규칙과 어휘정보를 이용한 한국어 문장의 구묶음(Chunking)

+김미영[0], 강신재, 이종혁,

포항공과대학교 컴퓨터공학과

Text Chunking by Rule and Lexical Information

Mi-Young Kim[0], Sin-Jae Kang, Jong-Hyeok Lee

Dept. of Computer Science & Engineering, POSTECH.

요약

본 논문은 효율적인 한국어 구문분석을 위해 먼저 구묶음 분석(Chunking) 과정을 적용할 것을 제안한다. 한국어는 어순이 자유롭지만 명사구와 동사구에서는 규칙적인 어순을 발견할 수 있으므로, 규칙을 이용한 구묶음(Chunking) 과정의 적용이 가능하다. 하지만, 규칙만으로는 명사구와 동사구의 묶음에 한계가 있으므로 실험 말뭉치에서 어휘 정보를 찾아내어 구묶음 과정(Chunking)에 적용한다. 기존의 구문분석 방법은 구구조문법과 의존문법에 기반한 것이 대부분인데, 이러한 구문분석은 다양한 결과들이 분석되는 동안 많은 시간이 소요되며 이 중 잘못된 분석 결과를 가려서 삭제하기(pruning)도 어렵다. 따라서 본 논문에서 제시한 구묶음(Chunking) 과정을 적용함으로써, 잘못된 구문분석 결과를 미연에 방지하고 의존문법을 적용한 구문분석에 있어서 의존관계의 설정 범위(scope)도 제한할 수 있다.

1. 서론

구문분석은 문장의 구문 구조를 주어진 문법 규칙하에서 찾아내는 작업이다. 대표적으로 많이 사용되는 문법은 구구조문법과 의존문법이다. 본 논문에서는 한국어의 특징을 잘 반영할 수 있는 의존문법을 이용하여 구문분석을 하고자 한다.

의존문법은 문장을 구성성분 단위로 나누어 분석을 하려는 구구조문법과는 달리, 한 구성요소와 다른 구성요소와의 문법적 관계를 밝혀서 문장을 분석한다[2]. 즉, 의존노드와 지배노드의 쌍을 찾아내고 그 관계를 정의함으로써 문장을 분석하는 것이다. 하지만 입력 문장에는 구조적 애매성이 존재하기 때문에 구문분석 후에 다양한 결과들이 많이 나타나게 된다. 본 논문에서 제안하고자 하는 것은 의존문법을 적용하기 이전에 구묶음(Chunking) 과정을 거쳐서 정확한 구문 구조와 동떨어진 구조의 생성은 미리 피하자는 것이다.

다음 문장을 살펴보자.

그들은 지혜를 얻을 만한 이 세 가지 길을 통해서 침범하고 들어오는 것입니다.

위의 예에서, 지배노드가 가능한 용언으로서,' 얻을', ' 만한',' 통해서',' 침범하고',' 들어오는', 것입니다' 등 6 가지가 존재한다. 하지만 문장을 좀더 살펴보면, ' 이' 와 ' 세 가지' 는 둘 다 명사 ' 길' 의 의존노드 역할을 하고 있다. 따라서,' 이' +' 세 가지' +' 길' 을 하나의 명사구로 묶게 되면(Chunking), 나머지 어절들과의 의존 관계의 설정이 간단해진다. 명사구 묶음의 또다른 예로서 ' 들어오는' 이라는 관형어를 들 수 있다. 위 문장에서 ' 들어오는' 이라는 어절을 살펴보면 뒤에 등장하는 ' 것' 이라는 명사의 수식어 기능을 하고 있으므로, ' 들어오는' +' 것' 을 하나로 묶은 후 문장 분석을 하면 훨씬 간단하다.

동사[1] 의 구묶음 과정도 앞서 설명한 명사구묶음 과정과 유사하다. 위 문장에서 ' 침범하고' 와 ' 들어오는' 이라는 두 동사를 살펴보자. 이 두 동사들은 공통적으로 ' 그들은' 이라는 어절을 주어로 하고 있고 ' 통해서' 를 종속연결어로서 지배하고 있으므로, 하나의 단위로 묶은 후 문장을 분석할 수 있다. ' 얻을' 이라는 본용언과 그 뒤를 따르는 보조용언인 ' 만한' 도 실제로 두 개의 어절을 구성하고 있지만 하나의 의미구를 형성하고 있으므로, 두 어절을 묶은 후 다른 어절들과의 의존 관계를 설정하는 것이 보다 간편하고 정확한 결과를 얻을 수 있다.

앞서 제시한 방법으로 위의 문장을 구묶음하고 나면, 구묶음(Chunks) 내의 지배소 어절들만으로 구성된 간단한 문장으로 바뀔 수 있다.

[1] 여기서 동사란, 형용사를 포함한 넓은 의미의 동사로 쓰인다.

그들은 지혜를 *얻을*_만한 이_세_가지_*길을* 통해서 *침범하고*_*들어오는*_*것*입니다.

2. 기존연구

구묶음(Chunk)의 개념은 1991 년 Abney 가 처음으로 도입했다. Abney 는 구묶음(chunks)에 대해 ' 문장에서 겹쳐지지 않는 덩어리(non-overlapping segments of a sentence)로서 구문트리에서 연결된 하위그래프(connected subgraph)를 형성하는 것' 이라고 논했다[2]. 또한 Abney 는 인간이 문장을 읽을 때 띄어 읽는 운율적 휴지의 단위와 구문 구조에 있어서의 단위가 상응한다고 판단했다[3].

Abney 를 처음으로 하여, 연도 순으로 기존의 구묶음 방법을 소개하면 다음과 같다.

1991 년 Abney 가 구문분석의 첫 단계로 서로 연관된 단어의 덩어리를 찾기 시작했고, 두 번째 단계에서 연결장치(attacher)를 사용해서 구묶음(Chunk)들을 결합하여 구문트리를 형성했다. 그 후, 구묶음_방법은 크게 문법을 기반으로 한 방법과 통계적 모델을 기반으로 한 방법으로 나뉘어서 활발히 고안, 적용되었다.

문법을 기반으로 하는 방법은, 주로 어휘 자료를 유한상태 오토마타(finite state automata)와 결합하거나 문법의 제약을 이용하여 묶는 방법이다. 1992 년 Bourigault 가 휴리스틱을 사용해서 최대 길이의 명사구를 찾은 후, 전문용어 명사구(terminological noun phrase)를 추출하기 위해 문법을 사용했다. 하지만 이 시스템에서의 구묶음 목적은 지배소 명사를 추출하는 것이기 때문에 명사구에 포함될 수 있는 관형어 등의 수식어는 무시되었다.

또한, 1993 년 Voutilainen 이 구묶음(Chunk)을 위한 새로운 태그를 도입하여 품사 태그에 첨가시켰다. 그리고 사전에 각 단어마다 가능한 구묶음 태그를 입력했다. 이 시스템은 재현율 98.5%, 정확률 95% 로 높은 성능을 보여 주었다고 기록되어 있으나 실제 테스트해 본 프로그램의 성능은 약간 떨어진다고 밝혀졌다. 이 방법도 위와 마찬가지로 많은 종류의 명사구 수식어들이 제외되었다고 판단된다.[4]

같은 해에, Kupiec 이 유한 상태 명사구 인식기(finite state NP recognizer)를 사용하여 구묶음하는 방법을 제안했으나, 알려진 성능 결과는 없다.

최근 국내에서도 구묶음 방법이 사용되고 있다. 신효필은 규칙을 기반으로 한 명사구를 묶는 과정을 수행한 후, 동사구 장벽 알고리즘을 이용하여 동사구 묶음을 수행했다. 따라서 규칙에 위배되는 구묶음들에 대한 처리가 부족한 단점이 있다[5][6]. 또한, 윤준태는 구묶음(Chunk)을 세 단계로 나눈 후, 간단한 문맥 무관 문법(Context Free Grammar)을 이용해서 명사구를 형성하고, 말뭉치를 통한 연어패턴 정보를 이용하여 동사구를 형성했다[7]. 이 방법도 또한 문법만으로 해결 불가능한 명사구의 처리가 부족하고, 데이터의 부족 문제로 연어패턴 정보의 효율적 이용이 어렵다.

구묶음의 또다른 방법인 통계적 모델을 사용한 예로서는 1988 년에 이루어진 Church' s Parts Program 을 들 수 있다. 이 프로그램은 우선 단어들의 품사정보를 알아낸 후 중심 명사구를 찾기 위해 괄호(bracket)를 도입한다. 괄호(bracket)는 Brown 말뭉치를 이용해 학습된 통계적 모델을 사용해서, 문장에 반자동적(semi-automatic)으로 추가되었다. 작은 크기의 문장 표본에서, 성능 테스트를 한 결과 98%의 재현율을 나타냈다.

그 후, 1995 년에 Rhamshaw 와 Mitchell P. Marcus 가 변형 기반의 학습을 이용해서 구묶음 방법을 문장에 적용했다. 구묶음을 태깅 문제로 간주하여 학습을 시킨 후, 컴퓨터가 자동적으로 명사구 묶음을 형성하는 것이 가능하게 하였다. 명사구에 대해서는 정확률과 재현율 모두 93%의 성능을 보였고, 명사와 동사가 결합된 구묶음에 대해서는 88%의 정확률과 재현율을 나타냈다.

Rhamshaw 와 Marcus 의 방법 이후로, Argamon(1998), Veenstra(1998) 등 메모리 기반의 학습(Memory based Learning)이 성행했다. [8]

이 논문에서는, 규칙을 기반으로 한 구묶음 방법을 제안하고, 더 높은 정확성을 위해 말뭉치에서 바이그램 어휘정보를 추출하여 이용한다.

3. 구묶음 방법

3.1 명사구 묶음

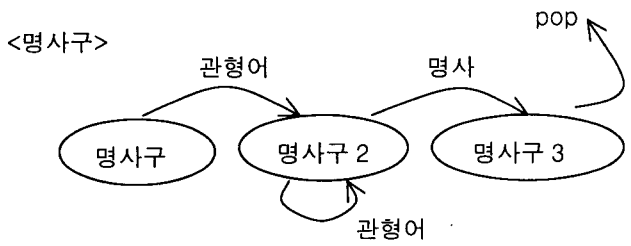
우선, 구묶음 과정의 첫번째로 명사구 묶음을 형성한다. 한국어는 어순이 자유로우나 명사구를 구성하는 " 관형어 + 명사(구) " 의 어순은 변화될 수 없으므로 이를 토대로 하여 규칙[2]을 형성한다.

<명사구>

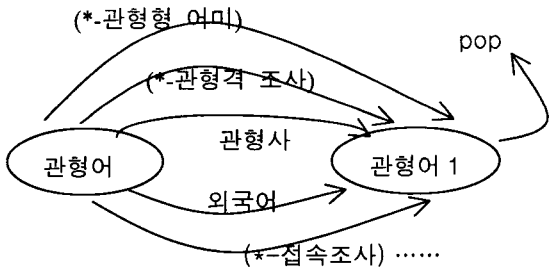
- (*-관형격 조사) * 명사(구)
- (*-관형형 어미) * 명사(구)
- (*-부사파생접사) * 명사(구)
- 관형사 * 명사(구)
- 명사 * 명사(구)
- 대명사 * 명사(구)
- 양수사 * 명사(구)
- 외국어 * 명사(구)
- (*-접속조사) * 명사(구)
- (*-접미사) * 명사(구)

[2] 이 규칙은 포항공대 KLE 연구실의 형태소 분석기 KOMA 의 형태소 분류표에 적합하게 만든 것이다.

규칙을 기술함에 있어서, (*- A) 의 뜻은, 어절이 A라는 형태소 끝남을 가리킨다. 위의 규칙들을 보기 쉽게 전이망(Transition Network)으로 나타내면 다음과 같다.



<관형어>



<그림 1> 명사구 묶음을 위한 전이망(transition network)

위의 규칙에서 부사파생접사로는 ' 적' 만을 취급하고 있으며, 접미사는 " 들" 과 " 네" 로만 한정하고 있다. 예로는" 사회적 동물", " 너희들 모두", " 명수네 가게" 등이 있을 수가 있다.

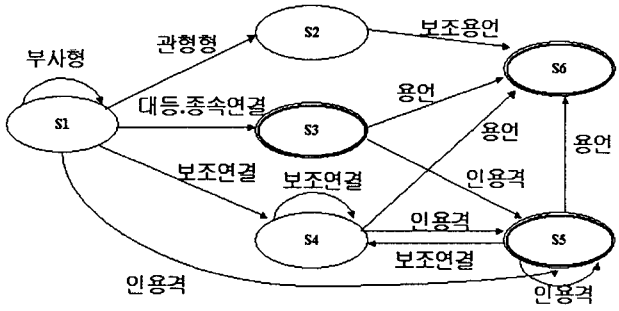
3.2 동사구 묶음

3.2.1 인접한 동사들의 묶음

문장에서 동사는 지배소 역할을 한다. 한 문장에 여러 개의 동사가 나타나면 지배소의 후보가 많아지므로 의존관계의 구조적 중의성이 더욱 커지게 된다. 그러므로 동사가 인접하여 나타나는 경우, 구묶음 과정을 거쳐서 묶어 주면 문장 구조가 훨씬 간단해진다. 인접한 동사들의 구묶음을 위한 오토마타는 <그림 2>와 같다.

이 그림에서 S3, S5, S6 는 수락 상태(final state)를 나타낸다. 동사구 묶음에서 더 길게 일치하는 경우가 보다 정확한 구묶음이므로 여러 개의 수락 상태 중에서 가장 길이가 길게 일치하는 수락 상태를 선택한다.

여기서 주의해야 할 점은, S1 -> S3 -> S6 으로 가는 과정에서, S3 -> S6 로의 용언은 관형형은 제외시킨다는 것이다.



<그림 2> 동사구 묶음을 위한 유한 상태 오토마타

예를 들어,

" 그들이 돌아가자 도망가던 도둑이 다시 돌아왔다."

라는 문장을 살펴보면, " 돌아가자" 와 " 도망가던" 은 비록 인접한 동사들이지만 어떤 의존관계도 존재하지 않기 때문에 구묶음 과정에서 제외시키는 것이 옳다. 이와는 반대의 예로,

" 골목을 빠져나와 걷던 그가 갑자기 멈췄다."

라는 문장에서는 " 빠져나와" 와 " 걷던" 은 공통적으로 " 그가" 를 주어로 취하므로 묶어주는 것이 편리하겠으나, 말뭉치에서 <예 1>과 같은 문장들의 빈도를 무시할 수 없으므로, 동사구 묶음에서 제외시킨다.

3.2.2 논항과 동사의 구묶음

실제 문장을 살펴보면, 관용적으로 쓰이는 논항과 동사의 패턴이 존재함을 쉽게 확인할 수 있다.

예를 들면," ~수 있(없)다", " 것 같다"," ~에 대하다"," ~를 위하다"," ~로 위장하다"," ~와 관련되다"," ~뿐 아니다"," ~기 위해" 등등이 있다.

이 예들 중 " ~로 위장하다"," ~와 관련되다", " ~로 인하다" 등의 서술패턴은 동사의 하위범주화 정보로 알

수 있다.

논항과 동사의 구묶음을 위한 패턴의 확인은 두 가지로 이루어질 수 있다. 양상 패턴의 리스트들은 따로 만들어서 참조하고, 서술 패턴은 각 동사의 하위범주화 정보를 사전에 입력하여 사용하는 방법이 있다. 이 논문에서는 현재 첫번째 방법만을 이용하고 있지만, 앞으로 용언의 결합가 정보를 구축해서 서술 패턴을 적극적으로 이용할 계획이다. 현재 사용하고 있는 (논항,동사) 패턴은 작은 크기의 말뭉치로부터 얻은 67 개와 김나리의 논문에서 발췌한 서술패턴 138 개이다[9].

3.3 구묶음의 예

앞서 언급했듯이 Abney 는 구묶음(Chunks)을 문장에서 겹쳐지지 않는 덩어리라고 했지만, 실제 명사구 내에 존재하는 관형어들은 용언이 많으므로 명사구와 동사구가 같은 어절을 공유하는 것도 허락한다. 아래의 <예 3>의 구묶음 결과에서 이를 확인할 수 있다.

<예 3>

다리를 건설해야 할 필요가 있다.

➔ 구묶음 결과

```
다리를
건설해야_할        ← 동사구 묶음 결과
할_필요가          ← 명사구 묶음 결과
있다.
```

4. 말뭉치에서 바이그램 어휘 정보의 추출

명사구 묶음에서의 대부분의 오류는, 주로 부사어 역할을 하는 명사가 명사구 내에 포함되기 때문에 생긴다. 아래의 예를 보자.

```
1)… … 요구될 때, 시위를 하지 않으면 안된다… .
2)… … 그의 국문에 대한 자각은 … … ..
3)… … 빤히 쳐다보자 당황한 듯 고개를 돌렸다. … .
4)… … .돌아서는 순간 그는 우뚝 섰다.
5)… … .그때 시간이… ..
```

2)는" 그의" 라는 관형어가 지배소와 인접해 있지 않으므로 발생하는 오류다. 실제 지배소는 ' 자각' 이지만, 바로 뒤 명사인' 국문' 과 잘못된 명사구를 이루고 있다. 2)를 제외한 나머지 예제는" 때" ," 듯" ," 순간" ," 그때" 와 같이 부사어 역할을 하는 명사들이 명사구 묶음 과정에서 제외되지 않아서 생기는 오류들이다. 이러한 명사들을 따로 처리하면 명사구 묶음에서의 오류는 대부분 해결이 가능하다. 하지만 이러한 명사는 문장의 종류에 따라 역할을 달리 할 수 있다. 예를 들면 다음과 같다.

```
1999 년 나는 학교에 가지 않았다.        (1)
1999 년 가을 이상한 일이 발생했다.      (2)
사실상 그의 얘기는 대부분 거짓말이다.  (3)
평상시 거의 외출은 안 하더니…          (4)
잠시 나간 동안 고양이가 들어왔다.       (5)
```

예 (1)에서는 명사구 묶음이 존재하지 않으나, 예(2)에서는 " 1999 년 가을" 라는 명사구 묶음이 존재한다. 따라서' 1999 년' 이 부사적 성질을 띠더라도, 뒤에 나오는 어절의 종류에 따라 부사의 역할을 하지 않을 수가 있다. 또 다른 예로서 (3),(4)를 보면, " 사실상" 과 " 평상시" 는 뒤에 오는 어절에 상관없이 항상 명사구 묶음에서 제외된다. 이와 같이 " ~상" ," ~시" 로 끝나면서 부사의 성질을 띤 명사는, 뒤 어절의 종류에 관계없이 부사어 역할을 한다. 또한 예문 (5)에서는 동안' 이라는 명사를 볼 수 있는데, 이것도 (3),(4)와 마찬가지로 항상 명사구 묶음에서 제외되는 어휘로서 뒤 어절을 확인할 필요가 없다.

그럼, 어느 정도 범위의 어절까지 고려해야 하는가? 위의 예제들을 조합한 결과, 이러한 명사를 얻기 위해 확인해야 하는 어절 수는 두 개를 넘지 않는다고 판단하고, 이 논문에서는 부사의 성질을 띤 명사 A 와, A 바로 뒤에 위치한 어절 B 까지만 확인한다.

이와 같이 부사어 역할이 가능한 명사들을 조합해 보면 시간에 관계된 명사가 대부분이다. 따라서 시간을 나타내는 특정 어휘인 경우 이러한 명사로 판단되는 것이 보통이나, 어절의 형태소 정보만을 이용하여 판단되는 수도 있다.

4.1 어휘에 의존적인 단어

대부분의 부사적 성질의 명사들은, " 봄" ," 낮" , " 밤" , " 가을" 등 특정 어휘들이다. 따라서, 이 어휘들에 대한 리스트를 따로 형성한다. 이들은 단독으로 쓰이는 경우 부사어 역할을 하지만, 비슷한 의미의 어휘들이 뒤에 등장하면 명사구 묶음 과정을 거쳐서 묶여야 한다. 그러므로 바이그램 어휘 정보에서 두 개의 어절 정보가 다 어휘에 의존적인 경우가 많다

4.2 형태소에 의존적인 단어

특정 어휘가 부사어 역할을 하는 경우 이외에, 특정 형태소들이 결합된 어절이 부사어 역할을 하는 경우가 있다. 현재, 말뭉치 학습 결과로는 단 하나의 예가 존재한다. 바로 <양수사+의존명사>가 그 예이다.

주로 시간에 관계된 의미를 지니는 4 월, 5 시, 39 분, 등등이 이에 속하는데, 이러한 어절들은 시간의 경과를 의미하는 어절이 뒤에 위치하는 경우를 제외하고는 부사어 역할을 하기 때문에 명사구 묶음에서 제외시킨다. 아래의 예를 참조하기 바란다.

<예>

1928년 멕시코 근방에서 태어났다. (X)
5시 반에 집에 왔다.
30분 뒤에 도착한다.
9월 중반에 들어서니, 날씨가 선선해진다.
70년대 말 사건이 터졌다.

4.3 바이그램 어휘 정보 추출

학습에 사용한 말뭉치는 Matec99[3]에서 제공받은 것으로서 설명문과 논설문으로 이루어진 4320 문장이 이용되었다. 잘못된 명사구 묶음에서 부사적 성질의 명사를 추출하는 과정은 손으로 이루어졌고, 총 152 개의 바이그램 어휘 정보가 추출되었다.

5. 실험 및 평가

5.1 실험방법

입력은 형태소 분석 및 태깅된 말뭉치들이고 출력은 명사구와 동사구 묶음이 이루어진 문장들이 된다. 여기서 형태소 분석기와 태거는 포항공대 KLE 연구실의 시스템을 이용했다.

제안된 시스템의 성능평가를 위해, 1999 년 ETRI 주최로 개최된 Matec99 에서 제공받은 말뭉치 중 설명문 200 문장을 테스트해 보았다.

그 결과는 <표 1>과 같다.

5.2 실험결과

	명사구 묶음		동사구 묶음
	어휘 정보 사용 전	어휘 정보 사용 후	
묶음수 (개)	781	770	201
정확률	97.31%	98.7 %	99.5 %
재현율	98.3 %	98.3 %	99.5 %

< 200 문장, 13.48 어절/문장>

< 표 1> 구묶음 실험 결과

동사구 묶음 과정에서, 정확률이 재현율이 각각 99.5%로 아주 높은 성능을 보였다. 인접한 동사구들을 묶을 때, 대부분의 오류는 <대등.종속연결어미 + 용언의 관형형>에서 발생하는데, 미리 언급했듯이 본 시스템에서는 이러한 경우를 구묶음 과정에서 제외시켰다. 따라서 거의 오류가 나지 않았다.

명사구 묶음 과정은, 어휘 정보의 사용유무에 관계 없이 재현율은 같다. 어휘 정보를 사용하지 않으면 잘못된 명사구가 더 발생하게 되므로, 정확률에만 영향을 주기 때문이다. 실제 어휘 정보의 사용이 유효하게 작용한 테스트 문장들을 추출해 보면 다음과 같다.

(1)… 예의 등 나름대로 개발한 생활양식이… (X)
(2)화해할 수 없을 만큼 갈등이 커져 버린다… (X)
(3)더 이상 화해의 가능성이 없어져 버렸다..(X)
(4)우리 나라의 경우 전자시대를… ..(X)

위의 잘못된 명사구 묶음들이, 어휘 정보 사용 후에는, 바르게 분석되었기 때문에, 어휘 정보 사용 후에 명사구 묶음 수는 줄었지만, 정확률은 높아졌다. 따라서, 어휘 정보의 사용이 구묶음의 성능 개선에 큰 역할을 함이 입증되었다.

6. 의존트리 생성

구문 분석 과정에서 실제 의존트리를 형성하기 전에 위와 같은 구묶음 방법을 거치면, 뒷 단계에서 해야 할 일은 크게 세 가지로 나뉘어질 수 있다.

1. 명사구 묶음 내에서의 의존관계 설정
2. 동사구 묶음 내에서 지배소의 설정
3. 구묶음 간의 의존관계 설정

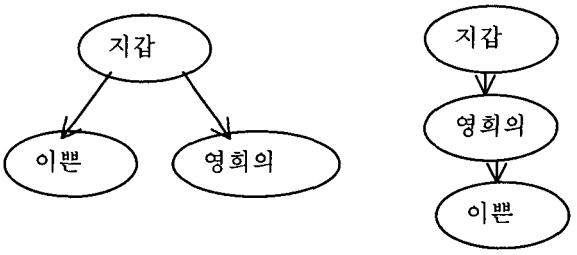
각 단계에서 하는 일에 대한 설명과 함께 간단한 예를 보도록 한다.

6.1 명사구 묶음 내에서의 의존관계 설정

한국어는 지배소가 후위에 위치하므로, 명사구 내에서 지배소는 가장 오른쪽에 위치한 명사라는 것을 알 수 있다. 하지만, 명사구 내에 지배소를 제외한 어절이 두 개 이상이라면, 각 어절들의 지배소가 서로 다른 경우도 존재한다. 예를 들어 " 이쁜 영희의 지갑" 이라는 명사구의 의존관계를 나타낸다고 해 보자. 명사구의 전체 지배소는 ' 지갑' 이라는 것이 명백하다. 또한, 한국어의 지배소 후위의 특징에 의거하여, ' 영희의' 의 지배소는 ' 지갑' 이라는 것을 쉽게 알 수 있다. 하지만, ' 이쁜' 의 지배소 후보는 ' 영희의' 와 ' 지갑' 두 가지가

[3] ETRI 에서 주최한 제 1 회 형태소 분석기 및 태거 평가 대회 (The first Morphological Analyzer and Tagger Evaluation Contest)

존재하므로, 이 명사구는 구조적 중의성을 가진다.



<그림 3> 명사구 내에서의 구조적 중의성

6.2 동사구 묶음 내에서의 의존관계 및 지배소 설정

의존문법의 특징은, 각 어절들이 하나의 노드를 형성하고 구(phrase) 개념이 없다는 것이다. 따라서, 의존문법에서 노드의 수는 문장에서의 어절의 수와 일치한다.

하지만 앞서 설명했듯이, 동사구 묶음은, 전체가 하나의 의미덩어리로 인식되는 경우도 있으므로, 이런 경우는 의존트리를 형성할 때에도 동사구 묶음을 분리시키지 않고 단 하나의 노드만을 부여한다. 하지만 의미에 관계없이 구조적 중의성을 줄이기 위해 여러 개의 동사가 묶어진 경우는, 기존의 의존문법의 특징을 반영하여 동사구 묶음 내의 각 어절마다 하나의 노드를 제공한다. 아래의 예를 보자.

<예 4>
그는 남편이 망했다고 생각하지 않았다

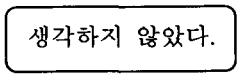
동사구 묶음 과정을 거치면, 위의 예는 다음과 같이 구묶음이 된다.

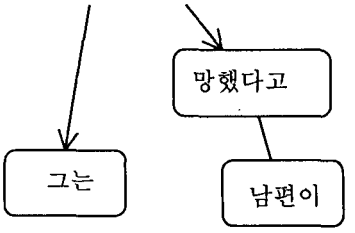
그녀는 남편이 망했다고_생각하지_않았다.

여기서, ' 생각하지' 와 ' 않았다' 는 본용언과 보조용언의 관계로서, ' 않았다' 가 ' 생각하지' 의 의미를 구체화시키는 역할을 할 뿐이다. 따라서, 이 두 어절은 하나의 의미 덩어리로 인식하여, 의존관계를 설정할 때도 구묶음 전체에 하나의 노드를 부여한다. 하지만 ' 망했다고' 와 ' 생각하지_않았다' 의 경우는 이와 다르다. ' 망했다고' 의 주어는 ' 남편이' 이고 ' 생각하지_않았다' 의 주어는 ' 그녀는' 이므로, 이 두 용언은 같은 어절을 의존소로 공유하지 않는다. 따라서, 두 용언이 각각 하나의 노드를 제공받게 된다.

하지만, 같은 동사구 묶음 내에 두 용언이 존재하므로, 명사구 묶음과 같은 방법으로 이 묶음 내에서의 지배소는 ' 생각하지 않았다' 가 되고 ' 망했다고' 는 ' 생각하지 않았다' 의 의존소가 된다.

<예 4>의 문장에 대해 의존트리를 형성하면 다음과 같다.





<그림 4> 의존 트리의 예

7. 결론

이 논문에서는 규칙과 어휘정보를 이용한 한국어 문장의 구묶음 방법에 대해 소개했다. 기존의 구문분석 방법은, 다양한 분석 결과들이 생성되는 동안 많은 시간이 소요되며, 그럴 듯한 분석결과와 동떨어진 결과들도 많이 생성되기 때문에 비효율적이다. 본 논문에서 제시한 구묶음(Chunking) 과정을 적용함으로써, 잘못된 분석결과를 미연에 방지하고 어절 사이의 의존관계 설정 범위도 제한할 수 있다. 또한 말뭉치를 통한 어휘 정보를 이용해서, 명사구 묶음 과정에 있어서의 정확률을 높일 수 있음을 입증했다.

본 논문에서는 부사어 역할을 하는 명사에 관해서만 어휘 정보를 이용하였으나, 뒷 단계의 트리 구성 과정에서, 논항과 용언의 의존관계 설정에 있어서 더 많은 어휘 정보가 필요할 것이라 예상된다. 또한 제시한 구묶음 과정 중, 동사구 묶음에서 관형형 용언과의 결합을 제외시켰는데, 실제 문장에서 이러한 용언들의 등장이 많으므로 이에 대한 대책이 필요하다. 현재는 작은 규모의 말뭉치를 통해 용언에 대한 서술 패턴 리스트를 획득하였기 때문에, 모든 동사에 대해 좀더 일반적인 서술 패턴을 적용하려면 사전을 이용한 결합가 정보의 도입이 큰 도움이 될 것이라 기대된다.

참고문헌

[1] Convington, M. A. 1990. " A Dependency Parser for Variable-Word-Order Languages" , Research Report AI-1990-01, Artificial Intelligence Programs, Univ.of Georgia,

[2] Steven Abney, 1991. " Parsing by chunks" , In *Principle Based Parsing*. Kluwer Academic Publishers,

[3] Steven Abney, 1995. " Chunks And Dependencies" , In *Computational Linguistics and the Foundations of Linguistic Theory*. CSLI.

[4] Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text chunking using transformation-based learning.

n *Proceedings of the Third ACL Workshop on Very Large Corpora.* Association for Computational Linguistics.
[5] 신효필. 1999 " 최소자원 최대효과의 구문분석", 제 11 회 한글 및 한국어 정보 처리 학술 대회, pp.242-248
[6] Hyopil Shin. 1999. The VP-Barrier Algorithm for a Robust Syntactic Parsing in Head-Final Languages, In *Natural Language Processing Pacific Rim Symposium*
[7] Juntae Yoon. 1999. Three Types of Chunking in Korean and Depenency Analysis Based on Lexical Association. In *Proceedings of the 18th International Conference on Computer Processing Languages (ICCPOL'99),* pp. 59-65, Tokushima, Japan, 1999, 3.
[8] Tjong Kim Sang, Erik F.. 2000. Noun Phrase Representation by System Combination. In *Proceedings of ANLP-NAACL 2000,* Seattle, Washington, USA. Morgan Kaufman Publishers,
[9] 김나리. 1997. 패턴정보를 이용한 한국어 구문분석. 서울대 컴퓨터공학과 박사학위논문.