圧縮テキストに対する文字列照合のための統一的枠組み

2G−2

喜田 拓也　柴田 裕介　竹田 正幸　篠原 歩　有川 節夫
九州大学大学院システム情報科学研究科

1. はじめに

圧縮された文書ファイルの中から必要なファイルを検索するために文字列照合を行う場合，展開の手間が伴うため余分な時間がかかる．そこで，圧縮されたテキストを展開せずに文字列照合を行なうという要求が生まれた．この研究は90年代初頭から始まり，いくつかの研究成果が報告されている．例えば我々のグループ[1, 4]や，チリ大のG. Navarroら[3]の最近の研究成果では，展開を伴う照合をよりも高速に照合が行える実用的アルゴリズムが示された．一方，理論的にはこれまで既存の各圧縮法に対し，個別に照合アルゴリズムが提案されてきたが，Navarroら[3]により初めてLempel-Ziv法 (LZ77,LZ78) に対する統一的な照合アルゴリズムが提案された．N, n, m, r はそれぞれ非圧縮時のテキスト長，圧縮時のテキスト長，パターンの長さ，パターンの出現回数とすると，彼らのアルゴリズムの時間計算量は平均 $O(\min(N, n\log m)+r)$，最悪時 $O(\min(N, mn)+r)$ である．

本論文では，辞書式圧縮法の本質をとらえた新しい枠組み（Collage system）を提案し，その枠組みにおける文字列照合アルゴリズムを示す．この枠組みでは，LZ法のみならず現在主流のほとんどすべての圧縮法を表現することが可能である．Collage systemとは，辞書に相当する変数定義の集合 $\mathcal{D}$ と符号化列に相当する変数列 $\mathcal{S}$ の組である．提案アルゴリズムは $O(||\mathcal{D}||+m^2)$ 領域を使用し，$O((||\mathcal{D}||+|\mathcal{S}|)\cdot height(\mathcal{D})+m^2+r)$ 時間でテキスト中のパターンの出現をすべて報告する．ここで $||\mathcal{D}||, |\mathcal{S}|, m, r$ はそれぞれ $\mathcal{D}$ の大きさ，$\mathcal{S}$ の長さ，パターン長，パターンの出現回数である．

2. Collage system とテキスト圧縮

テキスト圧縮は，原テキスト $\mathcal{T}$ を語の並び $\mathcal{T}=u_1u_2\ldots u_n$ に切り分け，各語 u_i の「表現」を保存し，語の並びをその表現の並びに変換する作業であると考えることができる．本節では，この視点に基づいた統一的枠組みであるCollage systemを紹介する．

Collage system とは以下で定義される組 $\langle\mathcal{D}, \mathcal{S}\rangle$ である．すなわち $\mathcal{D}$ は変数定義の列 $X_1=expr_1; X_2=expr_2; \cdots; X_n=expr_n$ で，$expr_k$ は次の形式のいずれかである．

a	$(a\in\Sigma\cup\{\varepsilon\})$,	文字代入
$X_i\cdot X_j$	$(i,j<k)$,	連結
${}^{[j]}X_i$	$(i<k, j$ は整数$)$,	前 j 文字切り落とし
$X_i^{[j]}$	$(i<k, j$ は整数$)$,	後 j 文字切り落とし
$(X_i)^j$	$(i<k, j$ は整数$)$,	繰り返し

各変数はそれを評価したときに得られる文字列に対応する．例えば，$X_1=a; X_2=b; X_3=X_1\cdot X_2; X_4=(X_3)^3; X_5=X_4^{[1]}$ のとき，変数 X_4 は文字列 *ababab* を，変数 X_5 は文字列 *ababa* を表している．以降では，変数 X_i とそれが表す文字列とを同一視する．$\mathcal{D}$ における変数代入の個数を $\mathcal{D}$ の大きさとし，$||\mathcal{D}||$ と表す．また $\mathcal{D}$ の変数 X の構文木を $\mathcal{T}(X)$ と書き表し，次のように定義する．$\mathcal{T}(X)$ の根は X でラベル付けされ，

- $X=a\in\Sigma\cup\{\varepsilon\}$ のとき，部分木を持たない．
- $X=Y\cdot Z$ のとき，部分木 $\mathcal{T}(Y), \mathcal{T}(Z)$ を持つ．
- $X=(Y)^i, {}^{[i]}Y, Y^{[i]}$ のとき，部分木 $\mathcal{T}(Y)$ を持つ．

変数 X の高さ $height(X)$ は構文木 $\mathcal{T}(X)$ の高さとし，$\mathcal{D}$ の高さを $height(\mathcal{D})=\max\{height(X)|X\in\mathcal{D}\}$ と定義する．

一方，$\mathcal{S}$ は $\mathcal{D}$ において定義された変数の列 $\mathcal{S}=X_{i_1}, X_{i_2}, \ldots, X_{i_k}$ である．$\mathcal{S}$ の長さを $\mathcal{S}$ 中の変数の個数 k とし，$|\mathcal{S}|$ と表記する．Collage system $\langle\mathcal{D}, \mathcal{S}\rangle$ は，$X_{i_1}, X_{i_2}, \ldots, X_{i_k}$ を連結して得られる文字列を表現する．すなわち $\mathcal{D}, \mathcal{S}$ はそれぞれ，圧縮法における辞書の表現と圧縮テキストに対応している．$\mathcal{S}$ は，$\mathcal{D}$ に含めることもできるので本質的には必要ないが，様々な圧縮法をうまくとらえるために分けている．実際の圧縮法においては，$\mathcal{D}$ や $\mathcal{S}$ は様々な方法で符号化され，圧縮率は $||\mathcal{D}||$ や $|\mathcal{S}|$ よりも $\mathcal{D}, \mathcal{S}$ の符号化の方法に左右される．

以下にCollage systemの例を挙げる．記述の簡便のために，複数の変数定義を連結し一つの変数定義で書き表すことを許している．

静的辞書符号化　$\mathcal{S}=X_{i_1}, X_{i_2}, \ldots, X_{i_n}$.

$$\mathcal{D}:\quad X_1=a_1;\quad X_2=a_2;\quad \cdots;\quad X_q=a_q;$$
$$X_{q+1}=w_1;\ X_{q+2}=w_2;\ \cdots;\ X_{q+s}=w_s.$$

ここで $w_k\in\Sigma^+, |w_k|>1$．$\mathcal{S}$ はハフマン符号化等で符号化される．$s=0$ のときは文字単位の符号化であり，圧縮率は $\mathcal{S}$ の符号化方法に依存する．$s>0$ のとき文書群の頻出語を辞書として独立して保存するのが一般的である．

A Unifying Framework for Compressed Pattern Matching
Takuya Kida, Yusuke Shibata, Masayuki Takeda, Ayumi Shinohara, Setsuo Arikawa
Department of Informatics, Kyushu University, Fukuoka, 812-8581 Japan

LZSS 圧縮法 [5]. $\mathcal{S} = X_{q+1}, X_{q+2}, \ldots, X_{q+n}$.

$$\mathcal{D}: \; X_1 = a_1; \quad X_2 = a_2; \quad \cdots; X_q = a_q;$$
$$X_{q+1} = \left(\left({}^{[i_1]}X_{\ell(1)} X_{\ell(1)+1} \cdots X_{r(1)} \right)^{m_1} \right)^{[j_1]} b_1;$$
$$\vdots$$
$$X_{q+n} = \left(\left({}^{[i_n]}X_{\ell(n)} X_{\ell(n)+1} \cdots X_{r(n)} \right)^{m_n} \right)^{[j_n]} b_n.$$

ここで $0 \le i_k, j_k, m_k,\ b_k \in \Sigma$ である.

LZW 圧縮法 [6]. $\mathcal{S} = X_{i_1}, X_{i_2}, \ldots, X_{i_n}$.

$$\mathcal{D}: \; X_1 = a_1; \quad X_2 = a_2; \quad \cdots; \quad X_q = a_q;$$
$$X_{q+1} = X_{i_1} b_{i_2}; \quad X_{q+2} = X_{i_2} b_{i_3}; \quad \cdots;$$
$$X_{q+n-1} = X_{i_{n-1}} b_{i_n}.$$

ここで $\Sigma = \{a_1, \ldots, a_q\}$ で, b_j は文字列 X_j の最初の文字を表現している. $\mathcal{S}$ は整数の列 $i_1, i_2, \ldots, i_n$ として符号化される.

3. 主結果

提案アルゴリズムの基本的な考えは, 非圧縮テキスト上での Knuth-Morris-Pratt(KMP) オートマトンの動作を Collage system 上で模倣することである. 文字列 $u = a_{i_1} a_{i_2} \cdots a_{i_n}, (a_i \in \Sigma)$ の i 文字目から j 文字目までの部分文字列を $u[i:j]$ で表す. このとき, パターン $P = P[1:m]$ に対する KMP オートマトンの遷移関数を $\delta_{\mathrm{KMP}} : \{0, 1, \ldots, m\} \times \Sigma \to \{0, 1, \ldots, m\}$ とする. この δ_{KMP} の定義域を $\{0, 1, \ldots, m\} \times \Sigma^*$ に拡張する. すなわち, $j \in \{0, 1, \ldots, m\}$, $u \in \Sigma^*$, $a \in \Sigma$ について, $\delta(j, \varepsilon) = j$, $\delta(j, ua) = \delta(\delta(j, u), a)$ とする. このとき, D を $\mathcal{D}$ で定義された語の集合とし, 関数 $Jump : \{0, 1, \ldots, m\} \times D \to \{0, 1, \ldots, m\}$ を

$$Jump(j, u) = \delta(j, u).$$

と定義する. また任意の組 $\langle j, u \rangle$ $(j \in \{0, 1, \ldots, m\}, u \in D)$ について, 集合 $Output(j, u) = \{ 1 \le i \le |u| \mid P$ が文字列 $P[1:j] \cdot u[1:i]$ の接尾辞$\}$ を定義する.

提案アルゴリズムはテキスト T を表現する Collage system$\langle \mathcal{D}, \mathcal{S} \rangle$ とパターン P が与えられたとき, $\mathcal{S}$ の変数列を前から走査しながらテキストに含まれるすべてのパターンの出現を報告する. その概略は図 1 の通りである. 証明は省略するが, 関数 $Jump$ と集合 $Output$ に関して以下の定理が得られた.

定理 1 関数 $Jump(j, X)$ は $O(\|\mathcal{D}\| \cdot height(\mathcal{D}) + m^2)$ 時間, $O(\|\mathcal{D}\| + m^2)$ 領域で構築可能である. また任意の $j \in \{0, 1, \ldots, m\}, X \in \mathcal{D}$ について $O(1)$ で値を返す. もし $\mathcal{D}$ が切り落とし操作を含まない場合は, 時間計算量は $O(\|\mathcal{D}\| + m^2)$ となる.

定理 2 集合 $Output(j, X)$ の要素を枚挙する手続きは, $O(\|\mathcal{D}\| \cdot height(\mathcal{D}) + m^2)$ 時間, $O(\|\mathcal{D}\| + m^2)$ 領域で構築可能である. また任意の $j \in \{0, 1, \ldots, m\}, X \in \mathcal{D}$ について $O(height(X) + \ell)$ 時間で動作する. ここで ℓ は集合 $Output(j, X)$ の要素の個数である. もし $\mathcal{D}$ が切り落とし操作を含まない場合は, $O(\|\mathcal{D}\| + m^2)$ 時間領域で構築でき, $O(\ell)$ 時間で動作する.

以上の定理から次の結果が得られた.

定理 3 *Collage system* に対する文字列照合問題は, $O((\|\mathcal{D}\| + |\mathcal{S}|) \cdot height(\mathcal{D}) + m^2 + r)$ 時間, $O(\|\mathcal{D}\| + m^2)$ 領域で解決できる. もし $\mathcal{D}$ が切り落とし操作を含まない場合は, $O(\|\mathcal{D}\| + |\mathcal{S}| + m^2 + r)$ 時間で解決する.

$\|\mathcal{D}\| + |\mathcal{S}|$ は圧縮テキスト長 n に対応するものと考えることができるので, 切り落としがない場合の時間領域計算量は $O(n + m^2 + r)$ と $O(n + m^2)$ となる. これは我々が LZW 圧縮法に対して示した計算量と一致する [2]. また Navarro と Raffinot が示した圧縮法 [3] は, 我々の枠組みで切り落とし操作を含まずに記述できる.

参考文献

1) T. Kida, M. Takeda, A. Shinohara, and S. Arikawa. Shift-And approach to pattern matching in LZW compressed text. In *Proc. 10th Ann. Symp. on Combinatorial Pattern Matching*, Lecture Notes in Computer Science. Springer-Verlag, 1999. to appear.

2) T. Kida, M. Takeda, A. Shinohara, M. Miyazaki, and S. Arikawa. Multiple pattern matching in LZW compressed text. In J. A. Atorer and M. Cohn eds., *Proc. of Data Compression Conference '98*, pp. 103–112. IEEE Computer Society, 1998.

3) G. Navarro and M. Raffinot. A general practical approach to pattern matching over Ziv-Lempel compressed text. In *Proc. 10th Ann. Symp. on Combinatorial Pattern Matching*, Lecture Notes in Computer Science. Springer-Verlag, 1999. to appear.

4) Y. Shibata, T. Kida, S. Fukamachi, M. Takeda, A. Shinohara, T. Shinohara, and S. Arikawa. Byte pair encoding: a text compression scheme that accelerates pattern matching. Technical Report DOI-TR-CS-161, Department of Informatics, Kyushu University, April 1999.

5) J. Storer and T. Szymanski. Data compression via textual substitution. *J. Assoc. Comput. Mach.*, 29(4):928–951, Oct 1982.

6) T. A. Welch. A technique for high performance data compression. *IEEE Comput.*, 17:8–19, June 1984.

Input. A collage system$\langle \mathcal{D}, \mathcal{S} \rangle$ and a pattern $P = P[1:m]$.
Output. All positions at which P occurs in T.

```
/* 前処理 */
Perform the processing required for Jump and Output;
/* 圧縮テキスト走査 */
let S = X_{i_1} X_{i_2} ... X_{i_n}.
ℓ := 0; state := 0;
for k := 1 to n do begin
  for each p ∈ Output(state, X_{i_k}) do
    Report a pattern occurrence at position ℓ + p − m + 1;
  state = Jump(state, X_{i_k});
  ℓ := ℓ + |X_{i_k}|
end
```

図 1 提案アルゴリズム.