専用ハードウェアを用いた形態素解析器の開発

福島俊一†

本論文では，形態素抽出から接続検定までを専用ハードウェアで実行する形態素解析器の構成・アルゴリズム・評価結果について述べる．本形態素解析器では，形態素解析処理を，(1)形態素抽出：単語辞書を検索してテキスト中に出現したと思われる単語（形態素）を抽出する過程，(2)接続検定：単語の隣接制約にもとづいて接続可能な単語を連結する過程，(3)候補選択：単語の組み合わせのなかから一番尤もらしいものを選択する過程，の3つに分け，(1)と(2)について専用ハードウェア（形態素抽出マシンMEX-IIと接続検定マシンMONC）を開発した．逐次型コンピュータを想定した従来の形態素解析アルゴリズムでは，ヒューリスティックスによる候補抑制（解析候補の切り捨て）を用いて高速化するため，処理速度と解析精度との間にトレードオフの関係が生じている．しかし，MEX-IIとMONCによると，候補抑制なしに形態素抽出から接続検定までを，8万語辞書の場合：約9.5千文字/秒，24万語辞書の場合：約5.6千文字/秒，49万語辞書の場合：約2.4千文字/秒の処理速度で実行できる．この処理速度は，パーソナルコンピュータ（CPU：80386，クロック：20 MHz）と比べて80倍以上高速である．これによって，解析候補の切り捨てを行うことなく，総当たり解析の精度を保ったまま高速化することが可能となった．

Development of a Hardware-based Morphological Analyzer

TOSHIKAZU FUKUSHIMA †

This paper describes a hardware-based morphological analyzer. Generally, morphological analysis methods for Japanese text consist of three processes: (1) a morpheme extraction process, (2) a morpheme network construction process, and a morpheme determination process. The morphological analyzer, designed in this paper, uses specific hardware, MEX-II and MONC. MEX-II is a morpheme extraction machine, and MONC is a morpheme network construction machine. MEX-II extracts all morpheme candidates from input text by searching through a morpheme dictionary. MONC examines all possible conjunctions of all the extracted morphemes to construct the morpheme network. MEX-II and MONC, in cooperation, can process 2,400-9,500 character Japanese text in a second. This performance is more than 80 times faster than a personal computer (CPU: 80386, clock: 20 MHz).

1. はじめに

コンピュータによって自然言語を解析する技術は，ワードプロセッサ（仮名漢字変換）や機械翻訳システムなどにおいて実用に結び付いている[1),2)]．特に，近年の日本でのワードプロセッサの急速な普及には，目を見張るものがある．しかし，人間が仕事や日常で手にしたり目にしたりするテキストの多量さに比べて，自然言語解析技術が実用化に結び付いている場面は，まだごくわずかである．応用を広げ，実用化を進めていくためには，より深い解析の追究による高精度化と並行して，処理の高速化を図っていく必要がある．

一般に実用システムにおいて処理の高速性が望まれることは言うまでもないが，自然言語解析の高速化は，具体的に次のような効果をもたらす．まず，自動通訳，仮名漢字変換，自然言語インタフェースなど，対話性の高い応用において実時間の応答を可能にする．一方，機械翻訳，校正支援，自動キーワード付けなどでは，より多量のテキストの一括処理を可能にする．さらに，別な見方をすれば，同一時間内に，より大規模な辞書・知識を参照したり，より多くの解釈（候補）を調べることが可能になり，高精度化にもつながる．

日本語テキストを対象とした応用を考えた場合，まず望まれるのは，自然言語解析の第一ステップである形態素解析の高速化である．単語単位に分かち書きされる欧米語に比べて，べた書きされる日本語テキストはコンピュータ処理に不利である．その結果，欧米語の世界では，既にスペルチェッカ（さらには文法チェ

† 日本電気株式会社情報メディア研究所音声言語研究部
Human Language Research Laboratory, Information Technology Research Laboratories, NEC Corporation

ッカ)が文書作成に不可欠なものとして普及し[3]，データベース検索のための自然言語インタフェースがヒット商品になっている[4]のに対して，日本語の世界では，その類の実用化は遅れている．このような日本語と欧米語との間のギャップを小さくし，日本語のコンピュータ処理の実用化を促進するために，形態素解析の高速化は重要である．

従来の逐次型コンピュータを想定した形態素解析アルゴリズムでは，処理速度と解析精度との間にトレードオフの関係が生じている．初期に考案され，現在も仮名漢字変換などの実用システムに多く用いられている最長一致法や二文節最長一致法[5]では，単語や文節の長さが一定の条件を満たす解析候補しか形成しない．そのため高速ではあるが，長い候補が常に正解とは限らないので，解析精度に上限が生じてしまう．一方，解析精度に重点を置いて，正解をできる限り落とさないようにしたのが，文節数最小法[6]，LAX法(CYK法による形態素解析)[7]，局所的総当たり法[8]などの総当たり型のアルゴリズムである．これらのアルゴリズムは，解析表を利用して重複解析を避けたり，字種境界を参照して組み合わせ範囲を限定するなど，総当たり解析の効率を高める工夫を加えてはいるが，処理速度の面では最長一致法の類に及ばない．さらに，総当たり型のアルゴリズムを改良・一般化したコスト最小法[9]，接続コスト最小法[10]，区間分割文節数限定法[11]などにおいて，処理速度と解析精度のトレードオフ関係はいっそう明確になる．これらのアルゴリズムでは，組み合わせの広がりや形成する文節数を段階的に抑制できる．抑制を加えない状態は総当たり解析と等価であり，抑制を強めるにしたがって解析候補は切り捨てられ，処理は高速になるが，解析候補から正解が洩れる危険性も高くなる．

逐次型コンピュータの枠組みから脱して，並列コンピュータや専用ハードウェアを用いるならば，解析候補を切り捨てずに高速化が達成し得る．並列コンピュータを想定した高速解析アルゴリズムは，最近，特に構文解析レベルで研究が盛んである[12)~14)]．形態素解析についても，前述のLAX法[7]が並列動作可能であるほか，LR状態遷移を利用した構文解析器[14]との連携[15]やマルチプロセッサによる形態素抽出[16]などが検討されている．しかし，並列コンピュータで十分な性能を得るには，今後，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克服されていく必要がある．また，システム規模が大きく，実用システムとして一般に普及するには，まだ時間がかかる．

筆者は，専用ハードウェア*を開発するアプローチを提案し，特に形態素解析の高速化のために，形態素抽出マシンMEX-I[17]・MEX-II[18]（Morpheme EXtraction machine），接続検定マシンMONC[19]（MOrpheme Network Construction machine）の開発を進めてきた．本論文では，MEX-IIとMONCをバックエンドマシンとして用いた形態素解析器の構成・アルゴリズム・評価結果について述べる．専用ハードウェアを用いることで，解析候補の切り捨てを行うことなく，総当たり解析の精度を保ったまま高速化が達成できた．

なお，MEXやMONCに近い専用ハードウェアの開発事例には，音声言語処理向けに浜口らが開発したもの（音素や単語をRAMのアドレス空間にマッピングすることで多候補を操作）[20),21)]がある．しかし，形態素解析器としての明確な設計方針にもとづいて統合可能な機能・構成を備えたものは，本論文で示すものが最初である．

本論文では，まず第2章で，従来のヒューリスティックスによる候補抑制の問題点を指摘し，専用ハードウェアを用いた高速な全候補形成という設計方針を述べる．続いて第3章と第4章では，専用ハードウェアとして開発した形態素抽出マシンMEX-IIと接続検定マシンMONCの各々について，構成とアルゴリズムを説明する．第5章では，MEX-II，MONC，および，それらを用いた形態素解析器の処理速度を中心とした評価結果を示す．

2. 形態素解析器の設計方針

2.1 ヒューリスティックスによる候補抑制の問題点

第1章で触れたように，従来の逐次型コンピュータを想定した形態素解析アルゴリズムでは，処理速度と解析精度との間にトレードオフの関係が生じている．本節では，この点をもう少し詳しく論じ，ヒューリスティックスによる候補抑制のもつ問題を明らかにする．

表1に日本語の形態素解析手法の分類を示す．表1では，字種境界を利用するか否か(2通り)と，解析候補の形成を抑制するか否か（段階的な抑制も含めて3通り）とで，形態素解析手法を2×3=6タイプに分類した．

字種境界を利用するものは，漢字仮名混じり文字列を解析する場合に，その特徴を生かして，仮名から漢

* 文章解析アクセラレータ（natural language parsing accelerators），あるいは，日本語解析マシン（Japanese text parsing machines）と呼んでいる．

表 1 日本語の形態素解析手法の分類
Table 1 Classification for Japanese text morphological analysis methods.

	字種境界非利用	字種境界利用
候補抑制あり（切り捨て型）	最長一致法 二文節最長一致法[5]	長尾らの手法[22]
候補抑制なし（総当たり型）	文節数最小法[6] LAX 法[7]	局所的総当たり法[8]
段階的抑制（段階型）	コスト最小法[9] 接続コスト最小法[10] 確率的形態素解析法[24),25]	区間分割文節数限定法[11]

字への変わり目** などで入力文字列を小さな区間に分割して，候補の組み合わせ範囲を限定する．その区間内で抑制を加えずに全解析候補を形成する「字種境界利用×総当たり型」の手法が局所的総当たり法[8]である．長尾らの手法[22]は，非仮名自立語と仮名付属語で区間をカバーできなかったときに限って仮名自立語を仮定するなどの候補抑制を加えており，「字種境界利用×切り捨て型」である．区間分割文節数限定法[11]は，区間内を最小文節数＋N 個以内でカバーする文節候補のみを形成するもので，N の値を変えることで候補数を段階的に抑制する「字種境界利用×段階型」である．

それに対して字種境界を利用しないものは，漢字仮名混じり文字列であるか，べた書き仮名文字列であるかを意識しない解析方針をとる．「字種境界非利用×切り捨て型」としては，単語や文節の長さを候補抑制条件に用いるものが知られている．最長一致法では，左から右へ解析を進めていく各時点で最長の候補のみを残す．二文節最長一致法[5]では，二文節としての長さが最長の候補のみを残す．一方，文節数最小法[6]や LAX 法[7]は，解析候補を解析表に登録することで重複解析を避けながら，可能な全候補を形成する「字種境界非利用×総当たり型」である．コスト最小法[9]や接続コスト最小法[10]は，解析表における広がりを抑制するパラメータを備えており，「字種境界非利用×段階型」である．また，単語・品詞・文字などの出現確率や連接確率を動的計画法（あるいは接続コスト最小法）の枠組みで扱った確率的形態素解析法[24),25]も，確率値の近似方法や閾値設定をパラメータと考えれば「字種境界非利用×段階型」の一種である．

一般的な傾向として，解析の途中で形成する候補の数が少ない方が処理速度は速い．しかし，通常，解析の途中で候補の切り捨て・抑制に使われる基準は，長い方が尤もらしいとか，自立語が仮名書きされることは少ないなど，確率的には正しいことが多くとも常に正しいとは限らないヒューリスティックスである．したがって，解析候補の抑制を強くすると，解析精度に上限が生じてしまう．

このとき，解析候補を抑制するための基準が，最終的に解析結果の尤もらしさを判定する基準と完全に一致するものであるならば，途中で候補を抑制しても解析精度は変わらないという見方もある．しかし，候補抑制に使われる基準が主に局所的* に判定可能なものであるのに対して，最終的な尤もらしさの判定では，係り受けや結合価・格構造などのやや広域的* な基準も用いられる[26)~28]．したがって，解析途中で候補はできる限り残しておいて，十分な情報がそろった最終段階で尤もらしさの判定を行う方が解析精度は高くなる．

2.2 専用ハードウェアを用いた全候補形成

本論文で述べる形態素解析器は，専用ハードウェアを用いた高速な全候補形成を設計方針とした．逐次型コンピュータの枠組みから脱することによって，第2.1節の最後で述べたような，解析途中で候補を切り捨てずに十分な情報がそろった最終段階で尤もらしさの判定を行う高精度な総当たり型の解析戦略をとったまま，高速化が可能となる．

上記の設計方針にしたがったハードウェア開発にあたり，以下のような機能分担を考えた．

入力された日本語テキストを単語列（形態素列）に分割し各単語の品詞を認定する形態素解析処理を，次のような3過程に分ける[17]（図1参照）．

（1）形態素抽出： 単語辞書を検索し，入力テキスト中に出現したと思われる単語（形態素）を抽出する．

** よく知られているように，仮名から漢字への変わり目は文節の切れ目である可能性が高い[23]．ただし，どんな場合でも分割するわけではなく，表1の字種境界を利用する手法では，「赤ん坊」「お嬢さん」のように変化点をまたがるような単語・文節がとれるときには分割しないようにしている．

* ここでいう「局所的」とは，隣合う2単語あるいは2文節程度の範囲を意味する．一方，「やや広域的」とは，1文程度の範囲を意味する．

(2) 接続検定： 接続表(二単語間の文法的な接続可否条件を記述したもの)を参照して，(1)で抽出された単語のうち接続可能なものを連結する。

(3) 候補選択： 構文・意味などの情報やヒューリスティックスを用いて，(2)で連結された単語の組み合わせのなかから，一番尤もらしいものを選択する。

このように分けた場合，(1)と(2)の2過程に関して，候補抑制を加えずに可能な解析候補はすべて形成するような高速な専用ハードウェアを開発することが，上述の設計方針を満たす．また，(1)(2)の過程は，文字列照合やテーブル参照など比較的単純な処理の多数回の繰り返しで構成されているため，ワイヤードロジックに適している（専用ハードウェアに向いている)．

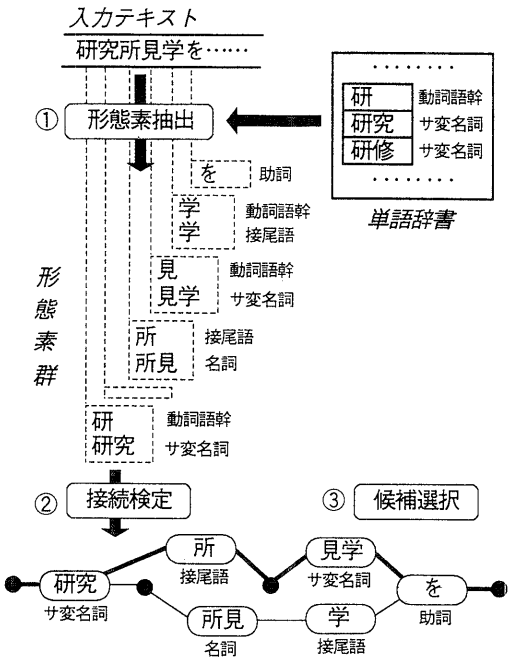


図1 日本語テキストの形態素解析処理
Fig. 1 Morphological analysis process for Japanese text.

(1)を高速化する専用ハードウェアが形態素抽出マシン MEX-II[18] (Morpheme EXtraction machine)で，(2)を高速化する専用ハードウェアが接続検定マシン MONC[19] (MOrpheme Network Construction machine) である．

3. 形態素抽出マシン MEX-II

3.1 構 成

図2に形態素抽出マシン MEX-II の構成を示す．MEX-IIは，テキストメモリ，単語辞書メモリ，インデックスメモリ，単語リストメモリ，シフトレジスタ，先頭文字切り換え器，アドレス生成器，比較器などから成る．

MEX-IIは，テキストメモリに書き込まれた漢字仮名混じりテキストのすべての文字位置で単語辞書を検索し，テキスト中の部分文字列と一致したすべての単語を単語リストメモリに書き込む．単語リストメモリは，抽出された各単語について，テキスト内位置（単語の先頭位置と単語長）と単語辞書から読み出した属性情報（品詞・読みなど）の組を，テキスト内位置の昇順に格納する．

単語辞書の各単語のデータは，見出し部分（表記）

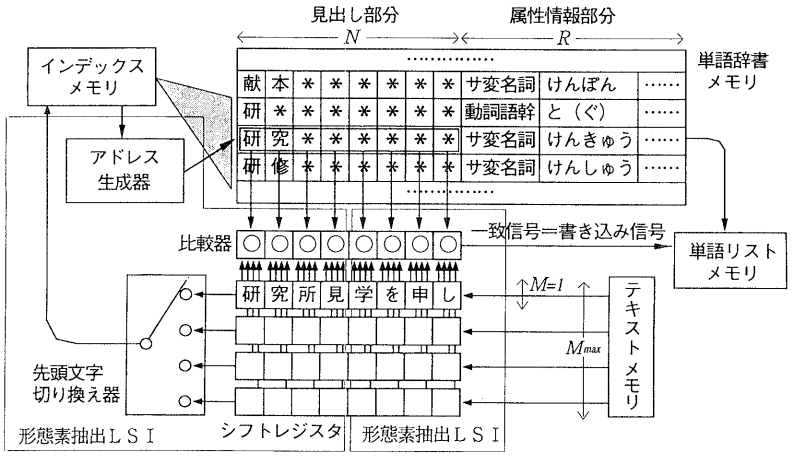


図2 形態素抽出マシン MEX-II の構成
Fig. 2 Architecture of morpheme extraction machine MEX-II.

と属性情報部分とから成る(図2参照)．見出し部分の長さは N 文字で，シフトレジスタの長さに一致する．属性情報部分は長さ R で，単語辞書は，サイズ $N+R$ のデータが一度に読み出せる構造になっている．単語辞書の見出し部分について，格納する表記の長さが N に満たない場合には，残余記号（図2では「＊」で表現）を詰めておく．

また，文字認識結果のようにテキストの各文字が複数の認識候補をもつ場合には，シフトレジスタを M 個用いる*（$M \leqq M_{max}$）．このとき，1文字当たりの認識候補数は M 個である．各比較器は，単語辞書側から読み出された1文字が，シフトレジスタ側から読み出された M 文字の1つにでも一致したら，一致信号を出す．その際，上記の残余記号には，任意の1文字と一致するワイルドカードの働きをもたせる．

3.2 アルゴリズム

MEX-IIで実行する形態素抽出アルゴリズムは基本的に，筆者が論文[17]で提案したものである．そこで，ここではアルゴリズムの正確な記載は省略し，概要を述べるにとどめておく．

テキストはシフトレジスタで順送りする．そして，シフトレジスタの各状態について，シフトレジスタの先頭文字に対応する単語辞書内範囲を読み出す．単語辞書から単語を1個読み出すごとに，N 文字分の比較器で同時照合を実行する．N 文字とも一致が検出されたら（前述のように残余記号「＊」はワイルドカードとして扱う），その単語に関する情報を単語リストメモリに書き込む．

この形態素抽出アルゴリズムは，従来，ソフトウェアで実現されているもの[30]をそのままハードウェア化したものではない．テキストの順送りを高速に行うためにシフトレジスタを用い，ワイルドカードを用いた同時照合により，見出し部分の照合を，その長さによらず1回で行えるようにしている．また，従来ソフトウェアでは，通常，見出し部分の照合を行って単語を検出した後，その次の段階の処理として，属性情報部分の読み出しを行う．しかし，MEX-IIでは，見出し部分を読み出すのと同時に，その単語の属性情報部分も読み出すようにしている．したがって，属性情報の読み出しのために，余分な時間を必要としない*．

4. 接続検定マシン MONC

4.1 構　　成

図3に接続検定マシンMONCの構成を示す．MONCは，単語条件リストメモリ，接続表メモリ，単語ネットワークメモリ，単語カウンタ，位置条件判定器，文法条件判定器などから成る．

MONCは，単語条件リストメモリに書き込まれた各単語について，文節の先頭/末尾になり得るか否か，および，直後に位置する単語と文法的に接続可能か否かを，接続表メモリを参照して調べ，その結果を単語ネットワークメモリに書き込む．

単語条件リストメモリは，図3に示すように，単語のテキスト内位置（単語の先頭位置と単語長）・前方接続属性・後方接続属性をテキスト内位置の昇順に格納したもので，MEX-IIの単語リストメモリのサブセットである．

単語ネットワークメモリは，単語条件リストメモリ

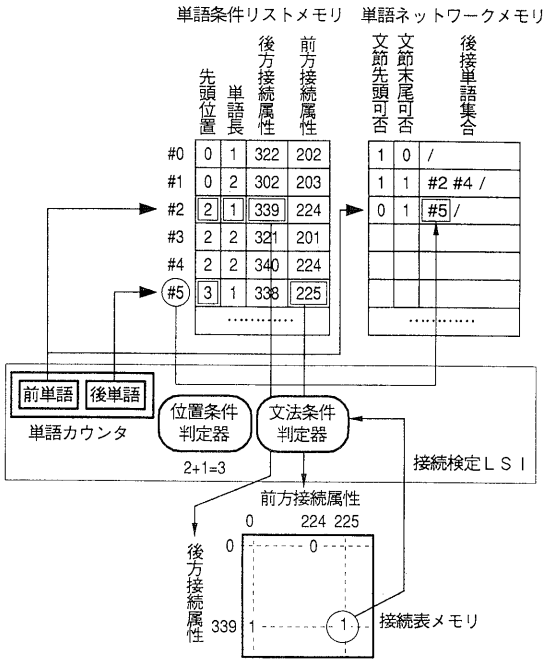


図3 接続検定マシンMONCの構成

Fig. 3 Architecture of morpheme network construction machine MONC.

* 各文字が複数の認識候補をもつ場合（$M>1$）の形態素解析は，個々の文字の認識処理では文字を決定できないときに，前後の文字を組み合わせて単語や文として判断することで正しい文字を決定する役目をもつ[29]．MEX-IIは $M>1$ のテキストに対応できるが，形態素解析が適用されるのは通常 $M=1$ の場合が多いので，以降の評価・説明では，基本的に $M=1$ として扱う．なお，$M>1$ のときは，$M=1$ のときよりも専用ハードウェアの効果が大きくなる[17]．

* 筆者が論文[17]で提案したアルゴリズムでは，単語の属性情報の読み出しまでは含めていなかったが，MEX-IIでは，形態素解析器に組み込むことを想定して，比較器での見出し部分の照合と同時に単語辞書から属性情報を読み出すように改良を加えた．

の各単語に対応させて，文節先頭可否フラグ・文節末尾可否フラグ・後接単語集合（単語条件リストのアドレスの集合として表現）を格納する（図3参照）．この単語ネットワークメモリに格納されたデータは，図1に示したような解析候補（接続検定結果）のネットワーク構造と等価である．

接続表は，文法的な接続可否条件を記述した2次元のビット配列で，行は単語の後方接続属性，列は単語の前方接続属性に対応する（図3参照）．テキスト内で単語Bが単語Aの直後に位置するとき，Aの後方接続属性の値をx，Bの前方接続属性の値をyとすると，接続表の要素(x, y)の値が1ならばAの直後にBは接続可能で，0ならば接続不可を意味する．また，行0と列0は各々，文節先頭と文節末尾に対応する．すなわち，接続表の要素$(0, y)$は単語Bの文節先頭可否を表わし，接続表の要素$(x, 0)$は単語Aの文節末尾可否を表わす．

単語カウンタは，接続可否判定の対象とする2単語のアドレスを指す．位置条件判定器は，単語条件リストメモリ内の先頭位置と単語長の値をもとに，2単語が隣接するかを判定する．文法条件判定器は，単語条件リストメモリ内の後方接続属性と前方接続属性をもとに，接続表メモリをアクセスして，文節先頭可否・文節末尾可否・2単語接続可否を判定する．

4.2 アルゴリズム

MONCは，次のようなアルゴリズムで接続検定を実行する．

□接続検定アルゴリズム

《メインプロシジャ》

［ステップ0.1］ 単語カウンタの前単語を単語条件リストの先頭の単語とする．

［ステップ0.2］ 前単語が単語条件リストの最後の単語を越えない間は，プロシジャ1を繰り返す．

《プロシジャ1》

［ステップ1.1］ 文法条件判定器が前単語に関する文節先頭可否を判定し，その結果（文節先頭可否フラグ）を単語ネットワークに書き込む．一方で，前単語のテキスト内先頭位置＋単語長の値Pを計算する．

［ステップ1.2］ 文法条件判定器が前単語に関する文節末尾可否を判定し，その結果（文節末尾可否フラグ）を単語ネットワークに書き込む．一方で，単語条件リストにおける前単語の次の単語を，単語カウンタの後単語とする．

［ステップ1.3］ プロシジャ2を繰り返し実行することで，前単語に対する後接単語集合を求め，その結果を単語ネットワークに書き込む．

［ステップ1.4］ 前単語をインクリメントする．

《プロシジャ2》（終了判定されるまで繰り返す）

［ステップ2.1］ 位置条件判定器が後単語のテキスト内先頭位置を値Pと比較する．それと同時に，文法条件判定器が前単語の後方接続属性と後単語の前方接続属性の値をもとに，接続表をアクセスして，文法的な接続可否を判定する．

［ステップ2.2］ 後単語のテキスト内先頭位置の値がPを越えた場合は，プロシジャ2を終了する．後単語のテキスト内位置の値がPに一致し，かつ，文法的に接続可能と判定された場合は，その後単語を現在の前単語に対する後接単語集合に加える．

［ステップ2.3］ 後単語をインクリメントする．後単語が単語条件リストの最後の単語を越えた場合は，プロシジャ2を終了する．

（接続検定アルゴリズム終わり）□

図3を参照して上記のアルゴリズムを補足説明する．

図3では，単語カウンタの前単語は#2を指している．前単語が#2のとき，ステップ1.1では，接続表の要素(0, 224)の値（=0）が読み出されて#2の文節先頭可否の値として書き込まれる．それと同時に，$P=2+1=3$が計算される．次のステップ1.2では，接続表の要素(339, 0)の値（=1）が読み出されて#2の文節末尾可否の値として書き込まれる．それと同時に，単語カウンタの後単語が，単語#2の次の#3にセットされる．

ステップ1.3では，前単語#2に対して，後単語を#3，#4，#5，……とインクリメントしながら，2単語の間の接続可否を判定していく．図3では，単語カウンタの後単語は#5を指している．ステップ2.1では，位置条件と文法条件を同時に判定している．#5の先頭位置3はPの値に一致するので，位置条件を満たす．また，接続表の要素(339, 225)の値は1なので，文法条件も満たす．したがって，ステップ2.2では，#2の後接単語集合に#5が追加される．

上記のアルゴリズムは，MEX-IIのアルゴリズムと比べると，従来のソフトウェアで実現されているアルゴリズムに近い．従来の処理の流れをベースに，独立した手続きを並行実行するようにした．例えば，2単語が接続するための位置条件と文法条件を同時に判定し

ている．また，ワイヤードロジックの適性を考慮して，ポインタでジャンプするような操作はなるべく避け，メモリを連続的にアクセスするようにして，高速化を図った．

5. 評　　価

5.1 形態素解析器の試作

専用ハードウェアを用いることによって高速な全候補形成が実現できることを検証するために，形態素抽出マシン MEX-II と接続検定マシン MONC，および，それらを用いた形態素解析器を試作した．**図4** にその外観を示す．

試作した MEX-II は，形態素抽出 LSI×2 個と，テキストメモリ，単語辞書メモリ，インデックスメモリ，単語リストメモリから成る 1 ボードマシンである（図2参照）．形態素抽出 LSI は，シフトレジスタ，先頭文字切り換え器，アドレス生成器，比較器を含む．MEX-II の諸元を**表2** に示す．

試作した MONC は，接続検定 LSI×1 個と，単語条件リストメモリ，接続表メモリ，単語ネットワークメモリから成る 1 ボードマシンである（図3参照）．接続検定 LSI は，単語カウンタ，位置条件判定器，文法条件判定器を含む．MONC の諸元を**表3** に示す．

形態素解析器は，パーソナルコンピュータ PC-9801 DA をホストとし，MEX-II と MONC をバックエンドマシンとしている．MEX-II と MONC を用いた形態素解析処理の流れは次のようになる．

① MEX-II と MONC の初期設定を行う*．
② 解析対象テキストをホストから MEX-II へ転送する（MEX-II のテキストメモリに書き込む）．
③ MEX-II で形態素抽出を実行する．
④ 形態素抽出結果を MEX-II からホストへ転送し（MEX-II の単語リストメモリの内容を読み出す），その部分情報をホストから MONC へ転送し直す（MONC の単語条件リストメモリに書き込む）．
⑤ MONC で接続検定を実行する．
⑥ 接続検定結果を MONC からホストへ転送する（MONC の単語ネットワークメモリの内容を読み出す）．
⑦ ホストにおいて，候補選択をソフトウェアで実行する．

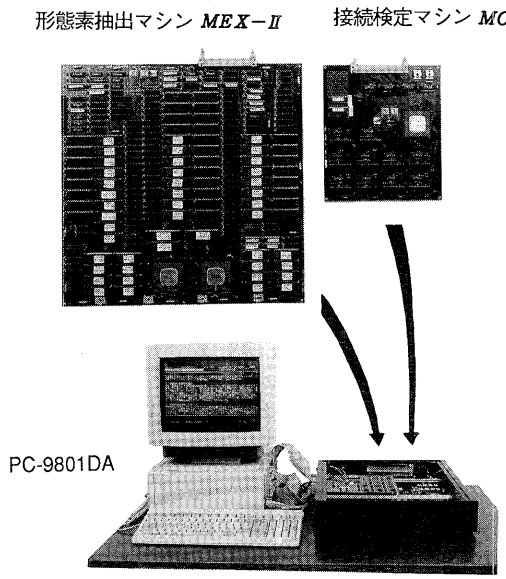


図4 試作した形態素解析器の外観
Fig. 4 Overall view of hardware-based morphological analyzer.

ソフトウェアで実現した候補選択処理は，表1の分類で言えば「字種境界利用×総当たり型」の基準を用いた．字種境界を利用して定めた区間ごとに，接続検定まで済んだ単語連鎖の各々について，個々の品詞あるいは品詞組に与えたコストの累積計算を実行し，累積コストが最良となるものを求める．単語連鎖が途切れた箇所には未知語を仮定する．品詞や品詞組のコストはヒューリスティクスにもとづいている．ただし，今回の評価では，専用ハードウェアによる全候補形成の高速性に着眼するので，高精度化のためのコストのチューニングなどは行っていない．

5.2 処理速度

表4 に示す3種類の単語辞書（8万語辞書，24万語辞書，49万語辞書）を用いて，形態素解析器の処理速度を評価した．形態素解析の対象テキストには5通りの2千文字テキスト N 1～N 5（いずれも新聞記事）を用いた．また，専用ハードウェアの高速性を確認するため，形態素抽出と接続検定をソフトウェアで実行した場合との比較も行った．その結果を**表5** に示す．また，表5の平均値をグラフ化したものを**図5** に示す．

ここで，ソフトウェアで実行した形態素抽出には桁探索法[30]を用いた．桁探索法は，二分探索法や順序ハッシュ法などのソフトウェア的探索手法のなかで，形態素抽出に用いた場合に最も高速な手法である[17]．一方，接続検定については，MONC のアルゴリズムと従来ソフトウェアで実現されていたアルゴリズムとで基

* 初期設定としては，ハードウェア的なリセットと，MONC の接続表メモリ内容のロードがある．今回は，MEX-II の単語辞書メモリとインデックスメモリには ROM を用いたのでロードの必要がない．ただし，RAM を用いて単語辞書を変更可能・ユーザ登録可能とした場合には，この初期設定の段階で，その内容をロードすることになる．

表 2 試作したMEX-IIの諸元
Table 2 Design parameters of MEX-II.

項目	内容
ボードサイズ	40 cm×40 cm
メモリサイズ	
●テキストメモリ	4 K バイト×4(SRAM)
●インデックスメモリ	5 バイト×64 K(EPROM)
●単語辞書メモリ	42 バイト/語×512 K 語(EPROM)
●単語リストメモリ	32 バイト/語×64 K 語(SRAM)
形態素抽出 LSI	
●パッケージサイズ	48.26 mm×48.26 mm(280 ピン PGA)
●プロセス技術	1.2 μm ルール，2 層アルミ配線，CMOS ゲートアレイ
●使用ゲート数	約 5 K ゲート
●動作周波数	12 MHz
アルゴリズムのパラメータ	$N=8$(16 バイト) $R=26$ バイト $M_{max}=4$

表 3 試作したMONCの諸元
Table 3 Design parameters of MONC.

項目	内容
ボードサイズ	20 cm×23 cm
メモリサイズ	
●単語条件リストメモリ	5 バイト×32 K(SRAM)
●接続表メモリ	256 ビット×351 ビット(SRAM)
●単語ネットワークメモリ	2 バイト×32 K+2 バイト×16 K(SRAM)
接続検定 LSI	
●パッケージサイズ	43.18 mm×43.18 mm(208 ピン PGA)
●プロセス技術	1.2 μm ルール，2 層アルミ配線，CMOS ゲートアレイ
●使用ゲート数	約 4.4 K ゲート
●動作周波数	12 MHz

表 4 評価に用いた単語辞書
Table 4 Word dictionaries.

辞書の種類	語数	辞書の内訳
8 万語辞書	78,988	仮名漢字変換用辞書
24 万語辞書	241,382	接辞付き語や複合語を中心に追加
49 万語辞書	493,902	固有名詞を大幅に追加

本的な流れは同じなので，第 4.2 節のアルゴリズムを逐次型にしたもの（位置条件と文法条件の判定のように並列実行部分を逐次実行にしたもの）をソフトウェア版とした．ソフトウェア版の形態素抽出と接続検定は，候補選択処理と同様に PC-9801 DA(CPU：80386，クロック：20 MHz）で実行した．

表 5・図 5 の結果にもとづいて，MEX-II，MONC，および，それらを用いた形態素解析器の処理速度についてまとめると，以下のとおりである*．

（1）MEX-IIを用いた形態素抽出の実行速度（1 秒当たり処理可能なテキスト量)は，次のとおりである．

- 8 万語辞書の場合：約 5.0 万文字/秒．
- 24 万語辞書の場合：約 1.8 万文字/秒．
- 49 万語辞書の場合：約 0.8 万文字/秒．

（2）MEX-II と MONC を用いた候補抑制を加えない形態素抽出から接続検定までの実行速度は，次のとおりである．

- 8 万語辞書の場合：約 9.5 千文字/秒．
- 24 万語辞書の場合：約 5.6 千文字/秒．
- 49 万語辞書の場合：約 2.4 千文字/秒．

（3）形態素抽出を PC-9801 DA 上のソフトウェア(桁

* 本論文では，前述のとおり，専用ハードウェアによる全候補形成の高速性の検証に主眼を置いているため，解析精度面でのチューニングは行っていない．参考の意味で，49 万語辞書を用いた場合の解析精度は，単語単位で区切りと品詞の両方が正しいものを正解としてカウントして約 98.5%であった（新聞記事 N 1 で評価）．

表 5 2千文字テキストに対する実行時間
Table 5 Implementation time for 2,000 character text.

(a) 8万語辞書

テキスト	T_1	T_1^{mex}	$\frac{T_1}{T_1^{mex}}$	N_{word}	T_2	T_2^{monc}	$\frac{T_2}{T_2^{monc}}$	N_{link}	T_3
N 1	16秒	0.04秒	417	6,487	16秒	0.17秒	93	3,484	16秒
N 2	15秒	0.03秒	442	5,565	14秒	0.14秒	97	2,954	16秒
N 3	16秒	0.04秒	411	6,757	17秒	0.19秒	92	3,727	18秒
N 4	16秒	0.04秒	440	7,098	18秒	0.19秒	93	3,683	17秒
N 5	16秒	0.04秒	416	6,621	17秒	0.18秒	96	3,564	18秒
平均	16秒	0.04秒	425	6,506	16秒	0.17秒	94	3,482	17秒

(b) 24万語辞書

テキスト	T_1	T_1^{mex}	$\frac{T_1}{T_1^{mex}}$	N_{word}	T_2	T_2^{monc}	$\frac{T_2}{T_2^{monc}}$	N_{link}	T_3
N 1	20秒	0.11秒	180	7,770	22秒	0.24秒	92	4,196	23秒
N 2	19秒	0,12秒	161	6,803	18秒	0.20秒	89	3,587	17秒
N 3	20秒	0.12秒	165	8,186	23秒	0.26秒	87	4,620	21秒
N 4	21秒	0.11秒	196	8,566	24秒	0.27秒	88	4,447	19秒
N 5	20秒	0.11秒	180	7,975	22秒	0.25秒	88	4,427	21秒
平均	20秒	0.11秒	176	7,860	22秒	0.25秒	89	4,255	20秒

(c) 49万語辞書

テキスト	T_1	T_1^{mex}	$\frac{T_1}{T_1^{mex}}$	N_{word}	T_2	T_2^{monc}	$\frac{T_2}{T_2^{monc}}$	N_{link}	T_3
N 1	26秒	0.25秒	104	12,711	48秒	0.59秒	82	7,183	37秒
N 2	25秒	0.24秒	105	10,950	39秒	0.48秒	82	6,045	25秒
N 3	27秒	0.26秒	105	13,133	50秒	0.61秒	82	7,922	32秒
N 4	27秒	0.25秒	109	14,031	53秒	0.65秒	81	7,366	29秒
N 5	26秒	0.25秒	106	13,251	50秒	0.60秒	83	7,542	33秒
平均	26秒	0.25秒	106	12,815	48秒	0.59秒	82	7,212	31秒

T_1：ソフトウェアによる形態素抽出所要時間　T_2：ソフトウェアによる接続検定所要時間
T_1^{mex}：MEX-IIによる形態素抽出所要時間　T_2^{monc}：MONCによる接続検定所要時間
N_{word}：形態素抽出結果として抽出された単語数　N_{link}：各単語の後接単語数の累積
T_3：ソフトウェアによる候補選択所要時間

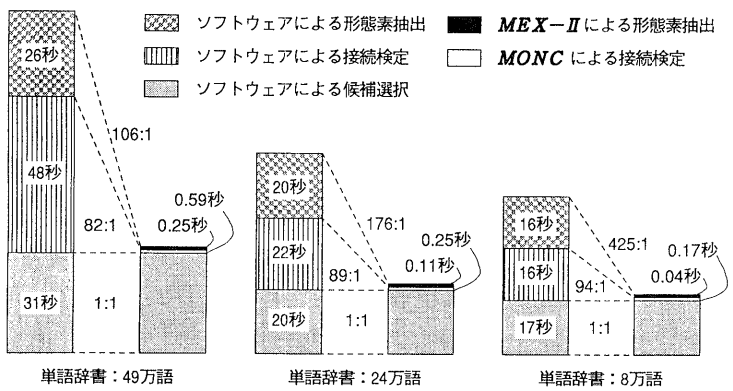


図 5 2千文字テキストに対する平均実行時間の比較
Fig. 5 Comparison of average implementation time for 2,000 character text.

探索法）で実行した場合とMEX-IIの処理速度を比較すると，MEX-IIの高速性は次のとおりである．

- 8万語辞書の場合：平均425倍高速．
- 24万語辞書の場合：平均176倍高速．
- 49万語辞書の場合：平均106倍高速．

（4）接続検定をPC-9801 DA上のソフトウェアで実行した場合とMONCの処理速度を比較すると，MONCが80～100倍程度高速である．

（5）図5から見て取れるように，専用ハードウェア（MEX-II，MONC）によって形態素抽出と接続検定に要する時間が大幅に短縮されたため，形態素解析の処理時間はほとんど候補選択処理のみで占められるようになった（その結果，総処理時間は約1/3に短縮された）．候補選択処理のソフトウェア（字種区切り利用×総当たり型）はチューニングしていないこともあるが，PC-9801 DAで54～125文字/秒程度である．

5.3 アルゴリズムの特性

MEX-IIで用いている形態素抽出アルゴリズムでは，Dを単語辞書の語数，Mをテキストの各位置当たりの平均候補文字数，Lをテキストの長さとすると，形態素抽出に要する時間T_1は，およそ次のような関係になる[17]．

$$T_1 \approx \alpha \cdot L \cdot M \cdot D$$

ここで，αは係数で，単語辞書の語数Dに対する各文字位置で単語辞書から読み出される単語数の割合（絞り込み率）に，1語の照合に要する単位時間を乗じたものである．D，M，Lは独立の関係にあるから，テキスト内各位置での平均処理時間（形態素抽出速度）T_1/Lの単語辞書の語数Dとの関係は$O(D)$である．

表5の結果にもとづいて単語辞書の語数Dと形態素抽出速度との関係をプロットしたのが**図6**である．単語辞書内の照合対象範囲は文字出現傾向に依存するので必ずしも一様ではなく，厳密には単語辞書によって絞り込み率の値は変化すると考えられるが，図6では$O(D)$の関係が粗く成立しているのがわかる．それに対して，桁探索法のソフトウェアの形態素抽出速度は$O(\log D)$である[17]．したがって，表5からも見て取れるように，単語辞書の語数Dが大きくなるほど，MEX-IIとソフトウェアとの処理速度の差は小さくなる．しかし，差が小さくなるといっても，D＝49万語の場合で100倍以上の差がある．

MONCで用いている接続検定アルゴリズムでは，Nを形態素抽出結果として得られた単語の総数，Lをテキストの長さとすると，接続検定に要する時間T_2は，およそ次のような関係になる．

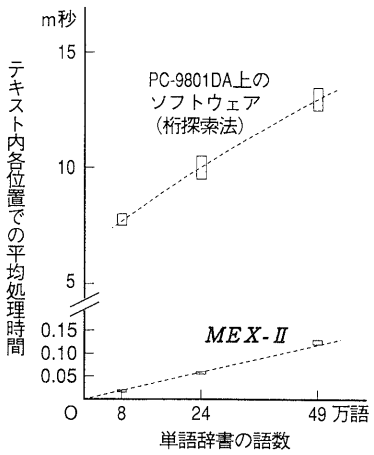


図6 単語辞書の語数に対する形態素抽出速度
Fig. 6 Morpheme extraction speed for dictionary size.

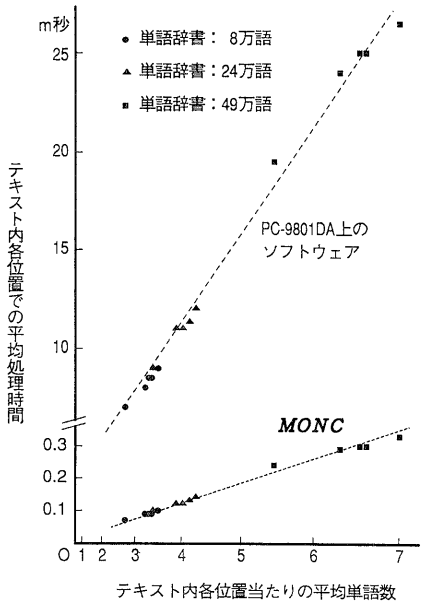


図7 テキスト内各位置当たりの平均単語数に対する接続検定速度
Fig. 7 Morpheme network construction speed for average number of extracted words.

$$T_2 \approx \beta \cdot N^2/L = \beta \cdot n^2 \cdot L$$

ここで，$n = N/L$はテキスト内各位置当たりの平均抽出単語数を表わし，βは係数である．したがって，テキスト内各位置での平均処理時間（接続検定速度）T_2/Lは$O(n^2)$である．

表5の結果にもとづいてテキスト内各位置当たりの平均抽出単語数nと接続検定速度との関係をプロットしたのが**図7**である．図7の横軸の目盛りはn^2に比例するように設定してある．MONCもソフトウェア

版も $O(n^2)$ の関係が粗く成立しているのがわかる．ただし，MONC とソフトウェアとでは，係数 β の大きさに 80～100 倍程度の開きがある．

6. おわりに

形態素抽出から接続検定までを専用ハードウェアで実行する形態素解析器を開発した．本形態素解析器では，可能なすべての解析候補を専用ハードウェアによって高速に形成することを設計方針としている．そのため，形態素解析処理を，(1)形態素抽出：単語辞書を検索してテキスト中に出現したと思われる単語（形態素）を抽出する過程，(2)接続検定：単語の隣接制約にもとづいて接続可能な単語を連結する過程，(3)候補選択：単語の組み合わせのなかから一番尤もらしいものを選択する過程，の3つに分け，(1)と(2)について，候補抑制せずに高速実行する専用ハードウェアを開発して用いている．

開発した専用ハードウェアである形態素抽出マシン MEX-II と接続検定マシン MONC によれば，候補抑制なしに形態素抽出から接続検定までを，8万語辞書の場合：約 9.5 千文字/秒，24 万語辞書の場合：約 5.6 千文字/秒，49 万語辞書の場合：約 2.4 千文字/秒の処理速度で実行できる．この処理速度は，パーソナルコンピュータ（CPU：80386，クロック：20 MHz）と比べて 80 倍以上高速なものである（8万語辞書の場合：約 89 倍，24 万語辞書の場合：約 116 倍，49 万語辞書の場合：約 152 倍）．

逐次型コンピュータを想定した従来の形態素解析アルゴリズムでは，処理を高速化するために解析候補の抑制（切り捨て）を行わざるを得なかったが，専用ハードウェアを用いるならば，候補抑制を加えずに高速化が可能であることを確認できた．

今後の課題としては，専用ハードウェア部分と候補選択処理との連携を強めることが考えられる．例えば，MONC に接続チェックと並行して接続コストも計算する機構をもたせてしまえば，候補選択処理を含めた総処理時間が短縮できる．また，MEX-II，MONC，候補選択ソフトウェアのパイプライン処理も考えられる．

総当たり解析を高速化するための別なアプローチとして，並列コンピュータ上でのプログラミングも今後は有効であろう．ただし，専用ハードウェアを開発するアプローチでは，構成要素の単純性から小型化が狙える．1チップ LSI に集積するなどにより，実用性を高めていくことも将来の方向の1つである．

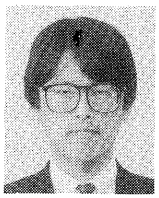
謝辞 形態素解析の高速化のために専用ハードウェアを開発するアプローチを共に検討・提案し，本研究においても多くの助言をいただいた日本電気株式会社関西 C & C 研究所宮井均部長，同社パーソナル C & C 開発研究所大山裕課長に深謝する．

参考文献

1) 石田晴久，木村　泉，安田寿明(編)：ワープロと日本語処理，bit 別冊，共立出版（1985）．
2) 野村浩郷，田中穂積(編)：機械翻訳，bit 別冊，共立出版（1988）．
3) Computer Aids for Authors and Editors—a Natural Extension of Word Processing and Typesetting, *The Seybold Report on Publishing Systems,* Vol. 13, No. 10 (1984).
4) Hendrix, G. G. and Walter, B. A.: The Intelligent Assistant, *Byte,* Vol. 12, No. 14, pp. 251-258 (1987).
5) 牧野　寛，木澤　誠：べた書き文の分かち書きと仮名漢字変換—二文節最長一致法による分かち書き—，情報処理学会論文誌，Vol. 20，No. 4，pp. 337-345（1979）．
6) 吉村賢治，日高　達，吉田　将：文節数最小法を用いたべた書き日本語文の形態素解析，情報処理学会論文誌，Vol. 24，No. 1，pp. 40-46（1983）．
7) 杉村領一，赤坂宏二，松本裕治：並列形態素解析システム LAX の実現，第 35 回情報処理学会全国大会論文集，pp. 1323-1324（1987）．
8) 宮崎正弘，大山芳史：日本文音声出力のための言語処理方式，情報処理学会論文誌，Vol. 27，No. 11，pp. 1053-1061（1986）．
9) 吉村賢治，武内美津乃，津田健蔵，首藤公昭：未登録語を含む日本語文の形態素解析，情報処理学会論文誌，Vol. 30，No. 3，pp. 294-301（1989）．
10) 久光　徹，新田義彦：接続コスト最小法による形態素解析の提案と計算量評価について，電子情報通信学会技術研究報告，NLC 90-8（1990）．
11) 福島俊一，佐々木伸太郎，赤石沢元博，竹元義美：日本語文書校正支援システム St. WORDS，第 45 回情報処理学会全国大会論文集，pp. 3-275-3-276（1992）．
12) 松本裕治：論理文法の並列構文解析，情報処理学会論文誌，Vol. 29，No. 4，pp. 335-341（1988）．
13) Rytter, W.: Parallel Time $O(\log n)$ Recognition of Unambiguous Context-free Languages, *Inf. Comput.,* No. 73, pp. 75-86 (1987).
14) 峯　恒憲，谷口倫一郎，雨宮真人：一般の文脈自由文法に対する効率的な並列構文解析，情報処理学会論文誌，Vol. 32，No. 10，pp. 1225-1237（1991）．
15) 峯　恒憲，谷口倫一郎，雨宮真人：日本語の並列形態素解析，第 40 回情報処理学会全国大会論文集，pp. 452-453（1990）．
16) 中村　修，田中明通，菊池英夫：形態素抽出アル

ゴリズムの高速処理方式，第37回情報処理学会全国大会論文集，pp. 1002-1003 (1988).
17) 福島俊一：形態素抽出ハードウェアアルゴリズムとその実現，情報処理学会論文誌，Vol. 32, No. 10, pp. 1259-1268 (1991).
18) 福島俊一：形態素抽出マシンMEX-II，情報処理学会自然言語処理研究会資料，NL-81-1 (1991).
19) 福島俊一：文章解析アクセラレータ(2)接続検定マシンMONCの試作と評価，第44回情報処理学会全国大会論文集，pp. 3-163-3-164 (1992).
20) 浜口重建，鈴木義武：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，電子情報通信学会論文誌D，Vol. J 70-D, No. 8, pp. 1589-1596 (1987).
21) 浜口重建，中津良平：連続音声認識における高速な言語処理のための接続検定LSI，電子情報通信学会技術研究報告，SDM 87-103 (1987).
22) 長尾　真，辻井潤一，山上　明，建部周二：国語辞書の記憶と日本語文の自動分割，情報処理，Vol. 19, No. 6, pp. 514-521 (1978).
23) 江川　清：漢字かな混り文の自動単位分割に関する一研究，計量国語学，No. 43/44, pp. 46-52 (1968).
24) 藤崎哲之助：動的計画法による漢字仮名混り文の単位切りと仮名ふり，情報処理学会自然言語処理研究会資料，NL-28-5 (1981).
25) Nagata, M.: A Stochastic Japanese Morphological Analyzer Using a Forward-DP Backward-A* N-Best Search Algorithm, *Proc. of COLING'94*, pp. 201-207 (1994).
26) 宮崎正弘：係り受け解析を用いた複合語の自動分割法，情報処理学会論文誌，Vol. 25, No. 6, pp. 970-979 (1984).
27) 大島義光，阿部正博，湯浦克彦，武市宣之：格文法による仮名漢字変換の多義解消，情報処理学会論文誌，Vol. 27, No. 7, pp. 679-687 (1986).
28) 山田洋志，大山　裕：結合価を用いたかな漢字変換，第36回情報処理学会全国大会論文集，pp. 1139-1140 (1988).
29) 西野文人：文字認識における自然言語処理，情報処理，Vol. 34, No. 10, pp. 1274-1280 (1993).
30) Knuth, D. E.: *The Art of Computer Programming Vol. 3 (Sorting and Searching)*, Addison-Wesley (1973).

(平成6年 8 月9日受付)
(平成6年12月5日採録)



福島　俊一（正会員）

1958年生．1982年東京大学理学部物理学科卒業．同年日本電気（株）入社．現在，情報メディア研究所音声言語研究部主任．本学会平成4年度論文賞，第45回全国大会奨励賞を受賞．自然言語処理・情報検索の研究分野において，日本語解析マシン，校正支援システム，日本語入力手法，全文検索システムなどの研究開発に従事．人工知能学会，言語処理学会，計量国語学会，情報知識学会，ACL各会員．