KIPS Tr. Software and Data Eng.
Vol.5, No.1 pp.21~28 pISSN: 2287-5905

http://dx.doi.org/10.3745/KTSDE.2016.5.1.21

Three-Phase English Syntactic Analysis for Improving the Parsing Efficiency

Sung-Dong Kim†

ABSTRACT

The performance of an English-Korean machine translation system depends heavily on its English parser. The parser in this paper is a part of the rule-based English-Korean MT system, which includes many syntactic rules and performs the chart-based parsing. The parser generates too many structures due to many syntactic rules, so much time and memory are required. The rule-based parser has difficulty in analyzing and translating the long sentences including the commas because they cause high parsing complexity. In this paper, we propose the 3-phase parsing method with sentence segmentation to efficiently translate the long sentences appearing in usual. Each phase of the syntactic analysis applies its own independent syntactic rules in order to reduce parsing complexity. For the purpose, we classify the syntactic rules into 3 classes and design the 3-phase parsing algorithm. Especially, the syntactic rules in the 3rd class are for the sentence structures composed with commas. We present the automatic rule acquisition method for 3rd class rules from the syntactic analysis of the corpus, with which we aim to continuously improve the coverage of the parsing. The experimental results shows that the proposed 3-phase parsing method is superior to the prior parsing method using only intra-sentence segmentation in terms of the parsing speed/memory efficiency with keeping the translation quality.

Keywords : 3-phase Syntactic Analysis, English-Korean Machine Translation, Intra-Sentence Segmentation, Rule-Based Machine Translation, Automatic Syntactic Rule Acquisition

영어 구문 분석의 효율 개선을 위한 3단계 구문 분석

김 성 동†

요 약

영어 구문 분석기는 영한 기계번역 시스템의 성능에 가장 큰 영향을 미치는 부분이다. 본 논문에서의 영어 구문 분석기는 규칙 기반 영한 기계번역 시스템의 한 부분으로서, 많은 구문 규칙을 구축하고 차트 파싱 기법으로 구문 분석을 수행한다. 구문 규칙의 수가 많기 때문에 구문 분석 과정에서 많은 구조가 생성되는데, 이로 인해 구문 분석 속도가 저하되고 많은 메모리를 필요로 하여 번역의 실용성이 떨어진다. 또한 쉼표를 포함하는 긴 문장들은 구문 분석 복잡도가 매우 높아 구문 분석 시간/공간 효율이 떨어지고 정확한 번역을 생성하기 매우 어렵다. 본 논문에서는 실제 생활에서 나타나는 긴 문장들을 효율적으로 번역하기 위해 문장 분할 방법을 적용한 3단계 구문 분석 방법을 제안한다. 구문 분석의 각 단계는 독립된 구문 규칙들을 적용하여 구문 분석을 수행함으로써 구문 분석의 복잡도를 줄이려 하였다. 이를 위해 구문 규칙을 3가지 부류로 분류하고 이를 이용한 3단계 구문 분석 알고리즘을 고안하였다. 특히 세 번째 부류의 구문 규칙은 쉼표로 구성되는 문장 구조에 대한 규칙으로 구성되는데, 이들 규칙들을 말뭉치의 분석을 통해 획득하는 방법을 제안하여 구문 분석의 적용률을 지속적으로 개선하고자 하였다. 실험을 통해 제안한 방법이 문장 분할만을 적용한 기존 2단계 구문 분석 방법에 비해 유사한 번역 품질을 유지하면서도 시간/공간 효율 면에서 우수함을 확인하였다.

키워드 : 3단계 구문 분석, 영한 기계번역, 문장 분할, 규칙 기반 기계번역, 구문 규칙 자동 획득

※ 본 연구는 한성대학교 교내학술연구비 지원과제임.
† 종신회원 : 한성대학교 컴퓨터공학과 교수
Manuscript Received : October 22, 2015
First Revision : November 30, 2015
Accepted : November 30, 2015
* Corresponding Author : Sung-Dong Kim(sdkim@hansung.ac.kr)

1. 서 론

규칙 기반 영한 기계번역 시스템의 구문 분석기는 많은 규칙을 이용하여 구문 분석을 수행하기 때문에 입력 문장이 길어질수록 구문 분석 시간이 급격히 증가하고, 많은 중간 결과를 생성하여 메모리를 많이 필요로 한다. 영한 기계번역 시스템이 실용적으로 사용되기 위해서는 시간/공간 효율 및 번역 품질의 개선이 필요하며, 영한 기계번역 시스템의 구성 요소 중 구문 분석기가 가장 성능에 영향을 많이 미치기 때문에 구문 분석기의 성능 개선을 위한 많은 연구들이 수행되어 왔다. 그 중에서 본 논문에서는 다단계 구문 분석 방법을 구문 분석기의 성능 개선 방법으로 적용하고자 한다.

주로 2단계 구문 분석 방법이 구문 분석의 효율성 개선 및 말뭉치 구축 등을 위해 제안되어 왔다. [1, 2]에서는 문장 분할(intra-sentence segmentation) 방법을 적용한 2단계 구문 분석 방법을 제시하였다. [3]에서는 한국어 구문 분석 말뭉치 구축을 위한 도구로서 2단계 구문 분석을 적용하였다. 정확한 구문 분석 결과를 부착(tagging)하기 위해 문장을 분할하고 각 분할에 대한 부분 구조를 생성한 후 이들을 결합하여 문장 구조를 생성하는 2단계 구문 분석을 수행하였다. [4]에서는 중국어 구문 분석을 위해 2단계 의존 구문 분석기(dependency parser)를 제안하였다. 여기서는 동사 오른편의 종속성 문제로 인해 종속 트리의 연결성을 보장하지 못하는 문제를 해결하기 위해, 동사 왼편 및 오른편의 종속성을 탐지하는 단계와 동사 오른편의 동사 종속 요소를 결정하는 2단계 분석을 수행한다. [5]에서는 2단계 구문 분석을 통한 자동 번역 장치를 제안하였는데, 동사구를 중심으로 구문 분석을 수행하여 부분 트리를 생성한 후, 다시 부분 트리들에 대해 품사 및 기본 명사구를 인식한 후 전체 문장에 대한 두 번째 구문 분석을 수행하는 방법을 제시하였다. 이 방법은 각 단계에서의 구문 분석 방법이 다르고 부분 트리에 대한 추가의 작업을 필요로 한다. 이와 같이 2단계 구문 분석 방법은 보다 정확한 분석 결과를 얻기 위해 적용되어 왔다.

본 논문에서는 구문 분석의 효율성 향상을 위해 3단계 구문 분석 방법을 제안한다. 제안하는 방법은 입력 문장의 분할 결과로 생성되는 문장 분할에 대해서 2단계의 구문 분석을 통해 각 분할의 구조를 생성하고, 문장 분할의 구문 분석 결과를 합성하기 위한 추가의 구문 분석을 수행하는 3단계로 구성된다. 이를 위해 구문 규칙을 3가지 부류로 분류하였는데, 기존 영어 구문 규칙을 2개의 부류로 분류하고 3단계 구문 분석을 위한 규칙을 말뭉치에 대한 문장 분석을 통해 획득하였다. 구문 분석의 각 단계는 차트 기반의 구문 분석 알고리즘을 동일하게 적용하고 단계별로 다른 구문 규칙 집합을 사용한다.

본 논문은 다음과 같이 구성된다. 2장에서는 규칙 기반의 구문 분석 방법의 2가지 방법을 설명하고 본 논문에서 대상으로 하는 영어 구문 분석기가 사용하는 구문 규칙을 제시한다. 3장에서는 3단계 구문 분석 알고리즘을 설명하면서 영어 구문 규칙의 분류 및 획득 방법을 설명한다. 4장에서는 구문 규칙의 분류 결과와 제안한 방법을 적용하였을 때의 시간/공간 면에서의 효율성 향상의 결과를 제시하고 5장에서 앞으로의 연구 과제를 제시한다.

2. 규칙 기반 구문 분석 방법

대표적인 구문 분석 방법에는 구성 성분 분석(constituency parsing)[5, 6]과 의존 관계 분석(dependency parsing)[7,8] 방법이 있다. 구성 성분 분석은 구 구조 문법(phrase structure grammar)을 이용하여 입력 문장을 구성하는 구(phrase)를 분석하여 문장의 구조를 생성한다. 의존 관계 분석은 의존 관계 문법(dependency grammar)을 이용하여 입력 문장에 있는 단어들 간의 관계를 분석한다.

구성 성분 분석 방법으로는 주로 차트 기반 분석(chart parsing) 방법이 사용된다. 차트 파싱 방법은 구문 분석 과정에서 생성되는 구성 성분들을 모두 차트에 기록하는 과정을 수행하여 입력 문장에 대한 최종 분석 결과를 생성한다. 결과적으로 백트랙킹(backtracking)을 하지 않아도 되므로 효율적인 구문 분석을 가능하다. 그러나 긴 문장을 분석할 경우, 구문 분석 과정에서 생성되어 차트에 기록되는 구성 성분들이 매우 많기 때문에, 시간/공간 면에서 복잡도가 크게 증가한다.

의존 관계 분석 방법으로는 전이 기반(transition-based) 방법과 그래프 기반(graph-based) 분석 방법이 있다. 전이 기반 방법은 전통적인 이동/축약(shift/reduce) 분석을 수행하는 규칙 기반 방법이며, 그래프 기반 방법은 의존 그래프에 대한 학습을 통해 단어 간의 의존 관계를 분석하는 통계적 방법이다. 의존 관계 분석 방법은 문장 의미에 대한 이해를 보다 잘 할 수 있으므로 최근에는 문장의 의미를 파악한 후 번역을 수행하려는 연구가 수행되었다. [9]에서는 간결하고 이해하기 쉬운 전체 문장의 의미 표현 방법으로 추상 의미 표현(abstract meaning representation)에 대한 프로젝트를 소개하였다. 그리고 [10]에서는 그래프 기반 방법을 확장하여 문장을 분석하여 추상 의미 표현을 생성하는 구문 분석 방법을 제안하였다. 이후 [11]에서는 [10]의 방법을 개선하기 위해 의존 관계 분석을 수행하여 의존 관계 분석 트리를 생성한 후 전이 기반 알고리즘을 이용하여 추상 의미 표현을 생성하는 2단계 방안을 제시하였다. 이들 방법들은 입력 문장으로부터 구문 구조 트리를 생성하고 이를 추상 의미 표현으로 변환한 후 번역을 수행한다. 최근에 이러한 의미적 구문 분석에 대한 연구가 활발하게 수행되고 있으며 [12]에서 현재 가장 좋은 성능을 보이는 추상 의미 표현에 대한 파서를 제시하고 있다. 그러나 의미 기반의 구문 분석을 위해서는 많은 정보를 구축해야 하는 노력이 필요한데, 현재 상황에서 유용한 영한 기계번역 시스템 개발을 위해서는 기존의 규칙 기반 방법을 적용하는 것이 현실적이며 따라서 본 논문에서는 규칙 기반 기계번역 시스템을 대상으로 한다.

Fig. 1은 구 구조 문법의 예와 이를 이용한 구성 성분 분석 트리(constituency tree)를 보여준다. 그리고 Fig. 2는 의존 관계 문법의 예와 의존 관계 분석 트리(dependency tree)를 보여준다. 그림에서 알 수 있듯이 구성 성분 분석 결과를 통해 입력 문장의 구조를 파악하는 것이 용이한 반면, 의존 관계 분석 결과는 문장에서 사용된 단어 간의 의존 관계, 즉 단어나 구의 역할을 파악하는 것이 용이하다. 의존 관계 분석 방법이 속도 면에서 유리하고 문장 의미 파악이 용이한 장점이 있으나, 단어 간의 의존 관계를 모두 나타내는 것이 어렵고 문법의 이해도가 낮아 의존 관계 분석을 위한 규칙을 구축하는 것이 쉽지 않다. 이에 비해 구성 성분 분석을 위한 규칙을 구축하는 것은 상대적으로 용이할 뿐만 아니라, 규칙의 수정 또한 쉽게 할 수 있기 때문에 본 논문에서는 구성 성분 분석 방법을 수행하는 영어 구문 분석기를 대상으로 한다. 그런데, 번역을 위해서는 단어나 구의 역할을 파악해야 하므로, 구성 성분 분석 결과를 생성할 때, 그것들의 역할을 분석할 수 있도록 의존 관계가 보강된(augmented) 구 구조 문법을 이용하여 차트 파싱을 수행한다. 즉 구 구조 문법에 구조간의 관계와 헤드(head) 정보를 추가하였다. 본 논문에서의 구문 분석기가 이용하는 보강된 구 구조 문법의 예는 Fig. 3과 같다.

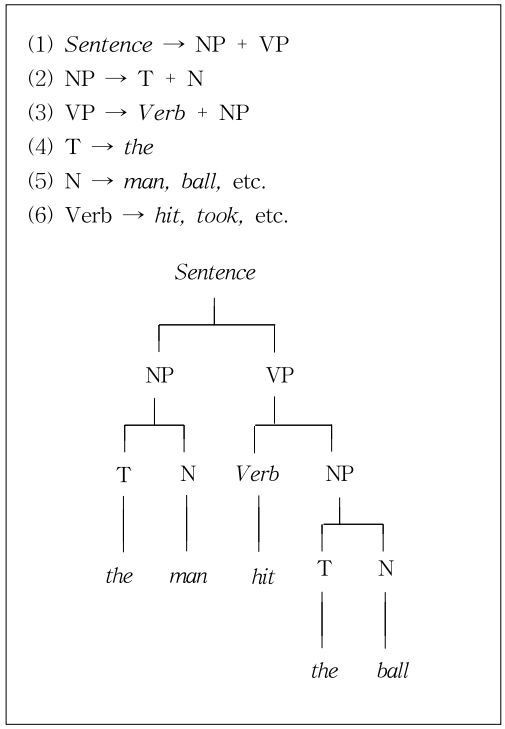

Fig. 1. Examples of Phrase Structure Grammar / Constituency Structure (cited from [13])

Fig. 3의 구 구조 문법은 문맥 자유(context-free) 형식으로 기술되었으며 규칙 왼편(left-had side: LHS)의 문법 기호는 괄호 안에 상위 구성 성분의 구조를 생성하기 위한 방법을 포함하는 행동 기술 영역(action description field)이 있고, 규칙 오른편(right-hand side: RHS)의 문법 기호는 괄호 안에 규칙 적용 가능성 검사를 위한 조건을 기술하는 조건 영역(condition field)을 포함한다. 즉 Fig. 3에서 제시한 문법은 NP(noun phrase)와 VP(verbal phrase)가 결합하여 SENT(sentence)를 구성함을 의미하며, LHS의 행동 기술 영역은 SENT의 중심(head)은 VP(%VP)이고 NP는 주어(SUBJ->NP)가 됨을 의미한다.

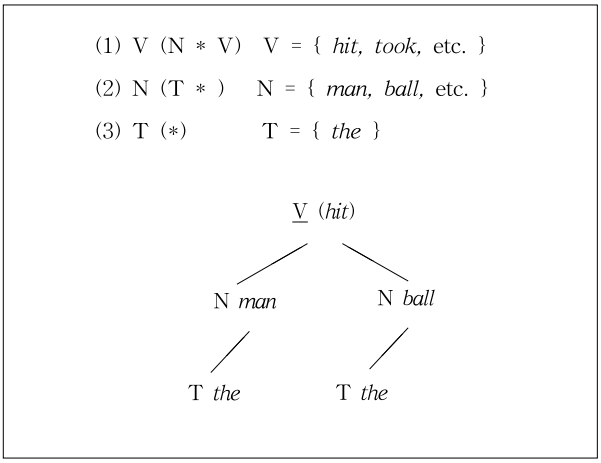

Fig. 2. Examples of Dependency Grammar / Dependency Relation Structure (cited from [13])

```
SENT(AgreeTest NP VP, CaseTest NP SUBJP,
    %VP, cat='SENT, func='DECL, SUBJ->NP)
→ NP( )
    VP( SUBJ->0 )
```

Fig. 3. Example of Phrase Structure Rule Augmented with Dependency Relation and Head Information

3. 3단계 영어 구문 분석 알고리즘과 영어 구문 규칙의 분류

보다 효율적인 구문 분석과 정확한 구문 구조 선정을 위해 기존의 전체 구문 분석(full parsing) 대신 문장의 일부를 분석하는 부분 분석(partial parsing) 방법이 사용되었다[14]. 또한 단순히 구성 성분만을 찾고 내부 구조를 파악하지 않는 얕은 파싱(shallow parsing, light parsing) 방법도 제시되었는데, [15]에서는 이를 의존 관계 구문 분석에 사용함으로써 문장의 구조를 파악할 수 있도록 하였다. 전체 구문 분석의 비효율성 때문에 부분 파싱 결과를 이용하여 전체 구문 분석 결과를 생성하는 방법이 널리 이용되고 있다. [16]은 전체 구문 분석의 비효율성을 극복하기 위해 부분 분석을 이용하는 방법을 설명하고 있다. 이들 방법들은 부분 분석과 부분 분석 결과 합성을 위한 2단계로 구성된다. 최근

에는 실시간 음성 번역(realtime speech translation)에서도 분할을 이용하여 보다 좋은 번역을 생성하려는 연구[17]가 진행되는 등, 분할은 보다 정확하고 빠른 번역을 얻기 위한 방법으로 널리 이용되고 있다.

본 논문에서는 부분 분석을 2단계로 수행한 후 결과 합성을 위한 추가의 단계를 수행하는 3단계 구문 분석 방법을 제안한다. Fig. 4는 3단계 구문 분석 방법이 적용된 구문 분석 과정을 보여준다. 입력 문장은 쉼표를 기준으로 복수의 문장 분할(segment)로 나뉘고, 각 분할은 독립적으로 분석된다. 각 분할은 2단계 분석 과정을 통해 분할의 구문 구조가 분석되는데, 1단계 분석을 통해 각 분할의 1차 구조가 생성되고 이들 1차 구조를 대상으로 2단계 구문 분석이 수행된다. 마지막으로 쉼표에 의해 분리된 각 분할의 구문 구조를 결합하여 입력 문장의 구문 구조를 생성하는 3번째 분석 과정을 통해 입력 문장의 구조가 생성된다.

일반적으로 긴 문장은 쉼표를 포함하게 되며, 본 논문에서는 쉼표에 의해 문장 분할을 수행한다. 쉼표는 분리(separation) 또는 나열(list)의 용도로 사용되는데 본 논문에서는 분리 용도로 사용된 쉼표만을 포함한 문장을 대상으로 하였다. 쉼표에 의해 분리된 분할이라 하더라도 많은 단어를 포함하는 경우, 구문 분석의 복잡도가 높아지고 많은 구조가 생성되므로 정확한 구조 선정이 어렵고 구문 분석 속도가 저하된다. 이러한 문제를 해결하기 위해 각 분할에 대해서 2단계 분석을 수행하며, 각각 다른 규칙의 집합을 적용하여 구문 분석 과정에서 생성되는 구조의 수를 줄이고 속도를 개선하고자 하였다. 각 분할의 결과를 합성하여 전체 문장의 구조를 생성하는 3단계 구문 분석에서는 쉼표를 포함하는 구문 규칙을 적용해야 하는데, 쉼표가 연결하는 구성 성분은 매우 다양하기 때문에 이러한 구조를 표현하는 구문 규칙을 모두 기술하는 것은 매우 어려울 뿐만 아니라 이들 규칙들이 구문 규칙 집합에 포함되면 규칙의 수가 매우 많아져 구문 분석의 복잡도를 크게 높일 수 있다. 본 논문에서는 이러한 문제를 해결하기 위해 쉼표로 구성된 문장들을 쉼표에 의한 문장 분할을 통해 구문 분석 한 후, 쉼표와 결합하여 문장을 만들 수 있는 경우를 수집하여 규칙화하였으며 3단계 구문 분석에서는 이들 규칙만을 사용한다. 따라서 3단계 구문 분석에서는 앞의 두 단계에서 사용한 규칙은 사용하지 않으며, 분할의 구문 구조 결과들과 쉼표가 어떻게 결합하여 문장 구조를 형성하는지에 대한 규칙들만을 사용하기 때문에 분석의 복잡도를 줄이면서 쉼표에 의해 결합될 수 있는 다양한 형식을 구문 규칙으로 포함할 수 있게 된다.

첫 단계 구문 분석은 문장 분할의 단어와 품사 정보를 이용하여 보다 상위 구성 성분인 구(phrase)를 찾는 과정이다. 따라서 기존 구문 규칙 중 특정 단어(word)와 품사 정보(part-of-speech: POS)가 포함된 구문 규칙을 추출한다. 즉 문맥 자유 형식으로 기술된 구문 규칙의 오른편(RHS)에 품사 기호(POS symbol)나 구 기호(phrase symbol) 중에서 특정 단어를 포함하는 구문 규칙을 추출한다. Equation (1)은 1단계 구문 규칙($Rules_{Class1}$)을 정의한다.

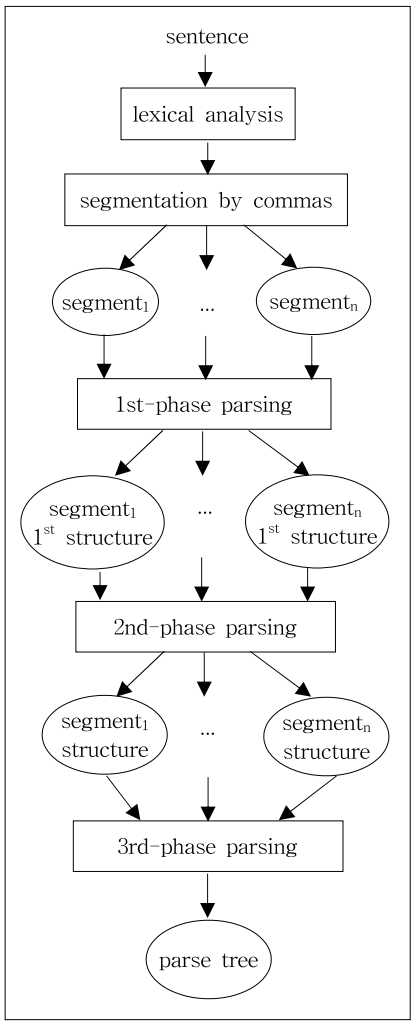

Fig. 4. Steps of 3-phase Parsing

$$Rules_{Class1} = \{r | r_{RHS_n} \in Symbols_{POS} \text{ or } w \in r_{RHS_n}(condition\_field)\}, \quad where\ Symbols_{POS} = \{\text{DET, NOUN, PRON, VERB, ADJ, ADV, PREP}\} \tag{1}$$

Equation (1)에서 조건 영역(condition field)에 특정 단어가 기술되었다면 해당 규칙은 단어에 종속적인 규칙으로 판단할 수 있다.

두 번째 단계의 구문 분석은 첫 단계 분석에서 찾은 구의 구성 성분을 결합하여 보다 상위의 구나 절(clause)에 대한 구조를 생성하는 단계이다. 전체 구문 규칙($AllRules$)에서 $Rules_{Class1}$을 제외한 규칙들 중 RHS가 쉼표(comma)를 포함하지 않는 규칙들로 구성된다. Equation (2)는 2단계 구문 규칙($Rules_{Class2}$)을 정의한다. 두 번째 단계가 종료되면 입력 문장 각각의 문장 분할에 대한 구조 분석이 완료된다.

$$Rules_{Class2} = \{r | r \in \text{AllRules} - \text{Rules}_{\text{Class1}} \text{ and } r_{RHS_n} \neq comma\} \tag{2}$$

세 번째 단계의 구문 분석은 문장 분할의 구문 분석 결과들을 합성하여 하나의 문장 구조를 생성하는 과정이다. 문장 분석을 통해 수집한 규칙들과 기존 규칙(*AllRules*)에서 쉼표를 포함하는 규칙들로 3단계 구문 규칙을 구성한다. 단 규칙의 RHS에서 쉼표에 해당하는 기호를 제외한다. 예를 들어 [SENT -> SUBCL COMMA SENT]라는 규칙은 [SENT -> SUBCL SENT]라는 형식으로 3단계 구문 규칙 집합에 포함된다.

$$Rules_{Class3} = \{r | r \in AllRules - Rules_{Class1} \text{ and } r_{RHS_n} = comma\} \cup \text{GeneratedRules} \quad (3)$$

Equation (3)에서 *GeneratedRules*는 말뭉치를 구문 분석하여 추출한 규칙의 집합을 의미하는데, 각 규칙의 LHS는 SENT이고 RHS는 Equation (4)와 같이 구성된다. 그리고 Fig. 5는 문장 분할의 구문 분석 결과를 이용하여 규칙을 생성하는 알고리즘을 보여준다.

$$r_{RHS} = \prod_{i=1}^{n} SyntacticCategory_i \quad (4)$$

```
Algorithm generateRules( SegmentParsingResult )
{
 GeneratedRules = { };
 foreach aResult from SegmentParsingResult
 {
 RHSs = r_RHS = ∏_{i=1}^{n} SyntacticCategory_i from aResult;
 new_Rule = [SENT -> RHSs] ;
 GeneratedRules = GeneratedRules ∪ new_Rule;
 }   // foreach
 return GeneratedRules ;
}
```

Fig. 5. Algorithm for Generating 3-phase Syntactic Rules using Segment Parsing Results

2단계 구문 분석 후에 각 문장 분할의 구조를 나타내는 파싱 트리(parsing tree)가 생성되는데, 이 때 각 파싱 트리가 나타내는 구문 범주가 Equation (4)에서 $SyntacticCategory_i$로 표현된다. 즉 $SyntacticCategory_i$는 i번째 문장 분할의 구문 분석 결과로 생성되는 파싱 트리의 구문 범주를 나타낸다. 이 때 각 문장 분할에 대해서 하나 이상의 파싱 트리가 생성될 수 있다. 따라서 구문 범주의 곱집합(product set)을 RHS로 하는 규칙들을 생성하여 *GeneratedRules*를 구축한다. 이후 *GeneratedRules*에 포함된 각각의 규칙들의 LHS에 구문 분석 결과 생성을 위한 행동(action)을 기술하고 필요한 경우 RHS의 기호들에 조건(condition)을 기술한다.[2] 그리고 구문 규칙 컴파일러를 이용하여 영한 기계번역 시스템의 규칙 적용 모듈로 변환한다[18]. 3단계 구문 규칙으로 분할 구조를 결합하여 하나의 문장 구조를 생성하는데, 만약 기존 구문 규칙으로 분할 구조들을 결합하지 못하는 경우에는 새로운 구문 규칙을 생성하고 이를 3단계 구문 규칙에 포함하도록 한다. 따라서 지속적으로 3단계 구문 규칙을 확장할 수 있다. Fig. 6은 문장 "Asiana Airlines, Korea's No.2 carrier, plans to increase its fleet to 88 aircraft by 2015, a company executive said yesterday."을 3단계 구문 분석을 적용하여 분석하는 과정과 그 결과를 보여준다.[3]

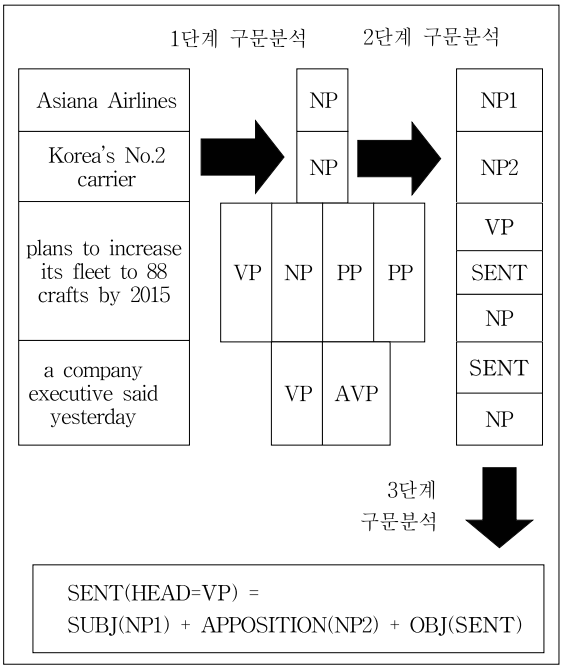

Fig. 6. Examples of Steps of 3-phase Syntactic Analysis

입력 문장은 3개의 쉼표에 의해 4개의 분할로 분리되고, 각 분할은 1, 2단계 분석을 통해 각각 NP, NP, VP/SENT/NP, SENT/NP로 분석된다.[4] 이들 4개의 구문 분석 결과를 3단계 구문 분석 과정에서 결합하여 다음과 같은 문장 구조를 생성한다: 3번 분할의 분석 결과 중 VP가 중심(HEAD)이 되고 NP1이 주어(SUBJ), NP2가 동격(APPOSITION), 그리고 4번 분할의 SENT가 목적어(OBJ)가 된다.

4. 실 험

본 절에서는 영어 구문 규칙의 분류 결과를 제시하고 3단계 구문 분석 방법과 기존의 문장 분할을 적용한 2단계 구

2) Fig. 3에 제시된 구 구조 문법의 형식에서 행동 기술 영역과 조건 영역을 기술해야 하는데, 이는 전문가의 노력을 필요로 하는 작업이다.

3) 최종 구문 분석 결과를 보여주기 위해 1, 2번 분할의 2단계 구문 분석 결과를 제시할 때 NP1, NP2로 구분하여 표기하였다.

4) 3, 4번 분할에 대해서 생성되는 복수의 구문 범주를 '/'로 분리하여 제시하였다. 즉 3번째 분할은 VP, SENT, NP로 4번째 분할은 SENT와 NP로 분석되었다. 그리고 최종 분석 결과에서 선택되는 VP와 SENT는 굵은 글씨로 그림에서 표현하였다.

문 분석 방법[19]을 번역 시간/공간 면에서 비교하였다.

Equation (1)을 이용하여 588개의 기존 규칙에서 쉼표를 포함하는 110개를 제외한 나머지 규칙들을 2개의 부류로 분류하였다. 그리고 Korea Herald 신문의 기술(technology) 영역의 기사에서 추출한 1,400 문장의 구문 분석을 통해 535개의 규칙을 생성하고 쉼표를 포함하는 규칙과 함께 3단계 규칙으로 분류하였다. 구문 규칙의 3가지 부류에 대한 통계는 Table 1과 같으며 Class3의 규칙은 지속적인 말뭉치 분석을 통해 그 수를 증가시킬 수 있다.

Table 1. Syntactic Rules Classification

	# of Rules
Class1 Rule Set	154
Class2 Rule Set	324
Class3 Rule Set	645

Table 2는 3가지 부류에 속하는 구문 규칙의 예를 보여준다.[5] 규칙 오른편(RHS)에 품사 기호를 포함하는 규칙들이 Class1에 속하고 Class2의 규칙은 오른편에 구와 절을 나타내는 기호만 포함한다. Class3에는 기존 규칙들 중 규칙 오른편에 쉼표(COMMA)를 포함하는 규칙들과 Fig. 5의 알고리즘에 의해 수집한 규칙들이 속하며 이 때 COMMA는 제외하고 기술된다.[6]

Table 2. Examples of Syntactic Rules

	Examples
Class1 Rule Set	1. NP(%NP) -> DET() NP(SINGULAR=1) 2. PP(%NP, cat='PP) -> PREP() NP(PLUR=1, OBJP=1)
Class 2 Rule Set	1. NP(%NP) -> AJP(RESTRIC=0) NP(DETS=0) 2. SENT(%VP, cat='SENT) -> NP(SUBJP=1) VP(SIMPLE=1) 3. SENT(ADVCL->SUBCL) -> SUBCL() SENT(func!='IMPR)
Class 3 Rule Set	1. SENT(ADVCL->SUBCL) -> SUBCL() SENT(func!='IMPR) 2. SENT(%VP, SUBJ->NP1, OBJ->SENT, NP1.APP->NP2) -> NP1() NP2() VP() SENT()

Table 3A와 3B는 기존의 2단계 구문 분석 방법과 본 논문에서의 3단계 구문 분석 방법을 두 가지 문장 집합을 이용하여 번역 시간과 메모리 사용량을 비교한 결과를 보여준다. Table에서 Imp.는 제안한 방법(3-phase)이 기존 방법(2-phase)에 비해 어느 정도 개선되었는지를 나타내며 Equation (5)로 계산된다.

Table 3A. Comparison of Translation Time/Memory Used by Different Parsing Methods

	Test Set 1 (58 sentences)		
Average Sentence Length	22.2 word / sentence		
Parsing Methods	2-phase	3-phase	Imp. (%)
Ave. Translation Time (sec)	0.283	0.267	5.7%
Ave. Memory Usage (KB)	732.6	702	4.2%

Table 3B. Comparison of Translation Time/Memory Used by Different Parsing Methods

	Test Set 2 (35 sentences)		
Average Sentence Length	30.3 word / sentence		
Parsing Methods	2-phase	3-phase	Imp. (%)
Ave. Translation Time (sec)	0.269	0.223	17.1%
Ave. Memory Usage (KB)	717	698.9	2.5%

$$Imp. = \frac{Time/Memory_{2-phase} - Time/Memory_{3-phase}}{Time/Memory_{2-phase}} \times 100 \quad (5)$$

문장 집합 1, 2 모두에 대해서 제안한 방법이 더 빠른 번역 속도를 보였다. 또한 문장 길이가 길어질수록 3단계 구문 분석 방법의 속도가 기존 방법에 비해서 더 많이 개선됨을 확인하였다. 또한 평균 메모리 사용량 면에서도 3단계 구문 분석 방법이 더 적은 양을 소요하였다. 결과적으로 제안한 방법은 긴 문장의 번역 속도 개선에 효과적이며 더 적은 메모리를 필요로 하므로 실용적 기계번역 시스템의 구현에 기여할 수 있을 것으로 판단된다.[7]

Table 3에서 사용한 93 문장에 대한 번역 결과를 비교하였다. 번역을 생성한 경우에는 2가지 방법이 거의 유사한 번역문을 생성하였으나, 문장 집합 2에 속하는 일부 문장들에 대

5) 규칙의 조건 및 행동을 나타내는 부분의 일부는 생략하고 기술하였다.

6) Class3 Rule Set의 규칙 1은 Class2의 규칙 3과 같으나, 기존 규칙에서 [SUBCL COMMA SENT -> SENT]로 기술된 것으로서, "COMMA"를 제외하고 기술한 것이다.

7) 20단어 정도의 문장보다 30단어 정도의 문장에 대해서 더 빠른 번역 속도를 보였는데, 후자의 경우 쉼표로 분리되는 분할이 많아 부분 분석의 단위가 더 작을 수 있기 때문인 것으로 판단된다.

해서 2단계 구문 분석은 문장 구조를 생성하지 못하거나 부자연스러운 번역을 생성하는 경우가 있었다. 이는 쉼표로 분리되는 문장 분할들을 결합하는 구문 규칙이 없기 때문이며, 제안한 방법은 구문 규칙을 말뭉치의 분석을 통해 지속적으로 획득할 수 있으므로 구문 규칙의 적용률을 높이는 데 기여할 수 있다. 예를 들어, Fig. 6에서 제시한 문장 “Asiana Airlines, Korea’s No.2 carrier, plans to increase its fleet to 88 aircraft by 2015, a company executive said yesterday.”을 2단계 구문 분석 방법에 의해서는 “Asiana Airlines, 2번 운반인, 2015 옆에서 88 항공기로의 그것의 함대를 증가시키는 계획, 회사 행정부는 어제 말했다.”라는 번역이 생성된 반면, 제안한 방법에 의해서는 “2번 운반인 Asiana Airlines는 2015 옆에서 88 항공기로의 그것의 함대를 증가시키려고 계획한다고 회사 행정부가 어제 말했다.”라는 번역을 생성하였다. 기존 구문 규칙에 없었던 [SENT -> NP1 NP2 VP SENT]라는 구문 규칙을 말뭉치를 통해 획득하였고 3단계 구문 분석 과정에서 이 규칙을 적용하여 정확한 문장 구조를 생성하고 보다 자연스러운 번역을 생성할 수 있었다.

5. 결 론

본 논문에서는 영어 구문 분석의 효율성 개선을 위해 각 단계별로 서로 다른 구문 규칙을 적용하는 3단계 구문 분석 방법을 제안하였다. 기존 구문 규칙을 2개의 부류로 분류하였으며 3번째 구문 분석 단계에서 적용할 규칙은 말뭉치의 구문 분석을 통해서 수집하였다. 제안한 구문 분석 방법은 먼저 입력 문장을 쉼표를 기준으로 나누어 문장 분할을 생성하고 각각의 문장 분할을 2단계의 구문 분석을 적용한 후, 문장 분할의 결과 합성을 위한 3번째 단계의 구문 분석을 수행한다. 1단계에서는 품사 기호만을 포함하는 규칙을 적용하고, 2단계에서는 구 기호를 포함하는 규칙을 적용하여 각 단계별 구문 분석의 복잡도를 줄이고자 하였다. 3단계 구문 분석은 쉼표로 분리된 문장 분할을 결합하는 단계로서, 쉼표를 포함하는 모든 규칙을 3단계에서만 적용하도록 하여 구문 분석의 복잡도를 줄일 수 있었다. 그리고 쉼표가 다양한 형식으로 문장 분할을 분리하기 때문에 쉼표에 의해 문장 구조를 구성하는 규칙을 사람이 모두 구축하기 보다는 말뭉치를 통해서 얻고자 하였다. 쉼표를 포함하는 다양한 문장 구조에 대한 규칙을 지속적으로 획득함으로써 구문 분석의 적용률(coverage)을 개선할 수 있다.

실험 결과는 문장이 길어질수록 제안한 3단계 구문 부석 방법이 속도 개선 및 메모리 사용량을 줄이는데 효과적임을 보여주었다. 또한 말뭉치 분석을 통해 획득한 3단계 구문 규칙은 기존 사람이 구축한 규칙에 비해 적용률이 높아서 기존 방법으로 분석하지 못했던 긴 문장도 분석할 수 있어 보다 높은 번역률에 기여할 수 있을 것으로 판단된다. 즉 본 논문에서 제안한 3단계 구문 분석 방법은 실제로 나타나는 긴 문장에 대해서 적절한 번역 속도를 보장할 수 있는 방법이라 판단된다. 또한 구문 규칙의 분류 및 획득 방법은 앞으로 지속적인 구문 규칙의 확장에 기여할 것으로 기대한다.

본 논문에서 쉼표는 문장의 구성 요소들을 분리하는 역할을 하는 것으로 간주하고 나열하는 역할을 하는 쉼표를 포함하는 문장은 제외하였다. 논문에서 제안한 방법을 실용화하기 위해서는 구문 분석 이전에 쉼표의 역할을 판단하고 처리하는 방법에 대한 연구가 필요하다. 그리고 구문 분석 복잡도를 줄이면서 구문 분석 정확성 개선을 위한 방법에 대한 연구는 실용적 영한 번역 시스템을 위해 필수적이라 할 수 있다. 또한 구문 분석 속도를 최대화하기 위해 입력 문장에 따라 2단계 및 3단계 분석을 선택적으로 적용할 수 있어야 한다. 이러한 연구를 통해 실제 생활에서 나타나는 긴 문장들을 보다 정확하고 빠르게 분석, 번역할 수 있을 것으로 기대한다.

References

[1] Sung-Dong Kim, Byoung-Tak Zhang, and Yung Taek Kim, “Learning-based Intrasentence Segmentation for Efficient Parsing of Long Sentences,” *Journal of Machine Translation*, Vol.16, No.3, pp.151-174, 2001.

[2] Sung-Dong Kim, “Intra-Sentence Segmentation using Maximum Entropy Model for Efficient Parsing of English Sentences,” *Journal of Korean Institute of Information Science and Engineering*, Vol.32, No.5, 2005.

[3] Hye-Kyum Kim, Kyung-Mi Park, Yeo-Chan Yoon, Hae-Chang Rim, and So-Young Park, “Tree Tagging Tool using Two-phrase Parsing,” *Proceedings of the 17th Annual Conference on Human & Cognitive Language Technology (HCLT 2005)*, 2005.

[4] M. Jin, M.-Y. Kim, and J.-H. Lee, “Two-Phase Shift-Reduce Deterministic De pendency Parser of Chinese,” in *Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP)*, 2005.

[5] Joseph Turial, “Constituent Parsing By Classification,” PhD dissertation, Computer Science Department, Sept., 2007.

[6] Xiao Chen, “Discriminative Constituent Parsing with Localized Features,” PhD thesis, City University of Hong Kong, 2012.

[7] J. Nivre and M. Scholz, “Deterministic dependency parsing of English text,” in *Proceedings of the 20th International Conference on Computational Linguistics*, pp.64-70, Geneva, Switzerland, 2004.

[8] A. Michael and A. Covington, “Fundamental Algorithm for Dependency Parsing,” in *Proceedings of the 39th Annual ACM Southeast Conference, ed.* John A. Miller and Jeffrey W. Smith, pp.95-102, 2001.

[9] L. Banarescu et. al., “Abstract meaning representation for sembanking,” in *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pp.178-186, 2013.

[10] J. Flanigan et. al., “A discriminative graph-based parser for the abstract meaning representation,” in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp.1426-1436, 2014.

[11] C. Wang et. al., “A transition-based algorithm for amr parsing,” in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics*, pp.366-375, 2015.

[12] M. Pust et. al., “Using Syntax-Based Machine Translation to Parse English into Abstract Meaning Representation,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp.1143-1154, 2015.

[13] An International Handbook of Contemporary Research, Edited by V. Agel, L.M. Eichinger, H.-W. Eroms, P. Hellwig, H.-J. Heringer, H. Lobin. Volume II. pp. 1081-1108. Mouton: 2006.

[14] S. Abney, “Part-of-Speech Tagging and Partial Parsing,” *Corpus-Based Methods in Language and Speech*, pp.118-136, 1996.

[15] B. Srinivas, “A lightweight dependency analyzer for partial parsing,” *Natural Language Engineering*, Vol.6, No.2, pp. 113-138, 2000.

[16] H. Faili, “From Partial toward Full Parsing,” *Proceedings of the International Conference on Recent Advances on Natural Language Processing 2009*, pp.71-75, 2009.

[17] Y. Oda et. al., “Optimizing Segmentation Strategies for Simultaneous Speech Recognition,” *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp.551-556, 2014.

[18] Sung-Dong Kim, “English Syntactic Rule Management System for Rule-Based English-Korean Machine Translation System,” *KIISE(Korean Institute of Information Science and Engineering) Transactions on Computing Practice*, Vol.20, No.7, pp.398-407, 2014.

[19] Sung-Dong Kim, “Intra-sentence Segmentation using Finite Automata for Efficient English Syntactic Analysis,” *KIISE(Korean Institute of Information Science and Engineering) Transactions on Computing Practices*, Vol.19, No.4, pp.186-193, 2013.

김 성 동

e-mail : sdkim@hansung.ac.kr

1991년 서울대학교 컴퓨터공학과(학사)

1993년 서울대학교 컴퓨터공학과(석사)

1999년 서울대학교 컴퓨터공학과(박사)

2001년~현 재 한성대학교 컴퓨터공학과 교수

관심분야 : Machine Translation, Natural Language Processing, Machine Learning, Data Mining