自然言語処理 75－9
（1990．1．19）

文章解析アクセラレータ(1)
－形態素抽出マシンの試作－

福島 俊一　大山 裕　宮井 均

日本電気株式会社　C＆Cシステム研究所

本稿では、文章解析処理の高速化への新しいアプローチとして、文章解析アクセラレータハードウェアを提案する。そして、その第1ステップとして試作した形態素抽出マシンについて報告する。形態素抽出処理は、単語辞書を検索することによって、入力されたテキストに出現した可能性のある全単語（全形態素）を抽出する処理である。文章解析の最初に不可欠な処理である上に、処理時間に占める比率が大きいことから、高速化が強く望まれていた。試作した形態素抽出マシンでは、8万見出しの単語辞書を用いた、1万文字のテキストに対する形態素抽出処理が、1秒程度で実行できる。この処理速度は、従来のPC9800上のソフトウェアと比べて、100～1000倍という高速なものである。本マシンは、①単語を構成する全文字の同時照合、②先頭文字による辞書範囲の絞り込み、③シフトレジスタによるテキストの順送り、などの手法を組み合わせた形態素抽出の新しいハードウェアアルゴリズムを実現している。各文字が複数の候補をもったテキストについても、候補数に対して線形オーダーの時間で、形態素抽出処理が行なえる。

Natural Language Parsing Accelerators (1)
- An Experimental Machine for Morpheme Extraction -

Toshikazu FUKUSHIMA, Yutaka OHYAMA and Hitoshi MIYAI
C&C Systems Research Laboratories, NEC Corporation
1-1, Miyazaki 4-chome, Miyamae-ku, Kawasaki City, Kanagawa 213, Japan
(fuku@tsl.cl.nec.co.jp, ohyama@tsl.cl.nec.co.jp, miya@tsl.cl.nec.co.jp)

This paper proposes natural language parsing accelerators, as a new approach to parsing speed-up. It also describes an experimental machine for morpheme extraction, which is designed as the first step toward achieving the natural language parsing accelerators. In the morpheme extraction process, all morphemes, which are considered probably used to construct input text, are extracted by searching a morpheme dictionary. Speed-up requirement for this process is very strong, because this is indispensable process for the parsing, and spends a large percentage of parsing time. The experimental machine can extract morphemes from 10000 character Japanese text by searching an 80000 morpheme dictionary in one second. It runs 100-1000 times as fast as morpheme extraction programs on personal computers. It realizes a new hardware algorithm, which uses parallel comparators, an index and shift registers. It can treat multiple text streams, which are composed of character candidates, as well as one stream of text. The algorithm is implemented on it in linear time for the number of candidates.

1　はじめに

1－1　文章解析処理の高速化の効果

近年、計算機による文章解析技術は、ワードプロセッサや機械翻訳システムなどのかたちで実用化され始めている。この実用化をさらに進めてゆくためには、より深い解析の追究による高精度化と並行して、処理の高速化を行なってゆく必要がある。文章解析処理を高速化することは、次の(1)～(3)のようなかたちで文章解析応用の拡大・実用化につながる。

(1)レスポンスの高速化
(2)処理対象ドキュメントの多量化
(3)処理の高精度化

(1)は最も直接的な効果である。自動通訳・音声認識における言語処理、ワードプロセッサのかな漢字変換、データベース検索の自然言語インタフェースなどのように、ユーザの入力に対する実時間の応答が要求される応用で望まれている。

一方、機械翻訳、ドキュメント検査（自動校正）、文字認識における言語処理（大量ドキュメント入力）、データベースへ登録するための自動キーワード付けなどの応用では、非常に多量のドキュメントの一括処理を行なう。(2)は、このような応用のメリットを高め（人手と比較した場合の所要時間を短縮し）、その実用化を促進する。

(3)は、間接的な効果である。処理が高速化できると、同一の時間内に、それまでより豊富な処理が行なえる。そこで、より大規模の辞書・知識ベースを参照したり、意味・文脈処理により多くの時間を割り当てたりすれば、一定の処理時間・レスポンス時間を保ったままで、解析精度を上げることが可能となる。

1－2　文章解析アクセラレータの提案

文章解析処理の高速化は、これまで、おもに、逐次型の汎用計算機の発展と、その上のソフトウェア／アルゴリズムの改良に支えられてきた。しかし、上記のような高速化の要求は、それ以上に強い。

このさらなる高速化の要求に対して、近年、汎用並列計算機が期待されている。並列論理型プログラミング言語で記述された構文解析プログラム[5]や、Ｐ－ＲＡＭ（Parallel Random Access Machine）を想定した並列構文解析アルゴリズム[6][7]などが考案されている。しかし、汎用並列計算機の実用化にはもう少し時間を要すると思われ、また、実際に実行した場合には、共有メモリに対するアクセス競合や、プロセス間通信のボトルネック問題などが発生して、十分な性能が得られない可能性がある。

一方、パターン認識の分野では、様々な専用ハードウェアが開発され、高速化が実現されてきた。そのなかには、文章解析処理へ適用できる可能性をもつものもある。例えば、音節の種類だけのアドレス空間をもつＲＡＭを用いた音声認識用の辞書照合方式[3]、マルチプロセッサにより複数テキスト位置で並列に辞書検索を行なう形態素抽出方式[4]、シストリックアレイを用いてＣＹＫ法やEarley法を並列化した文脈自由言語認識方式[8][9]などである。

筆者らは、今後、文章解析処理についても、ハードウェアアプローチが必要と考える。そこで、文章解析処理を高速化する専用ハードウェア（文章解析アクセラレータ：Natural Language Parsing Accelerators）を提案する。

2　形態素抽出処理のハードウェア化

2－1　形態素抽出処理の選択

形態素抽出処理は、単語辞書を検索することによって、入力されたテキストに出現した可能性のある全単語（全形態素）を抽出する処理である（図1参照）。通常、形態素抽出処理に引き続いて、文法的な情報をもとに単語間の接続可否を調べて文節を形成する接続検定処理が行なわれる。入力された文章（日本語ではべた書きされた文章）を単語列（形態素列）に分割し、

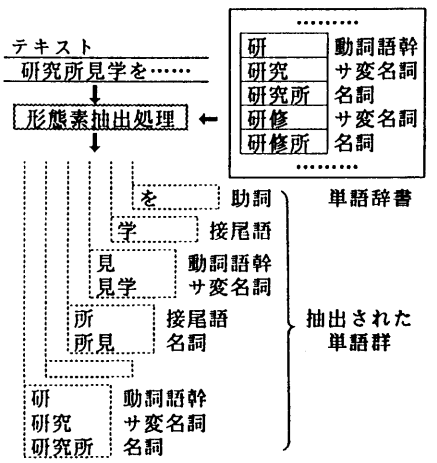


図1　形態素抽出処理

各単語の品詞を認定する形態素解析処理は、それら形態素抽出処理と接続検定処理をあわせたものを指す。文章解析処理は、さらに、構文解析処理・意味処理・文脈処理などを含んでいる。

形態素抽出処理は、文章解析の最初に不可欠な処理である上に、処理時間に占める比率が大きい（形態素解析では処理時間の７割を越える）ことから、高速化が強く望まれる。一方、ソフトウェア的には手法が確立されており、処理の大半が単純な照合処理の繰り返しであることから、ハードウェア化に適している。

また、近年、並列構文解析方式が研究され始めているが[5][6][7]、実際にインプリメントした場合には、共有メモリに置かれる辞書のアクセス時間が性能のボトルネックになることが予想される。この問題を軽減するためにも、形態素抽出処理のような辞書周辺処理の高速化は重要である。

以上の点から、高速化が望まれ、かつ、ハードウェア化に適した処理単位として、まず形態素抽出処理を取り上げ、そのハードウェア化を行なった[1]。

２－２　ハードウェア化の方針

ハードウェアを想定して形態素抽出処理を並列化する１つの方法として、テキストと単語辞書とを照合する際に、次のような多重化を行なうことが考えられる。

Ⓐ照合対象単語の多重化（図２参照）
Ⓑ単語抽出位置の多重化（図３参照）

従来のソフトウェアによる形態素抽出処理では、テキストのある位置から部分文字列を切り出して、その部分文字列に一致する単語を、単語辞書から検索する処理が繰り返される。その単語辞書検索の際には、部分文字列と単語との照合が、辞書内の単語ごとに逐次行なわれる（一対一の照合）。これに対して、Ⓐでは、テキストのある位置から切り出した部分文字列と、単語辞書内の複数の単語とを同時に照合する（一対多の照合）。一方、Ⓑは、テキストにおける形態素抽出を行なう位置を多重化するものである（多対一の照合）。すなわち、テキストの１文字目・２文字目・３文字目・……から同時に抽出処理を行なう。

Ⓐの方式は、従来、連想メモリ[10]を用いて実現されている。ただし、現状の連想メモリＬＳＩは１個の容量が数Ｋ～数十Ｋビットであるから、数万見出しの単語辞書を一度に登録するには、数千個を越える多数の連想メモリＬＳＩが必要になる。数個～十数個という現実的な数の連想メモリＬＳＩを想定し、単語辞書を１回で登録できるサイズの複数個に分割して、その登録・検索を繰り返すようにすると、今度は、繰り返し登録に要する時間が障壁となり、極めて多量のテキストを一括処理するときでないとメリットが出なくなる[2]。

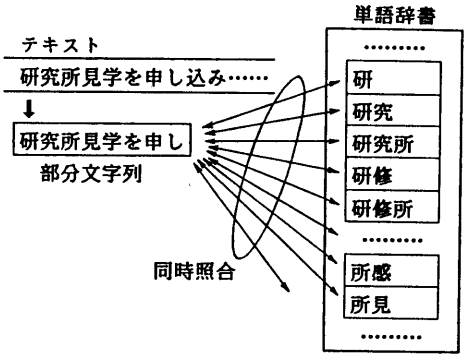


図２　照合対象単語の多重化

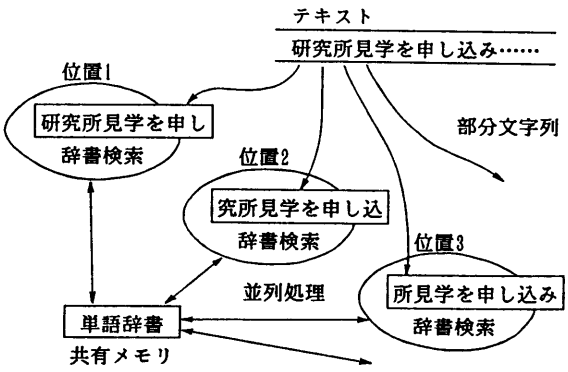


図３　単語抽出位置の多重化

Ⓑの方式は、従来、テキストの各文字位置に処理ユニットを対応付けたマルチプロセッサ形式で実現されている[4]。この場合、単語辞書は共有メモリに置かれ、複数処理ユニットからのアクセス競合の調整機構が設けられる。処理ユニット数を増加すると、処理速度は向上するが、それにつれてアクセス競合の頻度が増すので、処理速度の向上は頭打ちになる。文字列検索ＬＳＩ[11]に１文字ずつ位置をずらしてテキストを登録し、検索時に単語辞書を入力する方法で、Ⓑを実現することもできるが、その場合は、単語の抽出される順が位置的にランダムとなり、以降の処理（接続検定処理など）との整合が悪いという欠点がある[2]。いずれの実現方法とも、ハードウェアの構造が複雑になり、価格性能比の面では現実的でない。

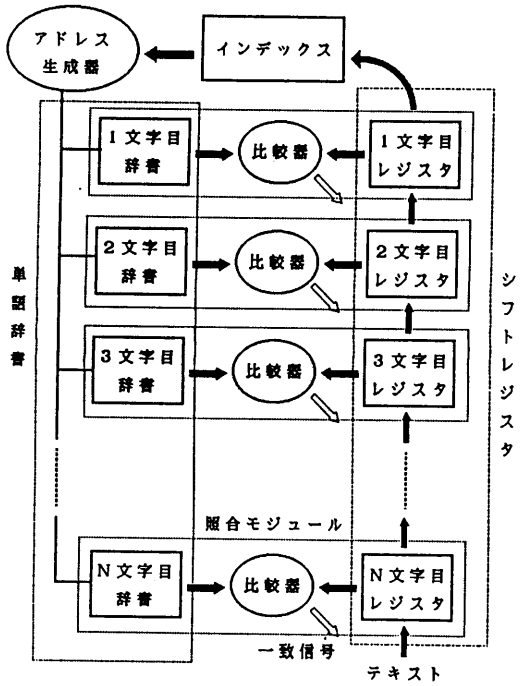


図４　基本的な構成

以上のような検討にもとづき、筆者らは、ⒶⒷのような多重化の方針はとらず、基本的な１回の照合処理のサイクルを可能な限り短縮するアプローチをとった。

３　形態素抽出ハードウェアアルゴリズム

３－１　基本的な構成・動作

考案した形態素抽出ハードウェアアルゴリズムを実行するための基本的な構成は、図４の通りである。おもな構成要素は、単語辞書、シフトレジスタ、インデックス、アドレス生成器、比較器である。

単語辞書は、図５に示すような形式で、１文字目辞書～Ｎ文字目辞書のＮ個に分割されている。長さがＮより短い単語については、Ｎに満たない部分に、あらかじめ定められた残余記号が格納されている（図５では「＊」が残余記号を表わしている）。

シフトレジスタは、１文字目レジスタ～Ｎ文字目レジスタで構成され、そのなかにはテキストのＮ文字分が格納される。シフトレジスタでは、図６に示すような文字列の順送りが行なえる。

インデックス（先頭文字表）からは、ある文字で始まる単語群の、単語辞書における存在範囲が得られる（図５参照）。

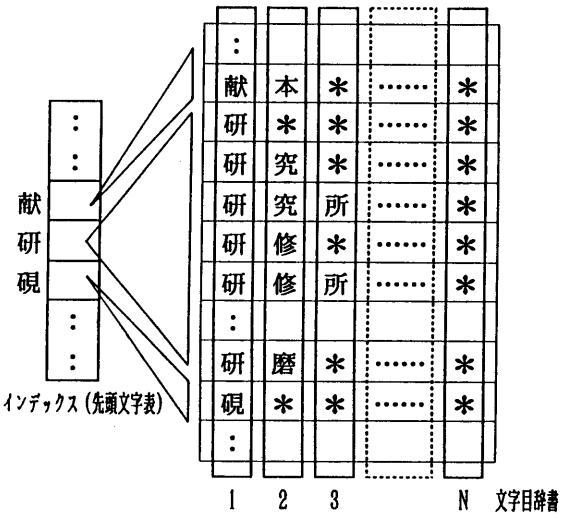


図５　単語辞書の構成とインデックスとの関係

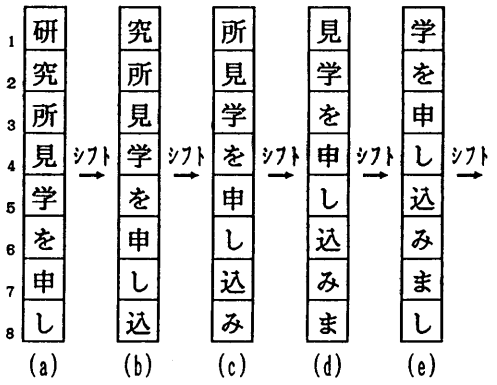


図６　シフトレジスタの動作（Ｎ＝８のときの例）

アドレス生成器は、１文字目辞書～Ｎ文字目辞書に対して、同一のアドレスを与え、かつ、そのアドレスを同時に変化させる。変化させるアドレスの範囲は、シフトレジスタの１文字目と同じ文字で始まる単語群の存在範囲であり、インデックスから得る。

比較器は、単語辞書とシフトレジスタの各文字位置を対応付けるように、全部でＮ個設けられている。ｎ文字目（１≦ｎ≦Ｎ）に対応する比較器は、ｎ文字目辞書から読み出された文字が、シフトレジスタのｎ文字目か残余記号のどちらかに一致したときに、一致信号を出力する（残余記号はワイルドカードの働きをする）。単語辞書内の１単語に対応するＮ文字、および、シフトレジスタ内のＮ文字は、同時に読み出せるようになっており、上記の照合処理は、Ｎ個の比較器で同時に行なわれる。全比較器から一致信号が出力された

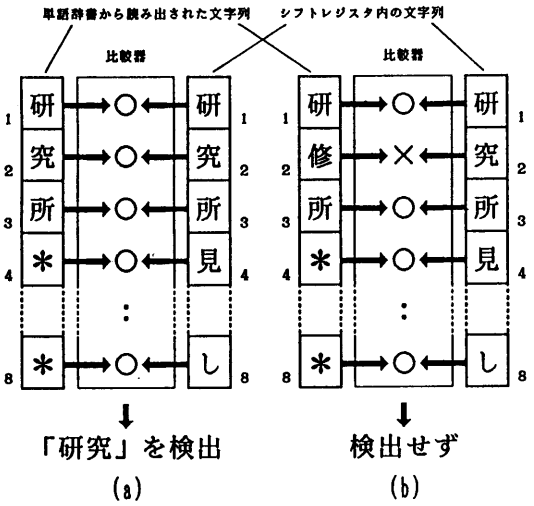


図7　同時照合による単語の検出（N＝8のときの例）

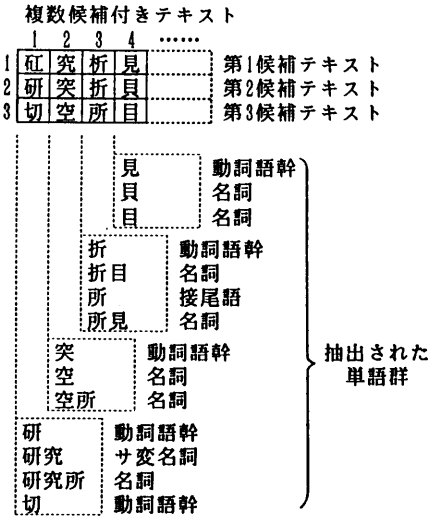


図8　複数候補付きテキストと形態素抽出結果
（M＝3のときの例）

ときに、単語（形態素）が1個検出されたことになる（図7参照）。

さて、上記のような構成にもとづく、形態素抽出のハードウェアアルゴリズムは、以下の通りである。

■メインプロシジャ：

【ステップ1】シフトレジスタに入力テキストの先頭のN文字をロードする。

【ステップ2】1文字目レジスタにテキスト末尾マークが到達するまで、プロシジャ1を繰り返す。

■プロシジャ1：

【ステップ1】1文字目レジスタと同じ文字で始まる単語群の単語辞書内アドレス範囲を、インデックスより得て、その先頭アドレスを単語辞書のカレントアドレスとする。

【ステップ2】カレントアドレスが上記のアドレス範囲内にある間は、プロシジャ2を繰り返す。

【ステップ3】シフトレジスタ内の文字列を、1文字分順送りする。

■プロシジャ2：

【ステップ1】カレントアドレスにおいて単語辞書から読み出されたN文字と、シフトレジスタ内のN文字とを、N個の比較器で同時に照合し、単語の検出を判定する。

【ステップ2】カレントアドレスをインクリメントする。

以下では、動作例を簡単に説明する。

「研究所見学を申し込みました……」という入力テキストに対して、メインプロシジャのステップ1の直後に、シフトレジスタの内容は、図6の(a)のようになる。プロシジャ1のステップ3ごとに、シフトレジスタ内の文字列は、(a)→(b)、(b)→(c)、(c)→(d)、(d)→(e)、……と移動する。プロシジャ1は、図6の(a)(b)(c)(d)(e)……の各々に対して実行される。

図6の(a)の状態では、シフトレジスタの1文字目が「研」である。そこで、プロシジャ1のステップ1では、単語辞書における「研」で始まる単語群のアドレス範囲が、インデックスより得られる。図5の単語辞書では、「研」「研究」「研究所」「研修」「研修所」……「研磨」が、そのアドレス範囲に該当する。そして、そのアドレス範囲内の各単語について、プロシジャ2のステップ1の照合処理が順次実行される。

以上を簡単にまとめると、本アルゴリズムは、次のような3つの手法を組み合わせたものである。

①単語を構成する全文字の同時照合
②先頭文字による辞書範囲の絞り込み
③シフトレジスタによるテキストの順送り

非常にシンプルな手法の組み合わせであるため、構成が簡単になり、2－2で述べた方針通り、照合サイクルの短縮による高速化が可能となる。

3－2　複数候補付きテキストへの拡張

本節では、3－1に示したアルゴリズムを、テキストの各文字が複数個（＝M個とする）の候補をもつ場

合に拡張する。複数候補付きテキストは、文字認識や音声認識の結果などにおいて発生する。図８に、複数候補付きテキストと、それに対する形態素抽出結果の例を示した。

図９は、拡張アルゴリズムを実行するための構成である。図９の構成は、図４の構成に対して、次の３点の修正が加えられている。

まず、シフトレジスタをM個設ける。M個のシフトレジスタは同期して、テキストの同一位置のM個の候補を同時に順送りする。

次に、ｎ文字目に対応する比較器は、ｎ文字目辞書から読み出された文字と、M個のシフトレジスタのM種類のｎ文字目とを照合するようにする。その比較器は、ｎ文字目辞書から読み出された文字が、残余記号、または、M個のシフトレジスタのいずれかのｎ文字目に一致したときに、一致信号を出力する。

第三に、切り換え器を設ける。切り換え器は、M個のシフトレジスタの１文字目を、順番に切り換えてインデックスへ渡す。

このような構成を用いて、複数候補付きテキストに対する形態素抽出を行なうアルゴリズムは、３－１に示したアルゴリズムにおいて、プロシジャ１を次のようなプロシジャに置き換えればよい。

■プロシジャ１：

【ステップ１】カレントレベルを第１候補とする。

【ステップ２】カレントレベルがM以下である間、プロシジャ1.5を繰り返す。

【ステップ３】シフトレジスタ内の文字列を、１文字分順送りする。

■プロシジャ1.5：

【ステップ１】カレントレベルのシフトレジスタの１文字目と同じ文字で始まる単語群の単語辞書内アドレス範囲を、インデックスより得て、その先頭アドレスを単語辞書のカレントアドレスとする。

【ステップ２】カレントアドレスが上記のアドレス範囲内にある間は、プロシジャ２を繰り返す。

【ステップ３】カレントレベルをインクリメントする。

図１０は、このような拡張を行なった場合の、単語の照合（プロシジャ２のステップ１）の例である。単語辞書から読み出された１個の単語の照合に要する時間は、候補数Mによらない。ただし、シフトレジスタの１文字目がM通りになるので、テキストの各位置で

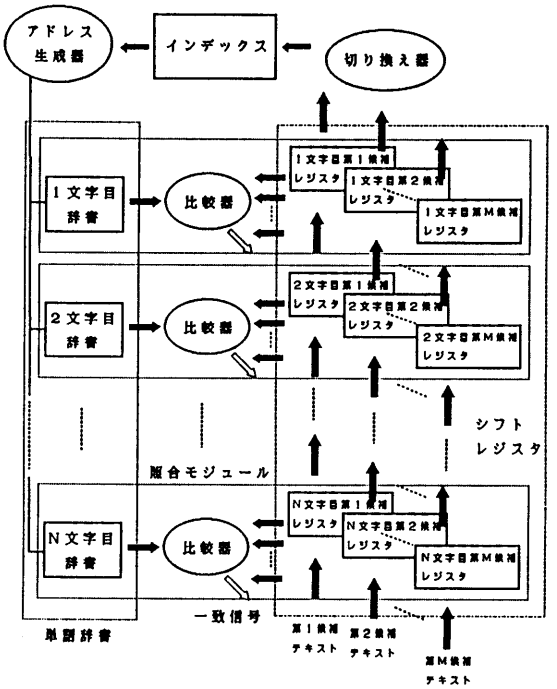


図９　複数候補付きテキストへ拡張した構成

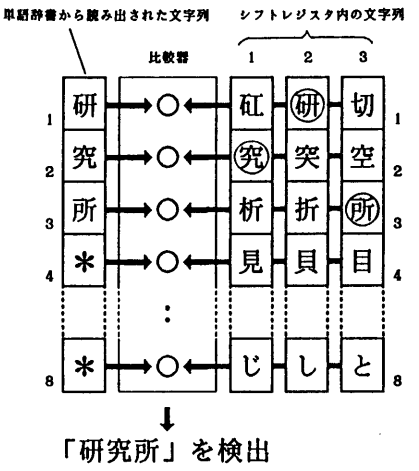


図１０　複数候補付きテキストでの同時照合
（Ｎ＝８、Ｍ＝３のときの例）

照合する単語数（アドレス範囲）は、平均してM倍になる。

従来のソフトウェアでは、候補数がM個になったときの形態素抽出の所要時間は、候補が１個の場合に比較して、Mの累乗オーダー（最悪でM^N倍）になる。それに対して、上記の方式では、所要時間がM倍（線形オーダー）に抑えられる。

本アルゴリズムは、音声認識を対象とした浜口らのアルゴリズム[3]と基本的な考え方は共通している。浜口らのアルゴリズムでは、比較器を用いず、文字（音節）の種類だけのアドレス空間をもつＲＡＭへ候補をマッピングして一致を判定する。したがって、音声のように種類が少なく候補数が多い対象には浜口らのアルゴリズムが適しているが、漢字を含み字種の多いテキストには筆者らのアルゴリズムが適している。さらに、シフトレジスタによるテキストの順送り機構を備えている点も、筆者らのアルゴリズムの特長である。

4　形態素抽出マシン

4－1　マシンの概要・諸元

試作した形態素抽出マシンは、３－２に示した形態素抽出アルゴリズムを実行する専用ハードウェアである。パソコン（ＰＣ９８００シリーズ）のバックエンドマシンとして動作する。約８０個のメモリＩＣ（総容量：約２ＭＢ）と約５００個のロジックＩＣを載せた、合計１２枚のボードから構成されている。図１１にその外観を示した。

アルゴリズムのパラメータは、Ｎ＝８（１６まで拡張可能）、Ｍ＝１～３とした。漢字かな混じりテキストを対象としており、扱う文字コードは１６ビットである（各メモリやレジスタのデータ巾も同様）。単語辞書は約８万見出しのものを用いている（自立語だけでなく付属語・接辞なども含む）。

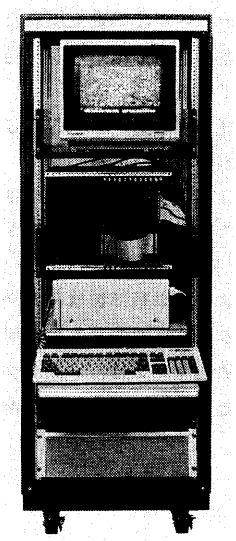
4－2　評価

本マシンは１０ＭＨｚのクロック（１００ｎ秒サイクル）で動作している。１単語の照合処理（３－１に示したプロシジャ２）は、３クロック（＝３００ｎ秒）で実行できる。このとき、形態素抽出の所要時間は、およそ次の式のようになる。

$$A \times D \times L \times M \times 300 \ [n秒]$$

ここで、Ｄは単語辞書の見出し数、Ｌはテキスト長、Ｍは文字位置当りの文字候補数、Ａはインデクシング係数（Ｄに対するインデックスによって絞り込まれた見出しの比率）である。

種々のテキストに関して、形態素抽出所要時間を測定した結果を、図１２に示す。単語辞書の見出し数Ｄは前述の通り約８万件である。この結果は、形態素抽



←試作したハードウェア
（形態素抽出マシン）

←ホスト（PC-9801VX）

図１１　システムの外観

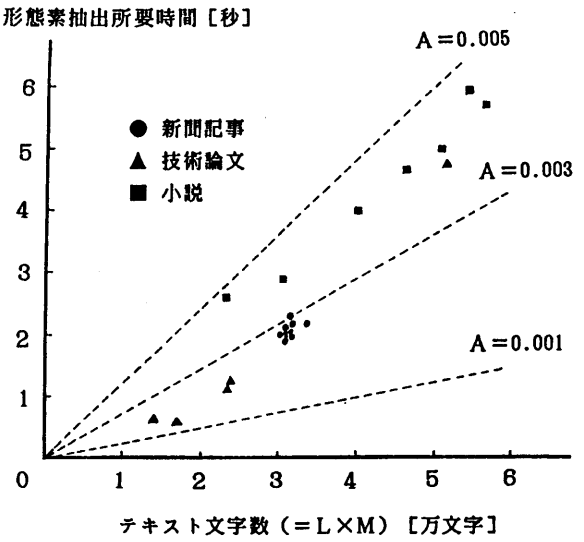


図１２　形態素抽出マシンの形態素抽出所要時間

形態素抽出方式		テキスト1	テキスト2	テキスト3	テキスト4
ソフトウェア	二分探索法[12]	564	642	615	673
	先頭文字表利用二分探索法	133	153	147	155
	順序ハッシュ法[13][14]	406	440	435	416
	木構造見出しの桁探索法[12]	52	56	54	54
形態素抽出マシン		0.56	0.50	0.51	0.44

（単位：秒）

図１３　５千文字テキストに対する処理時間の比較

出マシンによれば、８万見出しの単語辞書を用いた、１万文字のテキストに対する形態素抽出処理が、約１秒で実行できることを示している。

また、４種類のテキスト（いずれも、Ｌ＝５０００、Ｍ＝１）に対する形態素抽出の所要時間を、ソフトウェアと比較した結果を図１３に示す。単語辞書の見出し数Ｄは、これまでと同様に８万件である。ソフトウェアはＰＣ－９８ＸＬ²（ＣＰＵ：８０３８６、クロック：１６ＭＨｚ）上で動作させ、単語辞書はＲＡＭに置いた。パソコンとしてはかなり高性能なものを用いたにもかかわらず、形態素抽出マシンは、ソフトウェアの１００倍～１０００倍という高速な処理速度を達成している。

５　おわりに

文章解析処理の高速化への新しいアプローチとして、文章解析アクセラレータを提案し、その第１ステップとして、形態素抽出マシンを試作した。本マシンでは、８万見出しの単語辞書を用いた、テキスト１万文字に対する形態素抽出処理が、１秒程度で実行できる。この処理速度は、パソコン上のソフトウェアの１００倍～１０００倍という高速なものである。本マシンで実現した形態素抽出のための新しいハードウェアアルゴリズムは、各文字が複数の候補をもったテキストも対象とすることができ、かつ、候補数に対して線形オーダーの処理時間を達成している。

上記の高速性は、シンプルな構成により得られている。このシンプルな構成は、ＶＬＳＩ化に適しており、現在、その検討を進めている。形態素抽出ＬＳＩは、既に商用化されている機械翻訳システムなどの処理速度の向上をもたらすであろうし、多量のドキュメントを対象とした自動校正や自動キーワード付けなどの実用化を促進することも期待できる。また、人間の入力速度に追従するように各種ソフトウェア手法を導入しているかな漢字変換についても、ソフトウェアの負担が軽減され、さらに、従来よりもはるかに大語彙の辞書を用いることも可能になる。

今後は、文章解析アクセラレータのアプローチにしたがって、形態素抽出処理以外についても、ハードウェア化を検討してゆく予定である。文章解析の枠組みは長期的な研究課題であり、その途上においては、ソフトウェアのもつ柔軟性は必要である。しかし、今日、手法がある程度確立された部分もあり、それらについては、積極的なハードウェア化を考えてゆく。さらに、これまでのソフトウェアの枠組みのハードウェア化にとどまらず、専用ハードウェアの可能性を、文章解析の新しい枠組みに結び付けることも考えてゆきたい。

参考文献

[1] 福島・他、文章解析アクセラレータ(1)－形態素抽出マシンの試作－、情処39全大、2F-5、1989年。

[2] 福島・他、多重照合型形態素抽出方式に関する検討、情処39全大、1F-5、1989年。

[3] 浜口・他、音声日本語入力システムにおける高速な言語処理のための辞書照合アルゴリズム、信学論、J70-D(8)、1987年。

[4] 中村・他、形態素抽出アルゴリズムの高速処理方式、情処37全大、5B-9、1988年。

[5] 松本、論理文法の並列構文解析、情処論、29(4)、1988年。

[6] W.Rytter, Parallel Time O(log n) Recognition of Unambiguous Context-free Languages, Info. and Comput., 73, 1987.

[7] 峯・他、文脈自由文法の並列構文解析、情処研報、NL-73-1、1989年。

[8] K.H.Chu, et al., VLSI Architectures for High Speed Recognition of Context-free Languages, 9th Annu.Int.Sympo.Comput.Arch., 1982.

[9] Y.T.Chiang, et al., Parallel Parsing Algorithms and VLSI Implementation for Syntactic Pattern Recognition, IEEE Trans., PAMI-6(3), 1984.

[10] 小倉・他、連想メモリＬＳＩの現状と今後、信学誌、69(7)、1986年。

[11] 山田・他、文字列検索ＬＳＩ、信学技報、CAS87-25、1987年。

[12] D.E.Knuth, The Art of Computer Programming Vol.3: Sorting and Searching, Addison-Wesley, 1973.

[13] O.Amble, et al., Ordered Hash Table, Comput.J., 17(2), 1974.

[14] 横山・他、二次記憶上の大規模語彙を用いる自然言語処理システム、情処論、29(6)、1988年。