격 관계와 상호정보를 이용한 한국어 의존 파서

정석원°, 박의규, 나동열, *윤준태
연세대학교 전산학과
*다음소프트
{nanton,ekpark,dyra}@magics.yonsei.ac.kr, *jtyoon@daumsoft.com

A Study on Korean Dependency Parser Using Case Relation and Mutual Information

Seok-Won Jung, Eui-Kyu Park, Dong-Yul Ra, *Jun-Tae Yoon
Dept. of Computer Science, Yonsei University
*Daumsoft

요 약

본 논문은 의존 문법에 기반한 한국어의 구문 분석 시스템을 제안한다. 일반적으로 올바른 구문 구조를 얻기 위해서 많은 가능한 구문 구조를 생성하고 이 중에서 가장 좋은 것을 선택하는 방법을 사용한다. 이를 위하여 가능한 모든 구문 분석 구조를 생성하는 기법을 제안하였다. 이것은 모든 가능한 구문 구조에 관한 정보를 응축한 자료 구조를 구축한 다음 여기에서 구문 트리를 하나씩 추출하도록 하였다. 이 과정에서 의존 문법이 만족하여야 하는 모든 기본적인 제약 조건을 만족하는 트리 만이 효과적으로 추출되는 기법을 제안하였다. 그 결과 생성되는 트리의 수를 줄이게 되어 효율적인 구문 분석을 달성할 수 있게 되었다. 추출된 많은 트리 중에서 하나를 선택하는 작업에서 상호 정보가 이용되었다. 본 논문에서는 이러한 상호 정보를 구문 분석 중의성 해소에 효과적으로 사용하는 기법을 제시하였다. 제안된 기법의 타당성을 입증하기 위하여 구문 분석 시스템을 개발하고 여러 문장에 대한 분석을 실험하였다.

1. 서론

한국어는 비교적 어순이 자유롭고 격 조사 및 어미의 사용이 매우 발전되어 있는 언어이다. 또한 한국어는 문맥으로 파악할 수 있으면 주어나 목적어 등과 같은 필수적인 문장 요소까지도 생략할 수 있다[5]. 이러한 특징으로 인해 구 구조 문법(phrase structure grammar)으로 한국어를 표현하면 생성 규칙의 수가 지나치게 많아지므로, 구 구조 문법은 한국어 통사 구조의 분석에는 적합하지 않다고 주장하는 사람이 많다[12]. 따라서, 한국어 구문 분석에서는 의존 문법을 많이 이용한다.

의존 문법(dependency grammar)은 단어(word) 사이의 의존 관계에 중심을 두는 문법이다[2,3]. 주어진 문장 안의 단어 사이의 의존 관계(binary relation)를 파악하는 작업이 의존 문법에 의한 언어 분석의 중요한 작업이다[1]. 의존 관계의 파악은 한국어가 가진 수식 관계(의존 관계)의 특성에 기반을 두고 있다.

어순의 자유성, 생략 등 한국어의 특성을 잘 처리할 수 있다고 생각되어 의존 문법은 지금까지 한국어의 분석에 가장 많이 이용되어 왔다[7]. 본 논문에서도 이러한 점을 감안하여 의존 문법에 기반한 한국어 구문 분석 기법을 연구하고자 한다. 본 연구의 특징은 구문 분석 과정에서 발생하는 모든 구조적인 중의성에 대하여 이의 해소 기법을 제시하는 데 있다. 이를 위하여 격 정보와 대량의 말뭉치에서 추출한 통계 정보인 상호정보를 이용하고자 한다[8]. 지금까지의 다른 연구에서는 문장이 가지고 있는 모든 구조적인 중의성을 해소하겠다는 목표를 가진 것이 드물었고 이를 가진 연구에서도 효과적인 중의성 해소 기법을 제시하지 못하였다.

따라서, 본 연구는 주어진 문장에 대하여 모든 가능한 의존 관계를 나타내는 구문 구조 즉 의존 구조 트리를 생성하는 기법을 제안한다. 또한, 이 들 중에서 가장 좋은 하나를 상호정보에 기반하여 추출하는 기법을 제안한다. 제안된 기법의 타당성을 입증하기 위하여 구문 분석 시스템을 개발하고 많은 문장에 대하여 실험을 하였다.

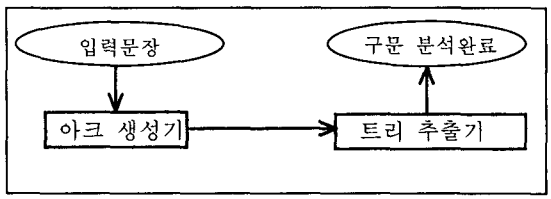
2. 구문 분석

2.1 격 관계를 이용한 구문 분석

의존 문법에 기반을 둔 한국어 구문 분석기를 구현하기

위해서는 먼저, 한국어 의존 규칙을 만들어야 한다[12]. 즉, 한국어를 문법적 범주에 따라서 지배소와 의존소로 나누고, 그들 사이에 의존 관계를 설정해 주어야 한다.

본 논문에서 제안한 구문 분석기는 어절들 사이의 의존 관계만을 분석한다. 그런데, 보통 한 어절은 둘 이상의 형태소들로 구성되어 있기 때문에 지배소 역할을 할 때와 의존소 역할을 할 때 영향을 주는 형태소가 다르게 된다. 즉, 지배소 역할을 할 때는 그 어절의 첫 번째 형태소(어간)가 그 어절의 특성을 지배하고, 의존소 역할을 할 때는 마지막 형태소(어미)가 그 어절의 특성을 지배한다. 따라서 둘 이상의 형태소로 구성된 어절의 경우, 지배소로 쓰일 때는 첫 번째 형태소의 문법적 범주를 따르고, 의존소로 쓰일 때는 마지막 형태소의 문법적 범주를 따른다. 본 논문에서는 마지막 형태소를 어절의 성격을 밝히는 수단(가능한 격 정보의 결정)으로 이용한다.



[그림 1] 구문 분석 시스템 구성도

본 논문에서 제안하는 구문 분석 시스템은 [그림 1]과 같이 크게 두 부분으로 나뉘어진다. 아크 생성기는 생성 가능한 지배소와 의존소 사이의 수식 아크를 모두 생성하는 것이다. 이 부분에서는 의존 문법의 제약 규칙을 적용하지 아니한다. 단지 격조사의 접속 격 정보표에 의해서 지배소와 의존소의 관계가 성립된다면 모든 아크를 생성한다. 트리 추출기는 생성한 모든 의존소와 지배소 사이의 수식 아크들 중 하나씩만을 선택하여 구문 분석 트리를 추출하는 부분이다.

[표 1] 격 조사의 접속 격 정보표

조사	주격	목적격	부사격	관형격	접속격
은/는	○	○	○	×	×
이/가	○	×	×	×	×
도	○	○	○	×	×
을/를	×	○	×	×	×
로/으로	×	×	○	×	×
서/에서	×	×	○	×	×
와/과	○	○	○	○	○
의	×	×	×	○	×

[그림 1]의 트리 추출기 부분에서 의존 문법 규칙들이 적용되어진다. 먼저 격조사의 가능한 접속 격 정보표를 보이면 [표 1]과 같다.

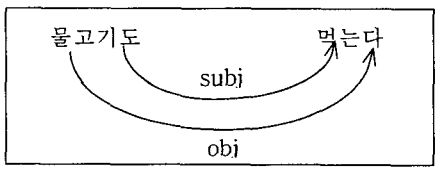
보통 한국어에서 의존소와 지배소의 관계는 명사와 용언의 관계라고 볼 수 있다. 단, 관형격과 부사격에서는 예외로 한다. 현재 어절이 의존소의 역할을 할 때, 우선 [표 1]의 접속 격 정보표에 의해서 가능한 격 관계(relation)를 나타내는 아크를 지배소로 향하게 연결한다. 의존소가 용언일 경우는 명사에게 아크를 생성하나, 의미상 아크의 방향은 명사가 용언을 수식하는 방향이다.

"엷집의 고양이가 마루에서 생선을 맛있게 먹었다."

다음 집합 D는 [의존소, 지배소, 격관계] 즉 아크의 집합을 나타낸다.

D={ [엷집의, 고양이가, adj],
[고양이가, 먹었다, subj],
[마루에서, 먹었다, adv],
[생선을, 먹었다, obj],
[맛있게, 먹었다, adv] }

각 관계 쌍을 살펴보면, 첫 번째 쌍은 의존소 격조사의 접속 정보표를 조사하면, '~의'는 관형격이 가능하므로 [adj]로 뒤의 명사를 수식하며, 두 번째 쌍도 마찬가지로 알아보면, 수식아크의 관계는 [subj], 계속해서 [adv], [obj], [adv]의 관계들로 이루어진다. 이는 [그림 3]과 같다. 위 예문에서는 다행히 각 어절마다 한 개씩의 아크만 존재하므로 아크를 생성함과 동시에 구문 분석 트리 까지도 완성되었다. 그러나 의존소 지배소의 관계 쌍은 같은 쌍임에도 여러 가지의 관계가 가능한 경우가 있다. [표 1]의 접속 격 정보표에서 '~도'와 같은 보조사는 주격, 목적격 모두 가능하다. 예를 들어, "물고기도 먹는다" 라는 문장에서 가능한 관계쌍의 수식 아크를 모두 표시하면, [그림 2]와 같다. 이 중에서 맞는 것만을 선택하는 것이 파서가 해야 할 큰 작업으로서, 이를 중의성 해소 작업이라고 부른다.

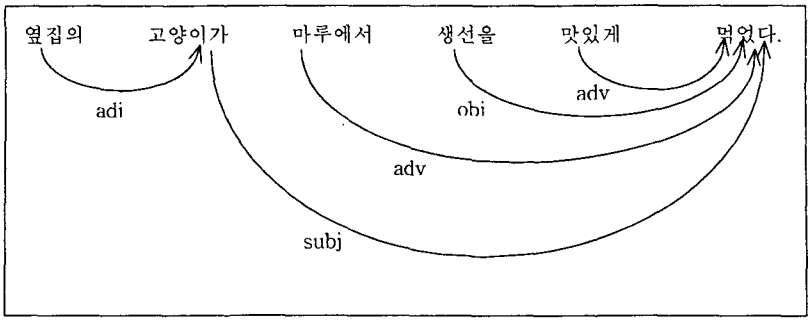


[그림 2] 가능한 격 관계 수식 아크

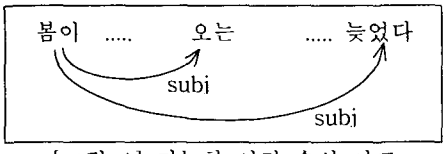
또 다른 중의성 현상은 다음 예문 에서 볼 수 있다. 여

"봄이 오는 소리를 듣느라고 좀 늦었다."

기에서 어절 "봄이" 가 수식할 수 있는 가능한 지배소의 후보로는 그 어절 다음에 나오는 모든 용언 어절들이라고 볼 수 있다. 이와 같은 현상은 [그림 4]에 나타나 있다.



[그림 3] 완성된 의존 트리



[그림 4] 가능한 어절 수식 아크

위의 가능한 아크 중 오직 하나만이 맞는 것으로서 이를 선택하여야 한다.

2.2 모든 가능한 수식 아크의 생성

입력 문장에 대하여 각 어절 사이의 가능한 모든 수식 아크를 생성하여 주는 단계이다. 여기서는 모든 생성 가능한 아크를 나타낸다. 예를 들어,

"철수가 길수도 아는 스테이크를 먹는 여자를 보았다"

에 대하여 생성된 모든 아크를 나타내면 [그림 5]와 같다.

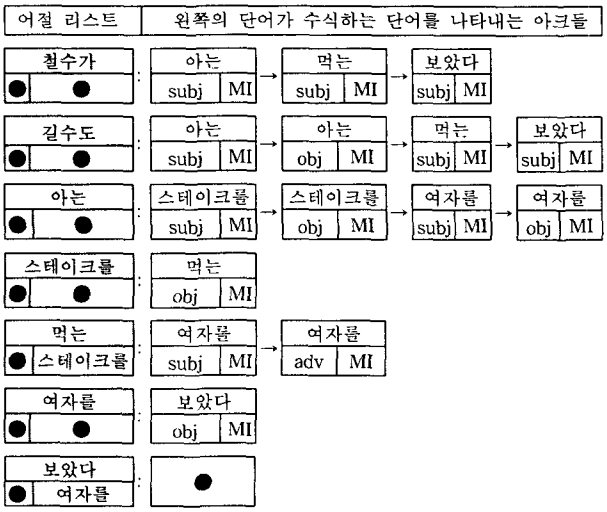
[표 2] 수식을 해야할 어절의 기준

용언을 수식하는 어절	부사. 격조사가 붙은 체언 어절.
명사를 수식하는 어절	조사가 붙지 않은 명사구 어절. 관형사. 관형격조사가 붙은 명사구어절. 관형형 어미를 가진 용언 어절.

본 논문에서 제안하는 아크 생성 알고리즘은 가능한 모든 수식 아크를 생성할 때 오른쪽에서 왼쪽, 즉 뒤 어절에서 앞 어절 방향으로 진행하여 나가며 아크를 생성한다. 이렇게 하는 이유는 한국어의 특성상 수식어가 피수식어의 앞(왼쪽)에 나타나므로 이 방법이 상당히 효율적이기 때문이다. 분석하기 전에 자신이 수식할 수 있는 어절을 파악한다. 먼저 용언을 수식하는 어절과 명사를 수식하는 어절을 정리하면 [표 2]와 같다.

마지막 어절인 "보았다"를 보면 자신은 마지막 어절이므로 뒤(오른쪽)에 이것이 수식할 수 있는 어절이 없다. 앞 어절인 "여자를"의 경우는 이것이 수식할 수 있는 어절은 "보았다"뿐이며 동시에 격 관계 또한 목적격만이 가능하다. 오직 한 개의 수식 아크만이 가능하므로 완전히 고정(permanent)된 아크로 만든다. 아크가 고정된 이유는 유일 필수격 원칙의 적용 때문이다. 예문에서 "여자를"이 필수격인 목적격으로 고정(permanent)된 아크를 생성하므로, 다른 어절은 절대 "보았다"의 목적격을 채워서는 안된다.

본 시스템에서는 필수격을 주격과 목적격으로 한정한다. 한국어의 특성상 부사와 관형사는 하나 이상이 한 지배소를 같이 수식할 수 있기 때문이다. 지배소의 필수격 중 채워진 격이 있다면, 유일 필수격의 원칙에 의해서 이것의 이 필수격을 채우려는 다른 아크를 더 이상 생성하지 않는다. 이와 같은 방법으로 각 어절이 어떤 어절들을 수식할 수 있는 지를 나타내면 다음 그림과 같으며 이를 의존 아크 풀(pool) 이라 부른다.



[그림 5] 의존 아크 풀

오른편은 의존 관계를 나타내는 아크의 리스트로서 그 구조는 [그림 6]과 같다.

[그림 6] 의존 관계 아크 구조

[그림 6]에서 상호정보(MI; mututal information)[1]란 의존소가 지배소를 특정 의존 관계로 의존하는 정도를 나타내는 값이다.

어떤 어절은 오직 한 어절만을 수식하도록 아크가 생성되는 데 이런 경우는 다음과 같다:

(1) 관형격 조사를 가지는 어절은 바로 뒤에 처음으로 나타나는 명사를 가진 어절을 수식한다.
(2) 부사격 조사를 가지는 어절은 바로 뒤에 처음으로 나타나는 용언을 가진 어절을 수식한다.[2]
(3) 조사가 없는 명사 어절 바로 다음 어절이 명사를 가진 어절이라면 앞의 어절이 뒤 어절을 수식하게 한다 (복합 명사 생성의 경우임).

위의 경우에서는 가능한 지배소가 오직 하나이므로 아크의 확률도 1이 된다.

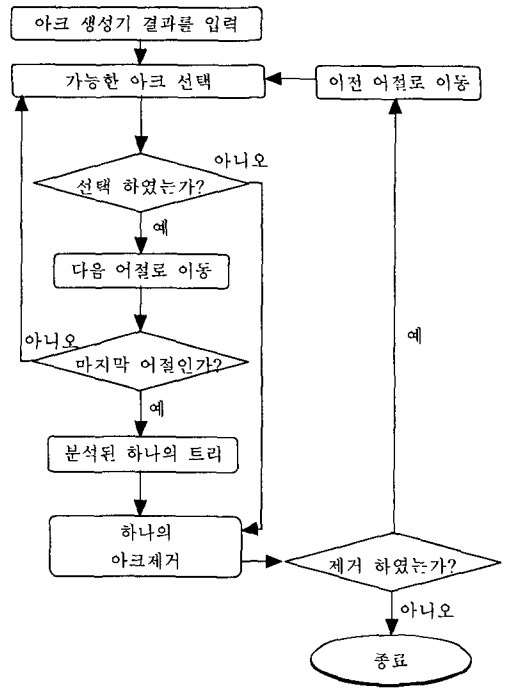
2.3 구문 분석 트리 추출의 알고리즘

각 어절은 의존소로 작용하여 다음에 나오는 어절을 수식한다. 이때 가능한 의존 관계는 하나 이상이다. 각 의존 관계는 아크로 나타내는데 위 [그림 5]에서 보듯이 한 어절이 취할 수 있는 지배소(및 격관계)는 여럿일 수 있다. 그러나 파스 트리에는 이 중 하나 만이 참여한다. 즉 한 의존소는 하나의 지배소만을 가진다 (지배소 유일의 원칙).

위의 의존 아크 풀은 동시에 여러 개의 파스 트리를 담고 있다고 볼 수 있다. 트리 추출기는 이 풀로부터 가능한 모든 파스 트리를 추출한다. 각 의존소에서 나가는 아크들 중 하나씩 만을 선택하여 하나의 파스트리를 만들게 된다. 이때 의존소마다 선택되어진 수식 아크를 저장하기 위해서 스택을 사용한다.

아크 생성기에서 생성된 모든 가능한 수식 아크들 중 한 의존소에서 하나의 아크만을 선택하여 비교차 규칙[3] 을 적용하여 구문 분석 트리를 추출한다.

추출된 여러 개의 트리들 중에서 아크들의 확률 정보를 이용하여 가장 큰 확률 값을 갖는 트리를 가장 좋은 구문 분석 트리라고 인식하여 선택한다. 본 논문에서 제안하는 트리 추출 알고리즘은 아크 생성기와는 반대로 문장의 순서대로 첫 어절부터 시작한다. 이유는 비교차 규칙을 적용하는데 있어서 순방향 알고리즘이 효율적이기 때문이다. 여기에서는 스택이 매우 유용하게 이용된다. 다음은 이 알고리즘의 개략적인 흐름을 나타낸다.

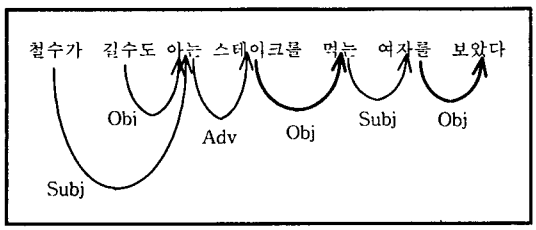


[그림 7] 구문 트리 추출기 흐름도

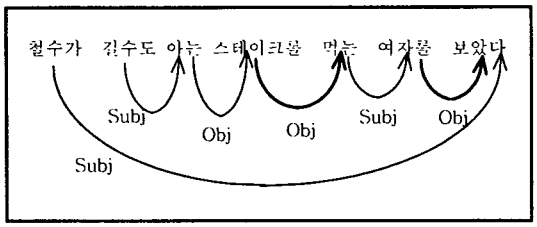
(1) 현재 어절에서 가지고 있는 노드들 중 의존 문법의 규칙에 어긋나지 않고 이전에 선택하지 않았던 하나를 선택하여 스택에 Push 한다. 지배소가 술어이고 선택되어진 아크의 격 관계가 필수격이면 지배소의 임시 공간에 격 정보를 기록한다. 단, 더 이상 선택할 노드가 없다면 (3)으로 이동.
(2) 다음 어절을 현재 어절로하여 (1)부터 재 수행. 다음 어절이 마지막 어절이라면 (5)로 이동.
(3) 스택에서 토큰 하나를 Pop 한다. Pop 된 정보에서 해당 의존소가 지배소의 임시 공간을 채웠다면 다시 비우고, 비교차 정보도 그 이전 어절의 정보까지로 고친다. 단, 더 이상 Pop 할 데이터가 없다면 (6)으로 이동.
(4) 이전 어절을 현재 어절로하여 (1)부터 재수행.
(5) 스택에 기록되어진 정보로 구문 분석 트리 생성. (3)으로 이동.
(6) 종료.

[그림 5]에서 추출할 수 있는 두 개의 파스 트리는 [그림 8, 9] 와 같다.

1) 상호정보의 자세한 설명은 3.3절 참조
2) 실제로는 뒤에 나오는 여러 용언 중 하나일 수 있으나 여기서는 시스템의 간단화를 위해서 제한을 둔 것임.
3) 비교차 규칙이란 파스를 구성하는 다른 두 개의 아크가 서로 교차할 수 없다는 원리이다.



[그림 8] 추출된 구문 트리 1



[그림 9] 추출된 구문 트리 2

3. 상호정보와 확률

3.1 트리의 확률

본 논문에서 제안하는 구문 분석을 위한 의존 문법에서는 각 의존 관계 즉, 각 아크마다 다음 3.3에서와 같이 확률을 부여한다[6]. 의존 트리는 이러한 아크의 집합이다. 의존 트리 D_j는 식 (3.1)과 같다.

$$D_j = \{ q_1, \cdots, q_{k_j} \} \tag{3.1}$$

의존 트리의 확률은 식 (3.2)와 같다.

$$P(D_j) = \prod_{i=1}^{k_j} P(q_i) \tag{3.2}$$

따라서 최종 선택되는 구문 분석 트리는 식 (3.3)과 같다.

$$D_{chosen} = \arg\max_{D_i} P(D_i) \tag{3.3}$$

3.2 아크에 대한 상호정보의 계산

수식 아크가 가지는 상호정보 값이란 해당 아크의 의존소가 지배소를 해당 격 관계로 수식할 확률을 말한다. 격관계는 보통 개입된 조사가 나타낸다. 조사마다 격관계가 다른 것으로 할 수 있으나 조사의 수가 많기 때문에 비효율적이다. 본 논문에서는 격 관계의 수[4]를 4가지로 제한하고 이에 따라 조사를 그룹화하였다. 예를 들면 주격을 나타내는 조사로는 '가', '이' 가 있으며 '는', '도' 등 보조사도 가능한 것으로 하였다.

하나의 아크, [의존소=d, 지배소=h, 격관계=r] 에 대한 상호정보는 이런 관계가 얼마나 잘 일어날 수 있는가의 정도를 나타내는 수치이다. 이것은 말뭉치에서 그러한 관계 쌍들이 얼마나 많이 나타났는가를 찾아냄으로써 알아낼 수 있다. 아크의 상호정보 M(d $\Rightarrow^r$ h) 는 식 (3.4)과 같다[8].

N : 훈련 말뭉치 안의 총 어절 수
$C(d)$: 의존소 d의 빈도수
$C(h)$: 지배소 h의 빈도수
$C(d \overset{r}{\Longrightarrow} h)$: 의존소 d가 지배소 h를 r의 격 관계로 수식한 빈도수

$$M(d \overset{r}{\Longrightarrow} h) = \frac{N \cdot C(d \overset{r}{\Longrightarrow} h)}{C(d) \cdot C(h)} \tag{3.4}$$

3.3 상호정보를 이용한 아크 확률의 계산

의존 트리 내의 아크 q 가 가지는 확률 값 $P(q)$는 현재 아크의 의존소에서 나가는 모든 가능한 아크들 중 q가 선택 될 확률이다. 각 의존소 i 가 가지는 아크들의 집합을 G 라 하자:

$$G = \{ q_{i,i_1}, \cdots, q_{i,i_k} \} \tag{3.5}$$

즉 아크들의 지배소의 위치는 $i_1, i_2, \cdots, i_k$ 가 된다고 하자. 각 아크 q_{i,i_j}의 상호정보는 식 (3.6)와 같다고 하자:

$$M(q_{i,i_j}) = m_{i,i_j} \tag{3.6}$$

각 아크 q_{i,i_j}의 확률은 식 (3.7)과 같이 구하여진다:

$$P\left(q_{i,i_j}\right) = \frac{m_{i,i_j}}{\sum_{h=1}^{k} m_{i,i_h}} \tag{3.7}$$

3.4 데이터 부족 문제

상호정보만을 이용하여 아크의 확률을 구할 경우에는 데이터 부족 문제가 발생할 수 있다[1,6]. 데이터 부족 문제를 해결해야만 정확하고 포용 범위가 넓은 구문 분석을 할 수 있다. 본 논문에서는 의존소의 빈도수 즉, $C(d)$의 값이 0일 경우에 대해서만 고려하였다. $C(d)$의 값이 없을 경우 식 (3.4)을 식 (3.8)과 같이 변경한다.

4) 4가지 격관계로 주격(subj), 목적격(obj), 관형격(adj), 부사격(adv)을 가정하였다.

$$M(T \xrightarrow{r} h) = \frac{N \cdot C(T \xrightarrow{r} h)}{C(T) \cdot C(h)} \quad (3.8)$$

T: 모든 의존소들(동일 품사)의 총 빈도수

그러나, 식 (3.8)을 그대로 쓰기에는 무리가 따른다. 이유는 오히려 위 확률 값이 매우 크게 나타나기 때문이다. 따라서, 어떠한 의존 관계 쌍의 확률 P를 구할 때는 back-off 처리를 하게 되어 식 (3.9)와 같이 나타내어진다.

$$P(d \xrightarrow{r} h) = \lambda \cdot \frac{N \cdot C(d \xrightarrow{r} h)}{C(d) \cdot C(h)} + (1-\lambda) \cdot \frac{N \cdot C(T \xrightarrow{r} h)}{C(T) \cdot C(h)} \quad (3.9)$$

λ: back-off 설정을 위한 상수

λ는 적절한 임의의 값을 설정하였다.

4. 실험 및 평가

본 실험에 쓰인 상호정보 데이터는 약 3백만 개의 수식 관계쌍을 이용하였다. 테스트 문장으로는 초등학교 읽기 교재에서 추출하였다. 실험한 전체 문장의 수는 100문장이었고 10단어 미만의 문장의 평균 길이는 7.5 이었고 10단어 이상의 문장의 평균 길이는 14.6이었다.

시스템의 성능을 나타내는 항목으로는 분석률과 정확률 그리고, 아크 정확률이 있다. 분석률을 나타내는 식은 식 (4.1)과 같다. 분석에 성공한 문장이란 문장의 전체에 대한 파스트리가 생성된 경우이다.

$$\text{분석률} = \frac{\text{분석에 성공한 문장의 수}}{\text{분석을 시도한 문장의 수}} \quad (4.1)$$

문장의 정확률을 나타내는 식은 식 (4.2)와 같다.

$$\text{문장정확률} = \frac{\text{정답과 구조가 일치하게 분석된 문장의 수}}{\text{분석에 성공한 문장의 수}} \quad (4.2)$$

문장의 아크 정확률을 나타내는 식은 식 (4.3)과 같다.

$$\text{아크정확률} = \frac{\text{정답안의 아크와 일치하는 아크의 수}}{\text{분석에 성공한 문장의 의존관계(아크)의 수}} \quad (4.3)$$

아크 정확률과 유사한 것으로 재현율이 있다. 재현율은 식 (4.4)와 같다.

$$\text{재현율} = \frac{\text{정답안의 아크와 일치하는 아크의 수}}{\text{정답 파스트리 안의 아크의 수}} \quad (4.4)$$

입력 문장의 문형은 형태소 분석이 정확하게 입력되었다고 가정한다. 입력 어절 수에 따른 각각의 평가 항목의 변화율은 [표 3]과 같다.

[표 3] 실험 결과

항목 \ 어절수	10어절 이내	10어절 이상
분석률	98%	98%
문장 정확률	77%	46%
아크 정확률	95%	90%
재현률	95%	90%

[표 3]에서 보듯이 문장의 분석률, 즉 구문 트리를 추출하는 부분에 대해서는 성능이 매우 좋다. 그러나 문장 정확률과 아크 정확률, 재현률에 있어서는 어절의 수가 늘어날수록 그 정확도가 낮아짐을 확인할 수가 있었다.

5. 결론

한국어 문장의 구문 분석을 위해서 구문 규칙이 필요 없는 문법 체계인 의존 문법을 이용하여 구문 분석을 시도하였다. 이러한 접근은 어순의 자유성에 의한 문제점을 해결하기가 쉬우며, 문장 구성 요소의 불연속성이나 생략 등의 현상에 큰 영향을 받지 않기 때문이다.

본 논문에서는 문장이 가진 모든 가능한 중의성을 고려하여 이를 해소하는 것을 목표로 하였다. 이를 위하여 먼저 가능한 모든 구문 분석 구조를 생성하는 기법을 제안하였다. 이것은 먼저 모든 가능한 구문 구조 트리에 관한 정보를 응축한 자료 구조를 구축한 다음 여기에서 구문 트리를 하나씩 추출하도록 하였다. 이 과정에서 조사와 격 사이의 관계, 필수격 유일의 원리, 비교차 원리, 지배소 유일의 원리 등 의존 문법이 만족하여야 하는 모든 기본적인 제약 조건을 만족하는 트리 만이 효과적으로 추출되는 기법을 제안하였다.

추출된 많은 트리 중에서 하나를 선택하는 것은 상호정보를 이용하였다. 즉 특정 명사가 특정 격 관계로 특정 용언과 관계를 갖는 정도를 나타내는 이 정보는 중의성 해소 정보로서 매우 유용하게 사용되었다. 본 논문에서는 이러한 상호 정보를 구문 분석 중의성 해소에 효과적으로 사용하는 기법을 제시하였다. 제안된 기법의 타당성을 입증하기 위하여 구문 분석 시스템을 개발하고 많은 문장에 대하여 실험을 하였다. 그 결과 매우 효과적으로 중의성을 해소하는 실용적인 가치가 있는 기법임이 밝혀졌다. 실험 결과 본 시스템은 상당한 수준의 분석률과 정확률을 보임을 알 수 있었다. 따라서 본 논문에서 제안한 의존 문법에 기반한 격 관계와 상호정보를 이용한 구문 분석 기법은 포용 범위가 넓고 정확도가 좋은 구문 분석 기법이라 할 수 있다.

참고 문헌

[1] M. Collins, "A new statistical parser based on bigram lexical dependencies," Proc. ACL'96, pp.184-191, 1996.

[2] P Hellwig, "Dependency Unification Grammar," Coling 86, pp 195~198, 1996
[3] I. A. Mel'cuk, Dependency Syntax: Theory and Practice, State Univ. of New York Press, 1988
[4] 권혁철, 최준영, "단일화 기반 의존 문법을 이용한 한국어 분석기" 한국정보과학회 논문지 19권5호, pp.467~476, 1992.
[5] 나동열, "한국어 구문 분석에 대한 고찰," 정보과학회지, 12(8), pp.33-46, 1994.
[6] 박의규, "의존문법을 이용한 통계적인 파싱에 관한 연구" 연세대학교 전산학과 석사학위 논문, 2001. 2.
[7] 류법모, 이태승, 이종혁, 이근배, "술어중심 제약전파를 이용한 2단계 한국어 의존파서", 한국정보과학회 1996 봄 학술발표논문집, pp.923-926, 1996.
[8] 양재형, 김영택, "다중 지식원을 이용한 한국어의 분석," 정보과학회 논문지, 21(7), pp.1324-1332, 1994.
[9] 엄미현, 신대규, 나동열, "한국어의 구조적 애매성," 정보과학회 1996 봄 학술발표 논문집, pp.911-914, 1996.4.
[10] 윤근수, 권혁철, "의존 문법과 대조 의미론을 이용한 한국어의 어휘적 중의성 해결 시스템," 인지과학, 3(1), pp.1-24, 1991.
[11] 이공주, 김재훈, 김길창, "한국어 구 구조 문법을 기반으로 하는 확률적 구문 분석" 한국정보과학회 가을 학술 발표논문집 1996 Vol. 23, No. 2, 1996.
[12] 홍영국, 이종혁, 이근배, "의존문법에 기반을 둔 한국어 구문 분석기," 정보과학회 1993 봄 학술발표 논문집, pp.781-784, 1993.