社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2006－CSEC－34 (58)
2006／7／21

情報埋込機能付データ圧縮ツールIH-ZIPの開発

吉岡 克成† 薗田光太郎† 滝澤 修† 中尾 康二† 松本 勉††

† 独立行政法人 情報通信研究機構 〒184-8795 東京都小金井市貫井北町 4-2-1
†† 横浜国立大学 大学院 環境情報研究院 〒240-8501 横浜市保土ヶ谷区常盤台 79-7
E-mail: †{katsunari_yoshioka,kotaro_sonoda,taki,ko-nakao}@nict.go.jp, ††tsutomu@mlab.jks.ynu.ac.jp

あらまし 我々は，LZ77符号化やハフマン符号化などの可逆データ圧縮において情報を埋込む方法を既に提案している．本報告では，LZSS符号化における情報埋込方式を，データ圧縮ツールとして広く利用されているZIPに適用した，情報埋込機能付データ圧縮ツールIH-ZIPの実装と性能評価について報告する．評価の結果，圧縮率の観点からは，IH-ZIPはパラメータを調整することにより，スライド辞書法を忠実に実装したオリジナルのZIPに準ずる効率を達成できることがわかった．さらに処理速度の観点からは，高速化の工夫により，オリジナルZIPと同程度の速度を達成できた．

キーワード 情報ハイディング，データ圧縮，LZ77，ZIP，改ざん検出

Development of IH-ZIP: A Data Compression Tool for Information Hiding

Katsunari YOSHIOKA†, Kotaro SONODA†, Osamu TAKIZAWA†, Koji NAKAO†, and Tsutomu MATSUMOTO††

† National Institute of Information and Communications Technology (NICT)
4-2-1, Nukui-Kitamachi, Koganei, Tokyo, 184-8795, Japan
†† Graduate School of Environment and Information Sciences, Yokohama National University
79-7 Tokiwadai, Hodogaya-ku, Yokohama, 240-8501, Japan
E-mail: †{katsunari_yoshioka,kotaro_sonoda,taki,ko-nakao}@nict.go.jp, ††tsutomu@mlab.jks.ynu.ac.jp

Abstract We have previously proposed several methods of information hiding for lossless data compression for LZ77 and Huffman coding, etc. This report focuses on the implementation and performance evaluation of an information hiding tool, called IH-ZIP, which we have developed based on our previous proposal on LZSS encoding. The evaluation showed that IH-ZIP can achieve the same level of performance with the original ZIP algorithm (with straight forward implementation of sliding dictionary) in terms of compression rate and compression speed.

Key words Information Hiding, Data Compression, LZ77, ZIP, Tamper Detection

1. はじめに

電子透かしやステガノグラフィなどの情報ハイディング分野では，画像や音声といった特定の種類または特定のフォーマットへの情報埋込が広く議論されているが，近年，汎用的な可逆データ圧縮において情報を埋込む方式が提案されている[1][3][6][7][8][9]．これらは，任意の入力データ(カバーデータ)に情報を埋込むという点でユニバーサルな情報ハイディング方式といえる．また，情報が埋込まれたデータ(ステゴデータ)から元のカバーデータが復元可能であることからロスレス情報埋込方式(lossless data embedding)[2]の一種と考えることができる．

Atallahらは，文献[1]において，LZ77符号化に変更を加えることで，通常の復号処理との互換性を保ちつつ，圧縮されたデータに秘密の情報を埋込む手法を提案し，提案方式をデータ圧縮アルゴリズムgzipに適用している．Shimらは，文献[3]において，LZW符号化において情報埋込を行う方法を提案している．畠田らは，文献[7]において，ロスレスデータ圧縮における情報埋込の理論的性能に言及し，Willems法[5]に関して，一定量までの埋込であれば，符号化の漸近的最適性が保たれることを示している．さらに，畠田らは，埋込情報としてカバーデータの一部を用いることで，情報埋込による圧縮性能の

向上を図る自己埋込と呼ばれる手法を提案している．横尾らは，文献[9]において，LSBへの情報埋込を定式化した情報理論的モデルを示し，提案モデルにおいて，埋込に必要なカバーデータサイズが漸近的に最小となる埋込法を示している．

筆者らは，文献[6]において，Atallahらの手法を改良することで，埋込可能情報量とステゴデータサイズのトレードオフを制御することが可能な方式を提案している．また，文献[8]において，ハフマン符号化において情報埋込を行う方法を示している．本報告では，筆者らが文献[6]で提案した方式を基に開発した，情報埋込機能付圧縮ソフトウェアIH-ZIPの実装とその性能評価について報告する．

2.節では，IH-ZIPを含む可逆データ圧縮における情報ハイディングのモデルを説明する．3.節では，我々が文献[6]で提案したLZSS符号化における情報ハイディング方式の概要について説明する．4.節ではIH-ZIPの実装について説明し，5.節では，IH-ZIPの性能評価について述べる．

2. モデル

本節では，可逆データ圧縮における情報ハイディングのモデルを示す．IH-ZIPによる情報埋込は本モデルに従う．

まず，情報埋込を伴わない，通常の圧縮符号化について定義する．長さkの系列$x^k \in A_D^k$を入力とし，系列$f(x^k) \in A_R^*$を出力する符号化fを考える．系列x^k, $f(x^k)$をそれぞれ入力系列，符号系列と呼び，$x^k = x_1 x_2 \cdots x_k$のように書く．また，$1 \leqq i \leqq j \leqq k$のとき，系列$x_i x_{i+1} \cdots x_j$を$x_{[i,j]}$と書く．ここで，$x_{[i,i]} = x_i$である．さらに，集合$A_D$と$A_R$をそれぞれ入力アルファベット，符号アルファベットと呼ぶ．符号化fは任意の入力系列$x^k \in A_D^k$に対して符号系列$f(x^k) \in A_R^*$を1対1対応で出力する．すなわちfはA_D^kからA_D^*への単射の写像と考える．一方，任意の$x^k \in A_D^k$について，符号系列$f(x^k)$から入力系列x^kに変換する復号f^{-1}を考える．復号f^{-1}はfの逆写像である．符号化と復号の流れを図1に示す．

次に，符号系列に情報を埋込むことが可能な符号化について定義する．まず，系列$x^k \in A_D^k$，鍵$\kappa \in \mathcal{K} = \{0,1\}^*$，埋込情報$e^a \in \mathcal{E} = \{0,1\}^a$の3つを入力とし，符号系列$f_{emb}(\kappa, e^a, x^k) \in A_R^*$を出力する符号化$f_{emb}$と，符号系列$f_{emb}(\kappa, e^a, x^k)$と鍵$\kappa$を入力とし，埋込情報$e^a$を出力する抽出$g$の，2つの写像を考える．ここで，$\mathcal{K}, \mathcal{E}$をそれぞれ鍵空間，埋込情報空間と呼ぶ．ここで2つの写像f_{emb}, gは以下を満たすとする:

全ての$(\kappa, e^a, x) \in \mathcal{K} \times \mathcal{E} \times A_D^k$について

- $f^{-1}(f_{emb}(\kappa, e^a, x)) = x$かつ
- $g(\kappa, f_{emb}(\kappa, e^a, x)) = e^a$.

すなわち，符号系列$f_{emb}(\kappa, e^a, x^k)$は，通常の復号$f^{-1}$により$x^k$に復号される．また，鍵$\kappa$を知るエンティティは，符号系列$f_{emb}(\kappa, e^a, x^k)$から鍵$\kappa$を用いて埋込情報$e^a$を抽出できる．情報ハイディングにおける通例に従い，埋込情報が埋込まれる対象である入力系列x^kをカバーデータと呼び，情報が埋込まれた後の符号系列$f_{emb}(\kappa, e^a, x^k)$をステゴデータと呼ぶ．図2に情報埋込を伴う符号化の流れを示す．

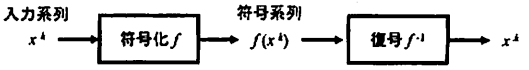

図1 通常の符号化．

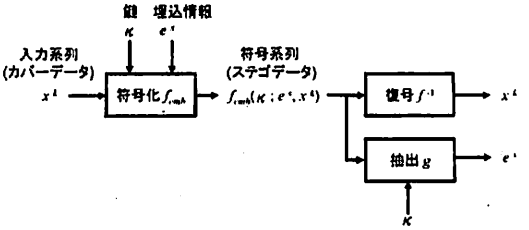

図2 情報埋込が伴う符号化．

本報告で注目するIH-ZIPにおいては，写像f, f_{emb}はそれぞれ，通常のZIP圧縮処理，IH-ZIPにおける情報埋込を伴う圧縮処理を指す．また，写像f^{-1}は通常のZIP復号処理(UNZIP)を指す．

3. LZSS符号への情報埋込

LZSS符号はZivとLempelによって提案されたLZ77符号から派生した圧縮符号化の1つで，実装の容易さが考慮されており，ZIPとLHAで利用されている．本節では，LZSS符号への情報埋込方式IH-LZSS[6]の基本アイディアを説明する．IH-LZSSは，IH-ZIPの核となるアルゴリズムである．まず，3.1節でオリジナルのLZSS符号について説明する．次に3.2節において，IH-LZSSの基本アイディアを説明する．

3.1 LZSS符号

LZSS符号では，入力系列のうち既に符号化処理が行われた，直近の長さlの系列を辞書系列として用いる．今，入力系列x^kのうち，$x_{[1,i-1]}$までは既に符号化の処理が行われており，系列$x_{[i,k]}$を符号化する状態を考える．このとき，辞書系列は$x_{[i-l,i-1]}$となる(注1)．次に$j = 1, \ldots, l$について，系列$x_{[i-j,i-1]}$と系列$x_{[i,k]}$が先頭から一致するシンボル数$len_{i,j}$を調べ，$len_{i,j}$が最大値len_i^{max}となるjと最大一致長len_i^{max}の組を出力する．ここでjをオフセットと呼ぶ．例外的に，最大一致長len_i^{max}が定められた値q以下の場合は，上記の組による出力を行わず，1シンボル分x_iをそのまま出力する(リテラル出力(注2))．このようにLZSS符号では，組(j, len_i^{max})による出力とリテラル出力の2通りがあるため，これを区別するために，上記の出力に先立ってフラグビットを出力する．復号の際は，フラグビットを読み込んだ後，フラグビットに従いリテラル出力(フラグ0)または組による出力(フラグ1)に対応した処理を行う．以下に，$q = 2$の場合のLZSS符号化の例を示す．入力系列は"ABBAABBABBAABB"とする．

A	B	B	A	ABBA	BBAABB
(0,A)	(0,B)	(0,B)	(0,A)	(1,4,4)	(1,7,6)

(注1)：ここでは簡単のため$l \leqq i$としている．
(注2)：ZIPとLHAでは$q = 2$である．

なお，ZIP では，辞書系列の長さは 2^{15}[シンボル](≃ 32[KB]) であるため，オフセットを一意に表現するために 15[bit] を確保している．さらに，最大一致長の限度を 256[シンボル] としており，一致長を表現するために 8[bit] を確保している．また，リテラル出力も同様に 8[bit] を用いる．これらの LZSS 符号の出力はさらにハフマン符号化され，ZIP 圧縮データとして出力される．詳細は 4. 節で述べる．

3.2 IH-LZSS [6]

本節では，筆者らが文献 [6] で提案した，LZSS 符号への情報埋込方式である IH-LZSS を概説する．3.1 節で述べた通り，LZSS 符号化では，辞書系列と現在の符号化対象系列が最長一致する位置の情報 (オフセット) と一致長の組により，入力系列を表現する．ここで，最長一致するオフセットが複数存在する場合を考える．このとき，いずれのオフセットを用いても正しく復号できるため，どのオフセットを用いるかを選択的に制御することで埋込情報を表現する．さらに，復号が正しく行われることのみを条件とし，圧縮率の低下を許容するならば，最長一致ではないオフセットも利用可能であるため，最大一致長よりも $len_{th}-1$[シンボル] 分小さい一致長をもつオフセットも選択肢に加えることで埋込可能情報量を増加させる．パラメータ len_{th} が大きいほど埋込可能情報量は大きくなるが，一方で圧縮率は低下し，ステゴデータサイズは大きくなる．

また，オリジナルの LZSS 符号では，上記のように最長一致するオフセットが複数存在する場合，最小のオフセット値を用いる．そのため，オフセットの確率分布は小さい値に偏りやすく，後段処理であるハフマン符号化が有効となる．一方，情報埋込を伴う符号化では，最長一致するオフセットが複数存在する場合，埋込情報に依存してオフセットが選ばれる．そのため，オフセットの分散は大きくなり，ハフマン符号化の効率が落ちる．そのため，文献 [6] では，埋込のために選択可能なオフセットの範囲を限定し，オフセットの分散を抑えている．具体的には最長一致するオフセットのうち最小のものからの距離が一定範囲 $offset_{th}$ に入る場合のみを選択可能とする．IH-LZSS を ZIP に適用する場合，パラメータ $offset_{th}$ が大きいほど埋込可能情報量は大きくなるが，一方で圧縮率は低下し，ステゴデータサイズは大きくなる．図 3 に IH-LZSS の基本アイディアを示す．

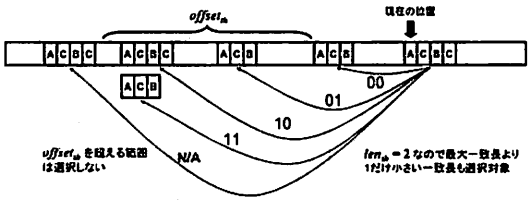

図 3 IH-LZSS の基本アイディア ($len_{th}=2$).

IH-LZSS の 2 つのパラメータは復号の際も必要であるため，IH-ZIP では，これらは鍵 κ の一部として，埋込側と抽出側で共有しているものとする．

4. IH-ZIP の実装

本節では，情報埋込機能付圧縮ソフトウェア IH-ZIP の実装について説明する．IH-ZIP には，任意埋込モード，改ざん検出モード，評価モードの 3 つの動作モードがあり，それぞれ目的，入出力，処理が異なっている．まず，3 つのモードに共通する構成要素として，IH-ZIP 情報埋込部と IH-ZIP 情報抽出部についてそれぞれ 4.1 節，4.2 節で説明する．次に 4.3 節で 3 つのモードについてそれぞれ説明する．

4.1 IH-ZIP 情報埋込部

IH-ZIP 情報埋込部は，IH-ZIP の核となる IH-LZSS 符号化機能を内蔵するコンポーネントであり，入力系列の圧縮を行うと共に，埋込情報を圧縮データに埋込む処理を行う．入力は，カバーデータ x^k，埋込情報 e^s，鍵 $\kappa=(len_{th}, offset_{th}, seed)$ である．ここで $len_{th}, offset_{th}$ は IH-LZSS の 2 つのパラメータであり，$seed$ は擬似乱数生成器の種である．種 $seed$ は情報埋込の際の埋込対応表作成に用いる．一方，出力はステゴデータ，すなわち，埋込情報 e^s が埋込まれた IH-LZSS 符号化の結果となる．IH-ZIP 情報埋込部の処理は次の 2 点を除き，通常の ZIP 圧縮処理と全く同一である．

- 前段の LZSS 符号化の代わりに IH-LZSS を用いる．
- 圧縮データに加えて，ログデータを出力する．

IH-LZSS 圧縮データは通常の LZSS 復号処理により復号可能であるから，IH-ZIP 情報埋込部により圧縮されたデータは通常の ZIP 復号処理，すなわち UNZIP により復号可能となる．次に，IH-ZIP が採用している埋込方法を説明する．

今，IH-LZSS 符号化において，カバーデータ x^k のうち，$x_{[1,i-1]}$ まで既に符号化している場合を考える．このとき，IH-LZSS では，以下を満たす全てのオフセット j が選択可能となる．

$$max(len_i^{max}-len_{th},q) < len_{i,j} \leqq len_i^{max}, \quad (1)$$

$$j-\xi \leqq offset_{th} \quad (2)$$

ここで $\xi = min\{j' | max(len_i^{max}-len_{th},q) < len_{i,j'} \leqq len_i^{max}, j'=1,...,l\}$ とする．すなわち，ξ は不等式 (1) を満足するオフセットのうち最小値を指す．今，上記の条件を満たすオフセットが p 種類存在する場合を考える．このとき，図 5 に示すように，葉の数 p を持ち，高さが最も小さい 2 分木を構成する．この 2 分木の分岐にそれぞれ 0 または 1 を割り当てる一方で，p 個の葉にそれぞれオフセットを 1 つずつ割り当てる．2 分木の根から各オフセットまでの経路に割り当てられているバイナリ値を連結することで，オフセットとバイナリ系列，すなわち埋込情報 e^s の部分系列を対応付けることができる．この割当方法は文献 [1] で用いられているが，IH-ZIP では，上記の処理に加えて，2 分木の各分岐へのバイナリ値の割当を擬似乱数列により行っている．この擬似乱数列は種 $seed$ から生成される(注3)ため，鍵 κ を知るエンティティは，オフセットから

(注3)：実際は，初期値 $seed$ を SHA1 に入力し得られるダイジェスト値を擬似乱数列として用いている．選択肢 p が大きく，乱数列が不足した場合は，既に得られた乱数列のダイジェスト値を生成しこれを新たな乱数列として用いる．

正しい埋込情報を復元することが出来る．

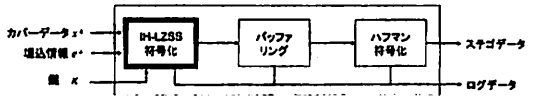

図 4　IH-ZIP 情報埋込部．太枠部の IH-LZSS とログデータ出力を除き通常の ZIP 圧縮と同様の処理である．

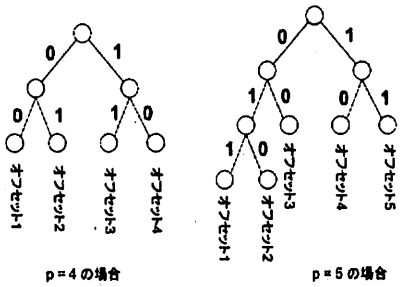

図 5　オフセットと埋込情報の対応付け．各分岐でのバイナリ値の割当は $seed$ から生成される乱数列により行う．

4.2　IH-ZIP 情報抽出部

IH-ZIP 情報抽出部は，IH-ZIP によって圧縮されたデータ (ステゴデータ) と鍵 κ を入力とし，入力ステゴデータに埋込まれた情報を抽出する．前段処理では，ハフマン符号の復号により IH-LZSS 符号系列に復号を行う．これは ZIP の復号処理，すなわち UNZIP の前段処理と同様である．次に，後段処理では，IH-LZSS 符号系列から情報を抽出する．抽出処理は，IH-LZSS 復号と並行して行われる．すなわち，復号によりカバーデータの一部を復元したのち，各 i, j における一致長 $len_{i,j}$ を求め，不等式 (1)(2) を満たす j のうち，いずれが選択されたかを判断する．さらに，鍵 κ を用いて図 5 の 2 分木を再構築し，オフセットと埋込情報の対応関係から埋込情報 e^* を得る．このように，抽出処理では，IH-LZSS の埋込処理とほぼ同様の処理を行うことに加えて復号を行う必要がある．そのため，抽出処理は埋込処理よりも煩雑であり処理時間も長くなることが予想される．

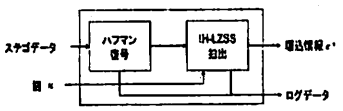

図 6　IH-ZIP 情報抽出部．

4.3　動作モード

本節では，IH-ZIP の 3 つの動作モードについて説明する．

任意埋込モードは，任意の単一ファイル (カバーデータ) に，任意の単一ファイル (埋込情報) を埋込むモードである．抽出処理においては，ステゴデータから埋込情報を抽出し，ファイルとして出力する．任意埋込モードの処理の流れを図 7 に示す．図から分かる通り，任意埋込モードは前節で説明した IH-ZIP 情報埋込部と IH-ZIP 情報抽出部をほぼそのまま用いて構成される．ただし，埋込情報ファイルには，抽出時にファイルとして復元するため，図 8 のような変換が施される．

改ざん検出モードは，任意のファイル (カバーデータ) の圧縮を行うと共に，圧縮データが改ざんされている場合には，それを検出する機能を有する．改ざん検出モードでは，カバーデータのダイジェストデータ (HMAC-SHA1 ダイジェスト値) を埋込んでおく．ステゴデータの検証の際には，ステゴデータを (通常の UNZIP により) 復号した結果のダイジェスト値と，埋込情報とを比較することで改ざんを検出する．このため，改ざん検出モードでは，鍵 (len_{th}, $offset_{th}$, $seed$) に加えて HMAC-SHA1 の鍵を共有する必要がある．図 9 に改ざん検出モードの処理の流れを示す．

最後に，評価モードでは，入力であるカバーデータに対して IH-ZIP が内部生成した乱数列を最大限埋込む．これにより，各カバーデータの最大埋込可能情報量を見積もることが出来る．

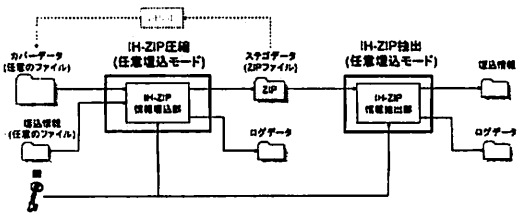

図 7　IH-ZIP の任意埋込モード．

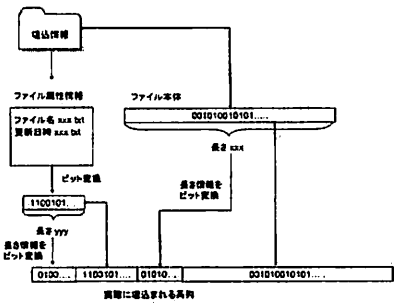

図 8　任意埋込モードにおける埋込情報の形式．

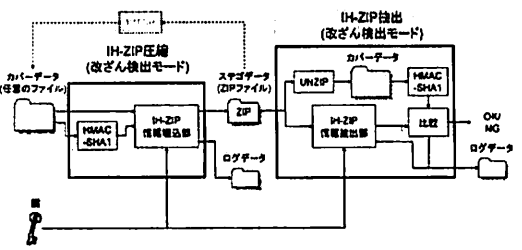

図 9　IH-ZIP の改ざん検出モード．

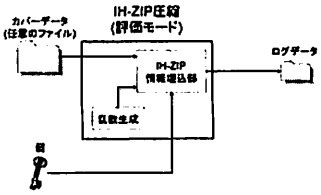

図 10 IH-ZIP の評価モード.

4.4 高速化

IH-ZIP では，IH-LZSS において最も負荷の高い，辞書系列とのパターンマッチ処理を工夫することで，高速化を図っている．今，入力系列 x^k のうち，$x_{[1,i-1]}$ までは既に符号化が行われており，系列 $x_{[i,k]}$ を符号化する状態を考える．このとき，シンボル $x_i \in A_D$ から始まる部分文字列が辞書系列のどの位置にあるかがわかっていれば，$j = 1, ..., l$ について全数探索的に $len_{i,j}$ を調べる必要はない．そこで，先頭 2 文字 $x^2 \in A_D^2$ について，このようなインデックス群を保持しておく．そのためには入力系列が符号化される度に，符号化された部分系列についてのみ部分的にサーチを行い，該当インデックス群を更新して行けばよい．このような高速化の手法は文献 [11] に詳しい．

5. 性能評価

本節では IH-ZIP の性能評価として，実際に情報埋込や抽出を行った結果を示す．本報告における評価実験は全て以下のスペックをもつ同一のマシンにより行った．

Microsoft(R) Windows(R) XP Professional(SP2)
Intel(R) Pentium(R) 4 Processor 670
(3.80 GHz, 2MB L2Cashe, 800 MHz FSB)
2GB DDR2 SDRAM Module (533MHz, 2DIMM)

入力サンプルデータは，圧縮アルゴリズムの評価テストベッドとして広く用いられている Calgary Corpus の 18 個のファイルと Canterbury Corpus の 11 個のファイル，合計 29 ファイルを用いた [10](表 1 参照).

まず，5.1 節では，評価モードを用いて，最大埋込可能情報量の埋込を行った際の IH-ZIP のパフォーマンスを報告する．さらに 5.2 節では，任意埋込モードにより，一定量の情報を埋込んだ場合のパフォーマンスについて述べる．

5.1 最大埋込可能情報量の埋込

表 2 に，全 29 個の入力ファイルに対して，高速 ZIP 圧縮，オリジナルの ZIP 圧縮を行った際の圧縮データサイズ，および，IH-ZIP の評価モードによって情報埋込を伴う圧縮を行った際の圧縮データ (ステゴデータ) サイズを示す．また表 3 に，このときに実際に埋込まれた情報量 (最大埋込可能情報量) を示す．表 4 は，上記の処理を行った際の各処理時間である．IH-ZIP のパラメータは $len_{th} = 1, 2, 5, 10, 100, 255$ の場合についてそれぞれ結果を示した．但し，パラメータ $offset_{th}$ については，$offset_{th} = 32768$(最大値) とし，固定値とした．また，高速 ZIP とは，LZSS 符号化における辞書系列とのパターンマッチの際に，同一文字列で始まる部分文字列をリンクしておくことで検索を容易にし，処理時間を短縮する手法である(注4)．オリジナル ZIP とは，スライディング辞書法を忠実に実装した場合である．

表 1 実験で用いた Calgary Corpus(bib～trans) と Canterbury Corpus(alice29.txt～xargs.1) のファイルの説明 [10].

bib	Bibliography (refer format)
book1	Fiction book
book2	Non-fiction book (troff format)
geo	Geophysical data
news	USENET batch file
obj1	Object code for VAX
obj2	Object code for Apple Mac
paper1	Technical paper
paper2	Technical paper
pic	Black and white fax picture
progc	Source code in "C"
progl	Source code in LISP
progp	Source code in PASCAL
trans	Transcript of terminal session
alice29.txt	text English text
asyoulik.txt	play Shakespeare
cp.html	html HTML source
fields.c	Ccrc C source
grammar.lsp	list LISP source
kennedy.xls	Excl Excel Spreadsheet
lcet10.txt	tech Technical writing
plrabn12.txt	poem Poetry
ptt5	fax CCITT test set
sum	SPRC SPARC Executable
xargs.1	man GNU manual page

表 2 より，オリジナル ZIP による圧縮データサイズが最も小さくなっており，最も高い圧縮率を示していることが分かる．IH-ZIP についてはパラメータ len_{th} が大きくなるにつれて，圧縮率が低下している．しかし $len_{th} \leq 2$ の範囲では，オリジナル ZIP と比較すると圧縮率は劣るものの，高速 ZIP と同等以上の圧縮率を示している．

一方，表 4 より，圧縮速度については圧倒的に高速 ZIP が速く，オリジナル ZIP および IH-ZIP はほぼ同程度の速度となっている．しかし，画像データである pic，ptt5 についてのみ，IH-ZIP の処理時間はパラメータ len_{th} に比例して大きくなる．これは，画像データの冗長性が高く，非常に長い一致長がしばしば現れるため，情報埋込のために選択可能なオフセットが大量に得られた結果，処理が遅くなっているものと思われる．

次に，代表的な 11 ファイル bib，book1，geo，obj1，paper1，pic，cp.html，grammar.lsp，kennedy.xls，ptt5，xargs.1 に関して図 11，図 12，図 13 にパラメータ len_{th} と圧縮率，埋込率，圧縮処理時間の関係をそれぞれ示す．ここで圧縮率=圧縮データサイズ/入力データサイズ，である．また，埋込率=埋込情報量/ステゴデータサイズ，である．

図 11 が示すとおり，全 11 ファイルについて，パラメータ len_{th} が大きくなるにつれて最大埋込可能情報量が増加する．画像データである pic，ptt5 を除き，最大埋込可能情報量の上限が確認できる．一方，冗長性に富む pic，ptt5 については，上記で述べたとおり，実装上の最大一致長である 255 を超える一致長が多く現れていることが予想され，最大埋込可能情報量の上限は [1, 255] の範囲では確認できない．同様に図 13 でも

(注4)：この手法は文献 [11] に詳細に述べられている．

表 3 IH-ZIP の最大埋込可能情報量．評価モードにおいてパラメータを $len_{th} = 1, 2, 5, 10, 100, 255$, $offset_{th} = 32768$ (最大値) とした場合．単位は全て [byte].

名前	$len_{th} = 1$	2	5	10	100	255
bib	2,115	4,374	8,515	11,784	14,020	14,016
book1	22,193	52,534	107,562	127,809	129,785	129,840
book2	13,718	30,034	63,094	85,562	94,212	94,098
geo	6,147	7,466	7,978	8,175	8,426	8,446
news	9,031	18,195	32,457	39,232	43,421	43,070
obj1	384	605	909	1,091	1,312	1,331
obj2	4,640	8,599	15,304	20,523	26,526	26,681
paper1	1,094	2,397	4,740	6,140	6,845	6,877
paper2	1,001	4,324	8,973	11,785	12,794	12,759
paper3	1,087	2,431	4,779	6,023	6,380	6,416
paper4	255	546	965	1,169	1,252	1,242
paper5	217	460	806	982	1,018	1,035
paper6	737	1,585	3,058	4,028	4,518	4,485
pic	3,745	6,310	11,292	17,500	47,052	59,330
progc	776	1,584	3,031	3,988	4,038	4,665
progl	1,138	2,297	4,429	6,424	9,201	9,368
progp	721	1,390	2,704	4,034	5,880	5,954
trans	1,123	2,130	3,991	5,833	9,171	9,461
alice29.txt	3,748	8,346	17,803	23,400	25,587	25,507
asyoulik.txt	3,419	7,868	16,081	19,237	20,023	19,977
cp.html	327	628	1,146	1,600	2,265	2,300
fields.c	147	308	584	848	1,122	1,150
grammar.lsp	50	94	178	245	310	310
kennedy.xls	29,980	77,650	145,060	228,892	229,339	228,924
lcet10.txt	10,125	22,463	47,780	63,255	69,373	69,372
plrabn12.txt	14,051	33,207	71,401	84,063	85,102	85,106
ptt5	3,738	6,327	11,399	17,535	48,187	59,269
sum	707	1,400	2,717	3,541	3,949	3,950
xargs.1	62	121	201	255	283	276

表 4 通常の ZIP(高速版とオリジナル ZIP) と IH-ZIP の圧縮処理時間．IH-ZIP については，評価モードにおいてパラメータを $len_{th} = 1, 2, 5, 10, 100, 255$, $offset_{th} = 32768$ (最大値) とした場合．単位は全て [s]．0.00 は処理時間が短すぎたため測定できないことを示す．

名前	高速 ZIP	オリジナル ZIP	IH-ZIP $len_{th} = 1$	IH-ZIP 2	IH-ZIP 5	IH-ZIP 10	IH-ZIP 100	IH-ZIP 255
bib	0.04	2.90	1.03	2.30	2.85	3.47	3.86	3.87
book1	0.37	32.71	14.26	21.33	28.10	30.35	30.55	30.90
book2	0.20	20.87	9.91	14.37	18.09	21.75	22.22	22.64
geo	0.07	6.03	4.40	4.96	5.27	5.04	5.17	5.56
news	0.15	13.56	7.25	9.52	11.96	12.72	14.11	13.61
obj1	0.01	0.39	0.66	0.86	0.77	0.77	1.10	1.49
obj2	0.07	7.32	5.27	6.03	7.91	8.90	9.56	9.64
paper1	0.01	1.34	0.88	1.22	1.66	1.90	1.82	1.93
paper2	0.03	2.37	1.32	2.02	2.53	2.88	3.29	3.21
paper3	0.01	1.14	0.87	1.31	1.50	1.84	1.93	1.93
paper4	0.01	0.12	0.38	0.50	0.42	0.58	0.40	0.55
paper5	0.00	0.09	0.35	0.45	0.38	0.49	0.56	0.41
paper6	0.00	0.82	0.69	0.94	1.06	1.31	1.27	1.43
pic	0.15	26.50	19.34	25.54	41.58	71.53	471.39	852.18
progc	0.01	0.71	0.65	0.84	1.02	1.27	1.35	1.50
progl	0.03	1.40	0.97	1.21	1.43	1.80	2.59	2.67
progp	0.01	0.65	0.72	1.00	1.13	1.39	2.30	2.15
trans	0.01	1.39	1.11	1.43	1.94	1.92	2.61	2.83
alice29.txt	0.07	5.07	2.84	3.82	4.86	5.55	6.09	6.12
asyoulik.txt	0.06	4.51	2.13	3.39	4.37	4.85	5.03	4.83
cp.html	0.00	0.25	0.43	0.55	0.66	0.61	0.88	0.80
fields.c	0.01	0.04	0.19	0.32	0.35	0.32	0.49	0.50
grammar.lsp	0.00	0.01	0.18	0.20	0.22	0.23	0.24	0.25
kennedy.xls	0.35	35.17	14.24	20.12	40.87	70.23	77.98	69.69
lcet10.txt	0.15	14.81	6.83	10.17	12.76	14.04	16.98	16.88
plrabn12.txt	0.21	20.51	9.33	13.32	17.96	19.69	20.14	19.90
ptt5	0.15	26.50	20.27	25.30	40.07	74.22	463.28	867.59
sum	0.01	0.82	0.96	1.08	1.24	1.80	1.50	1.57
xargs.1	0.00	0.01	0.15	0.17	0.18	0.19	0.23	0.22

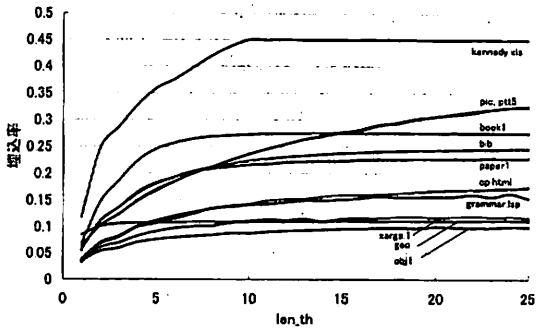

図 11 パラメータ len_{th} と埋込率の関係 (評価モード利用)．$offset_{th} = 32768$ (最大値) とした．

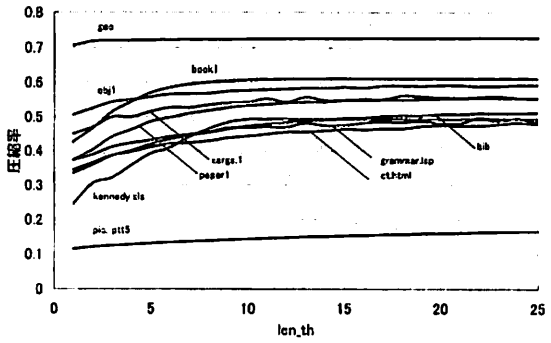

図 12 パラメータ len_{th} と圧縮率の関係 (評価モード利用)．$offset_{th} = 32768$ (最大値) とした．

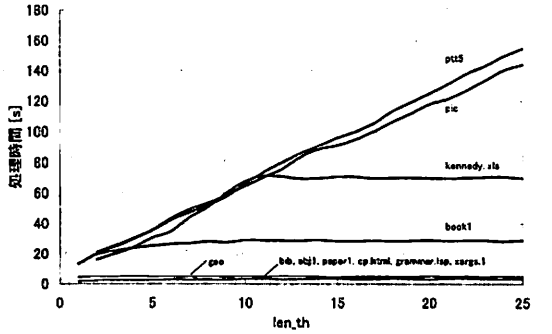

図 13 パラメータ len_{th} と圧縮時間 [s] の関係 (評価モード利用)．$offset_{th} = 32768$ (最大値) とした．

pic，ptt5 の圧縮 (埋込) 処理時間は単調増加を示し，上限は確認できない．

5.2 一定量 (20byte) の埋込

表 5 に，全 29 個の入力ファイルに対して，高速 ZIP 圧縮，オリジナルの ZIP 圧縮を行った際の圧縮データサイズ，圧縮処理時間，復号処理時間と，IH-ZIP の任意埋込モードによって 20[byte] の情報埋込を伴う圧縮を行った際の圧縮データ (ステゴデータ) サイズ，圧縮 (埋込) 処理時間，復号時間，埋込情報の抽出時間を示す．

高速 ZIP，オリジナル ZIP，IH-ZIP のいずれの圧縮方式においても，復号時間はほぼ一定となっている．また，圧縮時間については，オリジナル ZIP を上回る速度を示したが，情報抽出時間は圧縮時間の 2 倍以上になるケースもあり，4.2 節で予

表 2 ZIP(高速版とオリジナル ZIP) と IH-ZIP による圧縮データサイズの比較. IH-ZIP については, 評価モードにおいてパラメータを $len_{th} = 1, 2, 5, 10, 100, 255$, $offset_{th} = 32768$ (最大値) とした場合. 単位は全て [byte].

名前	圧縮前サイズ	高速 ZIP	オリジナル ZIP	IH-ZIP $len_{th}=1$	IH-ZIP 2	IH-ZIP 5	IH-ZIP 10	IH-ZIP 100	IH-ZIP 255
bib	111,261	43,471	35,204	37,386	39,957	46,786	52,651	58,052	58,214
book1	768,771	374,427	312,710	328,606	358,995	438,051	467,356	470,150	470,491
book2	610,856	253,739	206,278	218,551	237,191	294,312	330,925	346,618	346,688
geo	102,400	71,123	68,939	72,335	73,570	73,951	74,204	74,711	74,783
news	377,109	169,298	144,687	153,110	164,315	190,037	203,129	211,760	212,347
obj1	21,504	11,342	10,463	10,903	11,294	12,010	12,422	12,788	12,792
obj2	246,814	95,49	81,414	87,831	94,800	107,274	117,742	129,562	129,045
paper1	53,161	22,300	18,694	19,801	21,473	25,890	28,380	29,726	29,772
paper2	82,199	36,003	29,823	31,700	34,117	41,790	40,435	48,208	48,081
paper3	46,526	21,766	18,245	19,406	20,866	24,995	27,123	27,873	27,842
paper4	13,286	6,513	5,680	5,958	6,277	7,121	7,503	7,035	7,022
paper5	11,954	5,873	5,135	5,358	5,650	6,400	6,755	6,849	6,862
paper6	38,105	16,026	13,46	14,359	15,378	18,258	20,024	20,950	20,905
pic	513,216	63,017	52,992	59,178	61,931	67,156	73,870	107,395	120,057
progc	39,611	16,023	13,510	14,378	15,544	18,469	20,305	21,557	21,652
progl	71,646	20,329	16,362	17,582	18,932	22,958	26,321	31,318	31,750
progp	49,379	13,689	11,357	12,079	13,098	15,651	17,931	21,048	21,323
trans	93,695	24,055	19,130	20,426	21,959	25,417	28,928	35,434	36,149
alice29.txt	152,089	65,509	54,366	57,174	61,809	76,665	85,533	88,705	88,643
asyoulik.txt	125,179	58,572	49,027	51,891	56,368	68,662	73,526	74,793	74,932
cp.html	24,603	9,239	8,129	8,496	9,047	10,091	10,998	12,235	12,283
fields.c	11,150	3,902	3,274	3,397	3,644	4,224	4,729	5,295	5,314
grammar.lsp	3,721	1,587	1,384	1,395	1,453	1,631	1,741	1,844	1,844
kennedy.xls	1,029,744	231,763	208,801	253,584	314,389	404,540	508,411	508,892	508,493
lcet10.txt	426,754	176,756	144,429	152,028	164,335	205,274	231,074	241,415	241,486
plrabn12.txt	481,861	232,197	194,788	204,062	221,462	274,744	292,335	294,015	294,042
ptt5	513,216	63,021	52,996	59,168	61,939	67,313	73,827	107,854	120,091
sum	38,240	14,810	13,038	14,253	15,235	17,190	18,443	19,167	19,278
xargs.1	4,227	2,097	1,890	1,903	1,994	2,183	2,312	2,359	2,360

想したとおりの結果となった.

圧縮率についても, 前節と同様に, オリジナル ZIP が最も高い圧縮率を示し, IH-ZIP はオリジナル ZIP に近い圧縮率を示している.

6. おわりに

本報告では, 筆者らが文献 [6] で提案した方式を基に開発した, 情報埋込機能付圧縮ソフトウェア IH-ZIP の実装とその性能について説明した. 圧縮率の観点からは, IH-ZIP はパラメータを調整することにより, オリジナルの ZIP に準ずる効率 (高速 ZIP よりも高い効率) を達成できることがわかった. さらに処理速度の観点からは, 高速化手法の適用により, 高速 ZIP と比べては非常に遅いが, オリジナル ZIP と同程度の速度を達成することができた. 本報告では IH-ZIP の埋込パラメータの 1 つである len_{th} を可変とし, IH-ZIP のパフォーマンスを測定したが, 今後は, もう 1 つのパラメータである $offset_{th}$ も可変とし, 最適のパラメータを探索する. さらに, 抽出処理の高速化について検討を行う.

謝辞

IH-ZIP の開発全般に渡り貴重なご意見を頂いた, 日立 INS ソフトウェア株式会社インターネットソリューション事業本部ソリューションビジネス部 中村光宏氏, 高橋貴也氏の両氏に感謝の意を表する.

文 献

[1] M.J.Atallah and S.Lonardi, "Authentication of LZ-77 Compressed Data," ACM Symposium on Applied Computing, pp.282 - 287, 2003.

[2] J. Fridrich and M. Goljan, "Lossless Data Embedding for All Image Formats, " EI SPIE, Security and Watermarking of Multimedia Contents IV, vol. 4675, pp. 572 - 583, 2002.

[3] H. J. Shim and B. Jeon, "DH-LZW: Lossless Data Hiding Method in LZW Compression," PCM2004, LNCS vol. 3333, pp. 739 - 746, 2004.

[4] D. Salomon, "Data Compression: The Complete Reference - 2nd ed.," Springer, 2000.

[5] F. M. J. Willems, "Universal Data Compression and Repetition Times, " IEEE Trans. Inf. Theory, vol. 35, no.1, pp. 54 - 58, 1987.

[6] 松本勉, 吉岡克成, 滝澤修, "可逆データ圧縮における情報ハイディングに関する考察, " CSS2005.

[7] 畠田真吾, 横尾英俊, "再帰時間符号化データ圧縮法における情報埋込み, " 電子情報通信学会論文誌 Vol.J88-A No.5 pp.616-624, 2005.

[8] 吉岡克成, 滝澤修, 松本勉, "ハフマン符号へのデータサイズ不変型情報ハイディング " SCIS2006.

[9] 横尾英俊, 畠田真吾, "ユニバーサル無ひずみデータ埋め込みのための一般化 LSB データ埋め込みモデル," IT2005-85, ISEC2005-142, WBS2005-99, pp. 123-128, Mar. 2006.

[10] The Canterbury Corpus/The Calgary Corpus, (データ取得日 2006/06/26)
http://corpus.canterbury.ac.nz/

[11] 奥村晴彦, 山崎敏, "LHA と ZIP, " ソフトバンクパブリッシング, 2003.

[12] 情報理論とその応用学会 編, "情報源符号化=無歪みデータ圧縮," 培風館, 1998.

表 5　通常の ZIP(高速版とオリジナル ZIP) と IH-ZIP の圧縮データサイズ，圧縮時間，復号時間の比較．IH-ZIP については，任意埋込モードにおいてランダム文字列からなる 20[byte] のファイルを埋込んだ場合．パラメータは $len_{th} = 1$, $offset_{th} = 32768$ (最大値).

		高速 ZIP			オリジナル ZIP			IH-ZIP			
名前	圧縮前 [byte]	圧縮後 [byte]	圧縮時間 [s]	復号時間 [s]	圧縮後 [byte]	圧縮時間 [s]	復号時間 [s]	圧縮後 [byte]	圧縮 (埋込) 時間 [s]	復号時間 [s]	抽出時間 [s]
bib	111,261	43,471	0.04	0.06	35,204	2.90	0.04	36,490	1.11	0.05	3.86
book1	768,771	374,427	0.37	0.31	312,710	32.71	0.28	321,540	7.35	0.28	47.56
book2	610,856	253,739	0.20	0.26	206,278	20.87	0.21	212,512	5.57	0.20	28.69
geo	102,400	71,123	0.07	0.07	68,939	6.03	0.06	69,239	3.45	0.08	7.03
news	377,109	169,298	0.15	0.18	144,687	13.56	0.15	148,942	4.75	0.17	17.58
obj1	21,504	11,342	0.01	0.01	10,463	0.39	0.01	10,574	0.56	0.02	0.50
obj2	246,814	95,49	0.07	0.09	81,414	7.32	0.07	84,867	3.85	0.09	8.11
paper1	53,161	22,300	0.01	0.01	18,694	1.34	0.01	19,196	0.67	0.01	1.92
paper2	82,199	36,063	0.03	0.03	29,823	2.37	0.03	30,678	0.86	0.05	3.59
paper3	46,526	21,766	0.01	0.01	18,245	1.14	0.01	18,754	0.48	0.03	1.98
paper4	13,286	6,513	0.01	0.01	5,680	0.12	0.01	5,803	0.22	0.00	0.34
paper5	11,954	5,873	0.00	0.00	5,135	0.09	0.00	5,222	0.21	0.01	0.29
paper6	38,105	16,026	0.00	0.01	13,46	0.82	0.03	13,846	0.52	0.01	1.29
pic	513,216	63,017	0.15	0.14	52,992	26.50	0.09	56,697	17.97	0.11	23.28
progc	39,611	16,023	0.01	0.01	13,510	0.71	0.01	13,836	0.41	0.01	1.16
progl	71,646	20,329	0.03	0.03	16,362	1.40	0.03	16,851	0.55	0.02	1.93
progp	49,379	13,689	0.01	0.01	11,357	0.65	0.01	11,567	0.56	0.02	1.14
trans	93,695	24,055	0.01	0.03	19,130	1.39	0.03	19,717	0.73	0.03	2.17
alice29.txt	152,089	65,509	0.07	0.07	54,366	5.07	0.04	55,536	1.47	0.06	7.16
asyoulik.txt	125,179	58,572	0.06	0.06	49,027	4.51	0.03	50,350	1.17	0.06	6.60
cp.html	24,603	9,239	0.00	0.01	8,129	0.25	0.00	8,295	0.39	0.00	0 49
fields.c	11,150	3,902	0.01	0.01	3,274	0.04	0.01	3,301	0.17	0.01	0.26
grammar.lsp	3,721	1,587	0.00	0.00	1,384	0.01	0.00	1,373	0.17	0.00	0.17
kennedy.xls	1,029,744	231,763	0.35	0.35	208,801	35.17	0.32	222,267	6.66	0.27	48.05
lcet10.txt	426,754	176,756	0.15	0.15	144,429	14.81	0.14	147,800	3.98	0.14	20.77
plrabn12.txt	481,861	232,197	0.21	0.26	194,788	20.51	0.17	199,680	4.70	0.23	29.39
ptt5	513,216	63,021	0.15	0.10	52,996	26.59	0.14	56,699	18.54	0.11	22.27
sum	38,240	14,810	0.01	0.01	13,038	0.82	0.16	13,518	0.68	0.01	0.99
xargs.1	4,227	2,097	0.00	0.00	1,890	0.01	0.00	1,885	0.15	0.00	0.16