論文96-33B-5-15

일한 기계번역을 위한 On-Line 형태소 해석기 설계

(Design of an On-Line Morphological Analyzer for a Japanese-to-Korean Translation System)

姜 哲 壎 *, 崔 炳 旭 **

(Seok Hoon Kang and Byung Uk Choi)

요 약

본 논문에서는 On-line 방식의 일본어 형태소 해석기를 제안한다. 일본어는 문장의 중간에 공백이 없기 때문에 문장의 입력이 완전히 종료될 때까지 모호성이 누증되는 특징이 있다. 이 문제의 해결을 위해 변형된 차트 방식의 형태소 해석 앨고리듬을 제안한다. 또한 형태소 해석에서 사전 검색횟수는 중요한 문제이다. 일반적인 On-line 방식의 해석에서 N개의 문자로 이루어진 일본어 문장은 N(N+1)/2번의 사전 검색이 논리적으로 필요하며 이것은 최장일치에 기반한 Off-Line방식의 거의 두 배에 달하는 수치이다. 본 논문에서는 이를 위한 사전 형식의 변형으로 이 문제를 해결하고자 한다. 실험 결과 Off-line 방식에 접근한 해석 결과를 얻을 수 있고, 문장이 길어질 수록 해석 효율이 좋아짐을 알 수 있다.

Abstract

In this paper, an algorithm for On-Line rightward Japanese parsing is proposed. The ambiguity in On-Line parsing is accumulated until the input is completely finished, since there is not a space between words in the Japanese sentence. Thus the algorithm for morphological analysis, based on modified chart, is used in solving it. And the number of searching a word in dictionary for morphological analysis is also a puzzling problem. The Japanese sentence, consist of N characters, has logically its maximum number of N(N+1)/2 searches in the ordinary On-Line Analysis, which is nearly twice as many as normal Off-Line. In this paper, the matter is settled through the modification of dictionary format. In experiment, we can accomplish the rate of analysis which is nearly equal to that of Off-line parsing. And it becomes clear that the longer a sentence is, the better an analysis efficiency is.

Ⅰ. 서 론

기계번역의 연구는 문어체에 관해서 활발히 진행되어 왔으며 최근에는 구어체, 특히 대화체 언어에 대한 연구가 시도되고 있다.[11,13,14] 또한 컴퓨터 네트워 환경의 구축과 관련 기술의 발달로 이를 고려한 기계번역 시스템에 대한 연구와 개발도 필요한 실정이다. 이에 대한 연구로는, 대화체 문장의 번역뿐만 아니라 문장단위가 아닌 구(Phrase)단위의 번역, 그리고 실시간 입력에 대한 입력 처리기의 설계 등이 있을 수 있다. 컴퓨터 네트웍 상에서의 기계번역 뿐 아니라, 문서(혹은 FILE 형태)가 아닌 외부 입력장치에 의한 직접 입력을 이용한 기계 번역에서도 이의 특성을 고려한 입력기의 설계는 중요하다고 할 수 있겠다. 특히 다중 처리기 혹은 병렬처리가 가능한 시스템을 이용하는 기계

* 正會員, 東西大學校 컴퓨터工學科
(Dept. of Computer Engineering, Dongseo University)
** 正會員, 漢陽大學校 電子通信工學科
(Dept. of Electronic Communication Engineering, Hanyang University)
接受日字:1994年8月27日, 수정완료일:1996年4月13日

번역에서는 이런 입력 특성을 포함하는 언어 해석기의 설계가 필요하다고 할 수 있다.[1,5,6,12,17]

일한 기계 번역 시스템의 개발은 대상 언어의 당사자인 한일 양국에서 모두 개발할 수 있겠으나 최종 생성문의 대상이 한국어이므로 한국어를 위주로 개발하는 것이 보다 타당할 것이라고 판단된다. 이러한 관점에서 살펴보면, 한국어 생성을 전제로 일본어를 해석하는 것이 현명할 것이며, 이런 점이 일부에서 고려되고 있다.[20]

본 논문에서는 실시간 입력을 고려하여, 입력의 방향과 동일한 주사방향을 가지며 입력과 동시에 언어 해석을 진행할 수 있는 것을 On-Line 형태소 해석기라고 정의하고, 이 때의 처리 효율을 높이기 위한 형태소 해석 방법을 제안한다. 기존의 해석 방식은 문장의 완전한 입력이 이루어진 후에 해석을 개시하며 해석의 방식은 대부분 최장일치법에 근간을 두고 있는 것이 일반적이므로[2,3,4,9,10,13,16,18], 문장의 입력완료시까지 대기시간을 가지게 된다. 따라서 실시간 입력을 고려한다면 이는 시간적인 낭비를 초래할 수 있으며, 본 논문에서는 이를 해결하기 위해 문장의 입력방향을 해석 개시 방향과 동일하게 설정하여 입력완료까지의 대기시간을 없앨 수 있도록 하였다. 따라서 해석의 방향은 문장의 입력 방향과 동일한 방향인, 좌에서 우측으로 설정하여 해석목을 구성하며, 각 입력문자를 단위로 변형된 차트를 만들고, 이것을 단위로 규칙을 적용하여 해석목을 확장해 나가게 한다. 특히 사전처리의 효율 향상을 위해 개선된 사전 형식을 제안하며, 각각의 해석 단계가 상호 독립적으로 작용가능하게 하여, 다중처리기를 포함한 시스템에서는 보다 나은 해석 효율을 기대할 수 있을 것으로 판단된다. 아울러 기존의 논문들이 제시한 바 있는 Off-Line 방식의 형태소 해석에 의한 일본어 형태소 해석 방법과 해석 결과 및 사전 검색 효율, 그리고 형태소 해석 결과를 비교하여 본 논문이 제안하는 방식의 연구에 대한 타당성을 입증한다.

Ⅱ. 일본어 On-Line 해석기의 설계

1. On-Line 방식 형태소 해석의 문제점

On-Line 형태소 해석은 문장의 입력과 같은 방향으로 해석을 시도하는 것으로, 입력 중에 정보를 파악해 나갈 수도 있다는 장점이 있으나, 입력이 종료되지 않은 상태에서 해석을 개시하므로, 사전검색을 위한 단어의 수가 입력이 진행됨에 따라 누증되고, 해석목 생성의 가능성이 지수적으로 증가하는 등, 처리에 많은 어려움이 따르기도 한다. 특히 본 논문에서 입력대상으로 설정한 일본어의 경우, 문장의 중간에 공백이 발생하지 않으므로 단어의 구분이 없어지게 되고, 따라서 문장의 길이가 길어질수록 해석의 모호성은 더욱 증가하게 되며, 이 경우 형태소 해석 사전에 대한 검색 시도 횟수가 커다란 문제점으로 대두될 수 있다. 실제로 "めのまえ(*menomae*)"와 같은 단어는 "め+の+まえ" 혹은 "めのまえ"의 2가지로 최종 해석될 수 있고,[7,9,10,15] 시스템이 최단일치나 최장일치법을 따른다면 일반적으로 하나의 해석결과 밖에 얻을 수 없으므로 해석결과를 신뢰할 수 없게 된다.[7,9,10,12]

또한 "おしひろめる(*osihiromeru*)"라는 일본어 동사의 형태소 해석을 예로 들어보면, 하나의 단어를 여러 개의 형태소 집합으로 분리할 수 있으므로 형태소 해석용 사전에 의해 형태소만을 분리시킨다면 마치 여러 개의 단어가 결합된 것으로도 해석 가능하다.[9,10] 하나의 단어 수준이 아니라 문장 전체를 예상하고 일본어 문장 형식 자체의 모호성을 고려한다면, 해석 결과의 갯수는 기계 번역 시스템의 처리 과정에서 가장 난점이 될 수도 있을 것이다. 그리고 형태소 해석은 단순한 문자열의 분리 과정으로만 이루어지는 것은 아니므로 이를 고려한 형태소 해석기의 설계가 중요하다고 할 것이다. 앞의 일본어 동사 "おしひろめる"의 경우, 각각의 단어로 형태소 분리는 가능하지만 그 결과가 비문법적이면 이후에 진행되는 해석 결과 중 이것을 포함하는 전체 문장의 해석은 실패할 것이 분명하다. 즉, "ひろ"가 단위로만 사용되는 형태소이므로 "おし+ひろ"로 분석된 결과는, 이후에 어떤 분석이 이루어져도 올바른 해석이 될 수는 없는 것이다.

2. 우방향성 처리

On-Line 방식은 입력된 문장의 첫 번째 요소(첫 번째 문자)부터 차례로 처리하는 것을 가정한 것이므로, 효율적인 처리를 위해서는 Off-Line과는 반대인 우방향성 처리가 필요하다. 즉 일반적인 최장일치가 취하는 우에서 좌로의 주사(scan)가 아닌 좌에서 우측으로의 주사를 의미하는 것이다. 이 방법을 이용하면 프로세서의 할당 혹은 병렬처리로 문장이 입력되고 있는 동안에도 계속적으로 형태소의 분리와 결합작업을 수행할 수도 있으며 입력이 완료되는 시점에서 최종 분석 결

과를 생성해 낼 수 있는 시스템의 구현이 가능하다. On-Line 방식의 해석에서 입력 문자열에 대한 분석은, 여러 개의 가능한 결과를 만들어 내며 입력이 계속 진행되는 동안에는 어느 것 하나도 해석결과에 대한 우선권을 부여받을 수 없다. 예문 "おしひろめる"의 경우, On-Line 방식의 처리에 대한 입력은 최초상태가 "お"이며 "おしひろめる"라는 전체 단어가 최종상태가 된다. 문자와 문자의 사이가 시스템 처리의 입장에서 볼 때 시간적 지연이 발생할 수 있는 곳이며, 실제 형태소 분리를 위한 사전 검색과 분리 또는 결합에 대한 판단이 이루어져야 하는 곳이다.

3. 형태소 해석 사전

일본어의 특성과 본 논문의 대상이 On-Line 방식의 처리라는 것을 감안할 때, 일본어 해석에서 가장 중요한 부분 중의 하나가 형태소 해석 사전이다. 한국어나 영어가 하나의 기준 단어(공백에 의해 구분되는 하나의 구)에 대해 보통 1 ~ 3개 정도의 형태소를 가지며 다음의 기준 단어는 적어도 형태소 정보에 관해서는 이전 단어와의 연관성이 극히 작다고 할 때, 일본어의 경우에는 이와 반대의 현상이 있다. 즉, 하나의 형태소를 해석 사전에 의해 판별하였다고 하더라도 이후의 결합양상에 의해 단순한 검색 결과는 무시될 수도 있는 것

```
/* 0 */  "う", "", "う", "わ", "お", "", "っ", "い", "", "え", "え", "",
/* 1 */  "く", "", "く", "か", "こ", "", "い", "き", "", "け", "け", "",
/* 2 */  "ぐ", "", "ぐ", "が", "ご", "", "い", "ぎ", "", "げ", "げ", "",
/* 3 */  "す", "", "す", "さ", "そ", "", "し", "", "", "せ", "せ", "",
/* 4 */  "つ", "", "つ", "た", "と", "", "っ", "ち", "", "て", "て", "",
/* 5 */  "ぬ", "", "ぬ", "な", "の", "", "ん", "に", "", "ね", "ね", "",
/* 6 */  "ぶ", "", "ぶ", "ば", "ぼ", "", "ん", "び", "", "べ", "べ", "",
/* 7 */  "む", "", "む", "ま", "も", "", "ん", "み", "", "め", "め", "",
/* 8 */  "る", "", "る", "ら", "ろ", "", "っ", "り", "", "れ", "れ", "",
/* 9 */  "る", "", "る", "*", "*", "", "*", "*", "", "れ", "ろ", "よ",
/* 10 */ "る", "", "る", "*", "*", "", "*", "*", "", "れ", "ろ", "よ",
/* 11 */ "する", "す", "する", "せ", "し", "", "し", "", "", "すれ", "しろ", "しよ",
/* 12 */ "ずる", "ず", "ずる", "ぜ", "じ", "", "じ", "", "", "ずれ", "じろ", "じよ",
/* 13 */ "くる", "く", "くる", "こ", "", "", "き", "", "", "くれ", "い", "",
/* 14 */ "する", "す", "する", "さ", "し", "", "し", "し", "", "すれ", "しろ", "しよ",
/* 15 */ "い", "*", "い", "かろ", "", "", "かっ", "く", "う", "けれ", "", "",
/* 16 */ "だ", "*", "な", "だろ", "", "", "だっ", "な", "に", "なら", "", "",
/* 17 */ "い", "*", "い", "く", "から", "", "", "く", "かり", "けれ", "かれ", "",
/* 18 */ "だ", "*", "な", "なら", "", "", "", "に", "なり", "なれ", "なれ", "",
/* 19 */ "る", "", "る", "ら", "ろ", "", "っ", "り", "", "れ", "", "",
/* 20 */ "る", "", "る", "*", "", "", "", "*", "", "れ", "ろ", "よ",
/* 21 */ "る", "", "る", "*", "", "", "", "*", "", "れ", "ろ", "よ",
/* 22 */ "る", "", "る", "*", "", "", "", "*", "", "れ", "ろ", "よ",
/* 23 */ "る", "", "る", "*", "", "", "", "*", "", "れ", "ろ", "よ",
/* 24 */ "る", "", "る", "*", "", "", "", "*", "", "れ", "ろ", "よ",
/* 25 */ "す", "", "す", "", "しょう", "", "し", "", "", "", "", "",
/* 26 */ "*", "", "な", "", "ろ", "", "っ", "な", "", "なら", "", "",
/* 27 */ "す", "", "す", "せ", "しょう", "", "し", "", "", "すれ", "", "",
/* 28 */ "*", "", "*", "", "ろ", "", "", "", "", "ら", "", "",
/* 29 */ "い", "", "い", "", "かろ", "", "かっ", "く", "", "けれ", "", "",
/* 30 */ "い", "", "い", "", "かろ", "", "かっ", "く", "", "けれ", "", "",
/* 31 */ "だ", "", "な", "", "", "", "", "", "", "", "", "",
/* 32 */ "だ", "", "な", "も", "だろ", "", "だっ", "な", "に", "なら", "", "",
/* 33 */ "だ", "", "な", "", "だろ", "", "だっ", "な", "に", "なら", "", "",
/* 34 */ "だ", "", "な", "", "だろ", "", "だっ", "な", "に", "なら", "", "",
/* 35 */ "い", "", "い", "", "", "", "かっ", "く", "", "けれ", "", "",
/* 36 */ "り", "", "る", "ら", "", "", "り", "", "", "れ", "", "",
/* 37 */ "り", "", "る", "ら", "", "", "り", "", "", "れ", "", "",
/* 38 */ "", "", "る", "ら", "", "", "り", "", "", "れ", "", "",
/* 39 */ "し", "", "き", "く", "から", "", "く", "かり", "", "けれ", "", ""
```

그림 1. 용언의 어미 활용 테이블
Fig. 1. A conjugation table for a declinable word.

이다. 일본어 동사 "おしひろめる"에서 "ひろ(hiro)"라는 단어는 명사에 대한 형태소 정보를 가지지만, 문법적으로 단위(UNIT)의 형태로 사용되는 것이다. 그러나 그 이전에 판별/검색된 단어가 숫자적 정보가 없는 일반 명사의 경우로 처리되면, 두 단어를 접속 가능한 것으로 처리하는 것은 옳지 못하다. 결국 이 두 단어의 결합에 이어지는 형태소 해석은 옳지 못한 결과를 생성할 것이므로 이후의 해석을 진행하는 것은 의미가 없게 된다. 이를 위해서 형태소 해석 사전은 일본어 해석에 일반적으로 사용되는 접속 정보 이외에도, 이에 관련되는 정보를 아울러 가져야 한다. 본 논문에서 사용하는 접속 정보 중에서 용언의 어미에 관한 활용 정보는 그림 1과 같다. 이것은 동사와 형용사에 대한 어미 접속표를 기본형을 중심으로 각 활용형에 의해 분석한 것이며, 시스템에서는 이것을 유형별로 분석하여[8], 동사의 경우 그림 2와 같은 형식으로 활용형을 판별한다.

형식	단어	終止	否定	確定	命令	其他
10	燃す	~u	~aナイ	~iタ	~e	指す, 示す
11	死ぬ	""	""	-ンダ	""	呼ぶ,讀む
12	繼ぐ	""	""	-イダ	""	かぐ,こぐ
13	解く	""	""	-イタ	""	書く,引く
14	ゆく	""	""	イッタ	""	None
15	漏る	""	""	ッタ	""	いく,持つ
16	有る	""	()	""	""	None
17	なさる	""	~aナイ	ッタ	-イ	下さる
18	言う	""	~ワナイ	~ッタ	~e	思う,合う
19	問う	""	""	~ウタ	""	乞う, 逢う
1A	給う	""	("")	-ウタ	""	None
20	燃える	~ru	~ナイ	~タ	~ヨ/ロ	居る,見る
23	曳れる	""	""	""	~	None
22	うる	ウル	エナイ	エタ	エヨ	None
31	來る	~uru	~oナイ	~iタ	~oi	None
32	爲る	""	~iナイ	""	~eヨ/~iロ	☒する

그림 2. 동사 어미 활용
Fig. 2. A conjugation table of Japanese verb.

각각의 활용형태는 동사의 경우 그림 3과 같으며, 형용사의 경우 그림 4와 같이 구분한다. 예를 들어 그림 3의 동사 활용형에서 'A'라고 표시된 부분을 그림 2와 연계시키면 "呼ぶ(yobu)"의 연용형 형태이며 다음과 같은 동사 후속 부분에 접속할 수 있는 것으로 시스템에 기술된다. 즉,

A: -> デ ——→ [連用] 조사 "て(TE)"
ダ ——→ <陳述> "だ(DA)"
ill

의 형태의 접속에 대한 접속 위치를 지적하는 것이다. 이것은 "呼ぶ(yobu)"라는 동사가 조사 "て"와 결합할 때는 접속형 11번의 A와 같이 변형하고, 조사도 "で"로 변형된다는 것을 의미한다.

On-Line 방식으로 언어를 처리하기 위한 형태소 해석 사전은 기존의 언어 해석적 정보뿐만 아니라 모호성의 증가를 억제시키고, 효율적인 단어 검색을 위해 문자열에 관한 정보를 반드시 포함해야 한다.

		ス	サ	ソ	セ	シ	10
	ンA	ヌ	ナ	ノ	ネ	ニ	11
	ンA	ブ	バ	ホ	ベ	ビ	
	ンA	ム	マ	モ	メ	ミ	
	イ	グ	ガ	ゴ	ゲ	ギ	12
	イ	ク	カ	コ	ケ	キ	13
	ッ	ク	カ	コ	ケ	キ	14
	ッ	ク	カ	コ	ケ	キ	15
	ッ	ツ	タ	ト	テ	チ	
	ッ	ル	ラ	ロ	レ	リ	
	ッ	ル	ラ	ロ	レ	リ	16
イ	ッ	ル	ラ	ロ	レ	リ	17
	ッ	ウ	ワ	オ	エ	イ	18
		ウ	ワ	オ	エ	イ	19
		ウ	ワ	(オ)	エ	イ	1A
		ル			レ	λ	20
		ル			レ	λ	22
		ル			レ	λ	23

그림 3. 동사의 접속 규칙
Fig. 3. The combination rules of verbs.

	λ	ケレ	イ	カレ	カル	カッ	カロ	カラ	ク	5
	λ	ケレ	イ	カレ	カル	カッ	カロ	カラ	ク	51
	λ	ケレ	イ	カレ	カル	カッ	カロ	カラ	ク	52
	λ		イ					カラ	ク	55
キ									ク	59

그림 4. 형용사의 어미 활용 규칙
Fig. 4. The conjugation rules of adjectives.

본 논문에서 사용하는 형태소 해석용 사전의 형식은 다음과 같다.

```
struct    DICTIONARY{
            int          String_Continue;
            char         *Word_registerd;
            char         lexical_category;
            int          lexical_end_table;
            int          forward;
            int          *backward;
```

```
    int         homonym_flag;
    int         *generation_entry;
    int         reference_array;
    int         Noun_for_flag;
    char        refcat;
    char        *refdict;
};
```

Word_registered, lexical_category는 일반적인 어휘 정보를 저장하고, homonym_flag와 Noun_for_flag는 각각 동의어와 형태소정보에 포함되는 의미정보를 나타낸다. 즉 동음이의어의 경우 하나의 표제어에 여러개의 해석정보를 부가시키지 않고 각각을 별도의 어휘로 등록시키는데, 이는 검색과정에서 품사의 구별을 용이하게 하기 위함이다. forward와 backward는 그림 3과 그림 4에 나타낸 활용표와 여기에 이어지는 활용 접속 규칙을 연결시켜주는 역할을 하며 이것은 해석의 진행 중에 선택된다. String_Continue는 시스팀이 가지는 사전에, 현재 검색중인 문자열을 포함하는 문자열이 더 이상 있는지에 대한 정보를 가진다. 즉, 0의 값은 직후의 단어가 자신과 무관한 문자열임을 뜻하고, 1의 값은 자신을 포함한 문자열이 이어짐을 나타낸다. 앞서의 일본어 동사 "呼ぶ(*yobu*)"에 대한 형태소 해석 사전의 값은 다음과 같다.

```
((String_Continue=1)(Word_registered="呼")(lexic.
gory='v')
 (lexical_end_table=8,11)(forward=nil)(backward=nil)
 (homonym_flag=1)(generation_entry=23)(reference_
 array="")
 (Noun_for_flag=0)(refcat=nil)(refdict="yo"))
```

이 사전은 어간이 "呼"이고 8번 어미 활용(즉, "ぶ")을 하며, 접속 규칙 11을 가지는 동사로 음은 "yobu"이고 동음이의어를 가지고 있고, 직후의 단어는 "呼"라는 문자를 선두에 포함하고 있음을 나타낸다.

4. On-Line 방식의 형태소 해석

본 논문에서 설계하는 기본적인 On-Line 방식의 해석을 예문을 중심으로 표 1에 나타내며, 예문 "おしひろめる"의 입력에 대한 가능한 분석결과 중 일부를 보이고 있다. 입력의 방향은 입력 순서와 같은 좌->우로의 주사(scan) 방향을 가진다.

표 1에서는 제일 처음 문자의(예문의 경우"お") 입력 후, 그 자체가 단어를 이루는 경우(표 1의 예문에 대해서는 명사)와 임의의 단어의 첫머리라고 예상하는 경우의 두 가지로 분류하여, 각각 상태 '1'과 '2'로 나타내었다. 다음으로 두 번째 문자인 "し"가 입력되었을 때, 상태 '11'은 '1'의 후속하는 경우로 '명사 + 명사'로 해석하는 경우이고, 상태 '12'는 "し"로 시작되는 단어를 계속 검색중임을 '(+)'로 나타낸다. 상태 '21'의 경우는 상태 '2'에 후속하는 경우로, 상태 '2'가 단어의 계속을 나타내므로, "お"가 포함되는 단어, 즉 "おし"가 검색되었음을 나타내고, 상태 '22'는 "おし"가 포함된 단어를 계속 검색함을 나타낸다. 이와 같은 검색 및 분석 과정으로 일본어에 대한 On-Line 방식의 해석을 진행할 수 있다.

또한, 표 1에서 "+"는 각각의 글자에 대한 구분을 의미한다. 즉, "+"로 구분되는 하나의 글자가 한 순간에 대한 처리 대상이 됨을 의미한다. 그리고 이 때 사용되는 각 단어는 모두 형태소 사전에 등록되어 있는 것들이다. 표 1에서 왼편에는 각 단계마다 주사하는 문자를 나타내고 있다. 기본적으로 하나의 문자가 입력되었을 때 가정할 수 있는 경우는, 그 문자 자체가 형성하고 있는 단어를 검색하는 것과 그 문자가 포함된 단어를 검색하는 것이다. 표 1에서는 "(+)" 기호로 현재의 처리 대상 문자열이 포함된 단어를 계속 검색 가능함을 나타내었다. 즉, "(+)" 기호가 없는 문자열은 현재의

표 1. 일본어 동사 "おしひろめる"에 대한 입력 분석의 일례

Table 1. An analysis of a Japanese verb "おしひろめる"

(お + し + ひ + ろ + め + る)에 대한 입력 분석			
① お	1 お	2 お(+)	
② し	11 お - し 1	12 お-し(+) 2	21 おし 3
	22 おし(+) 4		
③ ひ	111 お - し - ひ	112 お - し - ひ (+)	121 お - しひ
	122 お - しひ(+)	211 おし - ひ	212 おし - ひ (+)
	~~221 おし ひ~~	222 おしひ(+)	
④ ろ	1111 お-し-ひ-ろ	1112 お-し-ひ-ろ (+)	(1121) お-し-ひろ X
	1122 お-し-ひろ(+)	1211 お-しひ-ろ	1212 お-しひ-ろ(+)
	2111 おし-ひ-ろ	2112 おし-ひ-ろ (+)	2121 おし-ひろ X
	2122 おし-ひろ(+)	2221 おしひろ	2222 おしひろ (+)
	1221 お - しひ (-) X X		
...	...		
	おしひろめる		

문자열로 단어를 만들 수 있음을 의미한다. "-" 기호는 검색된 단어들의 접속을 의미한다. 각각의 해석된 칸에 사용된 숫자는 상태를 나타내는 것이며, 상태 '1221'의 '(-)' 기호는 "しび"로 시작하는 문자열은 더 이상 형태소 해석 사전과 관련이 없으므로 이 상태 이후의 형태소 해석은 올바르지 않은 것을 의미한다.

일본어 문장에 대한 On-Line 방식의 처리는 그 특성상 입력이 종료될 때까지 단어의 검색과 해석된 노드의 생성을 중단할 수 없다. 즉, 하나의 입력문이 하나의 단어로 해석될 수도 있다는 것이다. 표 1에 사용된 예문의 경우도 입력이 종료되었을 때, 하나의 단어로 해석된 결과를 얻을 수 있다. 이것을 바꿔 말하면 사실상 어느것 하나도 단어 검색을 생략시킬 수 없음을 의미한다. 표 1의 상태 '221'은 "おしび"라는 단어가 사전에 없어서 이 문자열을 포함하는 해석은 더 이상 진전시키지 않아야 됨을 의미하는 것이다. 이는 상태 '1121' 등과는 구별되는 것으로 문법적 규범이 아닌 문자열 정보에 의한 것이다. 이를 처리하기 위해 해석 시스템은 상태 '221' 등과 같은 조건에서는 해석을 더 이상 시도하지 않아야 한다. 이것은 본 논문에서는 아직 미정의어의 처리는 포함하지 않음을 의미한다.

5. 형태소 해석 앨고리듬의 개선

전절에서 설계한 On-Line 방식의 형태소 해석에는 기본적인 문법적 정보와 사전적 정보를 이용하였다. 그러나 전절의 제 과정은 모든 문자열에 대해 단어로 해석될 가능성을 검사하는 것이므로 시스템의 처리에 난점이 많다. 따라서 본 논문에서는 2.3에서 제안한 사전정보로 보다 효율적인 검색과 해석을 위한 다음과 같은 앨고리듬을 제안한다.

Algorithm Make a Connection

```
type
    ContinueType = ( Yes, No, NIL );
var
    Node, StartNode, EndNode : NodeType;
    Continue, Continue1, Continue2 : Continue-
    nueType;
    Word, Word1, Word2 : String;

begin
  STEP1:
    PushMorphWord(Word [Node ], StartNo-
    de, EndNode, Continue, Complete)

  STEP2:
    GetOneWord(Node);
    if (EndNode = Node - 1) and (Continue =
    Yes) then
    Node = Node+1;
    Combine(Word [Node ], Startnode, End-
    node+1)
    Endnode = EndNode+1;
    Continue = NIL;
    PushMorphWord(Word [Node ], Start-
    Node, EndNode, Continue, Complete)
  end { if }

  PopMorphWord(Word [Node ], StartNo-
  de, EndNode, Continue, Complete)
  SearchWord(Word [Node ], StartNode,
  EndNode, Continue, Complete)

  if (Complete = 0) then
     goto STEP1;
  end { if }

     PopMorphWord(Word1, StartNode1,
     EndNoded1, Continue1, Complete1)
     PopMorphWord(Word2, StartNode2,
     EndNoded2, Continue2, Complete2)

  if (StartNode1 = EndNode2) then
     if (Continue2 = Yes) then
      PushMorphWord(Word2+Word1,Start
      Node2,EndNode1,Continue1,Complete1)
      goto STEP2;
     else if (Continue2 = No) then
      goto STEP1;
    end { if }
  end { if }

  if (CheckConnection(Word2, Word1) then
    Combine(Word, StartNode2, EndNode1)
  end { if }
  return TRUE;

end
```

이 앨고리듬은 먼저 하나의 문자를 입력받아 저장한다. 그 형식은 *PushMorphWord(Word, StartNode, EndNode, Continue, Complete)*와 같다. 여기서 Word는 처리 중인 문자열을 의미하며, StartNode는 입력 문자열 상에서의 시작 노드를 의미하고, EndNode는 마지막 노드를 나타낸다. 또 각 노드는 표 1에서의 "+" 위치와 동일하며 하나의 문자도 입력되지 않은 노드의 번호가 0이다.

처리된 문자열의 EndNode 값이 처리중인 문자열의 직전 노드와 같고, *PushMorphWord()*의 continue 값이 Yes이면, 즉 *PushMorphWord()*의 각 인자가 WORD1, StartNode, EndNode, Yes의 값을 가지면, 갱신한 노드로 *PushMorphWord(Word [Node], StartNode, EndNode, Continue, Complete)*의 값을 저장하여 새로운 해석의 기준으로 설정한다. 이는

문자열의 연속으로 하나의 단어가 계속 생성 중임을 의미하며, 이 때 Continue의 값이 NIL인 것은 아직 사전 검색을 완료하지 않았음을 의미한다.

PushMorphWord()의 값들을 꺼내어 Word 정보로 사전을 검색하고 Continue와 Complete의 값을 채운다. Complete의 값이 0이면 새로운 문자열을 입력받으며, 그렇지 않으면 이미 입력된 단어들의 접속을 검사한다. Continue의 값은 문자열이 이어질 때 Yes, 이어지지 않으면 NIL의 값을 가지며, 사전에 등록되지 않았으면 No의 값을 가진다. Complete의 값은 주어진 문자열이 사전에 등록된 단어일 경우 1, 그렇지 않으면 0이다.

이미 저장된 값에 대해, *PopMorphWord(Word2, StartNode2, EndNoded2, Continue2, Complete2)*와, 직전에 저장시킨 *PopMorphWord(Word1, StartNode1, EndNoded1, Continue1, Complete1)*에서 StartNode1 = EndNode2일 경우, Continue2의 값이 Yes이면 *PushMorphWord (Word2+Word1, StartNode2, EndNode1, Continue1, Complete1)*를 저장하고 나머지 문자열의 접속에 대한 검사를 반복한다. Continue1의 값이 0이면 단어 완성에 대한 검색을 실시하고 Continue1의 값이 No이면 새로운 문자열을 입력받는다. 이 때 직전에 저장시킨 *PushMorphWord(WORD1,,,,)*의 값은, 다중 처리기의 경우 EndNode 값이 EndNode1인 것을 의미한다. 그 후, *PushMorph Word (WORD2,,,,)* 와 *Push Morph Word(WORD1,,,,)*에 대해 접속 검사를 실시하고, 접속이 올바를 경우 *PushMorphWord(WORD2,,,,)*와 *PushMorphWord (WORD1,,,,)*을 결합한다. 문장 입력이 종료되고, 더 이상 결합할 것이 없을 때까지 이 과정을 반복한다. 이 때 앨고리듬의 진행은 *PushMorphWord()* 에 의한 두 형태소의 결합만을 고려해도 되므로 시간 복잡도는 $O(n^2)$이 된다. [20]에서 지적한 바와 같이 형태소 해석을 위한 어떤 앨고리듬도 $O(n^2)$보다 낮은 시간복잡도를 가질 수 없으므로 본 논문의 방식이 기존의 Off-Line 방식의 해석과 차이가 없음을 알 수 있다.

위의 앨고리듬으로 진행하는 형태소 해석의 구체적인 과정을 동사 "おしひろめる"를 중심으로 그림 5에 나타내고, 문장에 대한 해석의 과정과 결과를 그림 6에 나타낸다. 그림 5에서 상태 11은 노드의 추가와 생성이 모두 이루어졌으며 상태 11의 "お"+"し"는 두 개의 단어를 인식한 경우를 나타내는 것이고, 상태 12의 "お"+"し"는 단어 "お"만 인식한 후 "し"로 시작하는 단어의 인식을 계속 시도하는 경우를 나타내는 것이다.

상태	Morph-Word()	Result
1	("お",0,1,No,1)	노드 생성
2	("お",0,1,Yes,0)	
11	("し",1,2,No,1)	노드 추가
	("お"+"し",0,2,No,1)	노드 생성
12	("し",1,2,Yes,0)	
	("お"+"し",0,2,Yes,0)	
21	("おし",0,2,No,1)	노드 생성
22	("おし",0,2,Yes,0)	
...	...	...

그림 5. "おしひろめる"에 대한 해석의 일례
Fig. 5. A parsing result of "おしひろめる".

입 력 상 태	형태소 분석 ("くるまが左による."에 대하여...)
① く	1. く
② くる	1. く + る (FAIL) 2. くる
③ くるま	1. く + るま (FAIL) 2. くる (VP) + ま (NP) 3. くるま(NP)
④ くるまが	1. く + るまが (FAIL) 2. くる (VP) + ま (NP) + が (PP) 3. くる + まが (FAIL) 4. くるま (NP) + が (PP) 5. くるまが (FAIL)
...	...
⑨くるまが左による	1. くる + ま + が + 左 + に + よる VP NP PP NP PP VP 2. くるま + が + 左 + に + よる NP PP NP PP VP

그림 6. 형태소 해석 결과
Fig. 6. A result of a morphological analysis.

그림 6에서 입력 문장은 'くるまが左による(*kurumagahidariniyoru*)'로, 처음 입력되는 문자열은 'く'이고 마지막으로 입력되는 문자열은 8번째 글자인 'る'이다. ①의 상태에서는 'く'만 입력되었으므로 이것에 대한 해석 가능성만을 검사하게 된다. ②의 경우는 2번째 글자인 'る'가 입력된 상태로 이 때는 2가지로 나누어 생각할 수 있다. ② - 1의 경우, "く + る"는 "る"만의 형태소 분리가 불가능하여, 즉 "る"만으로 이루어지는 단어가 없으므로, 이후에 "く+る+"로 이어지는 형태소 분리는 제외시킨다. ② - 2의 경우 "くる"라는 단어는 해석이 가능하므로 노드에 삽입시킨다. 2

개의 글자가 입력된 상태의 결과는 "くる"라는 단어만이 해석가능한 것으로 남아있다. 다음 입력되는 문자는 "ま"이다. 이 경우는 다시 3가지로 세분하여 해석을 시도하게 된다. 새로운 문자가 입력되면 그 이전 상황에서 연결되는 경우가 있고(③-2 와 ③-3) 새롭게 시작되는 상황이 있는데, 후자의 경우가 바로 ③ - 1이다. "く + るま"의 경우는 이전에서 파생되는 노드가 아니기 때문이다. 이 때는 형태소 해석 사전에 "るま"로 시작하는 단어가 있는 한 계속 노드 생성 시도가 이루어진다.

그림 6의 입력상태는 표 1 등에 예시한 각 상태가 하나씩 결합된 형태이며, 그 오른쪽에 가능한 형태소 분리를 표시하였다. 한자(漢字)의 출현은 접두사를 제외하면 이전의 전/후 접속 규칙을 초기화시키는 효과가 있다. 이것은 일본어+한자의 형태는 발생하지 않는 것으로 간주하기 때문이며 이런 형태의 단어는 일본어 전체를 통틀어 대략 5% 이내라고 알려져 있고,[15] 접두어의 경우는 형태소의 분리 후 규칙에 의한 재결합 과정으로 처리가능하므로 처리 능력을 감소시키지 않을 것으로 판단된다. 그림 6에서 최종 입력상태가 ⑨인 것은 period를 포함한 문장부호가 입력의 종결을 알리는 것으로 간주하였기 때문인데, 이것은 시스템 구현에만 관련되는 문제이다. 최종적인 형태소 분리 결과, 2가지의 문법적인 문장이 가능하며, 이것은 한국어 생성 과정으로 그대로 전달시킬 수 있다.

6. 사전 검색횟수

On-Line 처리 시스템에서 가장 문제가 될 수 있는 것 중의 하나는 사전 검색횟수이다. 표 2에 On-Line 상태에서 문자수에 따른 사전 검색횟수 비교실험의 결과를 단어일 때에 대해서 나타내었다. 이는 품사에 구별을 두지 않고 임의로 형태소 해석사전에서 추출하여 실험한 결과이다. 여기에는 검색의 효율을 측정하기 위해 검색 효율치와 단어완성 확률이란 값을 사용하였다.

검색 효율치는

$$\left(1 - \frac{\text{검색 횟수}}{\text{절대 검색 횟수}}\right) \times 100\ (\%)$$

으로 정의하였으며, 이는 검색횟수를 상호 비교하기 위한 값이다. 이 값이 작으면 절대 검색 횟수에 가까워지는 것을 의미하므로 효율이 좋지 않은 것을 뜻한다. 단어 완성 확률은

표 2. 본 논문의 방식에서 글자수에 따른 사전 검색회수 비교(단어의 경우)

Table 2. A comparison of retrieval frequency. (in case of a word)

글자수	단어수	절대검색횟수	평균검색횟수	검색효율(%)
2	178	3	2.3	21.3
3	125	6	2.7	55
4	136	10	4.6	52
5	118	15	6.3	58
6	93	21	8.5	59.5

$$\frac{\text{분리 가능한 형태소의 갯수}}{\text{검색 횟수}} \times 100\ (\%)$$

의 형태로 정의하며 형태소 해석 효율을 상호 비교할 수 있는 값이다. 값이 작으면 검색 횟수에 비해 추출되는 형태소의 갯수가 적은 경우, 혹은 동일한 수의 형태소를 분리하였다면 검색횟수가 많은 경우이므로, 같은 조건하에서는 효율이 저하됨을 의미한다. 절대 검색횟수는 문자열의 길이가 N일 경우 N(N+1)/2로 정의되는 값이다.

실험에 의하면 2개의 글자로 이루어진 단어의 경우 검색 효율이 가장 낮고 점차 그 효율이 좋아지는 것을 알 수 있었다.(문자열의 길이가 3인 경우, 수치가 비교적 높은 것은 실험에 사용된 형태소 해석 사전에 3글자의 단어가 상대적으로 적었기 때문이라고 보인다.) 이것은 글자수가 긴 단어에 대한 분리를 시도할 경우 필요 없는 사전 검색횟수가 이전의 접속 정보에 의해 많이 줄어들 가능성이 높은 것을 나타내는 것으로, 형태소 해석 시스템이 이 정보를 충분히 활용해야 함을 의미한다.

Ⅲ. 실험 및 해석 효율

1. 사전 검색 횟수의 비교

Off-Line 방식의 해석은 일반적으로 형태소 분석과 해석, 그리고 문장 해석기로 시스템을 구성한다. Off-Line 방식의 해석은 완전히 입력된 문장을 대상으로 처리하는 것이며, 이 선형적인 입력 문자열을 사전을 참조하여 형태소로 분리하는 것으로 시작한다. 형태소 분석과 해석 과정에서는 표층의 언어 현상을 직접 처리하므로 번역의 전반적인 과정에서 매우 중요한 부분을 차지한다. 문장 해석부분은 분리된 일본어 문장을

적절히 재구성하여 가장 논리적인 문장을 처리하기에 간편한 형태로 생성한다. On-Line 방식의 형태소 해석과는 반대로 Off-Line 방식의 형태소 해석은 일반적으로 좌방향성이다. 형태소 분리 과정의 첫 입력은 시스템의 설계에 따라 다소의 차이는 있겠으나 전체 문장이 보통이며, 여기에서 분리 가능한 형태소를 하나씩 해석해 나간다. (NP1 NP2)로 분리 가능하며 NP3 -> NP1 NP2 가 가능한 어떤 명사가 있다면 On-Line 방식에서는 NP3로 먼저 판명한다. 해석목을 얻는 것만을 전제로 했을 때, On-Line 상에서 NP1, NP2가 각각 인식되고 NP -> NP NP 에 의해 최종적으로 NP가 되는 것에 비하면 Off-Line은 On-Line에 비해 신속한 처리 결과를 나타낼 수 있음을 알 수 있다.

표 3. 본 논문의 방식과 [13]에 의한 Off-Line에서 글자수에 따른 사전 검색회수 비교실험 평균치. (문장의 경우)
Table 3. A comparison of retrieval frequency. (in case of a sentence)

글자수	검색 횟수		분리한 형태소의 갯수		단어 완성 확률(%)	
	(A)	(B)	(A)	(B)	(A)	(B)
10	38	32	15	15	39.5	46.9
12	46	39	21	21	43.7	53.8
15	65	57	33	33	50.7	57.8
17	74	63	39	39	52.7	61.9
20	96	75	51	51	53	68

표 3은 On-Line 방식의 해석과 [13]에 의한 Off-Line 방식의 해석을, 각각 동일한 문장에 적용한 경우의 결과를 나타낸 것이다. (A)난의 수치는 On-Line 해석에 의한 실험치이고, (B)난의 수치는 [13]에 의한 Off-Line 방식의 해석에 의한 실험치이다. (A)와 (B) 두 가지 해석방법 모두 동일한 해석 사전을 사용하였으며, 최종 해석된 형태소의 갯수는 동일하였다. 문장의 경우, 단어와는 다르게 특정한 형태소의 분리는 가능하지만 그 결과치는 사용할 수 없는 경우가 빈번하다. 예를 들어 'ABCD'라는 4개의 형태소로 이루어지는 문장을 생각했을 때, 이 문장에서 'A', 'CD', 'BCD' 등의 3가지 단어가 검색 가능하다고 하면, 'CD'의 경우 앞의 'B'가 Fail되는 경우이므로 제외되어야한다. 다시 말하면 'A'와 'CD'는 해석이 가능하지만 이 방법으로는 'B'가 해석에서 제외되므로 이 문장은 'A' + 'BCD'라고 만 해석이 가능하다는 것이다. 이는 실제의 사용과 직결되는 부분이다. 실험의 일관성을 위해 한자(漢字)는 실험에서 제외하였는데 한자가 포함될 경우 단어 완성 확률은 더욱 높아진다. 본 논문에서 제안한 방법이 Off-Line 해석에 비해 단어 완성 확률이 다소 낮은 결과를 나타내고 있어 사전 검색 횟수가 많음을 의미하지만, 본 논문이 제안한 방식에 의한 실험이 Off-Line의 방식보다 정보가 적은 상태에서 실행된다는 것과 입력 대기시간을 줄이는 것을 감안하면, 글자수가 길어질수록 그 차이가 작아지는 것은 본 논문에서 제시한 방법의 효율을 입증하는 것이라 할 수 있다.

2. 해석 결과의 비교

기존의 논문에서 발표된 일본어 형태소 해석기는 대부분 문장의 입력이 종료된 후 해석을 개시하므로 본 논문의 방법과는 많은 차이가 있다. [19]와 [20]에서 제시하는 방법과 사전 형식으로 예문 "くるまが左による"를 해석한 경우, [20]의 방식은 그림 6의 두 가지 해석 결과를 모두 얻을 수 있었고, [19]의 방법은 두번째 해석 결과만을 생성하여, 2가지 해석결과를 모두 생성한 본 논문의 방식이 문장의 입력이 종료되지 않은 상태에서 해석을 개시하여도 신뢰할 수 있는 해석 결과를 얻을 수 있음을 알 수 있다.

표 4. 형태소 해석 결과의 비교
Table 4. A comparison of morphological analysises.
방법 1: 참고문헌 [13]의 방식
방법 2: 참고문헌 [20]의 방식
방법 3: 참고문헌 [19]의 방식

구분 \ 입력문	くるまが左による.	めのまえにある.
방법 1	2	3
방법 2	2	3
방법 3	1	1
본 논문의 방식	2	3

또한 [13]의 경우, 해석의 방향은 본 논문과 동일한 방향을 가지지만, 완전한 문장을 전제로 다단계 해석을 개시하므로, 본 논문의 방식과는 차이가 있다.그러나 중간 해석목과 해석된 결과는 동일하여 본 논문의 방식이 바른 해석을 실행함을 알 수 있다. 두가지 예문에 대해 기존의 논문의 방식과 본 논문의 방식을

해석목의 갯수를 중심으로 비교하여 표 4에 나타내며, 본 논문의 방식이 기존의 Off-Line 방식과 해석 결과가 다르지 않음을 보이고 있다.

표 5. "くるまが左による" 문장에 대한 분석 결과

Table 5. An analysis of a sentence "くるまが左による".

구 분	총 문자수	분석된 문장수	사전 검색횟수	분리된 형태소 갯수	단어 완성 확률
논리적 On-Line 해석 결과	8	2	36	11	30.6
본 논문의 해석 방식	8	2	25	11	44
참고문헌[13]에 의한 방식	8	2	22	11	50

표 5에는 예문 "くるまが左による"에 대한 문장 분석 결과를 보인다. 앨고리듬의 개선으로 사전검색 횟수는 줄어들고 단어완성 확률은 높아졌음을 보인다. 또한 [13]의 방식에 의한 해석결과와 비교하였을 때, 동일한 해석결과를 생성하고, 사전검색 횟수도 큰 차이가 없음을 보인다. 결국 본 논문이 입력과 동시에 해석을 진행하는 방식임을 감안하면, 사전 검색 횟수 및 단어 완성 확률이 기존의 Off-Line 방식에 근접하고 있음을 보이고 있다.

Ⅳ. 결 론

본 논문에서는 일한 기계 번역 시스템을 위한 On-Line 방식의 형태소 해석기를 제안하였다. On-Line 방식의 경우, 실제의 응용을 가정한다면 문장의 입력이 종료되기까지 입력없이 시간이 지연되는 경우도 있고, 입력 문장이 길 경우는 입력시간 자체도 길어지고 입력과 동시에 형태소 해석을 시작하므로,본 논문에서는 이를 고려하여 전체적인 언어처리의 속도를 단축시킬 수 있을 것으로 예상하여 이를 위한 해석 앨고리듬을 제안하였다. 제안한 해석 앨고리듬은 가능한 모든 해석목을 생성할 수 있었고, 문장의 입력이 종결되지 않은 상태에서 해석을 개시하여도 검색효율이 기존의 Off-Line 방식과 크게 차이 나지 않음을 보였다. 특히 제안한 앨고리듬은 각 단계가 독립적으로 운영 가능하므로, 각각에 대해 프로세서를 별도로 할당할 경우, 보다 향상된 처리 속도를 얻을 수 있을 것으로 기대된다. 일반적으로 On-Line 방식에서는 문장이 길어지면 모호성이 선형 이상으로 높아지지만 본 논문에서 제안하는 해석기에서는 불필요한 모호성을 억제하여 실시간 처리가 가능하다. 앞으로 이에 알맞은 언어 생성부와 이들을 보다 효율적으로 제어할 수 있는 병렬처리 앨고리듬이 완성되면 실시간으로 응용할 수 있는 On-Line 방식의 기계번역기가 가능할 것으로 사료되며, 본 앨고리듬은 약간의 개선으로 On-Line 방식의 한국어 형태소 해석에도 적용할 수 있을 것으로 보여 한국어 기계번역 시스템 등에도 적용 가능할 것으로 판단된다.

참 고 문 헌

[1] Daniel D. K. Sleator, Davy Temperley, "Parsing English with a Link Grammar ",

[2] Gosse Bouma, etc, "A Flexible graph-unification formalism and its appication to natural-language processing", *Journal of Research and Development*, 1988

[3] Masahiro Oku, "A Method for Analyzing Japanese Idioms"

[4] N. Maruyama, M. Morohashi, etc, "A Japanese sentence analyzer", *Jounal of Research and Development*, 1988

[5] Paola Velardi,Maria Teresa Pzienza, "How to Encode Semantic Knowledge", *Computational Linguistics* Vol. 17, No. 2, 1991

[6] Shalom Lappin,Michael McCord, "Anaphora Resolution in Slot Grammar", *Computational Linguistics* Vol. 16, No. 4, 1990

[7] T. Nishida, S. Dishita : "An English-Japanese machine translation system based on formal semantics of natural language", Proc. of COLING 82, 1982

[8] "日本語基本文法", 日本電子技術總合研究所研究報告

[9] 吉村賢治, 武內美津乃, 津田健藏, 首藤公昭, "未登錄語を含む日本語文の形態素解析", *情報處理學會論文誌* Vol. 30, No. 3, 1989

[10] 吉村賢治, 日高達,吉田浮, "文節數最小法を用いたべた書き日本語文の形態素解析, *情報處理學會論文誌* Vol. 24, No. 1, 1983

[11] 搏松明, "自動飜譯電話のための音聲處理と言語處理", *電子情報通信學會論文誌* D-II Vol. 75,

No. 10, 1992.

[12] 奥雅博, "日本文解析における述語相當の慣用的表現の扱い", *情報處理學會論文誌*, Vol. 31, No. 12, 1990

[13] 元吉文男, 大場健司, etc,"未定義語を含む文の多段階構文解析法", *電子情報通信學會論文誌*, D-II Vol. 72, No. 10, 1989.

[14] 日本人工知能學會, *人工知能ハンドバック*, オーム社, 1990

[15] 佐治圭三 外, 日本語學의 理解

[16] 池田尙志, "語法規則方式による日本語文の構文意味解析", *情報處理學會論文誌*, Vol. 26, No. 6, 1985.

[17] 平井誠,北橋忠安, "格の强度と述語の構文および意味屬性を用いた格構造の變換生成について", *情報處理學會論文誌*, Vol. 28, No. 3, 1987

[18] 平井章博,梶博行, 芦尺實, "機械翻譯向け前編集のための日本語係り受け構造の曖昧性檢出方式", *情報處理學會論文誌*, Vol. 31, No. 10, 1990

[19] 김영섬, 최병욱 "PC를 이용한 일한 번역 시스템 ATOM의 개발에 관한 연구 (I), - 해석 사전 구성과 형태소 해석을 중심으로 - ", 대한전자공학회논문지, Vol. 25, No. 10, 1988

[20] 김은자, 허남원,이종혁, "일-한 기계 번역 시스팀의 한국어 생성에서 양상류 의미자질을 이용한 술부처리" 제 5회 한글 및 한국어 정보처리 학술발표 논문집, 1993

저 자 소 개

姜 晳 壎(正會員) 1966년 8월 31일생. 1989년 2월 한양대학교 전자통신공학과 졸업(공학사). 1991년 8월 한양대학교 대학원 전자통신공학과 졸업(공학석사). 1995년 8월 한양대학교 대학원 전자통신공학과 졸업(공학박사). 1996년 3월 ~ 현재 동서대학교 컴퓨터공학과 전임강사

崔 炳 旭(正會員) 第 31卷 B編 第 12號 參照
현재 한양대학교 전자통신공학과 교수