Text Compression and Compressed String Mining

後藤, 啓介

https://doi.org/10.15017/1441265

（別紙様式２）

氏　　名　：　後藤　啓介

論文題名　：　Text Compression and Compressed String Mining
（テキスト圧縮と圧縮文字列マイニング）

区　　分　：　甲

論　文　内　容　の　要　旨

インターネット上を流通するデータ量が指数関数的に増大し，科学技術分野ではセンシング技術等の発達を背景に種々の観測・実験データが巨大化している．また，企業や機関等においては保管が必要な内部文書が増加の一途を辿っている．このような大規模データから価値ある知識を抽出し利活用したいという要求が，学界のみならず産業界でも高まっている．これらのデータの多くは定まった形式を持たない非定型データ，すなわち，文字列データと捉えることができる．

通常，大規模データの処理には膨大な計算資源が必要となるため，計算対象とするデータ量を制限せざるを得ず，データを十分に利活用できないというジレンマに陥る．そこで，本研究では，「データ圧縮」を核に据え，大規模文字列データマイニングの高速化と省領域化の両方を同時に達成する手法の開発に取り組んだ．

データ圧縮とは，データに含まれる規則性や統計的性質に着目することにより冗長性を除去し，データの表現長を短くする技術をいう．データ圧縮により記憶容量や通信コストを節約することができるが，その反面，データ利用時には伸張作業が必要となる．だがもし，圧縮データを伸張することなく直接処理できれば，このオーバヘッドは解消される．このような背景の下，「圧縮データ処理」の研究は，1990年代より今日まで世界の各所で行われている．この研究の目標は，

[目標１]　伸張時間　＋　非圧縮データ処理時間　＞　圧縮データ処理時間

であり，主にパターン照合問題を対象とし様々なデータ圧縮形式について目標１が達成されている．一方，本研究室では，より挑戦的な目標として，

[目標２]　　非圧縮データ処理時間　＞　圧縮データ処理時間

を掲げ，いくつかの圧縮形式に関してこの目標が達成できることを示した．これは，圧縮を高速化のための前処理と捉えるものであり，「圧縮による高速化」という新しい研究潮流に繋がっている．

しかしながら，圧縮データ処理の研究の多くはパターン照合問題やその変種に限られており，文字列データマイニングに関わる文字列処理については，ほとんど行われていない．そこで本研究では，文字列データマイニングに必要な文字列処理としてq-グラム頻度問題およびその変種に取り組んだ．ここで，q-グラムとは，テキストに出現する長さ q の部分文字列をいい，q-グラム頻度問題とは，テキストに出現する全 q-グラムの頻度を求める問題をいう．q-グラム頻度は文字列の特徴を表す重要な指標のひとつであり，機械学習や分類問題などに応用されている．圧縮による高速化と省領域化を同時に達成することを目指して，以下の２つの研究項目を置いた．

（A）圧縮テキスト上で動作する効率的な q-グラム頻度アルゴリズムの開発．

（B）高い圧縮率をもつ高速かつ省領域な圧縮アルゴリズムの開発．

圧縮データ形式として，直線的プログラム（Straight Line Program; SLP）を採用する．SLPは，単一

の文字列を導出するチョムスキー標準型の文脈自由文法を指す. SequiturやRe-Pair等の文法圧縮法はもちろん, Lempel-Ziv法などの既存圧縮法の多くは, 圧縮データフォーマット自体がこのSLPとみなせるか，または，容易にSLPに変換できることが知られている．

研究項目(A)のq-グラム頻度問題について，Inenaga & Bannaiは，サイズ n のSLPから q-グラム頻度を求める$O(\sigma^q qn^2)$時間アルゴリズムを提案した．ここに，σ はアルファベットサイズを表す．だが，このアルゴリズムの計算時間は，圧縮サイズ n に関して多項式であるものの q に関して指数的であるため, 実用からは程遠い代物であった. 本研究では, この先行研究を大きく上回る $O(qn)$ 時間アルゴリズムを開発した. 非圧縮データを入力とした線形時間アルゴリズムとの比較実験により，qが小さい場合，英語テキスト，XML，塩基配列をはじめとした様々な実データについて提案手法が高速であり，最大で5.7倍高速であることが判明した．すなわち，qが小さい場合には上述の目標2が達成できることを示したものである．

qが大きい場合，qnの値が非圧縮文字列長Nを超え，上述の$O(qn)$時間アルゴリズムは非圧縮データ上の線形時間アルゴリズムよりも遅くなってしまう．そこで，本研究では，この問題を解決すべく，さらに高速な$O(N - \mathrm{dup}(q, D))$時間アルゴリズムを開発した．ここで，$\mathrm{dup}(q, D)$は圧縮データ$D$において捉えられた$q$-グラムに関する冗長性の量を表している．すなわち，圧縮アルゴリズムによって捉えられた冗長性が多ければ多いほど高速化できることを明示的に示している．これはTに冗長さが含まれている場合($N > n$)，非圧縮文字列に対するq-グラム頻度計算アルゴリズムの下界である$\Omega(N)$時間を打破するという驚くべき結果であり，理論的に目標2を達成したものである．

さらに，q-グラムの出現を重複して数えない非重複q-グラム頻度問題にも取り組み，上で得られた知見を基に，SLP上で動作する$O(q^2 n)$時間アルゴリズムを開発した．

研究項目(B)については，与えられた文字列を導出する最小文法を求める問題はNP困難であることが知られており，これまで様々な近似アルゴリズムが提案されてきた．Rytterは入力文字列をいったんLZ77法で圧縮しそれから近似率$O(\log N)$のSLPへと変換する$O(z \log N)$時間近似アルゴリズムを提案した，ここでzはLZ77法で圧縮した時の圧縮サイズである．このアルゴリズムを大規模データに適用する際には，LZ77圧縮にかかる時間と領域がボトルネックとなる．そこで，高速かつ省領域のLZ77圧縮アルゴリズムを開発し，このボトルネックを解消した．

LZ77圧縮の核となるのは，入力文字列中の位置iについて，iから始まる部分文字列と最長一致する，i以前に出現する部分文字列の出現位置PrevOcc(i)とその一致長LPF(i)の計算である．既存手法の多くは，前処理においてこれらの情報を格納したPrevOcc, LPF配列をいかに効率的に計算するかに重きを置いていた．しかし，LZ77圧縮に必要なのは，これらの配列の一部であり，必ずしも配列全体を計算する必要はない．本研究では，PrevOcc, LPF配列の必要な部分のみを計算するというアイデアに基づき，アルゴリズム BG3, BG4, BG5 を開発した．作業領域は，それぞれ，$3N \log N$, $4N \log N$, $5N \log N$ビットである．これらのアルゴリズムを実装し，既存の線形時間アルゴリズムの中で最も高速なLZOGとの性能比較を行った. LZOGの作業領域は $3N \log N$ビットである. 実験結果により，BG3, BG4, BG5のいずれもが LZOG より高速であり，特に BG5では最大で2〜3倍高速であることが判明した．ほぼ同時期に，Karkkainenらは同様のアイデアを用いて作業領域 $2N \log N$ ビットのアルゴリズムを提案している．本研究では，さらに作業領域を $N \log N + O(\sigma \log N)$ ビットにまで減らしたアルゴリズムBG1の開発に成功した．