



Title	情報爆発時代のための新しい超高速アルゴリズム : <特集>情報爆発時代におけるわくわくするITの創出を目指して : パートI: 情報爆発時代における新しい基盤技術
Author(s)	宇野, 毅明; 湊, 真一; 竹田, 正幸
Citation	情報処理, 49(8), 897-903
Issue Date	2008-08-15
Doc URL	https://hdl.handle.net/2115/47391
Rights	ここに掲載した著作物の利用に関する注意 本著作物の著作権は情報処理学会に帰属します。本著作物は著作権者である情報処理学会の許可のもとに掲載するものです。ご利用に当たっては「著作権法」ならびに「情報処理学会倫理綱領」に従うことをお願いいたします。
Type	journal article
File Information	19_johoshori_2008.pdf



2. 情報爆発時代のための新しい超高速アルゴリズム

宇野 毅明^{*1}- 湊 真一^{*2}竹田 正幸^{*3}

*1 国立情報学研究所

*2 北海道大学

*3 九州大学

事実上の不可能を可能に

近年の急激な情報技術の発達により、人類はかつてないほど巨大なデータを扱うようになった。複数の店舗から集まるお客の購買した品目のデータは年に100万件を超え、近年読み取りが完了したヒトゲノムも3億文字以上の長さを持つ。Webページは現在100億以上あり、Webネットワークをひとつのデータとしてみれば、これは超巨大なデータベースといえるだろう。

一昔前は、データは手の中にあり、人の目ですべてを見ることができた。全体としてどのような傾向があるのか、どの部分が典型的なのか容易に知ることができた。しかし巨大データでは、そのような人の手による解析はほぼ不可能である。また、平均や分散といった単純な指標では、データの表面を見ることはできるが、部分的な特徴を含めた深いところまで見通すことはできないだろう。近年、このような巨大データの解析には、統計、データマイニングや機械学習、最適化などに代表される、組合せ的な要因を含み局所的な特徴を捉える解析手法が用いられている。

これらの解析ではデータに対する組合せ的な処理が必要となる。しかしデータが巨大になると、組合せ的な処理は莫大な時間を要する。たとえば、レコードが1億あるデータのすべてのレコードの組について、単に似ているかどうかを調べるには、類似性の判定がおおよそ $1 \text{ 億} \times 1 \text{ 億} \div 2 = 5000 \text{ 兆回}$ 必要である。仮に1秒間に1千万回比較できるとしても、5億秒、およそ20年かかる。100台のコンピュータで並列化してもまだ2カ月以上かかる。すぐれた計算手法・アルゴリズムなしには、データの解析は事実上不可能であり、将来的なデータ収集機器の性能の向上に伴い、蓄積されるだけで解析が行えない巨大データがあふれることになるだろう。

計算の理論、つまりアルゴリズムの研究においては、日本はいくつかの分野で世界有数のレベルを誇っている。最適化の1つである線形計画問題に対する現在主

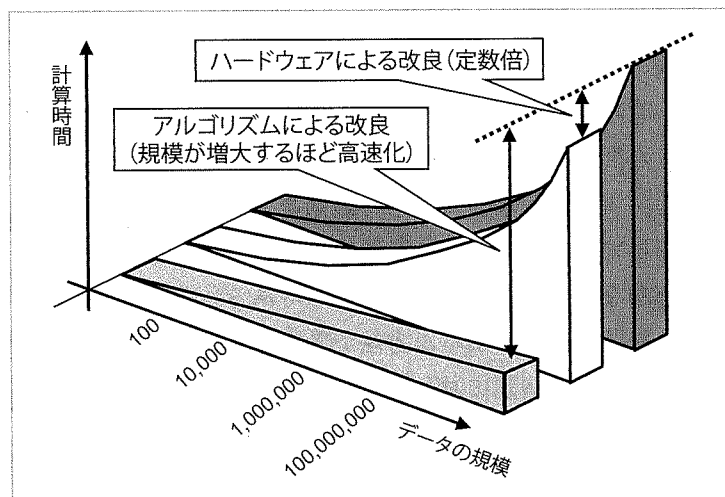


図-1 アルゴリズムの改良による高速化のイメージ

流となっている解法は日本人による設計であるし、幾何学的な構造を扱う計算幾何学の分野では、多くのアルゴリズムが日本人によって開発されている。簡潔な索引しか持たない高速圧縮検索アルゴリズムやデータマイニングの頻出パターン発見など、日本人が世界をリードしている分野が数多く存在している。解析できないとあきらめられている巨大データでも、洗練されたアルゴリズムを用いることで事実上の不可能が可能になったという事例は枚挙に遑がない。日本のソフトウェア産業への危機感の高まりが叫ばれているが、技術的には世界最高水準にある分野も多いのである。

計算時間の短縮にはハードウェアへの投資という方法もある。しかし、アルゴリズムの改良は、データサイズの増大に伴う計算時間の増加を小さくすることができるため、規模が大きくなればなるほど、高速化率が高くなる。たとえば、問題の大きさ n の2乗の時間がかかる上記の比較の問題に対して、クイックソートのような $n \log n$ 時間で終了するアルゴリズムを開発したとすると、1億レコードの比較の場合、多少のオーバーヘッドがあったとしても10万倍以上の高速化になるのである。

図-1で直観的な理解をしていただきたい。このような

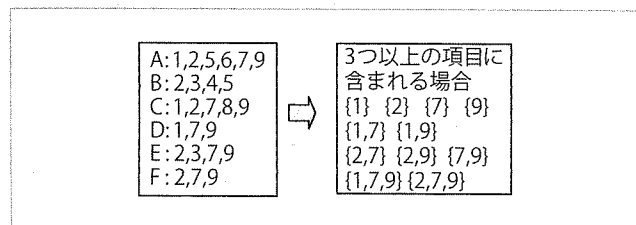


図-2 アイテム集合をレコードとするデータベースの例とその頻出集合

劇的な高速化を、大規模な設備投資なしに行えるところがアルゴリズム理論の強みである。また、アルゴリズムは計算方法の設計であるので、いったん新しい技術が開発されれば、以後同様の問題に対しては追加コストなしに技術の再利用ができる。

しかしながら、アルゴリズム理論は万能ではない。どのような問題にも効率の良い計算方法が見つかるわけではなく、実用で、場面に応じた最適なアルゴリズムの設計は簡単ではない。多種の問題に適用できる汎用性の高い効率的なアルゴリズムが重要なのである。研究者はどのような問題がどのような方法で効率良く解けるか、あるいは解けない問題に対してどのような代替手段が提供できるか、さまざまな方面から研究を行っている。近年の成果には、すぐに実社会に持ち込めるような技術も多い。以下では、情報爆発メンバの研究を中心に、日本での成果を紹介したい。

世界最速頻出パターンマイニング

データマイニングとは巨大データの特徴や性質を機械的に調べることであり、データ解析の手法として近年注目されている。中でもデータ中に多く現れるパターンを見つけ出す頻出パターンマイニングはその中心的な問題であり、実社会や学術分野での利用も盛んである。アルゴリズムの研究では日本が世界をリードしており、国際的なコンテストでも日本発のアルゴリズムが大差で優勝している。以下で簡単に紹介しよう。

図-2の左のような、各レコードがアイテム（ここでは1, ..., 9の数字）の集合であるデータを考える。このようなデータは、スーパー等の売上げデータ（各レコードが各客が買った商品の集合）や、アンケート結果のデータ（各レコードが各人が○をつけた欄の集合）など、実社会の多くの場面で見られる。分析のため、このデータの特徴的な構造を考えよう。今、アイテムは番号であり、それ自身に意味はないため、たとえば「差が2であるアイテムの組」といった構造は意味がない。よって、アイテムの組合せに注目するのが自然であるが、そうすると多くのレコードに含まれる（現れる）組合せを

特徴的と考えるのが適当であろう。このように、多くの、厳密にはある一定数（閾値）以上のレコードに含まれるアイテム集合を頻出パターン、あるいは頻出集合とよぶ。図-2の右側に示した集合はいずれも3つ以上のレコードに含まれる頻出集合である（閾値が3）。

頻出集合を用いると、局所的な部分に共通する構造が見られる。そのため、知識発見の分野で重宝されている。たとえば、頻出集合に含まれるアイテム同士は共起する確率が高いと考えられるため、「レコードが○と△を含むなら、□を含むことが多い」といったルール・法則を発見できる。正例と負例からなるデータで、正例に多く含まれ、負例にはあまり含まれないアイテム集合を見つけることで、正例と負例を分割する規則を機械的に発見し、正例や負例の要因の推測もできる。また、頻出集合は多くのレコードに含まれるため、エラーや異常値等に振り回されにくく、頻出集合をデータの統計量とみなしてデータ間の比較もできる。

これらの用途では、頻出集合を1つだけ見つけても意味がなく、多くの頻出集合を見つける必要がある。また、そのような集合はない、という保証を得るためには、すべての頻出集合を見つける必要がある。しかも、このような問題ではある程度以上の割合を見つけることは、本質的にすべてを見つけることと同程度の手間を必要とすることが分かっている。多量の解を短時間でを見つけるためには効率良いアルゴリズムの設計が必須であるが、学問の成り立ちの歴史からか、比較的データベース系の技術に注目が集まっている。最初に提案されたアルゴリズムはaprioriとよばれるもので、ディスク上のデータへのアクセス回数を最小化するようにデザインされている。その後、データがメモリに収まるようになると、メモリにデータの必要部分を格納し、深さ優先的に山登り探索を行う第2世代のアルゴリズムが提案された。現在の主流は、探索中に必要ないレコードやアイテムを再帰的にデータから消去して圧縮を行う第3世代のアルゴリズムである。

我々は、さらにアルゴリズム理論から効率的な計算手法を研究し、以下の4つの新しい技術を元に第4世代アルゴリズム LCM を開発した¹⁾。(1) 解空間上に数理的な探索路を定義し、メモリの爆発と計算コストの増大を起さずに重複の回避を行う方法を開発した。これは飽和集合というパターンを列挙する手法であり、従来は、完全性の保証のないヒューリスティックな枝刈りと既発見解のメモリへの保存が不可避であったところを、アイテムの追加によってパターンを大きくした際に prefix が変化することが完全な枝刈りを得る条件であることを証明し、同時にメモリを必要としない深さ優先型の探索を実現した。(2) 内部データベースの構築において、同

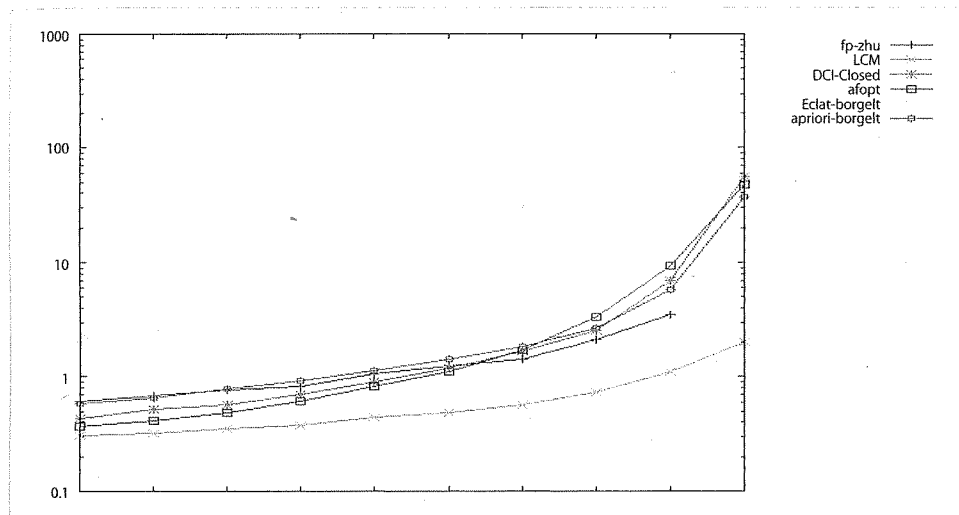


図-3 コンテストの結果の一部：横軸が閾値（解の数）、縦軸がlog スケールで表された計算時間

一のトランザクションを併合する操作がボトルネックとなっていたが、これを基数ソートをベースにした方法で行うことで、従来の検索をベースにした方法に比べ、計算量を $O(n \log n)$ から $O(n)$ に減少し、メモリなどのオーバーヘッドも減少させた。(3) 山登り探索では、各反復で前回追加したアイテムよりも大きな各アイテムを追加してデータの更新を行い、再帰呼び出しを行うが、このデータの更新をすべてのアイテムについて一度に行うことで計算の無駄を省いて高速化する方法を開発した。(4) さらに、再帰呼び出しを大きなアイテムの追加によるものから順に行うことで、データ更新に用いるメモリを再利用できるようにして計算効率を高めた。

これらの技術は他の研究には見られず、高速計算で優位である。現に、このアルゴリズムの実装は、2004年に行われた国際的なデータマイニングプログラミングコンテストである FIMI04 で優勝しており²⁾、その後も根本的にこれを上回るアルゴリズムは提案されていない。

図-3 に結果の一部を示す。一番下の線が LCM の計算時間を示すが、解の数が増えて計算の規模が増すほどに他のアルゴリズムとの計算時間の差が大きくなっていくことが分かるだろう。現実の大規模データにおける、アルゴリズム理論による高速化が効果を示したと言ってもよいだろう。

頻出集合マイニングアルゴリズムは、少し工夫をすると他の面白い用途にも使える。アイテム A と B の組が頻出集合であれば、それは A と B が共通して多くのレコードに含まれることを意味する。よって、大きさ 2 の頻出集合を見つけることで、共起性が高いアイテムの組をすべて発見できる。アイテムとレコードの関係を反転させれば、共通部分が大きなレコードの組や、似ているレコードの組をすべて発見できる。筆者の実装では、500 万の Web サイトから 10 分ほどで、リンク先が似ているサイトの組を 1 億ほど見つけた。総当たりの比

較なら 1 秒間に 1000 万回比較しても 20 日以上、それなりに工夫してもまる 1 日はかかる計算である。

近年では、ノイズや曖昧性に対応するため、曖昧性を考慮した包含関係を用いて頻出集合をモデル化する研究も進んでいる。これにより、今まで発見できなかったパターンの発見が可能となるが、曖昧性の導入により解の単調性がなくなるため、通常の上昇法による探索が不可能となる。そこで、逆探索という単調性に頼らない探索手法を用い、曖昧性を許容し、かつ探索が効率良くできるような頻出集合のモデルを構築し、効率的に解を列挙するアルゴリズムを構築した。これは、この種の問題に対する初の出力多項式時間アルゴリズムとなっている⁸⁾。

頻出パターンは XML データ、時系列データ、文字列データ等、データベースが構造を持つ場合にも拡張できる。構造を付加したパターンの発見にも、頻出集合に対する技術が拡張できることが多く、構造を持ったデータでの類似構造検出や特徴抽出が可能となるだろう。大規模データでは、計算コストの増大からついつい解析を断念してしまいがちだが、その前に効率良いアルゴリズムに目を向けることが重要である。

二分決定グラフ (BDD) を用いた パターンマイニングのさらなる高速化

以上で述べたように、頻出パターンマイニングのアルゴリズムは、近年急速に発展を遂げているが、最近になって、VLSI 設計自動化という異分野で発達した「二分決定グラフ」(BDD) と呼ぶ技法を組み合わせることにより、パターンマイニングをさらに高速化・効率化できることが分かってきた。以下にその概要を紹介する。

二分決定グラフ (BDD : Binary Decision Diagram)⁴⁾ は、1980 年代後半に VLSI 設計の分野で考案された論

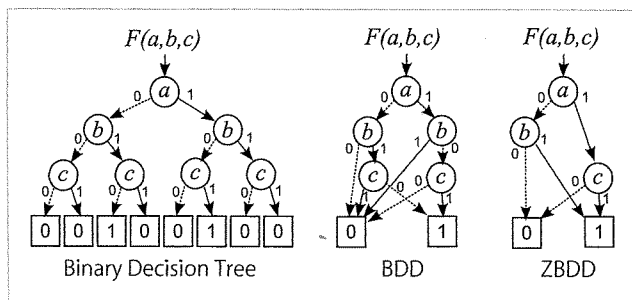


図-4 二分決定木の例、および対応する BDD, ZBDD

a	b	c	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

論理関数表現：
 $F: \bar{a}bc \vee \bar{a}\bar{b}c$

パターン集合表現：
 $F: \{ac, bc\}$

図-5 論理関数と組合せ集合の対応

理関数データのグラフによる表現である。これは、論理関数の値をすべての変数について場合分けした結果を図-4のように二分木グラフで表し、これを簡約化することにより得られる。このとき、場合分けする変数の順序を固定し、(1)冗長な節点を削除する、(2)等価な節点を共有する、という処理を可能な限り行うことにより「既約」な形が得られ、論理関数をコンパクトかつ一意に表せることが知られている。BDD によるデータ圧縮率は、論理関数の性質にも依存するが、例題によっては数十倍～数百倍以上もの圧縮率が得られる場合がある。また、2つの BDD を入力とし、それらの二項論理演算 (AND, OR 等) の結果を表す BDD を直接生成するアルゴリズムが知られており、ハッシュテーブルを巧みに用いることで、データが計算機の主記憶に収まる限りは、その記憶量にほぼ比例する時間内で論理演算を効率良く実行できるという特長を持っている (詳細は本学会誌 1993 年 5 月の小特集記事³⁾を参照)。

BDD は、元々は論理関数の表現法であるが、 n 種類のアイテムの組合せからなるパターン集合を、 n 入力の論理関数に対応付けることにより、パターン集合を表現することもできる。図-5に例を示す。BDD 表現は、類似する部分パターンが多く出現する場合にデータ圧縮効果が得られる。さらに、このようなパターン集合データの処理に特化した BDD として ZBDD (ゼロサプレス型 BDD) が知られている。ZBDD は、通常の BDD と異なる簡約化規則を持つ。すなわち、図-6に示すように、1- 枝が 0- 終端節点を直接指している場合に、この節点を取り除く。その代わりに、通常の BDD で削除されるような節点は削除しない。この簡約化規則は、特に疎な組合せの集合に対して顕著な効果がある。たとえば、アイテムの平均出現頻度が 1% であれば、ZBDD は BDD よりもさらに 100 倍コンパクトになる可能性がある。データマイニングで扱われる現実のデータベースでは、アイテムの平均出現頻度が非常に小さい場合が多い (たとえば、店舗での陳列商品の総数に比べて、顧客が 1 度に購入するアイテム数はきわめて少ない) ため、ZBDD による圧縮効果はきわめて大きい。ZBDD のア

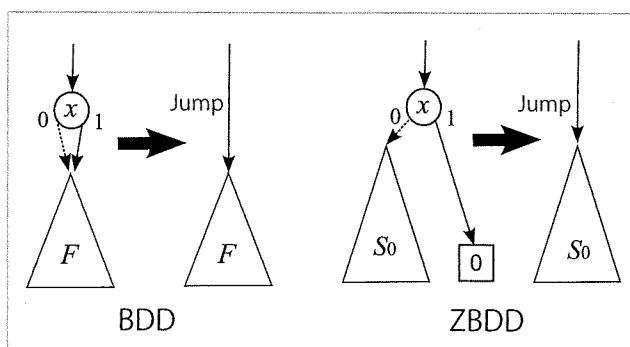


図-6 BDD と ZBDD の簡約化規則

イディアは、筆者の 1 人 (湊) により最初に提案されたものであり、現在も国内外でさまざまな用途に用いられている。

筆者らは、このような ZBDD の利点を活かして、世界最高速の LCM アルゴリズムをさらに改善した「LCM over ZBDDs」を開発した⁵⁾。一般に、頻出パターンマイニングでは、頻度の閾値を高く設定しすぎると自明なパターンしか見つからず、逆に閾値が低すぎると抽出パターン数が膨大になりすぎて、興味あるパターンの発見が困難になる。したがって、ちょうど良い適切な頻度を設定する必要があるが、例題に応じて実験的に決めるしかなく、それが本当に適切な値だったのかも確かめようがないという問題があった。

これに対して ZBDD を使えば、膨大な個数のパターンをひとまとめにしてコンパクトに圧縮して、計算機の主記憶上に表現することができる。さらに、抽出したパターン全体に対して ZBDD の集合演算を使って一定の制約条件を加えて、効率良く絞り込みを行うことができる。表-1 は、数百アイテム、数千～数万レコードを含む現実的なベンチマーク例題に対して、我々の手法を適用した実験結果を示している。この表に示す通り、比較的低い頻度の閾値を与えると膨大な数のパターンが抽出されるが、多くの実用的な例題では、これらのパターンは類似する部分パターンを多数含んでおり、これを ZBDD 処理系で自動的に検出して圧縮することにより、たとえば 10 億を超えるパターンを、わずか数万ノードの ZBDD で表現できるという事例が見られる。従来の

最小頻度 (閾値)	頻出	LCM over ZBDDs		従来 LCM
	パターン数	ZBDD	Time(s)	Time(s)
mushroom				
1,000	123,287	760	0.50	0.64
500	1,442,504	2,254	1.32	3.2
200	18,094,822	6,383	3.21	31.63
100	66,076,586	11,584	5.06	114.21
50	198,169,866	17,830	8.17	357.27
BMS-WebView1				
50	8,192	3,415	0.11	0.12
40	48,544	10,755	0.18	0.22
36	461,522	28,964	0.49	0.98
34	4,849,466	49,377	1.30	8.58
32	1,531,980,298	71,574	31.90	3,843.06

(2.4GHz Core2Duo E6600, 2 GB Memory)

表-1 従来の LCM と「LCM over ZBDDs」の比較

明示的な表現方法では、結果を出力するのにも膨大な時間がかかり、たとえ出力できたとしても、その後の解析処理（たとえば特定のアイテムによる絞込みや、因果関係の解析等）を現実的な時間で行うことは非常に難しい。それに対して、ZBDD で抽出結果をコンパクトに表現すれば、ZBDD 同士の集合演算により、さまざまな解析処理が ZBDD のサイズに依存する計算時間で効率良く実行可能となる。つまり、ZBDD を使うことにより、これまで正面から計算することを諦めていた規模の限界（現実的に計算可能な範囲）を、大きく広げることが可能になったと言える。

BDD（および ZBDD）は、類似する部分を持つきわめて多数の組合せの集合を、アルゴリズムの技術により自動的に圧縮して主記憶上にコンパクトに表現し、さらに、圧縮されたままの状態ですさまざまな集合演算を適用し、データの解析・加工を可能にする技法であると言える。

余談であるが、今年のチューリング賞（計算機科学のノーベル賞に相当する）は、米国 CMU の E. M. Clarke 氏らに与えられることとなった。彼らの業績は、ハードウェアやソフトウェアの設計の正しさを数学的に証明する「シンボリックモデルチェッキング」という技術を開発したことであるが、この中でも、膨大な場合分け処理を現実的な時間で実行するために不可欠な技術として BDD が利用されている。

常識破りの文書検索：オートマトン理論の実応用

ここまでは、組合せ的に膨大な計算量を要する問題に対していかに効率的に解を求めるか、という話をしてきたが、この章では、文書検索（document retrieval）という古典的な問題を扱う。この問題に対し、オートマ

トンで文書をべた読みするという、常識破りな手法を取りあげ、その実用システムへの応用事例と今後の展開について述べる。

● 文書検索と部分文字列サーチ

文書検索問題とは、最も単純には、文書（テキスト）の集合とキーワードが与えられたときに、そのキーワードを含む文書の集合を求める問題である。この問題に対する通常の解は、転置索引（inverted index）に基づくものである。転置索引とは、文書中の単語からその出現位置（もしくは文書の ID）のリストを効率的に得るためのデータ構造のことで、英文テキストのように単語の区切りが明らかなデータではこの方法が広く用いられている。しかし、日本語テキストのように明示的な単語の区切りを持たないデータや、DNA 配列やアミノ酸配列など、単語という概念自体が成立しないデータでは、この手法は必ずしも有効ではない。

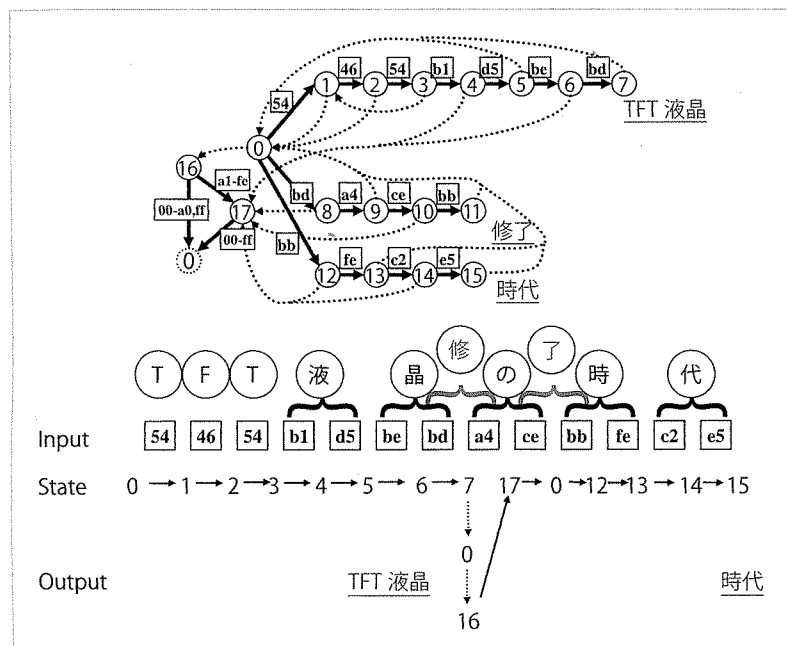
そこで、部分文字列サーチを考える。すなわち、キーワードが文書データの部分文字列として現れていればそれをキーワードの出現とみなす。もちろん、こうすれば、キーワード「書記」に対して「公文書記録」に含まれる「書記」も検出してしまう。しかし、日本語の形態素解析を 100% の精度で行えない以上、科学技術文献の検索や特許データの検索など、検索漏れが許されない場面、すなわち、精度よりも再現率が重視される応用においては、部分文字列としての出現をすべて求めることはきわめて重要である。

● オートマトンでテキスト全体を高速サーチ

部分文字列サーチのための索引構造の 1 つとして、接尾辞配列などが知られている。しかし、この構造は、テキストデータサイズに比例した主記憶領域を必要とする。この比例定数を小さくするための研究として、圧縮接尾辞配列やその変種が知られているが、テキストサイズとほぼ同等の主記憶領域が必要である。

ここで紹介する解は、キーワードから作成したパターン照合機械（PMM）を用いるものである。PMM は、一種の有限オートマトンで、テキストデータ全体をただ 1 度だけ走査する間に、すべてのキーワードのすべての生起位置を求めることができる。図-7 に日本語のテキストを扱う PMM とその動作を示した。

九州大学の有川節夫教授らのグループは、PMM の効率的な実装法に関する理論的・実証的な研究を徹底して行い、その成果に基づいて、SIGMA と名付けた汎用テキストデータベース管理システムを開発した。SIGMA は、同大学大型計算機センター（現情報基盤研究開発センター）において 1981 年より公開され、約 20 年の長



実線は通常の状態遷移を表し、破線は通常の状態遷移が行えないときのみ実行される ϵ -動作（ヘッドを固定したまま状態のみを変える）を表す。1バイト文字と2バイト文字が混在する日本語テキスト（EUC）の上を1バイト単位で動作する。1バイトのずれ読みによるキーワードの誤検出を避けるための機構がPMM内に組み込まれている。

図-7 日本語テキストを扱うPMM（上）とその動作（下）

きにわたって、文科系の原典研究者や昆虫学の研究者などから利用された。

テキスト長を n 、キーワード長を m とすると、接尾辞配列に基づく方法では $O(m \log n)$ 時間（または $O(m + \log n)$ 時間）でサーチを行える代わりに、 $O(n)$ の主記憶領域を要する。一方、PMMを用いた方式では、扱うキーワードの長さの合計に比例した領域量で済むが、処理時間が $O(n)$ かかってしまう。そこで、SIGMAでは、PMMの実装方法に関する研究成果に基づき、その比例定数を小さく抑えている。また、 k 個のキーワードをサーチする場合、索引を用いた方法では、基本的に k 回探索を行うことになるが、これに対して、SIGMAでは、 k 個のキーワードを同時に探索することができる。したがって、キーワードあたり $O(n/k)$ 時間ということになる。さらに、複数キーワードをブール演算子で結合した論理式を扱う場合、転置索引や接尾辞配列などの索引構造を用いて各キーワードごとに文書集合を求めたのちに集合演算を行うことが通常であり、そのための時間と領域が別途必要である。これに対して、SIGMAの方式では、文書を1つ1つ処理してゆくため、文書ごとにブール演算を行えばよく、集合演算が不要となる。このこともこの手法の大きな利点といえる。

● Shunsaku としての展開

SIGMAは、富士通（株）のテキスト型検索エンジ

ン Interstage Shunsaku Data Manager（以下 Shunsaku）においてメインエンジンとして採用されている。Shunsakuでは、データを数百MB程度に分割し、それぞれにCPUを割り当てて並列にサーチを行い、結果をマージする手法をとることで、大規模なデータに対応している。また、Shunsakuでは、複数の検索要求中に含まれるキーワードをまとめて同時に検索を行う。一度に数千個のキーワードを扱うことが可能である。もちろん、検索要求が一定量たまるまで待ち続けるのではなく、一定の時間で見切り発車し、それ以後に届いた検索要求は次の走査時に処理する。これが「山手線方式」とよばれるゆえんである。乗り損ねても次の電車はすぐに来る。

Shunsakuの導入事例として最も際立っているものが、国立遺伝学研究所と富士通との共同開発によるARSAである。ARSAは、3大国際DNAデータベースによって共同で構築された「DDBJ/EMBL/GenBank 国際塩基配列データベース」の検索サービスを行うシステムとして2004年に運用が開始されたが、その際には、COMPUTER WORLD誌に大きく取り上げられ「当該週世界で最も多く参照されたIT記事（Google調査）」と認定されるなど、世界中の注目を集めた^{☆1}。さらに、Shunsakuの専用ハードウェア機「Shunsaku Engine」の開発・販売開始を受けて、2007年には、Shunsaku Engine15台（2520CPUコア、630GBメモリ）を用いたシステムに刷新され、検索対象も、国際塩基配列データベースを含む24のデータベースに拡大している。ARSAは、合計約3300万件約100GB強（2008年5月現在）のデータに対して平均5秒、複雑なクエリでも10秒以内で応答が可能である。2520個のオートマトンがそれぞれの担当データ上を高速に走査している様子をイメージしていただきたい。

SIGMAおよびShunsakuは、オートマトン理論・形式言語理論の成果が現実社会において実用に使われている、最も素直な事例であろう。

● 今後の展開

上述のべた読み技術の今後の展開として、2つの方向性がある。1つは「圧縮による高速化」技術の組み込みである。これは、圧縮されたテキストを伸長せずにそのま

☆1 この年 Shunsaku は、(独) 情報処理推進機構および (財) ソフトウェア情報センターによる「ソフトウェア・プロダクト・オブ・ザ・イヤー 2004」を受賞している。

まパターン照合を行う「圧縮パターン照合」の研究をさらに一歩進めたものであり, Byte-Pair-Encoding (BPE) 法という圧縮法を用いた場合, 非圧縮テキスト上での通常のパターン照合と比較して, 照合時間を圧縮率と同程度の割合で短縮できる, という画期的なものであった(文献6)等を参照されたい). たとえていえば, クルマ(オートマトン)の速度を向上させるのではなく, 道路(テキスト)の長さを縮めようとするものである. しかし, BPE 法は gzip や bzip2 などと比べて圧縮率が低いという問題があり, このことが実用化に向けての大きな障壁となっていた.

この問題に対して, ごく最近の筆者らの研究⁷⁾により, 高速な圧縮パターン照合が可能で, かつ, 圧縮率の高い圧縮法を開発できた. この手法は, 従来の多くの圧縮法と同様, 文法変換(テキストをそれを生成する形式文法に変換)に基づく圧縮法であるが, 従来手法が文脈自由文法を扱うのに対し, 文脈依存文法を扱う点異なる. 詳細は省くが, たとえば, 典型的な英文テキストの場合, BPE 法の圧縮率が56%程度であったのに対し, これを33%程度にまで改善している. 照合時間についても同程度に改善している.

圧縮と高速化を同時に実現するこの技術は, Shunsaku で並列にサーチを行う際の CPU 数をも「圧縮」するであろう. また, 文脈依存文法変換は, 生まれたばかりの新しい概念であり, 今後この方向に沿った理論的・実用的な進展が期待される.

もう1つの方向は, XML などの半構造データへの拡張である. XML 文書の質問処理においては, タグによって表現された構造に関する構造パターンの照合をテキスト部分に対するキーワード照合と同時に行うことが要求されるが, この構造照合をいかに高速かつ軽量に行うかが開発のカギであった. 筆者らは, 2006~07年に, 実際の XML データの多くは, 構造がある程度安定していることに着目し, XML 木の根から葉へ向かうパス上のラベル列の全体を表す木構造であるパストライを事前に構築してから処理を行う XAXEN システム, さらには, 動的にパストライを構築する DXAXEN システムを開発した. 最も高速な XML ストリーム処理器として知られる XMLTK と比較して2~5倍高速であり使用メモリ量も5~20%程度である. この高速化・軽量化技術は, 現在開発中の Shunsaku の新しい版に組み込まれており, Shunsaku のさらなる高機能化が期待できる.

超高速アルゴリズムと情報爆発

本稿で紹介した3つの研究はそれ自体, 世界の研究と肩を並べられると考えているが, 実社会での利用や,

日本発のソフトウェアの育成に結びつけるためには, まだ多くの課題が残っている. 依然として, 実社会と理論, あるいは工学・自然科学と計算機理論の間には学術的にも産業的にも興味深い問題が多く転がっている. これら問題はこれからの学術研究の中心となってゆくであろうし, また産業面での成果も期待できるだろう. 本稿をご覧になった読者が, 学術産業の分野を問わず, アルゴリズム理論とその融合領域に興味を抱いていただければ幸いである.

参考文献

- 1) Uno, T., Asai, Y., Uchida, Y. and Arimura, H.: An Efficient Algorithm for Enumerating Closed Patterns in Transaction Databases, *LNAI* 3245, pp.16-31 (2004).
- 2) Goethals, B.: The FIMI Repository (2003), <http://fimi.cs.helsinki.fi/>
- 3) 湊 真一: 計算機上での BDD の処理技法, 情報処理, Vol.34, No.5, pp.593-599 (May 1993).
- 4) Bryant, R. E.: Graph-based Algorithms for Boolean Function Manipulation, *IEEE Trans. Comput.*, Vol.C-35, No.8 pp.677-691 (1986).
- 5) Minato, S., Uno, T. and Arimura, H.: LCM over ZBDDs: Fast Generation of Very Large-Scale Frequent Itemsets Using a Compact Graph-Based Representation, In Proc. of 12-th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2008), (May 2008). (to appear)
- 6) Takeda, M. et al.: Speeding Up String Pattern Matching by Text Compression, 情報処理学会論文誌, Vol.42, No.3 (40 周年記念論文特集号), pp.370-384 (2001).
- 7) Maruyama, Y., Tanaka, H., Sakamoto, H. and Takeda, M.: Context-Sensitive Grammar Transform: Compression and Pattern Matching, In Proc. of 15th International Symposium on String Processing and Information Retrieval (SPIRE 2008), (Nov. 2008) (to appear).
- 8) Uno, T.: Ambiguous Frequent Itemset Mining and Polynomial Delay Enumeration, Lecture Notes in Computer Science **5012**, Springer, pp.357-368 (2008).

(平成 20 年 5 月 23 日受付)

宇野 毅明 (正会員): uno@nii.jp

昭和 45 年生. 平成 10 年東京工業大学大学院総合理工学研究科システム科学専攻博士課程修了. 博士(理学)を取得. 同年同大経営工学専攻助手着任. 平成 13 年国立情報学研究所助教授に就任. 現在情報・システム研究機構国立情報学研究所准教授. アルゴリズム理論とその応用研究に従事. 特に, 離散アルゴリズム, 列挙アルゴリズム, データマイニング, 組合せ最適化など. 日本オペレーションズリサーチ学会会員.

湊 真一 (正会員): minato@ist.hokudai.ac.jp

昭和 40 年生. 平成 2 年京都大学大学院工学研究科情報工学専攻修士課程修了. 同年 NTT 入社. 以来, 大規模論理データの表現と処理アルゴリズムの研究開発に従事. 平成 7 年京大大学院博士課程(社会人)修了. 博士(工学). 平成 11 年 NTT 未来なつと研究所主任研究員. 平成 16 年より北海道大学大学院情報科学研究科 助教授(現在, 准教授). 著書『Binary Decision Diagrams and Applications for VLSI CAD』(Kluwer, 1995). IEEE, 電子情報通信学会, 人工知能学会各会員.

竹田 正幸 (正会員): takeda@i.kyushu-u.ac.jp

昭和 39 年生. 平成元年九州大学大学院総合理工学研究科情報システム学専攻修士課程修了. 同大助手, 助教授を経て, 現在同大学院システム情報科学研究科教授. 文字列アルゴリズム, パターン発見アルゴリズム, 発見科学とデータマイニングなどの研究に従事. 博士(工学). 人工知能学会会員.