自然言語処理 111－3
（1996. 1. 19）

文法および辞書評価支援のための言語解析システム

中山 拓也, 松本 裕治
奈良先端科学技術大学院大学 情報科学研究科

自然言語処理における曖昧性解消の手段として、シソーラスや格フレームのような語彙的知識(辞書)を用いる方法や、確率文法のような文法的情報を用いる方法が従来より提案されてきた。これらの言語情報は、単独では全ての曖昧性を解消できないため、実用的には統合的に用いることが必要になる。本稿では、さまざまな言語情報を統合的に利用した時、全体としての曖昧性解消への有効性という観点から言語情報を評価支援するための言語解析システムについて述べる。本システムは、基本的には曖昧性を解消しながら解析するパーサであり、曖昧性が如何に解消されるかをユーザに示すことで評価を支援する。また、評価支援の他に、本システムを言語情報の獲得のために利用することも考えている。

Language Analysis System for Evaluating Grammars and Dictionaries

Takuya Nakayama and Yuji Matsumoto
Graduate School of Information Science, Nara Institute of Science and Technology

Various types of linguistic knowledge, such as thesauri and case frames, and various types of grammatical knowledge, such as statistical grammars, have been studied and accumulated. Each of them can resolve some kinds of ambiguities included in a sentence, but cannot resolve all of them alone. Though they should be used in some integrated way, they have not been evaluated in such a way. This paper introduces a framework for evaluating various types of linguistic knowledge on a single platform. The system provides the user with an interface for observing the dynamic behavior of the parsing process.

1 はじめに

自然言語には多くの曖昧性が含まれる。計算機によって自然言語を解析する際に重要なことは、それらの曖昧性を効果的に解消する為の枠組を知ることである。

語義の曖昧性や構文的係り受けの曖昧性の解消方法の一つとして、シソーラスや格フレームのような語彙的知識(辞書)を用いる方法がある。これまでに、このような曖昧性解消に利用可能な語彙的知識を構築しようとする研究が数多くなされてきた。また、語彙的情報を用いる方法以外に、確率文法や文法的なヒューリスティクスのように、文法的情報を用いる方法も研究されてきた。これらの言語情報は単独では多様な曖昧性を完全に解消できないので、実際に利用する際には、複数の言語情報を統合的に用いて曖昧性解消を行なうことになる。

言語情報を手作業で構築したにせよ、コーパスなどから自動的に獲得したにせよ、作成した言語情報を評価することは重要なことである。我々は、さまざまな言語知識を、実際に曖昧性の解消に利用するという観点から評価する為のプラットフォームの作成を目指している。勿論、これまでの研究の中でも、曖昧性解消に用いることを目的とし、構築した言語情報をそのような観点から評価しているものがある。しかしそれらの評価は、それぞれの言語情報ごとに、ある言語現象に関して、個別に行なわれてきた。つまり、他の言語情報を同時に利用することを考え、それら相互の影響も考慮した評価はなされてこなかった。しかし、実際に統合的な利用を考えたとき、それぞれの言語情報を個別に評価するのではなく、さまざまな言語情報を全体として評価する必要がある。

本稿では、さまざまな言語情報を統合的に利用した

とき、全体としての曖昧性解消に関する有効性という観点から言語情報を評価支援するための言語解析システムについて述べる。この言語解析システムは、基本的には曖昧性を漸次的に解消しながら解析するパーサである。関連した研究としては [3] の KGW+p がある。KGW+p では言語情報を用いた解析過程の制御に重点を置いているが、我々のシステムでは、さまざまな言語情報を用いた時に、どれだけの曖昧性が解消されるかを調べることを主な目的としており、そのような調査を簡単に行なうためのユーザインタフェース部分に重点を置いている。

2 システム

2.1 目的

前述のように、言語情報を曖昧性解消の観点から評価するには、個別の評価ではなく、統合的に利用した際の全体としての評価が重要である。そこで、統合的評価を行なうために、次のような言語解析システムの作成を目的とした。

- さまざまな曖昧性 (語義の曖昧性、係り受けの曖昧性など) を、複数の言語情報を統合的に用いて解消する。
- 文全体としてだけでなく、その解析途中についても、どれだけの曖昧性が解消できたかを示す。
- 言語情報を改良するために利用できるデータを提供する。

我々は、このシステムを言語情報の評価支援だけでなく、逆に、コーパスからの言語情報獲得を支援することに利用することも考えている。

2.2 システム構成

システムは二つの部分から成る。一つは SAX パーサ [7] を拡張した言語解析器であり、Prolog と Perl を用いて実装されている。もう一つは、解析の最終的結果や途中経過をインタラクティブに表示するためのビジュアルインタフェース部分であり、Tcl/Tk を用いて実装されている (本稿では、この部分については説明しない)。現在のところ、対象言語としては日本語を考えており、文の形態素解析には JUMAN [8] を利用している。その他の言語の場合でも、わずかな変更を加えるだけで扱うことが出来る。

システムの概観を図 1 に示す。言語解析器は大きく分けて三つのモジュールから構成されている。

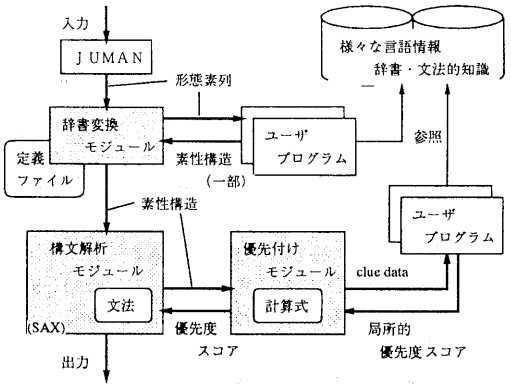

図 1: システムの概観 (解析器部分)

[A] 辞書変換モジュール

このモジュールは、さまざまな語彙的情報を、文解析の際に用いる素性構造の形式に変換するためのものである。現在のシステムでは文法的/意味的な情報を表現するために素性構造表現を用いている。

[B] 構文解析モジュール

このモジュールでは、文法と辞書変換モジュールで生成された素性構造を使って、入力文の構文解析する。結果として、入力文に対する解析木と、各部分木に対する素性構造が生成される。文法としては、DCG 形式で書かれた文法を扱うことができる。現在のところ、文法としては、HPSG[2] スタイルの単一化文法を用いることを考えている。

[C] 優先付けモジュール

このモジュールは、文解析の途中で何らかの曖昧性が生じた時に、構文解析モジュールから呼び出される。ここでは、曖昧性として認識されたそれぞれの解釈に対して、優先度スコアを計算する。優先度スコアの計算方法はユーザが設定をおこなう。この優先度スコアを用いることによって、システムは逐次的に曖昧性を解消する。具体的には、ある閾値を設定しておくことで、その閾値よりも優先度スコアの低い解析を切り捨てることによって行なう。

2.3 処理の流れ

言語解析器の処理の流れは次のようになる。

1. 文が入力されると、前処理として JUMAN で形態素解析を行ない、形態素列を得る。

2. 各形態素に対して、何らかの語彙情報を用いて素性構造を作成する (辞書変換モジュール)。このとき作成する素性構造は一つとは限らず、多義語の場合はそれに応じて複数の素性構造を作成する。すなわち、語に関する全ての曖昧性は、素性構造の集合という形で表現される。

3. 入力文を、bottom-up Chart 法によって解析し、各部分木ごとに言語的/意味的な情報を含む素性構造を生成する (構文解析モジュール)。

4. ある部分木の素性構造に曖昧性が発見された場合には、優先付けモジュールが呼び出される。ここでも、素性構造の曖昧性は、素性構造の集合という形であらわされる。

5. 優先付けモジュールでは、それぞれの素性構造に対して優先度スコアを計算し、それらを比較する。閾値よりも小さいスコアを持つものを切り捨て、残った素性構造を用いて解析を続ける。なお、閾値は部分木ごとの局所的な最大値に対して、相対的に決められる。

全ての曖昧性は、素性構造の集合という形で表現されるため、生成された素性構造の個数を数えることで、利用した言語情報の曖昧性解消に関しての有効性を簡単に調べることができる。言語情報を解析器へ組み込む方法については、以降で述べる。

2.4 素性構造への変換

辞書変換モジュールでは、文法的/意味的な情報を含めた素性構造を、二つの素性構造を単一化することによって生成する。一つは各品詞、各活用ごとに共通の情報を表したもので、主に文法的な、利用する語彙的情報に依存しない部分を定義する。もう一つは各単語ごとに固有の情報を表したもので、利用したい語彙的情報に依存した部分を定義する。前者は定義ファイルによって静的に、後者はユーザプログラムによって動的に与える。

ユーザは、利用したい語彙的情報ごとに、それを用いて決定できる部分的な情報を、定められた素性構造の形式で出力するプログラムを作成することによって、さまざまな語彙的情報をシステムで利用することができる。

複数の語彙的情報を同時に利用するために、ユーザプログラムは複数登録可能でなければならない。その

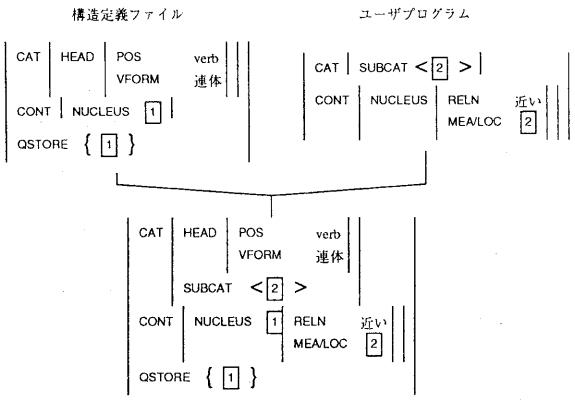

図 2: 素性構造生成の例

際、各ユーザプログラムからの情報を、如何に統合して素性構造を作成するかという問題が生じる。一例として、辞書 1 では「掛ける」の意味を二つ (例えば、「ハンガーに服を掛ける」「車にお金を掛ける」の二種類の意味) に区別しているが、辞書 2 ではそのような区別をしていない場合を考える。この場合、辞書 1 を用いたユーザプログラムは、例えば、次のような素性構造を出力することになる。

$$\left[\text{CONT}\quad \left[\text{RELN 掛ける}_1\right]\right] \vee \left[\text{CONT}\left[\text{RELN 掛ける}_2\right]\right]$$

ただし、記号∨ は選言演算子を表す (システム内では集合的に扱う)。他方、辞書 2 を用いたユーザプログラムからは、例えば、次のような素性構造を得る。

$$\left[\text{CONT}\quad \left[\text{RELN 掛ける}\right]\right]$$

これらの、最も望ましい統合方法は、以下のように素性構造の単一化によって扱う方法である。

$$([\text{CONT}\ [\text{RELN 掛ける}_1]] \vee [\text{CONT}\ [\text{RELN 掛ける}_2]]) \wedge [\text{CONT}\ [\text{RELN 掛ける}]]$$

$$\downarrow$$

$$([\text{CONT}\ [\text{RELN 掛ける}_1]] \wedge [\text{CONT}\ [\text{RELN 掛ける}]]) \vee ([\text{CONT}\ [\text{RELN 掛ける}_2]] \wedge [\text{CONT}\ [\text{RELN 掛ける}]])$$

$$\downarrow$$

$$\left[\text{CONT}\left[\text{RELN 掛ける}_1\right]\right] \vee \left[\text{CONT}\left[\text{RELN 掛ける}_2\right]\right]$$

このようにするためには、「掛ける$_1$」と「掛ける$_2$」が「掛ける」の下位クラスの概念であるという情報が

分かっていなければならない。しかしながら、そのような情報をシステムに与えるためには、利用する全ての語彙情報について調べ、それらの全ての項目についての階層関係をユーザが定義しなければならない。したがって、そのような情報をシステムが仮定するのは現実的でなく、そのような情報を使わずに複数の語彙情報を統合する枠組が必要である。

複数の語彙情報を、それぞれの項目間の関係を用いずに統合的に扱うための枠組として、我々のシステムでは、文献[1]で提案されている composite conjunction を導入し、素性構造単一化アルゴリズムを拡張している (appendix 参照)。本来 composite conjunction は等位接続表現を取り扱うために提案されたものであるが、ここでは、複数の語彙情報の統合を容易にするために利用している。

システムは composite conjunction を用いることによって、自動的に次のような混合素性構造 (composite feature structure) を生成する。

$$\left(\left[\text{CONT}\left[\text{RELN 掛ける}_1\right]\right]\text{A}\left[\text{CONT}\left[\text{RELN 掛ける}\right]\right]\right)$$
$$\text{V}\left(\left[\text{CONT}\left[\text{RELN 掛ける}_2\right]\right]\text{A}\left[\text{CONT}\left[\text{RELN 掛ける}\right]\right]\right)$$

ただし、記号A は composite conjunction 演算子を表す。このような構造を使うことによって、掛ける$_1$ (あるいは 掛ける$_2$) と 掛ける の情報を同時に利用した解析ができる。すなわち、辞書 1 と辞書 2 の情報を統合して扱うことができる。混合素性構造は、作成する際に階層構造のような特別な知識を必要とせず、また、混合素性構造のために拡張された単一化アルゴリズムを用いることで、文法などには変更を加えることなく扱うことができる。この様にすることで、ユーザは、自分が利用したいもの以外の辞書については知識が無くても、それらを同時に用いた解析を試すことができる。

2.5 優先度スコア付け

優先度スコアを用いた曖昧性解消は、優先付けモジュールにユーザプログラムを登録することにより実現できる。優先付けモジュールは、優先度スコアを計算する際に、素性構造から単純な形式のデータ (clue data) を作り出し、その clue data を登録されたユーザプログラムに伝達する。ユーザプログラムは clue data を引数とし、それに対して (局所的な) スコアを返す関数の役割をする。

例えば、「お金をかける」を解析してできる素性構造 (図3 参照) の場合では、「かける」の語義の曖昧性を解消する為に利用できる情報としては、(1)「かける」がどの意味で用いられることが多いかという確率的な情報や、(2)「お金」をヲ格に取りやすいのはどの場合かといった情報が考えられる。これらそれぞれの手掛かりごとに必要な情報を、最小限に含む単位的なデータが clue data である。(2) の場合、「お金」「ヲ格」と「かける」の意味の一つが与えられると、その意味についての優先度を計算できるので、それら三項からなる形式の clue data があれば良いことになる。

なお、ある素性構造から作り出された clue data は、全体として、素性構造内に含まれる優先度スコア計算に必要な手がかりを全て含むものとする。つまり、全ての clue data に対して優先度の評価を行なって、それらを一つにまとめることによって素性構造全体の優先度スコアが計算できるようにしている。

このように、優先度スコア計算の基本単位として clue data を用いるのには二つの理由がある。一つは、ユーザプログラムでの処理を軽減することで、ユーザの負担を減らすためであり、もう一つは、システムを言語情報の獲得のために利用するためである (3節参照)。現在のところ、clue data の種類として考えている形式は、以下の三つがある。ただし、これで十分かどうかについては、今後検討する余地がある。

Form-1: $(term)$[1] ···(単項引数形式)
この形式は、個々の単語に関する確率的な優先付けを扱うためのものである。例えば、単語のコーパス内における生起確率をもとに優先度を決めたいような場合、この形式の clue data に対して局所的優先度スコアを計算するユーザプログラムを作成すれば良い。

Form-2: $(term_1)/relation/(term_2)$
···(三項引数形式)
この形式は、$term_1$ と $term_2$ の関係 ($relation$) を評価するためのものである。 例えば、(お金)/ヲ/(掛ける) を「お金」と「掛ける」のヲ格における共起確率を使って評価する場合などに用いる。

Form-3: $\alpha \rightarrow \beta$ ···(句構造規則形式)
この形式は、句構造規則の優先度を評価するためのものである。例えば、確率文法に用いられる確率値を優先度スコアの中に組み込むときに利用する。

呼び出されたユーザプログラム側では、それぞれの clue data に対して、何らかの言語情報を用いて局所的な優先度スコアを計算する。その後、ユーザが自由

[1] $term$ に何が入るかは、素性構造をどのように定義するかによる。$term$ の部分には混合素性構造が入ることがあるので、ユーザプログラムを書く際には注意が必要である。

に定義できる計算式[2] を用いることで、これらの clue data ごとに求めた局所的優先度スコアから、一つの素性構造全体としての優先度スコアを計算する。曖昧性解消は、この優先度スコアを利用して行なわれる。

また、ユーザプログラムは複数登録が可能である。各言語情報を用いてスコアをそれぞれ計算するプログラムを作成し、それらのスコアを統合するための計算式をユーザが設定することで、さまざまな言語情報を統合的に用いた曖昧性解消を試みることができる。

図 3 に優先度スコア計算の例を示す。ユーザプログラムは、それが参照している言語情報で clue data を評価できない場合、スコアを出力しないように規定されている。その場合は、あらかじめ設定しておいたデフォルトスコアを局所的優先度スコアとして用いる。

素性構造	clue data	ユーザプログラムからのスコア		優先度スコア
		prog1	prog2	
[賭ける ガ _ ヲ お金]	お金	1.0	--	0.2
	賭ける	0.2	--	
	___ ガ 賭ける	--	--	
	お金 ヲ 賭ける	--	1.0	
[掛ける ガ _ ヲ お金]	お金	1.0	--	0.15
	掛ける	0.5	--	
	___ ガ 掛ける	--	--	
	お金 ヲ 掛ける	--	0.3	
default score		1.0	1.0	

$$\text{計算式} = \prod_{x \in \{clue\ data\}} (score_1(x) \times score_2(x))$$

図 3: 「お金をかける (賭ける/掛ける)」スコア付け例

2.6 ユーザの負担

さまざまな言語情報を本システムに組み込むために、ユーザがしなければならないことをまとめると次のようになる。

(1) 利用したい語彙情報を素性構造の形式に変換するプログラムの作成と、品詞/活用形ごとに共有な素性構造の定義。

(2) 言語情報を用いて、clue data に対する局所的優先度スコアを計算するユーザプログラムの作成と、そのデフォルトスコアの設定。

(3) 局所的優先度スコアをまとめ、素性構造全体としての優先度スコアを計算するための計算式の定義。

(4) 優先度スコアを計算するために必要な情報を含む素性構造を、各部分木に対して作成しながら文を解析するための文法の作成。

実際には、(1)-(4) 全てをする必要は無く、必要なものについてのみ変更を加えれば良いので、比較的容易に、様々な言語情報を用いることができる。

3 言語知識獲得への利用

本システムは、様々な言語情報の評価支援を基本的な目的としたものであるが、言語知識獲得に有用なデータを、コーパスから抽出する為にも利用可能である。

例えば、「お金をかける」の「かける」の解釈として、「掛ける」、「賭ける」、「架ける」、「駆ける」などの中から「賭ける」と「掛ける」があり得る解釈だと分かれば、次の不等式に相当するデータを得ることができる (ただし、clue data として Form-2 のみを考えた場合)。

$$score(\text{お金/ヲ/掛ける}) > score(\text{お金/ヲ/架ける})$$
$$score(\text{お金/ヲ/賭ける}) > score(\text{お金/ヲ/架ける})$$
$$score(\text{お金/ヲ/掛ける}) > score(\text{お金/ヲ/駆ける})$$
$$\vdots$$

重要な点は、このデータは「お金をかける」の「かける」の曖昧性を正しく解消するために必要十分な情報を持っているということであり、その性質は他の文を解析してデータを増やしても変わらない。つまり、このデータを用いて何らかの言語知識を構築した場合、逆にその言語知識を用いて正しく曖昧性を解消できなければ、その知識の構築方法に不備があるか、その利用方法に問題があることが指摘できる。この点において、従来から行なわれてきた統計的手法を用いたコーパスからの学習とは異なる。統計的手法が「統計的によく現れるものが、より尤もらしい解釈になる」という仮定に基づいているのに対し、ここで得られるデータはそのような仮定なしに利用することができる。

実際には複数の型の clue data が存在するので、得られるデータはもう少し複雑で、

$$score(\{\text{お金}, \text{掛ける}, \text{お金/ヲ/掛ける}, (s \to p, v), (p \to n, p)\})$$
$$> score(\{\text{お金}, \text{架ける}, \text{お金/ヲ/架ける}, (s \to p, v), (p \to n, p)\})$$

[2] 計算式には Perl で扱うことのできる演算子/関数を含めることができる。

のようなものになる。この場合でも、スコアの計算式を適当なものに固定すれば、パラメータ学習アルゴリズムを用いて、各 clue data に対する最適な局所的優先度スコアを得ることができる。こうして得たデータは、さまざまな種類のデータ間の関係を情報として含むため、他種の言語知識と矛盾しないような言語知識を構築することができると思われる。

問題点としては、clue data、それを作り出すための素性構造記述、スコア付けによる曖昧性解消手法、文法などのシステム部分が不十分な場合、本システムによって得られるこれらのデータに、矛盾したデータが含まれる可能性がある点が挙げられる。しかしながら逆に言えば、矛盾点を調べることで、システムの不備の改良に役立つ情報が得られる。

4 問題点と今後の課題

本システムの問題点としては次のようなものがある。

(1) 評価に利用できる例文は、文法が正しく解析できるものに限られる。したがって、より広範囲に評価実験を行なうためには、文法の coverage をより広げる必要がある。

(2) 曖昧性解消に対する有効性を評価するためには、例文の解析結果の正解 (可能解) を知っておく必要がある。

現在、HPSG を基に記述した小規模な日本語文法と、IPAL 辞書 [5][6] と分類語彙表 [4] を利用したスコア付けユーザプログラムを用いて、システムのテストをしている。その過程で文法を改良して行くことにより、(1) の問題の改善を試みている。

(2) の問題点については、同じ例文でも、使用する言語情報によって正解に含まれる情報が異なるので、あらかじめ正解付きコーパスを作成するというよりは、ユーザが正解を容易にシステムに与えられるようなインタフェースの整備に重点を置いている。

今後の予定としては、ある程度の規模の文法ができた時点で、その他のさまざまな言語情報を取り込んだ評価実験を行なう。また、3節で述べた言語知識獲得の為のデータについて、統計的データとの比較実験を行なうことで、その有効性を調べてみる予定である。

5 おわりに

本稿では、さまざまな言語情報を統合的に用いて曖昧性を解消しながら文を解析するシステムについて述べた。このシステムは、どれだけの曖昧性が解消できるかという観点から言語情報やその利用方法を評価することを目的としている。特に、複数の言語情報を統合的に用いることができるので、それぞれの言語情報の整合性も含めた評価が可能である。

また、既存の言語情報の評価だけでなく、言語情報獲得の目的にも、本システムが利用できる可能性について述べた。

A 混合素性構造の単一化

$$\left(\begin{bmatrix}\mathrm{A}: & \begin{bmatrix}\mathrm{B}: & c_1\end{bmatrix} \\ \mathrm{D}: & e_1\end{bmatrix} \wedge \begin{bmatrix}\mathrm{A}: & \begin{bmatrix}\mathrm{B}: & c_2\end{bmatrix} \\ \mathrm{D}: & e_2\end{bmatrix}\right) \wedge \begin{bmatrix}\mathrm{A}: & \boxed{1} \\ \mathrm{D}: & \boxed{2} \\ \mathrm{F}: & \boxed{3}\end{bmatrix}$$

$$\Downarrow$$

$$\begin{bmatrix}\mathrm{A}: & \begin{bmatrix}\mathrm{B}: & c_1\end{bmatrix} \\ \mathrm{D}: & e_1 \\ \mathrm{F}: & \boxed{3}\end{bmatrix} \wedge \begin{bmatrix}\mathrm{A}: & \begin{bmatrix}\mathrm{B}: & c_2\end{bmatrix} \\ \mathrm{D}: & e_2 \\ \mathrm{F}: & \boxed{3}\end{bmatrix}$$

$$\boxed{1} = \begin{bmatrix}\mathrm{B}: & c_1\end{bmatrix} \wedge \begin{bmatrix}\mathrm{B}: & c_2\end{bmatrix}$$

$$\boxed{2} = e_1 \wedge e_2$$

参考文献

[1] Richard P. Cooper. Coordination in unification-based grammars. In *Proceedings of the 5th Conference of the European Chapter of the ACL*, pp. 167–172. ACL, April 1991.

[2] Carl Pollard and Ivan A. Sag. *Head-Driven Phrase Structure Grammar*. The University of Chicago Press, 1994.

[3] J. Tsujii, Y. Muto, Y. Ikeda, and M. Nagao. How to get preferred readings in natural language analysis. In *COLING-88*, Vol. 2, pp. 683–687, 1988.

[4] 国立国語研究所. 分類語彙表. 国立国語研究所資料集 6. 秀英出版, 1993.

[5] 情報処理振興事業協会技術センター. 計算機用日本語基本動詞辞書 IPAL(Basic Verbs) -解説編-. 情報処理振興事業協会, 1987.

[6] 情報処理振興事業協会技術センター. 計算機用日本語基本形容詞辞書 IPAL(Basic Adjectives) -解説編-. 情報処理振興事業協会, 1990.

[7] 松本裕治, 伝康晴, 宇津呂武仁. 構文解析システム SAX 使用説明書 version 2.1. 奈良先端科学技術大学院大学 松本研究室, 1994.

[8] 松本裕治, 黒橋禎夫, 宇津呂武仁, 妙木裕, 長尾真. 日本語形態素解析システム JUMAN 使用説明書 version 2.0, 1994.