様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成　３０　年　　６　月　２０　日現在

機関番号：　１７１０２
研究種目：　基盤研究(A)（一般）
研究期間：　2013～2017
課題番号：　２５２４０００３

研究課題名（和文）情報爆縮基盤技術

研究課題名（英文）Information Implosion Foundational Technology

研究代表者
竹田　正幸（Takeda, Masayuki）

九州大学・システム情報科学研究院・教授

研究者番号：５０２１６９０９

交付決定額（研究期間全体）：（直接経費）　　35,600,000円

研究成果の概要（和文）：爆縮とは工学用語で，爆発の圧力が;外側ではなく内側へ集中する現象をいい，通常では得難い物理現象を発生させるために利用される．本研究では，膨れ上がるデータを爆発的に凝縮することにより，(i) データ量削減，(ii) データ処理の高速化，(iii) 知識獲得の三つを達成する基盤技術の確立を目指し，これを情報爆縮（information implosion）と名付けた．情報爆縮基盤技術の確立のために，(A)高速データストリーム圧縮アルゴリズム，(B)圧縮データ上の高速データ処理アルゴリズム，(C)大規模データ解析アルゴリズムという３つの研究項目をおいて研究開発を行い多くの成果を得た．

研究成果の概要（英文）：Implosion is a term used to describe a physical process in which an object violently collapses inward, used to generate phenomena which would not be possible in a normal environment. In this research, we aimed to lay the foundations of the technology which simultaneously realize the following three functionalities: (i) space-efficient data representation, (ii) time-efficient data processing, and (iii) knowledge acquisition. We name this scheme information implosion. We worked on developing (A) efficient compression algorithms for streaming data, (B) efficient string processing algorithms over compressed data, and (C) efficient data analysis algorithms for large-scale data, which will be key to realizing a new generation of information systems.

研究分野：　アルゴリズム

キーワード：　データ圧縮　圧縮データ構造　簡潔データ構造　文字列パターン照合　文字列パターン発見　文字列データ解析　文字列アルゴリズム

１．研究開始当初の背景

人類によって生み出されるデータの増加はとどまるところを知らない．インターネット上を流通するデータ量は指数関数的に増大しており，科学技術分野では，センシング技術の発達等を背景に種々の観測データが巨大化している．企業や機関においては，SOX 法施行後，証跡提出要求への対応のために保管が必要な内部文書は増加の一途をたどっている．米国では，これら膨大な量のデータを最大限に活用し，国家が直面する喫緊の課題への取り組みに役立てることを目的とした 2 億ドル規模の「ビッグデータ研究開発イニシアティブ」を 2012 年 3 月に発表し，国家の重要戦略目標に位置づけている．

データの巨大化に伴い，データベース（DB）管理システムはより多くの計算資源（ストレージ，主記憶，演算能力）が要求される．このため，DB への格納は多くの機関や企業にとって費用対効果の面から事実上不可能となりつつある．今後，ハードウェア技術の進展により記憶容量の増大や処理速度の向上が期待されるが，その向上は ICT 機器の利用を促しデータ量の増大に拍車をかけ，この問題はいっそう顕在化するであろう．

２．研究の目的

データ圧縮は，データの規則性に基づいて冗長性を除去することによりデータの記述長を減らす技術をいう．また，知識発見は，データに潜む規則性を計算機によって見出す技術である．したがって，「圧縮」と「発見」の間には深い関係がある．実際，機械学習・機械発見の分野では，最小記述長原理や正規化圧縮距離などデータ圧縮に基づく手法がよく用いられる．一方，上述の「圧縮による高速化」は，入力データ中の冗長性を除去することにより計算そのものを高速化する．すなわち，「圧縮」と「高速計算」の間にも密接な関係がある．そこで，申請者らは，「圧縮＝高速計算＝発見」という視点から，膨れ上がるデータを爆発的に凝縮することによって，高圧縮率，高速データ処理，大規模データ解析，という３つの課題を同時に解決できるとの着想に至り，この技術を情報爆縮(Information Implosion)と名付けた．

本研究では，研究代表者らの「圧縮による高速化」技術を中心に据え，簡潔データ構造や文法圧縮の技術を有機的に結合させ，データ圧縮・アルゴリズム・知識発見の各分野の最新の研究成果を援用しながら，情報爆縮のための基盤技術を確立し，低コストでデータを格納・管理・検索・解析できる新しい情報システムの実現を目指す．

３．研究の方法

本研究では，「圧縮＝高速計算＝発見」という独自の視点に立ち，膨れ上がるデータを爆発的に凝縮し効率的に処理することの可能な情報爆縮基盤技術の確立を目指す．初めの２年間は記号列データを対象とし，３年目以降より対象を数値データ及び半構造データに広げてゆく．

基本となる圧縮スキーマとして文法圧縮を採用する．文法圧縮とは，与えられた入力記号列から，その記号列のみを生成する文脈自由文法(SLP)を生成し符号化する圧縮スキーマである．これを採用する理由は次の３つである．(i)高度反復集合に対し高圧縮率が期待できること．(ii)高速なランダムアクセスが可能であること．(iii)圧縮によるデータ処理の高速化が期待できること．LZ77 圧縮法の拡張である相対 LZ 圧縮法は，(i),(ii)をある程度満たすが，(iii)を満たさない．

次の３つの研究項目をおいて研究を行う．(A)高速データストリーム圧縮アルゴリズム．(B)圧縮データ上の高速データ処理アルゴリズム．(C)大規模データ解析手法．

４．研究成果

本研究では，5 年の研究期間に数多くの優れた成果をあげることができた．ここでは，３つの研究項目ごとに成果の概要を述べる．

(1)高速データストリーム圧縮アルゴリズム．既に開発していた ESP に基づくストリーム型の文法圧縮アルゴリズムについて高速化と省領域化を行った．さらには，アルゴリズムのハードウェア化に成功し，VLDB workshop 2015 Best Paper Award の受賞など，高い評価を得ている．

また，LZW 圧縮にヒントを得た，LZD 文法圧縮アルゴリズムと呼ぶ新しい実用的圧縮アルゴリズムを考案した．

(2)圧縮データ上の高速データ処理アルゴリズム．制約付き最長共通部分列の計算，連計算，アーベル規則性の抽出など，様々な文字列処理に対して，RLE 圧縮法による高速化に成功した．また，(1)で述べた ESP に基づく文法圧縮アルゴリズムに基づき，世界初のオンライン圧縮索引を実現している．さらに以下では，動的圧縮データ構造に関する研究成果について述べる．

情報システムに格納されたデータは頻繁に更新され，それに伴ってデータ構造更新の必要が生じる．たとえば，接尾辞配列は文字列データに対する索引構造として広く知られているが，任意の位置への文字列の挿入・削除といったデータ更新が生じた際，配列の更新に最悪の場合再構築と同等の時間を要する．そこで，効率的な更新の可能な「動的データ構造」の研究が行われている．しかし，その多くは圧縮されないデータ構造に対するもので，効率的更新の可能な圧縮データ構

造の研究は，ほとんど行われていない．

本研究では，Signature Encoding (SE)と呼ばれる文字列表現法に着目しこれを文法圧縮の一種と捉え，文字列の SE 表現に基づいて基本的なクエリを効率よく実行できる静的および動的な圧縮データ構造の開発に取り組んだ．その結果，LCE クエリに対する動的な圧縮データ構造を開発し，これに基づいて，動的な圧縮索引構造の開発に成功した．

T を圧縮の対象のテキストとする．M を $M \geq |T|$を満たす整数とし，ワード RAM モデルの下で M 以下の非負整数を定数時間で扱えるものと仮定する．このとき，T の SE のサイズ w は，$w = O(\min\{z\log|T|\log^* M, |T|\})$を満たすことが知られている．ここに，$z$ は T に対する自己参照なしの LZ77 分解の項数である．

本研究では，SE に基づくデータ構造に関して以下の①～⑤の成果を得た．

①SE の効率的な構築．本研究では，効率的な SE 構築アルゴリズムを開発した．既存研究との比較を下表に示す．ここに，$f_A = O(\min\{\text{loglog}M\text{loglog}\, w / \text{logloglog}\, M, (\log w/\text{loglog}\, w)^{1/2}\})$ である．

	構築時間	作業領域（ワード）
単純な手法	$O(f_A\|T\|\log^*\|T\|)$	$O(\|T\|)$
Alstrup et al.	$O(f_A\|T\|\log^*\|T\|)$	$O(\|T\|)$
	$O(f_A\|T\|\log\|T\|\log^*\|T\|)$	$O(w)$
Cormode &Muthukrishnan	$O(\|T\|\log^*\|T\|)$	$O(\|T\|)$
本研究	$O(\|T\|)$	$O(\|T\|)$
	$O(f_A\|T\|)$	$O(w)$

また，テキスト T が圧縮形式で与えられた場合について次の結果を得た．

【定理】T が自己参照なし LZ77 形式で与えられたとき，T の SE を $O(z\, f_A \log|T|\log^* M)$ 時間と $O(w)$領域で構築できる．ここに，z は T の LZ77 分解の項数である．

【定理】 Tが SLP 形式で与えられたとき，Tの SE を $O(n\, f_A \log|T|\log^* M)$ 時間と $O(w)$領域，または，$O(n\text{loglog}(n\log^* M)\log|T|\log^* M)$時間と $O(n\log^* M+w)$領域で構築できる．ここに，n は SLP のサイズである．

②動的テキストに対する SE の更新．テキスト T の任意の位置に対する部分文字列の挿入・削除が生じた場合に，T の SE G を効率的に更新する問題にも取り組み，次の成果を得た．ここに，$M \geq 4|T|$とする．

【定理】サイズ w の SE G に対し，T の任意位置に対する長さ y の部分文字列の挿入・削除に伴う G の更新を $O((y+\log|T|\log^* M)f_A)$時間で実行可能なサイズ $O(w)$のデータ構造を，$O(w\, f_A)$時間で構築できる．

③SE に基づく動的圧縮データ構造．圧縮データ構造は圧縮データベースシステムにおいてカギとなる役割を果たす．圧縮データベースシステムにおいては，テキストデータの更新に伴うデータ構造の更新作業を効率的に行う必要がある．更新の頻度が低い場合にはその都度再創成すれば良い．しかし，ある程度頻繁に更新が生じる場合には，効率的な更新が可能な圧縮データ構造が必要である．

④SE に基づく動的圧縮 LCE データ構造．LCE クエリは，文字列を扱う多くの問題の副問題として現れるため，その効率的なデータ構造に関する研究が盛んに行われている．そこで本研究では，動的圧縮 LCE データ構造の研究に取り組んだ．文字列 x, y の最長共通接頭辞を $\text{lcp}(x, y)$で表す．整数 $i, j \in [1..|T|]$ に対して文字列 $\text{lcp}(T[i..], T[j..])$ の長さを返す基本クエリを $\text{LCE}(i, j)$とする．

【定理】テキスト T の SE G を用いることにより，クエリ $\text{LCE}(i, j)$ に $O(\log|T|+\text{logs}\log^* M)$ 時間で応答可能である．ここで，s は クエリ $\text{LCE}(i, j)$ に対する解である．

⑤SE に基づく動的圧縮索引．$\text{Find}(P)$ を $P \in \Sigma^+$に対し，P の T における出現位置の集合 $\text{Occ}_T(P)$ を返す基本クエリと定める．次の結果を得た．

【定理】 T の SE G が与えられたとき，以下の性質を満たすサイズ $O(w)$のデータ構造を構築できる．

- クエリ $\text{Find}(P)$に $O(|P|f_A + \log w\log|P|\log^* M (\log|T|+\log|P|\log^* M)+|\text{Occ}_T(P)|\log|T|)$ 時間で応答可能．
- T の任意位置への文字列の挿入・削除に対し $O((y+\log|T|\log^* M)\log w\log|T|\log^* M)$時間で更新可能．ここに y は挿入・削除する文字列の長さである．

(3)<u>大規模データ解析手法</u>．圧縮データおよび非圧縮データ上での様々な解析手法の研究を行った．引用文献⑤の成果は，学習アルゴリズムに対し圧縮による高速化を図った世界初の試みで，WALCOM 2018 Best Paper Award の受賞など，国内外で高く評価されている．以下では，最適パターン発見問題に関する実用的な成果について述べる．

Π をパターン族とする．

【定義】[Π に関する最適パターン発見問題]
入力: Σ^+の有限部分集合 S, T $(S \cap T \neq \varnothing)$.
出力: $\text{score}(P; S, T)$を最大にする $P \in \Pi$.

S の元を正例，T の元を負例とよぶ．目的関数 score には，正例のほとんどに合致しか負例のほとんどと合致しない（またはその逆）パターンに高い値を割り当てるような関数を用いる．パターン $P \in \Pi$ と合致する正例数，負例数を，それぞれ，x_P, y_P とする．

$$\text{score}(P; S, T) = f(x_P, y_P)$$

とおき，

$$f(x,y) = -(|S|+|T|)^{-1}((x+y)I(x/(x+y))+(x'+y')I(x'/(x'+y')))$$

と定める．ここで，$x' = |S| - x$, $y' = |T| - y$ であり，

不均衡度関数 I: $[0,1]\to\mathbb{R}$ は，$I(0)=I(1)=0$, $I(1/2)>0$ を満たし，$[0, 1/2]$で単調増加し$[1/2, 1]$で単調減少するという性質をもち，情報エントロピー関数，分類エラー，Gini 指標などが用いられる．

パターン族 Π として，部分文字列パターンを用いた場合，最適パターン発見問題を解く線形時間アルゴリズムが存在する．しかし，部分列パターンおよび VLDC パターンに対する最適パターン発見問題は NP 困難であることが知られている．そこで，この問題を実用的時間内で解くための手法が望まれる．

Hirao ら[②]は部分列パターン族に対して最適パターン発見問題を解く実用的手法を提案している．また，Inenaga ら[①]はその手法を VLDC パターン族へ拡張している．このアルゴリズムは，基本的に，可能なすべての VLDC パターンを枚挙し，各々の score を求めながら score 最大のパターンを求める．すなわち，パターンを節点とする仮想的な探索木を深さ優先探索する．この探索において有効な枝刈手法を提案している．この枝刈手法は，score 関数の定義に用いた関数 f が単峰性をもつ(unimodal)ときに有効である．不均衡度関数 I として上述の情報エントロピー関数，分類エラー，Gini 指標を用いる場合，関数 f は単峰性をもつことが示されている．VLDC パターン全体の集合を Π_{vldc} とし，統語的に定義される Π_{vldc} 上の自然な半順序を $\preceq_{\mathrm{vldc}}$ で表す．このとき，次の定理が成り立つ．

【定理】関数 f: $[0,|S|]\times[0,|T|]\to\mathbb{R}$ が単峰関数ならば，任意の P, $Q\in\Pi_{\mathrm{vldc}}$ に対して以下が成り立つ．

$P\preceq_{\mathrm{vldc}} Q \Rightarrow$
$f(x_Q, y_Q)\le \max\{f(x_P,y_P), f(x_P,0), f(0,y_P), f(0,0)\}$

この定理に基づき枝刈を行う．

本研究では，最適な飽和パターン[③]を出力する最適飽和パターン問題を定義し，VLDC パターンの族 Π_{vldc} に対し，有効なパターン発見手法を開発した．

Π をパターン族とし，$\preceq$, $\equiv$ を，それぞれ，Π 上の半順序，同値関係とする．パターン $P\in\Pi$ が飽和であるとは，P が $[P]_{\equiv}$ における $\preceq$ に関する極大元であるときをいう．任意の P, $Q\in\Pi$ について，以下を仮定する．

$P\equiv Q \Rightarrow$ score(P)=score(Q)

【定義】[$<\Pi, \preceq, \equiv>$に関する最適飽和パターン発見問題]

入力: Σ^+の有限部分集合 S, T ($S\cap T\neq\varnothing$).

出力: score(P; S, T)を最大にする飽和パターン $P\in\Pi$.

本研究では，I ら[④]の飽和パターン枚挙アルゴリズムと上述のパターン発見手法を結合することにより，新たな最適飽和パターン発見アルゴリズムを得た．すなわち，すべての飽和パターンを枚挙し各々の score を求め，score を最大にするパターンを求める．

計算機実験によりアルゴリズムの性能比較を行った．実験においては，I らの用いた Π_{vldc} 上の同値関係のうち，$\equiv_{\mathrm{MXG}}$ を採用する．その理由は以下のとおりである．

- 対称性と単調性をもつこと．引用文献②で用いる $\equiv_{\mathrm{B}}$ は対称性を欠くこと．
- 引用文献④で示された対称性をもつ同値関係のうち，$\equiv_{\mathrm{MXG}}$ が最も粗いこと．

アルゴリズムを C 言語で実装し，計算機実験を行った．計算機環境は MacPro (Early2008), 3.2 GHz Quad Core Xeon x 2 = 8 CPU, 18GB Memory (DDR2 FB-DIMM 800MHz), OS は Mac OS X 10.6.8 である．コンパイラには g++ (MacPorts gcc48 4.8.1 3) 4.8.1 を用いた．

入力データとしては，以下の 2 種類のテキストデータを用いた．

SmallData: アプリ通信ログの各レコードのデータ項目を，Host と User-Agent の 2 項目に絞ったもの(約 564KBytes).

LargeData: 項目を絞らず全項目をそのまま用いたデータもの(約 2.6 MBytes)

SmallData に対する計算時間を表 1 に示した．なお，パターン中のワイルドカードの個数を高々3 個に制限している．

表 1: SmallData に対する計算時間(sec).

アプリ	正例	負例	非飽和パターンの枝刈 無	非飽和パターンの枝刈 有	非飽和パターンの枝刈 有
			単峰性に基づく枝刈 有	単峰性に基づく枝刈 無	単峰性に基づく枝刈 有
a1	695	4259	68.64	985.98	12.13
a2	440	4514	145.66	992.22	19.80
a3	384	4570	>300min	986.21	674.52
a4	240	4714	>300min	991.61	681.88
a5	201	4753	>300min	987.44	675.59

表には，２つの枝刈手法のうち少なくとも一方を使用した３つの場合について計算時間を示している．

正例とするアプリが異なっても，正例と負例を併せたデータ全体は変わらない．このため，単峰性に基づく枝刈を行わない場合には，同一のデータに対して飽和パターンの枚挙を行っているにすぎない．したがって，枝刈を行わない場合の計算時間は正例によらずほぼ一定となっている．一方，単峰性に基づく枝刈を行う場合の計算時間は，正例によってばらついている．しかし，いずれの場合も枝刈の効果が現れている．このように，単峰性に基づく枝刈手法は，最適飽和パターン発見においても有効に働くことがわかる．

なお，単峰性に基づく枝刈のみを用いた場合については，圧倒的に計算時間を要したため，計算を途中で打ち切っている．このことから，少なくともこの実験においては，飽和性のチェックに要するオーバヘッドよりも，不飽和パターンに対する枝刈の効果が上回っていることが確認された．

LargeData に対する実験結果については，詳細は省くが，提案手法では，速いもので 8 分程度，遅いものでも 140 分程度で計算が完了した．

（4）その他の研究成果．時系列データ処理において重要となる順序保存パターン照合問題を，木構造や DAG 構造に拡張する研究を行った．索引構造として著名な CDAWG の効率的構築や，CDAWG の節点に対応することが知られる「部分文字列同値類」を，接尾辞配列を用いて効率的に計算するアルゴリズムの開発にも成功している．

文字列は，そのままでは陽には構造をもたない記号の連鎖であるから，そこに潜む規則性を同定することは極めて重要である．規則性として，回文，周期，連，被覆などがある．そこで，これらに関わる研究として，回文分解に関する NP 完全性の証明，連定理の証明，異なる全てのスクエアを求める線形時間アルゴリズム，α-ギャップ付き反復や回文に関する最適アルゴリズム，最長共通回文部分列に関するアルゴリズム，最短ユニーク部分文字列の個数のバウンドに関する研究，最短ユニーク回文部分文字列クエリに対する最適アルゴリズム，Lyndon 分解や Lyndon 木に関する逆問題，最長片腕ギャップ付き回文を求めるアルゴリズムなどの成果を得ている．特に，連定理の証明は，15 年来の未解決問題にエレガントな解を与えたものとして高く評価されている．

＜引用文献＞

① S. Inenaga, et al.: Discovering best variable-length-dont-care patterns. *Proc. DS 2002*, 86–97, 2002.

② M. Hirao, et al.: A practical algorithm to find the best subsequence patterns. *Theor. Comput. Sci.*, **292**:465–479, 2003.

③ H. Arimura and T. Uno: Mining maximal flexible patterns in a sequence. *Proc. New Frontiers in Artificial Intelligence*, 307–317, 2007.

④ T. I, et al.: General algorithms for mining closed flexible patterns under various equivalence relations. *Proc. ECML/PKDD 2012*, 435–450, 2012.

⑤ T. Fujita, K. Hatano, E. Takimoto: Boosting over Non-deterministic ZDDs. *Proc. WALCOM 2018,* 195-206, 2018.

５．主な発表論文等

〔雑誌論文〕（計 120 件）すべて査読あり

① J. Fischer, T. I, D. Köppl, K. Sadakane: Lempel-Ziv factorization powered by space efficient suffix trees. *Algorithmica*, **80**(7), 2048-2081, 2018. DOI:10.1007/s00453-017-0333-1

② S. Fukunaga, Y. Takabatake, T. I, H. Sakamoto: Approximate frequent pattern discovery in compressed space. *IEICE Transactions*, **101-D**(3):593-601, 2018. http://search.ieice.org/bin/summary.php?id=e101-d_3_593

③ K. Narisawa, H. Hiratsuka, S. Inenaga, H. Bannai, M.Takeda: Efficient computation of substring equivalence classes with suffix arrays. *Algorithmica*, 79(2):291-318, 2017. DOI:10.1007/s00453-016-0178-z

④ H. Bannai, T. I, S, Inenaga, Y. Nakashima, M. Takeda, K. Tsuruta: The "runs" theorem. *SIAM J. Comput.*,**46**(5): 1501-1514, 2017. DOI:10.1137/15M1011032

⑤ Y. Tanimura, T. Nishimoto, H. Bannai, S. Inenaga, M. Takeda: Small-space LCE data structure with constant-time queries. *Proc. MFCS 2017*, 10:1-10:15, 2017. DOI:10.4230/LIPIcs.MFCS.2017.10

⑥ T. Takagi, S. Inenaga, K. Sadakane, H. Arimura: Packed compact tries: a fast and efficient data structure for online string processing. *IEICE Transactions*, **100-A**(9):1785-1793, 2017. http://search.ieice.org/bin/summary.php?id=e100-a_9_1785

⑦ Y. Takabatake, T. I, H. Sakamoto: A space-optimal grammar compression. *Proc. ESA 2017*, 67:1-67:15, 2017. DOI:10.4230/LIPIcs.ESA.2017.67

⑧ T. I, Y. Nakashima, S. Inenaga, H. Bannai, M. Takeda: Faster Lyndon factorization algorithms for SLP and LZ78 compressed text. *Theor. Comput. Sci.*, **656**: 215-224, 2016. DOI:10.1016/j.tcs.2016.03.005

⑨ Y. Matsuoka, T. Aoki, S. Inenaga, Hideo Bannai, M. Takeda: Generalized pattern matching and periodicity under substring consistent equivalence relations. *Theor. Comput. Sci.*, **656**: 225-233, 2016. DOI:10.1016/j.tcs.2016.02.017

⑩ Y. Tanimura, T. I, H. Bannai, S. Inenaga, S.J. Puglisi, M. Takeda: Deterministic sub-linear space LCE data structures with efficient construction. *Proc. CPM 2016*, 1:1-1:10, 2016. DOI:10.4230/LIPIcs.CPM.2016.1

⑪ T. Nishimoto, T. I, S. Inenaga, H. Bannai, M. Takeda: Fully dynamic data structure for LCE queries in compressed space. *Proc. MFCS 2016*, 72:1-72:15, 2016. DOI:10.4230/LIPIcs.MFCS.2016.72

⑫ T. Nishimoto, T. I, S. Inenaga, H. Bannai, M. Takeda: Dynamic index and LZ factorization in compressed space. *Proc. Stringology 2016*, 158-170, 2016. http://www.stringology.org/event/2016/p14.html

⑬ T. I, W. Matsubara, K. Shimohira, S. Inenaga, H. Bannai, M. Takeda, K. Narisawa, A. Shinohara: Detecting regularities on grammar-compressed strings. *Inf. Comput.*, **240**:74-89, 2015. DOI:10.1016/j.ic.2014.09.009

⑭ T. I, T. Nishimoto, S. Inenaga, H. Bannai, M.

Takeda: Compressed automata for dictionary matching. *Theor. Comput. Sci.*, **578**: 30-41, 2015.
DOI:10.1016/j.tcs.2015.01.019
⑮ K. Goto, H. Bannai, S. Inenaga, M. Takeda: LZD factorization: simple and practical online grammar compression with variable-to-fixed encoding. *Proc. CPM 2015*, 219-230, 2015.
DOI:10.1007/978-3-319-19929-0_19
⑯ P. Bille, G. M. Landau, R. Raman, K. Sadakane, S. Rao Satti, O. Weimann: Random access to grammar-compressed strings and trees. *SIAM J. Comput.*, **44**(3), 513-539, 2015.
DOI:10.1137/130936889
⑰ Y. Takabatake, Y. Tabei, H. Sakamoto: Online self-indexed grammar compression. *Proc. SPIRE 2015*, 258-269, 2015.
DOI:10.1007/978-3-319-23826-5_25
⑱ J. Yamamoto, T. I, H. Bannai, S. Inenaga, M. Takeda: Faster compact on-line Lempel-Ziv factorization. *Proc. STACS 2014*, 675-686, 2014.
DOI:10.4230/LIPIcs.STACS.2014.675
⑲ G. Navarro, K. Sadakane: Fully functional static and dynamic succinct trees. *ACM Trans. Algorithms*, **10**(3):16:1-16:39, 2014.
http://doi.acm.org/10.1145/2601073
⑳ H. H. Do, J. Jansson, K. Sadakane, W.-K. Sung: Fast relative Lempel-Ziv self-index for similar sequences. *Theor. Comput. Sci.*, **532**:14-30, 2014.
DOI:10.1016/j.tcs.2013.07.024

〔学会発表〕（計 83 件）すべて査読あり

① H. Bannai, T. I, S. Inenaga, Y. Nakashima, M. Takeda, K. Tsuruta: A new characterization of maximal repetitions by Lyndon trees. *Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms* (SODA 2015), San Diego, CA, USA, January 4-6, 2015.

〔図書〕（計 4 件）

① 定兼邦彦: 簡潔データ構造. 共立出版, 2018 (215 ページ).
② M. Takeda, A. Shinohara: Pattern matching on compressed text. *Encyclopedia of Algorithms*, 1538-1542, Springer, 2016.
③ H. Bannai: *Grammar compression*. *Encyclopedia of Algorithms*, 861-866, Springer, 2016.
④ G. Navarro, K. Sadakane: Compressed tree representations. *Encyclopedia of Algorithms*, 397-401, Springer, 2016.

６．研究組織

(1)研究代表者

竹田　正幸（TAKEDA，Masayuki）
九州大学・大学院システム情報科学研究院・教授
研究者番号：５０２１６９０９

(2)研究分担者

定兼　邦彦（SADAKANE，Kunihiko）
東京大学・大学院情報理工学系研究科・教授
研究者番号：２０３２３０９０

坂本　比呂志（SAKAMOTO，Hiroshi）
九州工業大学・大学院情報工学研究院・教授
研究者番号：５０３１５１２３

瀧本　英二（TAKIMOTO，Eiji）
九州大学・大学院システム情報科学研究院・教授
研究者番号：５０２３６３９５

坂内　英夫（BANNAI，Hideo）
九州大学・大学院システム情報科学研究院・准教授
研究者番号：２０３２３６４４

稲永　俊介（INENAGA，Shunsuke）
九州大学・大学院システム情報科学研究院・准教授
研究者番号：６０４４８４０４

喜田　拓也（KIDA，Takuya）
北海道大学・情報科学研究科・准教授
研究者番号：７０３４３３１６

畑埜　晃平（HATANO，Kohei）
九州大学・基幹教育院・准教授
研究者番号：６０４０４０２６

成澤　和志（NARISAWA，Kazuyuki）
東北大学・情報科学研究科・助教
研究者番号：４０５８３３２３

井　智弘（I，Tomohiro）
九州工業大学・若手研究者フロンティア研究アカデミー・特任助教
研究者番号：２０７７３３６０

中島　祐人（NAKASHIMA，Yuto）
九州大学・大学院システム情報科学研究院・助教
研究者番号：８０８０４６８２