Lempel–Ziv 項の距離を高次情報量で表現する符号

Dominik Köppl[1,a)]　Nicola Prezza[2,b)]　Gonzalo Navarro[3,c)]

概要：入力されているテキストの接頭辞の colex 順序に基づいて，LZ 分解 (Lempel–Ziv 77) で作られた項の距離の符号化が提案される．距離の符号化は k 番目の経験的エントロピーを寄せる．行った実験によると，提案される符号化は別の符号化より適切である．

キーワード：可逆データ圧縮，Lempel–Ziv 77 分解，符号化

序論

可逆圧縮は，圧縮のうち，復元前の文字列を完全な形で取り出すことが可能な圧縮方法である．

文字列の可逆圧縮における一つの専門的分野，LZ 分解 (Lempel–Ziv 77 分解) [4] がよく知られている．LZ は任意文字列 T に対し，T を項に分解する．このとき，LZ 分解は任意の項がその項の開始位置より前に出現を持つように分解する．各項を長さと前の出現の開始位置の二つ組に表現できる．ただし，後半を参照先と呼ぶ．すなわち，項を二つ組で表現した場合でも，元文字列を復元できる．二つ組の列による圧縮表現は，universal code [1] で符号化された後，元文字列より小さい場合に与えられる．しかし，長いテキストに考えれば，参照先の値は大きくなりつつ，圧縮率は悪くなるという現象がよく見られる．値を減らすため，参照先の変わりに，項の開始位置と参照先の距離を格納すれば，圧縮率を改善できる．複数参照先の候補があれば，項の一番近い参照先を選択することは最適である．その選び方を rightmost (parsing) [2] と呼ぶ．しかし，圧縮したファイルを見えて，距離はまだ一番大きい部分になってしまう．その距離のサイズを減らすために，本研究 [3] で，項の距離について新鮮な表現を提案する．

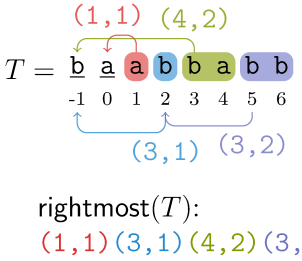

図 1　T = abbabb の LZ 分解．前処理で T の頭に追加した文字は下線で強調されている．その文字は参照先候補であるが，項には含まれることはない．項は円やかな長方形で現れるし，各項の参照先は矢で表現される．下で rightmost の二つ組列で表現される．

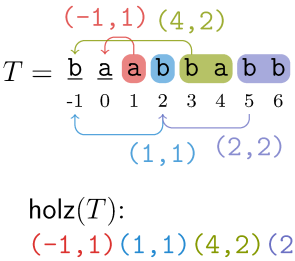

図 2　LZ　分解されている T = abbabb の項の holz 表現．

準備

Σ をアルファベットとし，Σ^* の要素を文字列と呼ぶ．文字列 T の長さを $|T|$ と表記する．文字列 $T = XYZ$ について X, Y, Z をそれぞれ文字列 T の接頭辞，部分文字列，接尾辞と呼ぶ．$T[i]$ は T 中の i 文字目の文字，$T[i..j]$ は T の i 文字目から j 文字目までの部分文字列を表す．$j = |T|$ のとき，$T[i..j]$ は開始位置 i を持つ接尾辞であり，

1　東京医科歯科大学, 東京
2　Ca' Foscari University, Venice, Italy
3　University of Chile, Santiago, Chile
a)　koeppl.dsc@tmd.ac.jp
b)　nicola.prezza@unive.it
c)　gnavarro@dcc.uchile.cl

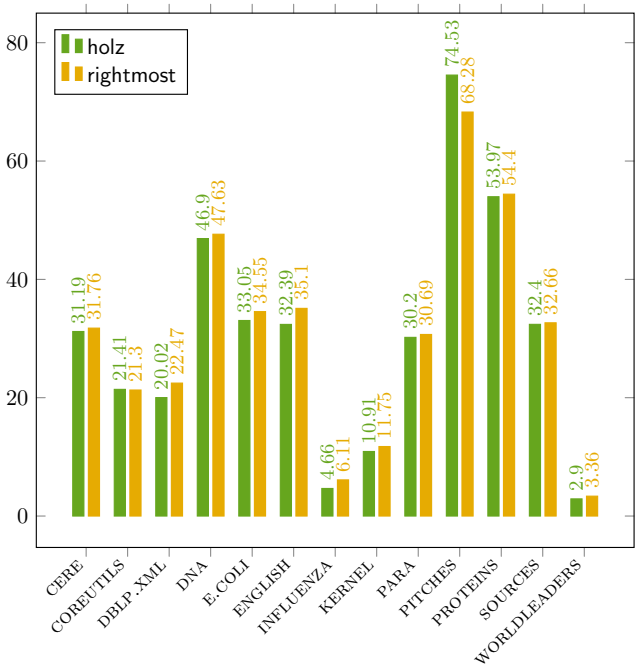

図 3 各 Pizza & Chili データセットの LZ で圧縮されている表現の圧縮率．横軸はデータセット，縦軸は圧縮率 (低いほうが良い)．

表 1 実験で使った Pizza&Chili corpus のデータセット．各のデータセットは 20 MB の接頭辞に縮められた．H_k と z は k 番目の経験的エントロピーと LZ の項の個数をそれぞれに示す．

dataset	σ	z [K]	H_0	H_2	H_4
CERE	5	8492	2.20	1.79	1.78
COREUTILS	235	3010	5.45	2.84	1.31
DBLP.XML	96	3042	5.22	1.94	0.89
DNA	14	12706	1.98	1.92	1.91
E.COLI	11	8834	1.99	1.96	1.94
ENGLISH	143	5478	4.53	2.89	1.94
INFLUENZA	15	876	1.97	1.93	1.91
KERNEL	160	1667	5.38	2.87	1.47
PARA	5	8254	2.17	1.83	1.82
PITCHES	129	10407	5.62	4.28	2.18
PROTEINS	25	8499	4.20	4.07	2.97
SOURCES	111	4878	5.52	2.98	1.60
WORLDLEADERS	89	408	4.09	1.74	0.73

$T[i..]$ とも書く．ϵ は空文字列を示す．colex 順序というのは逆文字列の辞書式順序に従う順序である．例えば，辞書式順序に aab $\prec$ bba であるが，colex 順序に aab $\succ$ bba である．

以降，文字列 T を入力として固定する．$\sigma := |\Sigma|$ と $n := |T|$ は アルファベットサイズと $T[1..n] = T$ の長さをそれぞれ示す．簡略のため，Σ の要素はすべて T の中に出現すると仮定する．前処理として， テキストの頭ですべての異なる σ 文字を追加し，T は $T[-\sigma+1..n] = \sigma\cdots 1T[1]\cdots T[n]$ になる．ただし，$\Sigma = [1..\sigma]$ と見られる．

LZ 分解　LZ 分解は $T[1..n] = F_1\cdots F_z$ を $F_1,\ldots,F_z$ の項に分解する．ただし，F_x は $T[-\sigma+1..\sum_{y=1}^{x-1}|F_y|]$ の範囲で始まる接尾辞の最長の共通接頭辞である．$T[-\sigma+1..0]$ はすべての文字を格納するので，その共通接頭辞の長さは 1 以上である．その共通接頭辞が持つ接尾辞は $T[j..]$ とすると，$j \le \sum_{y=1}^{x-1}|F_y|$ を満たすし，$\sum_{y=1}^{x}|F_y| - j$ は項 F_x の距離になる．

例 1. T = abbabb と $\Sigma = \{$a, b$\}$ とすると，前処理に， T に ba を追加し，$T[-\sigma+1..n]$ = baabbabb になり，LZ 分解は $T[1..n]$ を b · ba · bb に分解する．

符号化　LZ の 項を圧縮するとき，様々の方法が存在するが，各の項を参照先までの距離と長さと成り立つ二つ組で表現されるのは最も自然な方法である．項の参照先が複数にある可能性があるが，universal code を利用すれば，最小の距離を満たす参照先を選択することは最適である．一番近い参照先の選び方は rightmost parsing (rightmost) と呼ぶ．図 1 で例文が挙げられた．

本研究

この研究で，項の距離について圧縮率の方が良い表現を提案する．概念で言えば，LZ 分解を実行しながら，処理済みのテキストの接頭辞を colex 順序で保ち続け，項の距離を計算するとき，テキストの距離ではなく，ソートされている接頭辞の中の距離を利用する．

距離の表現　より詳細には，任意 $i \in [-\sigma..n]$ に対し， $T_i := T[-\sigma+1..i]$ は長さ $\sigma+i$ を持つ接頭辞を示す．ただし，$T_{-\sigma} = \epsilon$．下記では，任意 $p \in [1..n]$ に対し，$T[1..p-1]$ までの LZ 分解を計算でき，$T[p]$ から始まる新しい項 F_x の距離を計算したいと仮定する．F_x の距離を計算するため，$T_{j_1} \prec T_{j_2} \prec \ldots \prec T_{j_{p+\sigma}}$ を colex 順序でソートされている接頭辞 $T_{-\sigma},\ldots,T_{p-1}$ とする．その並んだ接頭辞の列の中に，r_{p-1} は T_{p-1} の位置を示す，すなわち，$j_{r_{p-1}} = p-1$．t_{p-1} は $T[j_{t_{p-1}}+1..j_{t_{p-1}}+|F_x|] = F_x$ を満たす r_{p-1} の一番近い位置，すなわち，$|r_{p-1}-t_{p-1}|$ は最小である．そのとき， $\mathsf{off}_x := r_{p-1} - t_{p-1}$ とする．ただし， off_x は負数になる可能性がある．計算されている二つ組列 $(\mathsf{off}_x, |F_x|)$ を $\mathsf{holz}(T)$ と呼ぶ (**h**igh-**o**rder entropy coding of **LZ** parsing)．図 2 で例文が挙げられた．

復元　$T[1..p-1]$ まで復元できた，かつ，$T[p]$ から始まる項 F_x の二つ組は $(\mathsf{off}_x, |F_x|)$ と前提すると，上記に従って，$p-1$ までの接頭辞を colex 順序でソートされている列で格納したら，off_x から参照先を計算できる．ただし，参照先はもう復元できたテキストの部分に指すので，F_x を計算できる．

実験

提案した表現 holz の有効性を実践的に強調するために，実験を行った．Pizza&Chili corpus のデータセットを取り

(表 1)，各のデータセットの長さ 20 MB が持つ接頭辞を LZ 分解の入力テキストに扱い，計算されている圧縮表現を 図 3 で表す．ただし， universal code として Elias γ code [1] が利用されている．pitches といった経験的エントロピーが高いデータセットで，holz は理想的できないが，他のより低いエントロピーが持つデータセットで，holz の圧縮率は rightmost より強い．

参考文献

[1] P. Elias. Efficient storage and retrieval by content and address of static files. *J. ACM*, 21(2):246–260, 1974.

[2] P. Ferragina, I. Nitto, and R. Venturini. On the bit-complexity of Lempel–Ziv compression. *SIAM J. Comput.*, 42(4):1521–1541, 2013.

[3] D. Köppl, G. Navarro, and N. Prezza. HOLZ: high-order entropy encoding of Lempel-Ziv factor distances. In *Proc. DCC*, pages 83–92, 2022.

[4] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Trans. Information Theory*, 23(3):337–343, 1977.