LZW の圧縮感度

三神 摩周 *　　クップル　ドミニク *

概要

文字列圧縮に対する 1 文字編集時の影響を評価する指標として，加法的感度・乗法的感度がある．既存研究では LZ77/LZ78 などが解析されているが，LZW の理論解析はまだ十分に行われていない．本稿では LZW に対し，加法的感度が $\Theta(n)$ であることを示す．さらに乗法的感度について，下界 $\Omega(n^{1/4})$ と上界 $O((n/\log n)^{2/3})$ を与える．

1　はじめに

文字列圧縮アルゴリズムは，通信，データ保存，情報検索など幅広い分野で利用されている．従来の評価は，静的な文字列に対する圧縮率や計算量に主眼が置かれてきた．しかし実際のデータは，編集操作や誤り訂正により動的に変化することが多い．そのため，**1 文字のわずかな編集が圧縮結果にどの程度影響を与えるか**を評価することは重要である．

Akagi ら [1] は，文字列圧縮法および反復性指標に対する「感度（sensitivity）」の概念を導入した．さらに Fujie ら [2] は，LZ 系圧縮 [5] および文字列アトラクタに対する加法的感度の厳密な上界・下界を与えている．

本研究では，これらの先行研究を基盤として，LZ78 [4] の改良版であり実用上広く用いられている **LZW 圧縮アルゴリズム**に着目し，1 文字編集に対する加法的感度および乗法的感度を理論的に解析する．

*山梨大学コンピュータ理工学科

2　基本概念

2.1　記号の定義

本稿では，以下の記号を用いる．

c 圧縮手法（LZW，LZ78 など）

T 入力文字列

T' T に 1 文字の編集操作を施した文字列

$c(T)$ 圧縮手法 c により T を factor 分解した結果

$z_c(T)$ 圧縮手法 c を T に適用した際の factor 数

2.2　漸近記法

関数 $f(n), g(n)$ に対し，

- $f(n) \in O(g(n))$：十分大きな n に対し $f(n)$ が $g(n)$ の定数倍以下
- $f(n) \in \Omega(g(n))$：十分大きな n に対し $f(n)$ が $g(n)$ の定数倍以上
- $f(n) \in \Theta(g(n))$：$f(n) \in O(g(n))$ かつ $f(n) \in \Omega(g(n))$

が成り立つとき，それぞれ上界，下界，タイトな評価と呼ぶ．

2.3　感度の定義

加法的感度

$$\mathrm{AS}_{\mathrm{edit}}(c, n) = \max_{T, T'} |z_c(T') - z_c(T)| \qquad (1)$$

乗法的感度

$$\mathrm{MS}_{\mathrm{edit}}(c,n) = \max_{T,T'} \frac{z_c(T')}{z_c(T)} \tag{2}$$

加法的感度は factor 数の差分，乗法的感度はその比率を評価する指標である．

表 1: 既存研究における LZ 系圧縮法の感度

圧縮手法	加法的感度	乗法的感度
LZ77	$\Theta(n^{2/3})$	2
LZ78	$\Theta(n)$	$\Omega(n^{1/4}) \sim O\left(\left(\frac{n}{\log n}\right)^{2/3}\right)$

既存研究により，LZ 系圧縮法に対する 1 文字編集時の感度はすでに一部が解析されている．[1, 2].

2.4 上界・下界評価の考え方

本研究では，文字列圧縮法の感度を評価するために，加法的感度および乗法的感度について，上界と下界の双方を与えることを目的とする．

上界とは，任意の文字列 T に対する 1 文字編集により，factor 数が最悪の場合にどの程度増加しうるかを評価したものである．一方，下界とは，特定の文字列 T を構成し，実際に 1 文字編集を行った際に，factor 数がどの程度増加するかを示すものである．

上界と下界が一致したとき，当該圧縮手法の感度は漸近的にタイトに評価されたといえる．

3 LZW 圧縮アルゴリズム

LZW 圧縮 [3] は，入力文字列を左から右へ逐次処理し，現在位置から始まる最長の辞書語を選択することで factor 分割を行う辞書型圧縮アルゴリズムである．初期辞書にはアルファベット Σ に含まれるすべての単一文字が登録されており，処理の過程で生成された辞書語は直ちに以降の factor 分割に利用される．

以下に，LZW 圧縮の動作を具体例で示す．アルファベット $\Sigma = \{a, b\}$，入力文字列を

$$T = \texttt{abaaaaba}$$

とする．初期辞書は $\{a, b\}$ とする．LZW は現在位置から始まる最長の辞書語を選択し，その直後の文字を用いて新たな辞書語を辞書に追加する．処理の過程を 表 2 に示す．

表 2: 文字列 abaaaaba に対する LZW の処理例

入力語	次の文字	追加される辞書語
a	b	ab
b	a	ba
a	a	aa
aa	a	aaa
ab	a	aba

この例から分かるように，一度生成された辞書語は直ちに再利用され，入力の構造に応じて factor の長さが変化する．そのため，入力文字列の途中で 1 文字が変更されると，辞書の構築順序が変化し，後続部分の factor 分割が大きく影響を受ける可能性がある．本研究では，この性質に着目し，LZW 圧縮に対する 1 文字編集時の感度を解析する．

4 LZW の加法的感度の解析

4.1 下界構成に用いる文字列

以下の文字列 T を考える．

$$T = \prod_{i=1}^{p} (a_i b_i a_i b_i c_i) \prod_{i=1}^{p} (a_i b_i c_i)$$

このとき，長さは $n = 8p$ である．

1 文字編集後の文字列 T' を次で定義する．

$$T' = \prod_{i=1}^{p} ((a_i b_i a_i b_i c_i) \# b_1 c_1 \prod_{i=2}^{p} (a_i b_i c_i)$$

ここで $\#$ は T に含まれない新しい文字である．

4.2 LZW による factor 分割

$$\begin{aligned}&\mathrm{LZW}(T) = \\ &\quad |a_1|b_1|a_1b_1|c_1|\cdots|a_p|b_p|a_pb_p|c_p| \\ &\quad |a_1b_1c_1|a_2b_2c_2|\cdots|a_pb_pc_p|\end{aligned}$$

$$\begin{aligned}&\mathrm{LZW}(T') = \\ &\quad |a_1|b_1|a_1b_1|c_1|\cdots|a_p|b_p|a_pb_p|c_p| \\ &\quad \#\,|b_1|c_1a_2|b_2|c_2\cdots|c_{p-1}a_p|b_p|c_p|\end{aligned}$$

編集点以降では既存辞書が利用できず，短い factor に分割されている．

4.3 下界評価

前節で与えた構成に対し，LZW による factor 数を具体的に数える．

まず，文字列 T に対する LZW 分解では，各 $i = 1,\ldots,p$ に対して

$$|a_i|,\ |b_i|,\ |a_ib_i|,\ |c_i|,\ |a_ib_ic_i|$$

の 5 つの factor が生じ，したがって $z_{\mathrm{LZW}}(T) = 5p$ が成り立つ．

一方，編集後の文字列 T' では，後半部分が短い factor に細分化され，# 自身と各 i に対して 6 個の factor が生成されるため，

$$z_{\mathrm{LZW}}(T') = 6p + 1$$

となる．以上より，

$$\mathrm{AS}_{\mathrm{edit}}(\mathrm{LZW}, n) \geq p + 1$$

ここで，文字列長は $n = 8p$ であるから，

$$\mathrm{AS}_{\mathrm{edit}}(\mathrm{LZW}, n) \in \Omega(n)$$

4.4 上界評価

一方，最悪の場合でも各文字が独立した factor になる以上には分割されないため，

$$\mathrm{AS}_{\mathrm{edit}}(\mathrm{LZW}, n) \in O(n)$$

が成り立つ．したがって，

$$\mathrm{AS}_{\mathrm{edit}}(\mathrm{LZW}, n) \in \Theta(n) \tag{3}$$

5 LZW の乗法的感度の解析

5.1 下界評価

本節では，LZW 圧縮に対する 1 文字編集時の**乗法的感度**の下界を与える．そのために，LZW による factor 数が大きく増加するような文字列 T を構成し，その factor 分解の様子を解析する．

アルファベットを

$$\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_{3k}\}$$

とする．まず，T を次の 2 つの部分に分けて定義する：

$$T = T^{(1)}\,T^{(2)}.$$

前半 $T^{(1)}$　前半部分は

$$T^{(1)} = \prod_{i=1}^{k}\Big((\sigma_1\sigma_2\cdots\sigma_i)\,\sigma_{k+i}\Big)$$

で定義される．すなわち

$$(\sigma_1)\sigma_{k+1}\ (\sigma_1\sigma_2)\sigma_{k+2}\ \cdots\ (\sigma_1\cdots\sigma_k)\sigma_{2k}$$

である．

この部分に対する LZW の factor 分解は，次のように与えられる：

$$\begin{aligned}\mathrm{LZW}(T^{(1)}) = (\sigma_1)\mid\sigma_{k+1}\mid(\sigma_1\mid\sigma_2)\mid\sigma_{k+2}\mid \\ (\sigma_1\sigma_2\mid\sigma_3)\mid\sigma_{k+3}\mid\cdots\mid(\sigma_1\cdots\sigma_{k-1}\mid\sigma_k)\mid\sigma_{2k}\end{aligned}$$

その結果，$T^{(1)}$ 全体では 2k-1 個の factor が生成される．

後半 $T^{(2)}$　次に後半部分を定義する．各 $j \in \{1,2,\ldots,k\}$ に対し，整数 l_j を

$$\frac{1}{2}l_j(l_j-1)+1 \leq j \leq \frac{1}{2}l_j(l_j+1)$$

を満たす最大の整数として定め，さらに

$$2 + j + l_j - 1 \equiv y_j \pmod{l_j}$$

を満たす整数のうち，k を超えない最大のものを y_j とする．このとき

$$P_j = (\sigma_1\sigma_2\cdots\sigma_{y_j}\,\sigma_{k+j})$$

とおき，後半部分を

$$T^{(2)} = \prod_{j=1}^{k}\Big(P_j\,\sigma_{2k+j}\,P_j\Big)$$

と定義する．すなわち

$$(P_1\sigma_{2k+1}P_1)\,(P_2\sigma_{2k+2}P_2)\,\cdots\,(P_k\sigma_{3k}P_k)$$

この部分では，1 ブロックあたり 4 個の factor が生成される．したがって，$T^{(2)}$ 全体では 4k 個の factor が生成される．文字列 T 全体に対する LZW の factor 数は次にようになる．

$$z_{\mathrm{LZW}}(T) = (2k-1) + 4k = 6k - 1$$

編集後の後半 $T'^{(2)}$　$P_j = (\sigma_1\cdots\sigma_{y_j}\sigma_{k+j})$ とする．

$$T'^{(2)} = \Big(P_1'\,\sigma_{2k+1}\,P_1\Big)\prod_{j=2}^{k}\Big(P_j\,\sigma_{2k+j}\,P_j\Big)$$

で定義する．ただし

$$P_1' = (\#\,\sigma_2\sigma_3\cdots\sigma_{y_1}\sigma_{k+1})$$

であり，# は T に含まれない新しい記号である．編集点より前の $T^{(1)}$ に対する factor 分解は T と同一であり，

$$z_{\mathrm{LZW}}(T^{(1)}) = 2k - 1$$

一方，$T'^{(2)}$ に対する factor 分解は次のようになる：

$$\begin{aligned}
&\mathrm{LZW}(T'^{(2)}) = \\
&\quad(\# \mid \sigma_2 \mid \cdots \mid \sigma_{y_1} \mid \sigma_{k+1} \mid \sigma_{2k+1} \mid \\
&\qquad\sigma_1\cdots\sigma_{y_1} \mid \sigma_{k+1}) \\
&\quad(\sigma_1 \mid \sigma_2\sigma_3 \mid \sigma_4\sigma_5 \mid \cdots \mid \sigma_{y_2} \mid \\
&\qquad\sigma_{k+2} \mid \sigma_{2k+2} \mid \sigma_1\cdots\sigma_{y_2} \mid \sigma_{k+2}) \\
&\qquad\quad\vdots \\
&\quad(\sigma_1\cdots\sigma_{y_k} \mid \sigma_{2k} \mid \sigma_{3k} \mid \sigma_1\cdots\sigma_{y_k}\sigma_{2k})
\end{aligned}$$

編集点以降（すなわち $T'^{(2)}$）で生成される factor 数は

$$z_{\mathrm{LZW}}(T'^{(2)}) = (y_1+3) + 5k + \sum_{j=2}^{k}\frac{y_j - (j+1)}{l_j}. \quad (4)$$

ここで，第 1 ブロックは # の導入により y_1 に比例して細分化され，$j \geq 2$ の各ブロックは l_j ごとに規則的な細分化が生じるため，(4) が得られる．

以上より

$$\begin{aligned}
z_{\mathrm{LZW}}(T') &= z_{\mathrm{LZW}}(T^{(1)}) + z_{\mathrm{LZW}}(T'^{(2)}) \\
&= 9k - 3 + \sum_{j=2}^{k}\frac{y_j - (j+1)}{l_j} \\
S &:= \sum_{j=2}^{k}\frac{y_j - j - 1}{l_j}
\end{aligned}$$

$l_j = L$ となる j の範囲は

$$\frac{1}{2}L(L-1) + 1 \leq j \leq \frac{1}{2}L(L+1)$$

で与えられる．これを

$$j_L^{\min} := \frac{1}{2}L(L-1) + 1, \qquad j_L^{\max} := \frac{1}{2}L(L+1)$$

と書く．このとき

$$S = \sum_{L=2}^{m}\sum_{j=j_L^{\min}}^{j_L^{\max}}\frac{y_j - j - 1}{L} = \sum_{L=2}^{m}\frac{1}{L}\sum_{j=j_L^{\min}}^{j_L^{\max}}(y_j - j - 1)$$

ただし $m := \max\{L \in \mathbb{N} \mid \frac{1}{2}L(L+1) \leq k\}$.

この定義より $m = \Theta(\sqrt{k})$ が成り立つ．

$l_j = L$ を固定すると，$y_j \equiv j + 1 \pmod{L}$ かつ $y_j \leq k$ より，$j = j_L^{\min}, \ldots, j_L^{\max}$ に対し，y_j は k から $k - L + 1$ までを一巡する．よって

$$\sum_{j=j_L^{\min}}^{j_L^{\max}} y_j = k + (k-1) + \cdots + (k-L+1) = Lk - \frac{1}{2}L(L-1)$$

また

$$\sum_{j=j_L^{\min}}^{j_L^{\max}} j = \frac{L}{2}\big(j_L^{\min} + j_L^{\max}\big) = \frac{L}{2}\big(L^2 + 1\big)$$

$$\sum_{j=j_L^{\min}}^{j_L^{\max}} 1 = L$$

以上より

$$\sum_{j=j_L^{\min}}^{j_L^{\max}} (y_j - j - 1) = L\Big(k - \frac{1}{2}L(L+1) - 1\Big)$$

したがって

$$\begin{aligned} S &= \sum_{L=2}^{m}\Big(k - \frac{1}{2}L(L+1) - 1\Big) \\ &= (m-1)k - \Theta(m^3) \end{aligned}$$

よって $S = \Theta(k^{3/2})$ であり，また，構成した文字列 T の長さは $n = \Theta(k^2)$ であるため，

$$z_{\mathrm{LZW}}(T') = 9k - 3 + S \in \Omega(k\sqrt{k})$$

$$\mathrm{MS}_{\mathrm{edit}}(\mathrm{LZW}, n) \geq \frac{z_{\mathrm{LZW}}(T')}{z_{\mathrm{LZW}}(T)} \in \Omega(\sqrt{k}) = \Omega(n^{1/4}).$$

5.2 上界評価

上界については，LZ 系圧縮と最小文法サイズとの関係に関する既存結果 [6] を用いる．LZ78 を含む辞書型圧縮アルゴリズムによって生成される文法サイズが，最小文法サイズに対して

$$\mathrm{O}\left(\left(\frac{n}{\log n}\right)^{2/3}\right)$$

で近似されることを示している．LZW は LZ78 の改良版であり，生成される辞書構造は LZ78 と同様に文法圧縮として解釈できるため，同様の上界評価が適用可能であり，以下の結果が得られる．

$$\mathrm{MS}_{\mathrm{edit}}(\mathrm{LZW}, n) \in \mathrm{O}\left(\left(\frac{n}{\log n}\right)^{2/3}\right)$$

6 まとめと今後の研究

本研究では，LZW 圧縮アルゴリズムに対して，1 文字編集に対する加法的感度，乗法的感度を理論的に解析した．今後の研究課題として，LZD など他の辞書型圧縮への拡張を考える．

参考文献

[1] Tooru Akagi, Mitsuru Funakoshi, and Shunsuke Inenaga. Sensitivity of string compressors and repetitiveness measures. *Inf. Comput.*, 291:104999, 2023.

[2] Yuto Fujie, Hiroki Shibata, Yuto Nakashima, and Shunsuke Inenaga. Tight additive sensitivity on LZ-style compressors and string attractors. In *Proc. SPIRE*, volume 16073 of *LNCS*, pages 109–123, 2025.

[3] Terry A. Welch. A technique for high-performance data compression. *IEEE Computer*, 17(6):8–19, 1984.

[4] Jacob Ziv and Abraham Lempel. Compression of individual sequences via variable-rate coding. *IEEE Trans. Information Theory*, 24(5):530–536, 1978.

[5] James A. Storer and Thomas G. Szymanski. Data compression via textual substitution. *Journal of the ACM*, 29(4):928–951, 1982.

[6] Moses Charikar, Eric Lehman, Ding Liu, Rina Panigrahy, Manoj Prabhakaran, Amit Sahai, and Abhi Shelat. The smallest grammar problem. *IEEE Trans. Information Theory*, 51(7):2554–2576, 2005.