

ゴリズムの計算量に対する低減量はマイクロプロセッサの最大十数年程度の進歩に相当するので，提案アルゴリズムは有効である．

今後の課題として，転置ファイルの圧縮，復号・検索の実験を行い，本論文の結果と比較したいと考えている．

謝辞 早稲田大学でご指導いただいた平澤茂一教授に深く感謝いたします．有益な助言をいただいた早稲田大学の松嶋敏泰教授，図書館情報大学の石川徹也教授，藤井敦博士，早稲田大学の平澤研究室，松嶋研究室の各氏に深く感謝いたします．

参 考 文 献

1) 石井啓豊：オンライン検索を利用する，情報管理，Vol.35, No.6, pp.519–534 (1992).
2) 高木義和：インターネットにおける情報検索，情報管理，Vol.38, No.10, pp.891–900 (1996).
3) Moffat, A. and Zobel, J.: Parameterized compression for sparse bitmaps, *Proc. 15th ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pp.274–285 (1992).
4) 情報処理学会編：情報処理ハンドブック, p.2000, オーム社 (1997).
5) Bell, T.C., Moffat, A., Nevill-Manning, C.G., Witten, I.H. and Zobel, J.: Data compression in full-text retrieval systems, *Journal of the American Society for Information Science*, Vol.44, No.9, pp.508–531 (1993).
6) 林 雅樹，小出東洋，武田正幸，松尾文碩：情報検索システムにおける高頻度キーワードの文書参照ファイルの圧縮について，情報処理学会第55回全国大会予稿集，Vol.3, pp.141–142 (1997).
7) Bookstein, A. and Klein, S.T.: Flexible compression for bitmap sets, *Proc. IEEE Data Compression Conference*, pp.402–410 (1991).
8) Schuegraf, E.J.: Compression of large inverted files with hyperbolic term distribution, *Information Processing and Management*, Vol.12, pp.377–384 (1976).
9) Jakobsson, M.: Evaluation of a hierarchical bit-vector compression technique, *Information Processing Letters*, Vol.14, No.4, pp.147–149 (1982).
10) Choueka, Y., Fraenkel, A.S., Klein, S.T. and Segal, E.: Improved hierarchical bit-vector compression in document retrieval systems, *Proc. 9th ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pp.88–96 (1986).
11) 弘田正雄，溝渕昭二，獅々堀正幹，青江順一：大規模文書データに対する用例文の効率的検索アルゴリズム，情報処理学会論文誌，Vol.38, No.10, pp.2004–2013 (1997).
12) Chien, R.T. and Frazer, W.D.: An application of coding theory to document retrieval, *IEEE Trans. Information Theory*, Vol.IT-12, No.2, pp.92–96 (1966).
13) Ancheta, Jr., T.C.: Syndrome-source-coding and its universal generalization, *IEEE Trans. Information Theory*, Vol.IT-22, No.4, pp.432–436 (1976).
14) 鈴木輝暁，今井秀樹：単純マルコフ情報源に対するシンドローム情報源符号化について，電子通信学会論文誌，Vol.J62-A, No.10, pp.736–743 (1979).
15) Liu, C. and Yu, C.: Data compression using word encoding with Huffman code, *Journal of the American Society for Information Science*, Vol.42, No.9, pp.685–698 (1991).
16) 今井秀樹：符号理論，p.341，電子情報通信学会，東京 (1990).
17) 程 亜非，桧垣泰彦，池田宏明：順次インデックスファイルに対する差分圧縮法の具体的提案，情報処理学会論文誌，Vol.36, No.9, pp.2175–2182 (1995).
18) Zipf, G.K.: *Human Behaviour and the Principle of Least Effort*, Addison-Wesley, Cambridge, Mass. (1949).
19) 高野文雄，佐藤 誠：JICST ファイルの用語統計分析とシソーラス作成への利用，情報管理，Vol.29, No.12, pp.1035–1052 (1987).
20) 小野脩一，小野寺夏生，富永 勲，鳥海 剛：日本情報の国際流通のための JICST の活動，情報の科学と技術，Vol.40, No.7/8, pp.468–479 (1990).
21) 新井智久，矢野陽一：プロセッサ技術—マイクロプロセッサ，電子情報通信学会誌，Vol.81, No.11, pp.1107–1112 (1998).

付 録

漸近的場合（$N_0 = \infty$）の近似計算

最小距離 $D = 3$ の場合に，長さ N のサブブロックのキーワード出現数が 1 である確率 q_1，キーワード出現数が 2 以上となる確率 q_2 は以下のようになる（式 (16)，(17)）：

$$q_1 = Np\{1-p\}^{N-1}. \tag{51}$$

$$q_2 = 1 - \{1-p\}^N - Np\{1-p\}^{N-1}. \tag{52}$$

また，VG 限界式（式 (49)）を満足する最小のシンドローム長 $N-K$ は次式となる．

$$N - K = \lfloor \log_2 N \rfloor + 1. \tag{53}$$

漸近的な場合（$N_0 = \infty$）には木の大きさは転置ファイルの大きさに比べて無視できる．転置ファイルの圧縮率を ρ_2 とすれば次式が成り立つ（式 (19)）：

$$\begin{aligned}\rho_2 &= R_2/N_0M\\ &= \{2-\{1-p\}^N\}/N\\ &\quad + p\{1-p\}^{N-1}\{\lfloor \log_2 N\rfloor + 1\} + 1\\ &\quad - \{1-p\}^N - Np\{1-p\}^{N-1}\\ &\le \{2-\{1-p\}^N\}/N\\ &\quad + p\{1-p\}^{N-1}(\log_2 N + 1)\\ &\quad + 1 - \{1-p\}^N - Np\{1-p\}^{N-1}\\ &\equiv \rho_{2,upper}. \end{aligned} \tag{54}$$

上界式を $\rho_{2,upper}$ とおいた．トレード・オフ曲線上では $pN = 0.01$〜0.5 なので $pN \ll 1$ と仮定すると，圧縮率 $\rho_{2,upper}$ は近似的に次のように表せる．

$$\begin{aligned}\rho_{2,upper} &= (1+pN)/N\\ &\quad + p(1-pN)(\log_2 N + 1)\\ &\quad + pN - p^2N^2/2 - pN + p^2N^2\\ &\approx 1/N + p^2N^2/2 + p\log_2 N\\ &\quad - p^2N\log_2 N + 2p. \end{aligned} \tag{55}$$

計算量については $pN \ll 1$ の条件下では $\{q_1 + q_2\}^{M_q}$ が 1 よりもずっと小さくなるので，計算量 C_2 の式 (31) の括弧内の第 3 項，第 4 項は無視できる．したがって，第 1 項と第 2 項だけをとって近似的に次式を得る．

$$C_2 = \alpha N_0 M_q\{1/NM_q + p\}. \tag{56}$$

式 (54) の圧縮率 $\rho_{2,upper}$ をサブブロック長 N に関して偏微分する．

$$\begin{aligned}&\partial\rho_{2,upper}/\partial N\\ &= -\{2-\{1-p\}^N\}/N^2\\ &\quad - \{1-p\}^N\log_e(1-p)/N\\ &\quad + p\{1-p\}^{N-1}\log_e(1-p)(\log_2 N + 1)\\ &\quad + p\{1-p\}^{N-1}/N\log_e 2\\ &\quad - \{1-p\}^N\log_e(1-p) - p\{1-p\}^{N-1}\\ &\quad - Np\{1-p\}^{N-1}\log_e(1-p). \end{aligned} \tag{57}$$

$pN \ll 1$ の条件のもとでは以下の式を得る．

$$\begin{aligned}&\partial\rho_{2,upper}/\partial N\\ &\approx -1/N^2 - p^2\log_2 N + p/N\log_e 2\\ &\quad + p(1-pN) - p\{1-p(N-1)\} + Np^2\\ &\approx -1/N^2 + Np^2. \end{aligned} \tag{58}$$

これより，圧縮率 $\rho_{2,upper}$ はサブブロック長 $N \approx N_{min} = p^{-2/3}$ で最小値をとる．サブブロック長 $N = N_{min}$ のとき式 (7)，(55) より次の近似式が成り立つ．

$$R_2/R_0 = N_0M\rho_2/R_0 \approx -3/2p^{1/3}\log_2 p. \tag{59}$$

式 (12)，(56) より検索キーワード数 M_q が $M_q \ll p^{-1/3}$ を満たすならば，次の近似式を得る．

$$C_2/C_0 = -1/\{M_qp^{1/3}\log_2 p - p^{-2/3}\}. \tag{60}$$

最小の圧縮率を与える点（サブブロック長 $N \approx N_{min}$）の近傍での圧縮率と計算量の関係を調べるためにサブブロック長 $N = N_{min}(1+\delta) = p^{-2/3}(1+\delta)$ ($|\delta| \ll 1$) とおくと圧縮率 $\rho_{2,upper}$ の式（55）と計算量 C_2 の式（56）は δ の 2 乗の項までとると近似的に以下のようになる：

$$\begin{aligned}&\rho_{2,upper}\\ &\approx 3p^{2/3}/2 - 2p\log_2 p/3 + 2p^{4/3}\log_2 p/3\\ &\quad + 2p + \delta(p/\log_e 2 - p^{4/3}/\log_e 2\\ &\quad + 2p^{4/3}\log_2 p/3)\\ &\quad + \delta^2(3p^{2/3}/2 - p/2\log_e 2 - p^{4/3}/2\log_e 2)\\ &\approx 3p^{2/3}/2 - 2p\log_2 p/3 + p\delta/\log_e 2\\ &\quad + 3p^{2/3}\delta^2/2. \end{aligned} \tag{61}$$

$$C_2 \approx \alpha N_0M_q\{p^{2/3}(1-\delta+\delta^2)/M_q + p\}. \tag{62}$$

メモリ量比 R_2/R_0 と計算量比 C_2/C_0 は以下のようになる：

$$\begin{aligned}R_2/R_0 &\approx -3/2p^{1/3}\log_2 p\\ &\quad + 2/3 - \delta/\log_e p - 3\delta^2/2p^{1/3}\log_2 p. \end{aligned} \tag{63}$$

$$\begin{aligned}C_2/C_0 &\approx -\{(1-\delta+\delta^2)/M_qp^{1/3} + 1\}\\ &\quad /\{\log_2 p - 1/M_qp\}. \end{aligned} \tag{64}$$

メモリ量比 R_2/R_0 と計算量比 C_2/C_0 がともに δ の一次関数として近似できるときはメモリ量比 R_2/R_0 と計算量比 C_2/C_0 の関係は直線になる．このための条件は式 (63)，(64) で δ^1 の項が δ^2 の項よりも十分に大きいことである．この条件は $\delta \ll p^{1/3}$ と表せる．割合 $p^{-2/3}\Delta\delta(\ll p^{-1/3})$ だけサブブロック長 N が変化するときのメモリ量比と計算量比の変化を $\Delta(R_2/R_0)$ と $\Delta(C_2/C_0)$ とおけば式 (63)，(64) より近似的に次式が成り立つ．

$$\begin{aligned}&\Delta(R_2/R_0) \approx -\Delta\delta/\log_e p.\\ &\Delta\{C_2/C_0\} \approx \Delta\delta/M_qp^{1/3}\log_2 p\{1\\ &\qquad - 1/M_qp\log_2 p\}. \end{aligned} \tag{65}$$

式 (65) よりサブブロック長 $N = N_{min}$ の近傍でのメモリ量比と計算量比の直線の傾きは次式で与えられる．

$$\begin{aligned}&\Delta\{C_2/C_0\}/\Delta\{R_2/R_0\}\\ &\quad = -\log_e 2/M_qp^{1/3}\{1 - 1/M_qp\log_2 p\}. \end{aligned} \tag{66}$$

(平成 12 年 9 月 1 日受付)
(平成 14 年 2 月 13 日採録)

二神 常爾（正会員）

1963 年生．1986 年東京大学理学部地球物理学科卒業．1988 年同大学大学院理学系研究科修士課程修了．1991 年同大学院理学系研究科博士課程修了．理学博士．東京大学研究生，日本原子力研究所専門研究員，東京大学宇宙線研究所研究員等を経て 1995 年早稲田大学大学院理工学研究科博士後期課程入学．横浜商科大学非常勤講師を経て現在関東学園大学非常勤講師．情報検索，文献検索に関する研究に従事．オフィス・オートメーション学会会員．