# 규칙에 기반한 한국어 부분 구문분석기의 구현

이 공 주† · 김 재 훈††


## 요 약

본 논문에서는 문법검사기나 기계번역과 같은 실제 응용 시스템을 위한 한국어 부분 구문분석의 처리 대상을 정의하고, 규칙에 기반한 한국어 부분 구문분석기의 구현에 대해서 논의한다. 부분 구문분석기는 기본적으로 여러 개의 형태소나 단어가 구문적으로 하나의 구조에 속할 경우 이를 하나의 덩어리로 묶어주는 역할을 수행하며, 동시에 부가적인 작업을 수행할 수 있다. 또한 부분 구문분석기는 다양한 형태로 표현된 부분 구조를 표준 형태소 구조로 바꾸어 줌으로써, 상위 모듈의 처리에서 그 결과를 용이하게 사용할 수 있도록 한다. 본 논문에서는 한국어 부분 구문분석을 위해서 수동으로 작성된 140여 개의 규칙을 이용하였으며, 각 규칙은 일반적인 규칙과 마찬가지로 조건부와 행위부로 구성되었다. 부분 구문분석의 효율성을 관찰하기 위해서 일반적인 구문분석과 부분 구문분석을 포함한 구문분석을 비교하였다. 실험을 통해서 전자가 후자에 비해 약 두 배의 레코드 수가 요구됨을 알 수 있었다.


# Implementing Korean Partial Parser based on Rules

Kong Joo Lee† · Jae-Hoon Kim††


## ABSTRACT

In this paper, we present a Korean partial parser based on rules, which is used for running applications such as a grammar checker and a machine translation. Basically partial parsers construct one or more morphemes and/or words into one syntactical unit, but not complete syntactic trees, and accomplish some additional operations for syntactical parsing. The system described in this paper adopts a set of about 140 manually-written rules for partial parsing. Each rule consists of conditional statements and action statement that defines which one is head node and also describes an additional action to do if necessary. To observe that this approach can improve the efficiency of overall processing, we make simple experiments. The experimental results have shown that the average number of edges generated in processing without the partial parser is about 2 times more than that with the partial parser.


**키워드 : 한국어 부분 구문분석기(Korean partial parser), 표준 형태소 구조(canonicalized morphological structure)**

## 1. 서 론

한국어는 영어와는 달리, 문장을 구성하는 구성성분들의 순서가 전체적으로 매우 자유로운 것 같으나, 부분적으로는 매우 엄격하다. 예를 들면, 명사구 "사과 한 개"는 매우 엄격한 순서구조를 가지고 있으며, 이 명사구에 속하는 단어의 어순을 변경하면 한국어의 문법에 벗어나게 된다. 본 논문에서는 한국어 문장 구조 분석을 위해서 이와 같이 엄격한 순서구조를 가진 구들에 대한 구문을 먼저 분석하고, 그 분석 결과를 이용하여 전체 문장의 구조를 분석하고자 한다. 이와 같은 과정을 일반적으로 부분 구문분석(partial parsing)이라고 한다[3, 9]. 부분 구문분석은 복잡도가 매우 높은 구문분석 문제를 단계적으로 해결하고자 하는데 그 목적이 있다. 즉, 가장 분명한 구문구조를 먼저 분석하고, 분석된 구조를 이용해서 좀더 복잡한 구조를 분석하는 방법이다. 응용분야에 따라 부분 구문분석의 정의는 조금씩 다르지만, 구들 중에서 연속적이고, 비재귀적인 구성성분에 해당하는 단순한 구를 인식하는 것으로 정의할 수 있다[3, 9]. 이처럼 부분 구문분석은 구문분석의 전처리 과정으로도 볼 수 있지만, 그 자체 결과만으로도 여러 분야에 응용될 수 있다. 예를 들면, 정보추출이나 질의응답[15, 20], 말뭉치 분석 도구[21, 23], 개체명 인식과 같은 특정 표현 인식[22, 31] 등에서 부분 구문분석이 이용되고 있다.

일반적으로 부분 구문분석의 처리범위가 명확하지 않기 때문에 여러 가지 방법으로 그 범위를 명시하려는 노력이 있었다[3, 10]. 그러나 앞에서 언급했듯이 어떠한 용도로 부분 구문분석을 사용하느냐에 따라 그 정의와 처리범위가 조금씩 달라질 수 있다. 부분 구문분석의 수행 방법이나 부분 구문분석 구조를 표현하는 방법도 또한 부분 구문분석의 목적에 따라 다를 수 있다. 본 논문에서는 문법검사나 기계번역과 같이 실제 응용 시스템을 위한 부분 구문분석

※ 이 연구는 2003학년도 이화여자대학교 교내연구과제 지원에 의한 연구임.
† 정 회 원 : 이화여자대학교 컴퓨터학과 전임강사
†† 정 회 원 : 한국해양대학교 컴퓨터공학과 교수
논문접수 : 2003년 3월 16일, 심사완료 : 2003년 5월 26일

〈표 1〉 부분 구문분석 시스템의 요약

| 시 스 템(명) | 방 법 론 | 분 석 단 위 | 재 현 률(%) | 정 확 률(%) |
|---|---|---|---|---|
| (Bourigault, 1992)[12] | 단순규칙 | 명사구 | | |
| Fidditch[21] | 수정된 구구조 문법 | | | |
| NPTool[29] | 수정된 구구조 문법<br>유한상태 오토마타 | 명사구 | 97.2 | 96.1 |
| PARTS[16] | HMM | 비재귀적명사구 | 98.0 | |
| Chunk Tagger[26] | HMM | 말덩이 | | 81.3 |
| (Chen and Chen, 1994)[17] | HMM<br>유한상태 오토마타 | 말덩이<br>최장명사구 | 96.0 | 95.0 |
| (Brants, 1999)[13] | CMM | 말덩이 | 84.8 | 91.4 |
| (Skut and Brants, 1998)[27] | 최대 엔트로피 모델 | 말덩이 | | 84.2 |
| (Daelemans et al, 1999)[18] | 메모리기반 학습<br>유사도기반 인식 | 명사구<br>동사구 | 94.0 | 93.7 |
| (Cardie and Pierce, 1998)[14] | 메모리기반 학습<br>최장일치 규칙 | 기저명사구 | 94.0 | 94.0 |
| (Ramshaw and Marcus, 1995)[25] | 변환기반 학습<br>변환규칙 | 기저명사구<br>명사구/동사구 | 92.0<br>88.0 | 92.0<br>88.0 |
| (Dagan and Krymolowski, 2001)[19] | 메모리기반 학습 | NP<br>VP with NP | 76.1<br>82.6 | 91.3<br>61.5 |

에 대해서 논의할 것이다.

문법검사나 기계번역 시스템과 같은 응용 시스템에서 다루어야 하는 문서에는 괄호, 한자, 숫자 등과 같은 각종 기호들이 자주 나타난다. 이러한 기호들은 문서분석을 어렵게 한다. 예를 들어, '"경전(慶典)"으로'와 같은 어절의 경우, 모두 7개의 형태소[1]로 구성된다. 그러나 의미적으로 이 어절에 대한 형태소 분석은 '경전+으로'만으로도 충분하다. 즉, 예에서 본 바와 같이 기호들로 인해 복잡해진 형태소 분석 구조를 의미의 변화 없이 단순한 형태소 구조로 변환하는 기능이 부분 구문분석에서 필요로 한다. 본 논문에서는 이와 같이 단순화된 형태소 구조를 **표준 형태소 구조**(canonicalized morphological structure)라고 한다. 수식표현(numerical expressions)의 경우에도 표준 형태소 구조로의 변환이 필요하다. 예를 들면 어절 '삼백육십'은 '360'으로 변환함으로써 '360'을 표현하는 여러 표층 표현('삼백육십', '3백6십', '360')을 하나로 다룰 수 있다. 본 논문의 부분 구문분석기는 부분 구문분석의 기본 기능인 여러 개의 형태소나 어절을 하나의 단위로 묶는 작업 외에도 부분 구문 분석된 구문을 표준 형태소 구조로 변환하는 작업을 동시에 수행한다.

본 논문에서는 규칙에 기반해서 이러한 고려사항들을 만족할 수 있는 부분 구문분석기를 구현하고자 한다. 규칙 기반의 부분 구문 분석을 수행함으로써 문법 검사기나 기계번역과 같이 정확도가 중요 고려 사항이 되는 응용 분야에 적합한 부분 구문 분석기를 구현할 수 있을 것이다. 2장에서는 부분 구문분석과 관련된 여러 선행 연구에 대해서 기술하고 3장에서는 부분 구문분석을 위한 규칙에 대해서 기술하고, 4장에서는 3장에서 정의된 규칙을 이용해서 본 논문에서 다루는 부분 구문분석 규칙을 기술하고 5장에서 시스템의 구현 방법과 실험 결과를 기술한다. 끝으로 6장에서 앞으로의 연구 방향과 결론을 내리고자 한다.

## 2. 관련 연구

부분 구문분석의 기본 단위는 매우 다양하게 정의되어 왔으나, 가장 일반적으로 널리 사용되는 것은 말덩이(chunk)다[3, 8]. 말덩이는 중의성을 가지는 문의 구성성분에 대해서 중심어에 부착하지 않은 상태의 구문분석 조각이다[8]. 즉, 말덩이는 반드시 연속적이어야 하며, 말덩이 내에 다른 말덩이가 포함될 수 없다. 영어의 경우, 명사구, 전치사구, 수식어구, 접속어구, 동사의 필수 성분 등이 모두 여기에 속한다. 부분 구문분석의 분석 단위는 주로 응용 분야에 따라 말덩이(chunk), 기저명사구(baseNP)[25], 최장명사구(maximal length NP)[12], 구문요소[6] 등 매우 다양한 용어로 정의되어 왔다.

부분 구문분석 방법은 크게 규칙 기반 방법, 통계 기반 방법 그리고 기계학습에 의한 방법으로 나누어 볼 수 있다[2]. 규칙을 이용한 방법에는 주로 수정된 구구조 문법[30]과 오토마타[11]를 사용하여 부분 구문분석을 수행하였다. 오토마타를 이용한 경우는 주로 다층형의 유한상태 오토마타를 이용하고 있다. 통계기반의 부분 구문분석의 대표적인 방법으로 [26]을 들 수 있다. 이 방법은 주어진 단어열에 대해서 구조적 관계열을 찾는 문제로 부분 분석을 해결하고자 하였다. 기계학습의 경우에는 메모리 기반 학습 방법[18, 28]을 비롯한 여러 종류의 학습 알고리즘을 이용해서 부분 구문분석에 적용하고 있다[24]. 이를 종합하면 <표 1>과 같

1) 어절 '"경전(慶典)"으로'를 형태소로 분리하면 ("", 경전, (, 慶典, ), "", 으로)이 된다.

이 요약할 수 있다.

이와 같은 다양한 부분 구문분석 방법들은 다중단어, 전문용어, 공기정보, 선택제약정보 등과 같은 다양한 어휘 정보를 추출하는 데 사용되어 왔으며 전체 구문분석의 전처리기로도 개발되어 사용되어 왔다. 최근에 와서는 부분 구문분석만으로도 정보추출이나 질의응답[15, 20], 말뭉치 분석 도구[21, 23], 개체명 인식과 같은 특정 표현 인식[22, 31] 등의 분야에서 응용되고 있으며 그 응용범위는 앞으로도 더욱 증가될 것이다.

한국어의 경우에는 영어의 경우와 같이 부분 구문분석에 대한 연구가 활발하지는 않지만, 몇몇 연구자들에 의해서 진행되어 왔다[3, 5, 6]. 부분 구문분석은 구문요소라는 용어로 처음 시작되었다. 구문요소는 문장의 구성성분을 말하며, 이는 구문분석의 기본 단위가 된다[6]. 구문요소는 문의 의미를 알리는 구문정보를 지니고 있어야 하며, 구문요소는 주어, 서술어, 목적어, 부어, 관형어, 부사어, 독립어로 분류된다. 구문요소는 하나의 단어로 구성되는 경우도 있지만, 경우에 따라서는 구나 절이 될 수도 있다. [6]에서는 후자와 같이 광범위한 구문요소를 다루는 것이 아니라, 하나의 의미를 가지는 최소단위를 구문요소로 정의하여 처리하였다. [3]은 한국어 부분 구문분석의 단위를 정의했으며, [5]은 문법검사기를 위해서 부분 구문분석을 사용하며, 주로 문장의 의존관계를 확인하기 위해 사용되었다.

## 3. 부분 구문분석을 위한 규칙

### 3. 1. 규칙의 형식

본 논문은 수동으로 작성된 규칙을 기반으로 부분 구문분석을 수행한다. 본 논문에서 사용되는 기본적인 규칙의 형식은 (그림 1)과 같으며, 형식언어(formal language)의 문맥자유 문법(restricted context-free grammar)과 같은 형식이다. 즉, 본 논문의 노드((그림 1)의 $NODE_1$, $NODE_2$, $NODE_3$)는 문맥자유 문법의 비단말기호(non-terminal symbol)와 같은 역할을 한다. 기본 규칙은 두 개의 노드($NODE_1$, $NODE_2$)를 결합하여 하나의 노드($NODE_3$)를 생성하는 이진규칙(binary rule)이며, 예외적으로 규칙의 오른편(RHS)에 하나 또는 세 개의 노드가 올 수 있다. $NODE_i$는 품사 혹은 문장의 구성성분이 될 수 있으며, 필요에 따라서는 규칙기술자(rule descriptor)가 정의한 임의의 심볼이 될 수도 있다. $expression_i$는 $NODE_i$가 만족해야 할 조건이며, 여기서 조건은 간단한 논리식이나 연산식이며, 복잡할 경우에는 다른 함수를 호출하여 조건을 검사할 수도 있다. *statements*는 행위부(action part)로서 규칙의 오른편 두 노드가 결합되었을 때, 처리해야 하는 작업들을 기술한다. 이 규칙의 조건이 만족 되었을 때, $NODE_1$과 $NODE_2$는 새로이 생성된 $NODE_3$의 자식이 되며, $NODE_3$은 다시 다른 부분 구문분석 규칙의 적용을 받을 수 있다. 부분 구문분석기의 *statements*에서는 $NODE_i$과 $NODE_2$로부터의 적절한 정보를 $NODE_3$으로 넘겨야 하며, 부가적인 동작이 필요할 경우, 이를 수행하도록 한다.

```
RULE_NAME:
        NODE1 ( expression1 )
        NODE2 ( expression2 )
        →
        NODE3 { statements ; }
```

(그림 1) 부분 구문분석을 위한 규칙의 형식

### 3.2 형태소 분석 결과

부분 구문분석기의 입력은 형태소 분석 결과이다. 본 논문에서 사용되는 형태소 분석기[7]의 결과는 (그림 2)와 같은 레코드들의 격자구조(lattice)로 표현된다. (그림 2)는 어절 "직업군인"과 어절 "움직여보았다"에 대한 형태소 분석 결과이다. 각각의 레코드는 실질어(*cont*)와 기능어(*func*) 그리고 사전정보(*dictinf*)로 구성되어 있으며, 하나의 어절은 레코드들의 나열로서 표현될 수 있다.

| | | | |
|---|---|---|---|
| ***cont*** : 직업/noun<br>***func*** : NULL<br>***dictinf*** : | ***cont*** : 군인/noun<br>***func*** : 의/josa<br>***dictinf*** : Humn Gen | ***cont*** : 움직이/verb<br>***func*** : 어/ending<br>***dictinf*** : Concat | ***cont*** : 보/verb<br>***func*** : 쓰다/ending<br>***dictinf*** : Past Decl |

(그림 2) 형태소 분석 결과 예제

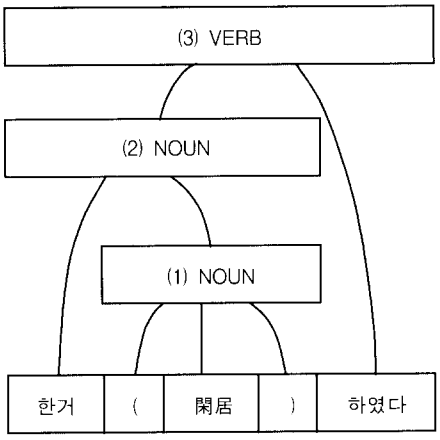
형태소 분석 결과의 각 레코드는 부분 구문분석 규칙의 조건부 노드에 대응되어 각각의 *expression*을 만족하는지 검사 받게 된다. *expression*에서는 레코드가 갖고 있는 기본 자질 - 실질어, 기능어, 사전정보 - 외에도 그 레코드의 표층어, 레코드의 바로 앞이나 바로 다음에 위치한 레코드 등 그 레코드와 관련된 모든 정보에 대해서 검사가 가능하다. 레코드의 조건이 만족되면 부분 구문 규칙에 의해 새로운 노드가 생성된다.

### 3.3 규칙 기술의 예

(그림 3)은 입력어절 '한거(閑居)하였다'에 대한 부분 구문분석 결과이다. 이와 같은 부분 구문분석 결과를 생성하기 위해서 3개의 부분 구문분석 규칙이 필요하다[2]. 여는 괄호와 닫는 괄호를 처리하는 규칙(BrackNounBrack1)에 의해서 노드 (1)의 NOUN이 생성된다. 노드 (2)는 복합어를 처리하는 규칙(NounBracketedNoun1)에 의해 생성되며, 이때의 표준 형태소 구조는 ***cont*** 정보로써 '한거'를 갖게 되며, 괄호 명사는 부가정보로써만 저장된다. 마지막으로 (그림 4)의 CC_NOUN_VERB1에 의해서 노드 (3)의 VERB가 생성되었다. 이때에는 사전에 '한거하다'라는 단어가 있는지를 검사하여 있을 경우, 그 사전 정보를 노드 (3)에 전달한

2) (그림 3)을 위한 실제 부분 분석 규칙이 (그림 9)에 제시되고 있다.

다. 노드 (3)의 표준 형태소 구조는 ***cont*** 정보가 '한거하다', ***funct*** 정보가 '쓰다'가 된다. 최상위 노드에 해당하는 노드 (3)만이 전체 구문분석에 전달된다. 따라서 전체 구문분석의 입력 노드를 크게 줄일 수 있고 수행속도도 크게 개선할 수 있다.




(그림 3) 부분 구문분석의 결과 형태

```
CC_NOUN_VERB1 :
    NOUN (has_open_close_bracket() & funct = NULL)
    VERB (cont in ? set { '당하' '만들' '하' '짓' '시키' '되' } )
    →
    NOUN { head = VERB ;
           cont = cont(NOUN) + cont(VERB) ;
           funct = funct(VERB) ;
           If (has_dict_info (cont(NOUN) + cont(VERB)) {   // '한거하'
           dictinfo = dictinfo (cont) ;
           }
    }
```

(그림 4) (그림 3)을 위한 부분 구문분석 규칙

## 4. 부분 구문분석의 적용 범위

본 논문에서는 한국어 구문분석의 기본 단위가 될 수 있는 문장의 구성성분을 인식하는 것을 주목적으로 하며, 그 처리 대상을 표준 형태소 구조라고 한다. 본 논문에서 사용하는 140여 개의 부분 구문분석 규칙은 크게 일곱 종류(괄호 처리, 인용부호 처리, 복합어 처리, 인명 처리, 수사 및 숫자 표현 처리, 문법 요소 및 고정된 표현 처리, 각종 심볼 처리)로 나눌 수 있다. 괄호처리와 인용부호 처리 규칙이 모두 20개, 복합어 처리 규칙이 35개, 인명 처리 규칙이 7개, 수사 및 숫자 표현 규칙이 19개, 문법 요소 및 고정된 표현 처리 규칙이 34개, 심볼 처리 규칙이 16개, 그리고 그 외의 기타가 10개로 구성되어 있다. 이하의 절에서 이들 규칙에 대해서 하나씩 살펴보고자 한다.

### 4.1 괄호 처리

문서에서 어떤 단어의 의미를 분명히 하고자 할 경우, 괄호를 사용한다. 예를 들면, 원어(한자, 영어, 일본어 등), 연대, 나이, 주소, 주석, 설명 등을 표현할 때, 괄호를 사용한다. 이 경우 일반적으로 앞말에 붙여 쓰고, 괄호 다음에 조사가 오면 단어처럼 조사는 괄호 뒤에 바로 붙여 쓴다[1]. 괄호 내에는 단순한 단어뿐 아니라, 구, 절, 문장이 모두 가능하기 때문에 부분 구문분석 단계에서 괄호 처리의 범위를 어디까지 다루어야 하는가를 우선 결정해야 한다.

<예 제>

가) 중용(中庸)은

나) 영하 24.0℃(4시 10분)입니다.

다) 유가의 내성성덕지교(내적으로 성인이 되어 덕을 완성하도록 하는 가르침)를

라) 가면을 쓴 (또는 벗어 던진) 힘과 힘의 대결

본 논문에서 다루고 있는 괄호는 우선, 부가적인 설명을 위해서 단어 뒤에 띄어쓰기 없이 나타나는 여는 괄호와 그에 따라 나타나는 닫는 괄호를 의미한다. 즉, 예제 (라)는 본 논문의 부분 구문분석 괄호 처리의 범위를 벗어난다. 본 논문에서의 부분 구문분석기는 여는 괄호와 닫는 괄호 사이에 한 개의 처리 단위가 존재할 경우에만 여는 괄호와 닫는 괄호의 쌍을 맞추어 처리를 수행하도록 처리 범위를 정의하였다. 여기서 하나의 처리 단위라고 하는 것은 기본적으로 한 개의 어절만을 의미하며, 여러 어절일 경우라도 그 여러 어절이 부분 구문분석의 다른 규칙에 의해서 하나의 단위로 묶일 경우를 의미한다. 이렇게 괄호 처리의 범위를 축소시킨 이유는 예제 (다)와 같이 괄호 사이에 여러 어절이 존재할 경우, 괄호 사이에는 단순 명사절뿐만 아니라 하나의 문장까지도 올 수 있기 때문이다. 즉 괄호 사이의 단어들을 처리하기 위해서 부분 구문분석기가 문장을 처리할 수 있는 규칙까지 가지고 있어야 하게 된다. 그렇기 때문에 본 연구에서의 부분 구문분석기는 괄호 사이에 하나의 처리 단위만을 가지고 있는 경우에 한해서 괄호를 묶어주는 역할을 수행하도록 하였다. 위의 예제에서는 예제 (가)와 예제 (나)만이 괄호쌍을 맞추어 하나로 묶일 수 있는 경우이다. 예제 (가)의 표준 형태소 구조는 '중용은'과 동일하며, 한자 부분은 부가적인 정보로만 저장된다. 예제 (나)의 '4시 10분'은 두 어절로 구성되어 있으나, 부분 구문분석기에서 하나의 단위 - 시간 표현 - 로 분석된다.

### 4.2 인용 부호 처리

인용부호는 대화, 인용, 특별어구 따위를 나타낼 때 사용한다[1]. 인용 부호는 주로 큰따옴표나 작은따옴표로 구성되며 여러 어절에 걸쳐서 나타나므로 그 짝을 맞추는 작업은 부분 구문분석 단계의 범위를 넘는다. 본 논문에서의 부분 구문분석 단계에서는 한 어절에 나타나는 인용부호 처리만을 그 대상으로 삼는다.

<예 제>

마) 그는 '동정'과 '경의'의 마음을…

바) "경전(慶典)"으로

사) "맑스는 사회주의는 … 공산주의로의 과도기라는 것을 보여준다"라는 …

예제 (마)와 (바)가 처리 대상이 된다. 예제 (마)의 경우에는 형태소 분석 단계에서 하지 못했던 닫는 따옴표 뒤의 분간조사[3] '과'에 대하여 앞 명사 '동정'과의 결합 여부(받침의 유무)도 검사한다. 예제 (바)의 처리 결과 표준 형태소 구조는 '경전으로'와 동일하며 괄호정보, 따옴표 정보가 부가적으로 함께 저장된다. 예제 (사)의 경우에는 여는 따옴표와 닫는 따옴표를 함께 처리하지 못하고, 대신 따옴표가 결합되어 있는 어절에 대해서만 따옴표 처리를 수행한다.

### 4.3 복합어 처리

본 논문에서 복합어는 한 어절 내에 여러 개의 명사나 동사, 또는 그 외의 성분들이 결합되어 있는 경우를 의미한다. 처리 범위와 해당 예제는 다음과 같다.

| 처 리 대 상 | 예 제 |
|---|---|
| 심볼(영어/한자/숫자) + 조사 | 90% + 까지 |
| 명사 + 명사 | 황금 + 박쥐 |
| 명사/대명사 + 괄호명사 | 경전 + (慶典) |
| 명사/대명사 + 접사 | 10대 + 들 김용수 + 님 |
| 접사 + 명사 | 반 + 체계적 |
| 동사/형용사 + 동사/형용사 | 자고 + 싶었다 |
| 명사/대명사 + 동사/형용사/서술격조사 | 신문 + 읽는 |
| 명사/대명사 + 부사 | 그녀 + 없이 |

가장 기본적으로 복합명사의 경우, 복합명사들을 하나의 단위로 묶어주고 최종 결과 명사구절의 중심어와 기능어를 정의해야 한다. 명사 뒤에 '들'이나 '님'과 같은 접미사가 결합했을 경우에는 복수(plural)나 존칭(honorific)에 관련된 부가적인 정보를 첨가해야 한다. (그림 5)에는 명사 뒤에 접사 '님'이 결합했을 경우의 처리 규칙을 간략하게 보이고 있다.

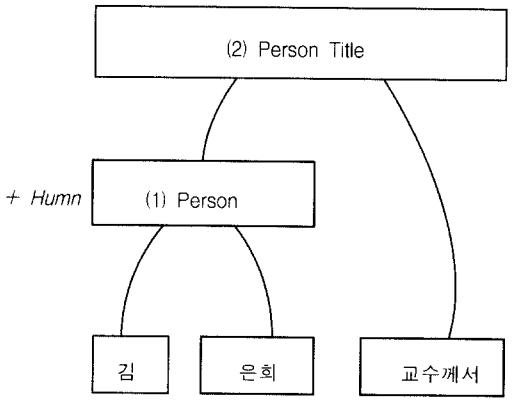
```
CC_NOUN_HONR :
        NOUN (func = NULL)
        NOUN (cont = '님')
        →
        NOUN { head = NOUN#1 ;
                cont = cont(NOUN#1) ; func = funct(NOUN#2) ;
                dictInf = dictInf(NOUN#1) ++ 'Honr' ;
        }
```

(그림 5) 접사 '님' 처리 예제

3) 앞 체언의 받침 유무에 따라 분간하여 쓰이는 조사.

### 4.4 인명 처리

본 연구에서 사용하는 사전은 120개의 한국인 성과 5,400여 개의 한국인 이름 정보를 갖고 있다. 형태소 분석기는 이와 같은 성과 이름의 정보를 부분 구문분석기에 넘겨준다. 부분 구문분석기는 이를 이용하여 다양한 인명 표현을 처리하여 전체 구문분석기로 넘긴다. 성과 이름은 붙여쓰는 것이 원칙이나, 성과 이름을 띄어 쓴 경우에도 처리하도록 하였다.




(그림 6) 인명 처리 예제

(그림 6)에는 '김은희 교수께서'의 부분을 처리한 결과를 보여주고 있다. 노드 (1)에서 성과 이름을 인식하여 'Person'이라는 노드를 생성하며, 이때 '*Humn*'이라는 부가적인 정보를 첨가한다. 노드 (2)에서는 노드 (1)과 그 뒤의 호칭/직위 명사를 함께 묶어서 Person_Title이라는 새로운 노드를 생성해 내고, 이것이 전체 구문분석 단계에 참여하게 된다. 7개의 인명 처리 부분 규칙이 사용되고 있으며, 이 규칙들이 처리할 수 있는 형태는 다음과 같다.

<처리 예제>

철수씨가, 최군이, 김모양은, 김 선생에게, 이인국, 김훈,
황선영은, 박헌영선생, 박헌영씨를, 김대중(金大中)씨,
김진호전무님
최진호 박사는, 화백 장욱진은, 화백 장욱진 선생은

### 4.5 수사 및 숫자 표현 처리

수사는 수량을 가리키는 양수사와 차례를 가리키는 서수사로 나뉘며, 이들은 각각 고유어와 한자어로 된 것이 있다. 본 연구의 부분 구문분석기는 고유어와 한자어의 양수사 및 서수사를 모두 처리한다. 수사는 만단위로 띄어 쓰는 것이 원칙이나, 잘못 띄어 쓴 경우에도 모두 인식할 수 있도록 구현하였다. 처리하는 대상은 다음과 같다.

<처리 예제>

'칠천구백삼십', '마흔아홉', '삼천 이백 칠십 육', '서넛',

'제 3의', '1,200,000', '-23.06'

본 논문의 부분 구문분석기는 한글 표현의 수사를 단순히 묶어주는 역할 이외에도 수사를 의미 있는 값으로 바꾸어 주는 역할도 수행한다. 예를 들어, '천구백삼십칠'과 같은 표현에서 이를 실제 값 1,937로 바꾸는 작업도 동시에 수행한다. 이렇게 하는 이유는 실제 응용 프로그램에서 값의 범위를 점검하여 특정 동작을 수행할 지의 여부를 결정할 수 있거나 또는 기계번역기에서는 한글 표현을 실제 값으로 변환한 후 다른 언어로 생성해 내야 하기 때문이다. 수사뿐만 아니라, 숫자 '1,200,000'나 '-23.04'과 같은 표현을 하나의 단위로 묶어줌과 동시에 이를 실제 값으로 환산하는 작업도 수행한다.

### 4.6 문법적 요소 및 고정된 표현들

여러 어절에 걸쳐 나타나는 고정된 표현들을 처리한다. 이와 같이 여러 어절에 걸쳐서 나타나는 고정된 표현의 경우, 이를 한 노드로 묶어주면서, 이 노드의 타입을 결정해 준다. 이 처리에 해당하는 대표적인 예들은 다음과 같다.

<처리 예제>

'~수 있다/없다', '~ㄹ 터이다/참이다', '~기/게 마련이다', '~기 일쑤이다', '~ㄹ 것 같다'

'1998년 3월 27일', '오전 8시 10분', '오후 9시 정각', '5학년 2반'

### 4.7 각종 심볼 처리

문서에 나타날 수 있는 모든 종류의 심볼을 처리한다. 문장 부호, 수학 기호, 약어 표현, 도량형 표현 등이 여기에 해당한다. 이와 같은 심볼들은 적절한 정보를 부가한 NOUN으로써 처리한다. (그림 7)에는 약어를 처리하는 간단한 규칙의 예가 보이고 있다.

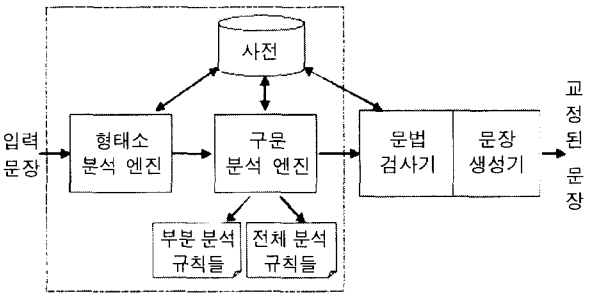
```
Abbrev1 :
    NOUN ( pure_alpha ( Word ( Lex ) ) & CltoB )
    CHAR ( cont == "." & CltoM )
    NOUN ( pure_alpha ( Word ( Lex )) )
    → NOUN {head = NOUN#2 ; + Factoid ; + KeepIt ;
            + Abbrev ;
    }
```

(그림 7) 약어 처리하는 규칙 예제

## 5. 한국어 부분 구문분석기의 구현 및 실험

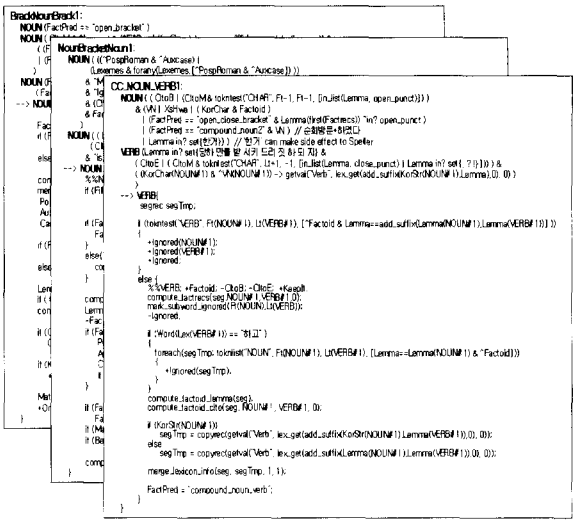
### 5.1 한국어 부분 구문분석기

본 논문에서 다루고 있는 부분 구문 분석기는 한국어 문법 검사기 시스템의 일부로써 구현되어 있다(그림 8). 입력 문장에 대해 형태소 분석과 구문 분석을 하게 되면, 구문 트리가 결과로 도출된다. 문법검사기는 구문 트리에 대해서 문법적으로 수정해야 할 부분이 있는 지를 검사하여, 수정할 부분이 있을 경우, 문장 생성기를 통해 수정된 문장을 생성해 낸다.




(그림 8) 전체 시스템 구성

(그림 8)에서 전체 시스템 구성 중, 본 연구에서 다루고 있는 부분을 점선으로 표시하였다. 부분 구문 분석과 전체 구문 분석은 동일한 분석 엔진을 사용하고 있으며, 어떤 규칙을 사용하느냐에 따라 부분 분석을 하거나 전체 분석을 수행하게 된다. (그림 9)에는 실제 시스템이 사용하는 부분 분석 규칙의 예제를 보이고 있다.



(그림 9) 실제 시스템이 사용하는 부분 분석 규칙의 예제

### 5.2 실험 및 평가

본 논문에서 제안한 부분 구문분석기는 모두 141개의 규칙을 사용한다. 각 규칙은 모두 수동으로 일일이 작성되었다. 부분 구문분석기의 성능을 평가하기 위해서 세종코퍼스[4]를 사용하였다. 실험 대상으로 삼은 코퍼스는 570,546문장의 약 700만 어절로 구성되었으며, 문장당 평균 어절 수는 11.7어절이다. 부분 구문분석기의 평가 측도로서 수행속도와 구문분석에서 필요한 노드 수(레코드 수)를 사용하며, 그 결과는 <표 2>과 같다. <표 2>에서 구문분석 (A)는 형태소 분석 결과를 입력으로 받아 부분 분석 규칙을 적용하

고, 그 결과를 이용해서 다시 전체분석규칙을 적용한 경우이다. 구문분석 (B)는 형태소 분석 결과를 입력으로 받아, 부분 분석 규칙과 전체 분석 규칙을 모두 한꺼번에 적용하여 분석을 수행한 결과이다.

〈표 2〉 실험 결과

| | 구문분석기 (A) : (부분분석규칙 → 전체분석규칙) | 구문분석기 (B) : (부분분석규칙 + 전체분석규칙) |
|---|---|---|
| 파싱 수행 시간 | 0.13226 sec | 0.20718 sec |
| 파싱 성공 문장 | 201,191 개 | 200,607 개 |
| 레코드 평균 갯수 | 457.9492731 개 | 907.378707 개 |
| 평균 트리 갯수 | 7.013315388 개 | 7.612805039 개 |

전체 문장 570,546 중 약 35% 정도만이 구문분석에 성공했으며 평균 트리 개수는 두 경우 모두 7개 정도만이 만들어졌다. 결과 트리 개수가 이렇게 적은 이유는 구문분석기가 사용하는 규칙이 모두 수동으로 작성되어 있으며 중의성을 해소하고자 많은 튜닝 작업을 거쳤기 때문이다. 구문분석이 실패하는 대부분의 경우는 다양한 한국어 문장을 모두 분석할 수 있는 전체 구문 분석 규칙이 부족하기 때문이었다. 구문분석기 (B)에서 문장 당 만들어진 평균 레코드 개수는 구문분석기 (A)에 비해 약 1.98배 정도 많았고, 파싱 수행시간도 구문 분석기 (A)의 경우가 더 적게 걸렸다.

## 6. 결 론

부분 구문분석기의 처리 대상과 그 범위는 실제 그 부분 구문분석기가 사용될 시스템의 목적에 따라 달라질 수 있다. 본 논문에서는 문법 검사기와 같은 실제 응용 시스템을 위한 부분 구문분석기의 구현에 대해서 소개하고, 그 성능 평가를 위한 실험을 수행하였다. 부분 구문분석 과정은 전체 구문분석 과정을 두 단계로 분리시켜 줌으로써 전체 구문분석의 성능을 향상시킬 수 있었다.

본 논문에서 다루고 있는 부분 구문분석기는 고유명사에 대해서는 현재 한국인 인명만을 다루고 있다. 좀더 유용한 부분 구문분석기가 되기 위해서는 고유명사에 대한 심도 있는 연구가 필요할 것이다.

## 참 고 문 헌


[1] 국어어문규정집.

[2] 김재훈, 부분 구문분석 방법론, 정보처리학회지, 제7권 제6호, pp.83-96, 2000.

[3] 김재훈, 한국어 부분 구문분석의 단위와 그 표지, 한국해양대학교, 컴퓨터공학과, KMU-NLP-TR-2000-006, 2000.

[4] 김홍규 외, 현대국어 기초 말뭉치 개발, 문화공보부, 2002.

[5] 박수호, 권혁철, "확장된 어휘적 중의성 제거 규칙에 따른 부분 문장 분석에 기반한 한국어 문법검사기", 제13회 한글 및 한국어 정보처리 학술대회 발표논문집, pp.516-522, 2001.

[6] 안동언, 기계번역을 위한 한국어 해석에서 형태소로부터 구문요소의 형성에 관한 연구, 한국과학기술원, 전산학과, 석사학위논문, 1987.

[7] 이중영, 신병훈, 이공주, 김지은, 안상규, COM 기반의 다목적 형태소 분석기를 이용한 명사 추출기, 제 1회 형태소 분석기 및 품사태거 평가 워크숍 논문지, pp.167-172, 1999.

[8] Abney, S., "Chunk and dependencies : Bringing processing evidence to bear on syntax," Computational Linguistics and the Foundations of Linguistic Theory, CLSI, 1995.

[9] Abney, S. "Partial Parsing via Finite-State Cascades," J. of Natural Language Engineering, 2(4), pp.337-344. 1996.

[10] Abney, S. Chunk Stylebook, http://sfs.npil.unituebingen.de/~abney/Papers.html#98i, 1996.

[11] At-Mohtar, S. and Chanod, J. P., "Incremental Finite-State Parsing," Proceedings of ANLP '97, Washington, pp.72-79, 1997.

[12] Bourigault, D., "Surface grammatical analysis for the extraction of terminological noun phrases," Proceedings of COLING-92, pp.977-981, 1992.

[13] Brants, T., "Cascaded Markov Models," *Proceedings of EACL-99*, Bergen, Norway, 1999.

[14] Cardie, C. and Pierce, D., "Error-driven pruning of treebank grammars for base noun phrase identification," *Proceedings of COLING-ACL-98*, 1998.

[15] Cardie, C., Ng, V., Pierce, D. and Buckley, C. "Examining the role of statistical and linguistic knowledge sources in a general-knowledge question-answering system," Proceedings of the Sixth Applied Natural Language Processing Conference(ANLP-2000), pp.180-187, 2000.

[16] Church, K., "A stochastic PARTS program and noun phrase parser for unrestricted texts," *Proceedings of ANLP-88*, Austin, Texas, 1988.

[17] Chen K.- H. and Chen H. H., "Extracting noun phrase phrases from large scale texts : Hybrid approach and its automatic evaluation," *Proceedings of ACL-94*, pp.234-241, 1994.

[18] Daelemans, W., Buchholz, S. and Veenstra, J., "Memory-Based Shallow Parsing," Proceedings of CoNLL-99, Bergen, Norway, 1999.

[19] Dagan, I. and Krymolowski, Y., "Compositional partial Parsing by memory-based sequence learning, Data- oriented Parsing," Rens Bod, Remko Scha, and Khalil Sima'an(eds), CSLI publications, 2001.

[20] Hobbs, J., Appelt, D., Bear, J., Israel, D., Kameyama, M., Stickel, M. and Tyson, M., "FASTUS : a cascaded finite -state transducer for extracting information from natural -language text," Finite State Devices for Natural Language

Processing, E. Roche and Y. Schabes, eds., Cambridge M A : MIT Press, 1996.

[21] Hindle, D., User manual for Fidditch, Technical Memorandum, #7590-142, Naval Research Laboratory, 1983.

[22] http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnofftalk/html/office01022003.asp.

[23] INUI, T. and INUI K. "An application of Probabilistic Partial Parsing : Detection of Syntactic-Tag Errors in Treebanks," IPSJ SIGNotes Natural Language Abstract, No.134-003, 1999.

[24] Joshi, A. and Hopely, P., "A parser from antiquity : an early application of finite state transducers to natural language parsing," *Extended Finite State Models of Language*, Kornai, A. eds, Cambridge University Press, pp.6-15, 1999.

[25] Rawshaw, L. A. and Marcus, M. P., "Text chunking using transformation-based learning," Proceedings of the 3rd Workshop.on Very Large Corpora, MIT, pp.82-94, 1995.

[26] Skut, W. and Brants, T., "Chunk tagger-statistical recognition of noun phrases," Proceedings of the ESSLLI Workshop on Automated Acquisition of Syntax and Parsing. Saarbrcken, Germany, 1998.

[27] Skut, W. and Brants, T. "A maximum-entropy partial parser for unrestricted text," *Proceedings of the Sixth Workshop on Very Large Corpora*. Montreal, Canada., 1998a.

[28] Tjong Kim Sang, "Noun phrase representation by system combination," *Proceedings of ANLP-NAACL 2000*, Seattle, Washington, USA, 2000.

[29] Voutilainen, A. and Padro, L., "Developing a hybrid NP parser," Proceedings of ANLP-97, 1997.

[30] Voutilainen, A., "NPtool, a detector of English noun phrases," The Computation and Language E-Print Archive (http://arXiv.org/), cmp-lg/9502010, 1995.

[31] Zhang, T., Damerau, F. and Johnson, D. "Text Chunking based on a Generalization of Winnow," Journal of Machine Learning Research, Vol.2, pp.615-637, Mar., 2002.

**이 공 주**

e-mail : kjlee007@ewha.ac.kr

1992년 서강대학교 전자계산학과(학사)

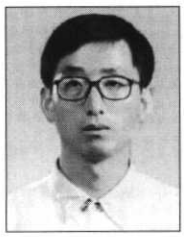
1994년 한국과학기술원 전산학과(공학석사)

1998년 한국과학기술원 전산학과(공학박사)

1998년~2003년 (주)한국마이크로소프트 연구원

2003년~현재 이화여자대학교 컴퓨터학과 전임강사

관심분야 : 자연언어처리, 자연어인터페이스, 기계번역, 정보검색



**김 재 훈**

e-mail : jhoon@mail.hhu.ac.kr

1986년 계명대학교 전자계산학과(학사)

1988년 한국과학기술원 전산학과(공학석사)

1996년 한국과학기술원 전산학과(공학박사)

1988년~1997년 한국전자통신연구원 선임 연구원

1997년~1999년 한국해양대학교 컴퓨터공학과 전임강사

2000년~2002년 한국과학기술원 첨단정보기술연구센터 연구원

2001년~2002년 USC, Information Sciences Institute, 방문연구원

1999년~현재 한국해양대학교 컴퓨터공학과 조교수

관심분야 : 자연언어처리, 한국어 정보처리, 정보검색, 정보추출