論文2002-39CI-6-2

단어통사론을 위한 계산 모형

(A Computational Model for the Word-Syntax)

金 東 柱 *, 金 漢 宇 *

(Dong-Joo Kim and Han-Woo Kim)

요 약

한국어 형태론에 대한 기존의 전산모형은 선형적인 것들로 단어 내부구조 분석보다 형태소 분리 문제에만 관심을 두고 있다. 이러한 선형적 전산모형을 구문 분석 과정과 통합적으로 고려할 경우, 구문 단위 요소의 형성을 위해 형태소 분석 결과를 묶어야만 하는 추가적인 과정이 필요할 뿐만 아니라 의미적 직관성을 얻기도 어려웠다. 본 논문에서는 형태소 분리와 구문 요소 형성뿐만 아니라 단어의 구조 분석까지도 통합적으로 다룰 수 있는 단어통사론적 시각에 따른 전산 모형을 제안한다. 먼저 형태소 분리와 변형 문제를 다루기 위해 2단계형태론의 형식화를 도입하고, 품사 문맥을 반영하기 위해 기능성 구분문자를 제안한다. 그리고 형태소의 통사적 결합 검사를 위해 GLR에 기반한 변형 알고리즘을 제안한다.

Abstract

Computational models up to now for Korean morphology have been linear in that it deal with only segmentation of morphemes rather than formation of the internal structure of a word. When integrating a linear computational model with syntax analysis, it requires an additional interface component between this model and the syntax to bind morphemes into sentence constituents. Furthermore the linear model is not semantically intuitive. In this paper, based on word-syntactical viewpoint, we propose an integrated computational model that deals with morpheme segmentation, formation of syntactic element (sentence constituent), and even internal structure of word. Formalism of two-level morphology is employed to cope with morpheme segmentation and alternation problems, and functional diacritics are proposed to incorporate categorial context into the two-level formalism. A modified GLR-based algorithm is also proposed to check syntactical constraint of morphemes.

Keywords : 전산형태론(computational morphology), 단어통사론(word-syntax), 2단계형태론(two-level morphology), GLR 알고리즘(GLR algorithm)

Ⅰ. 단어통사론과 연구의 범위

언어학에서의 통사론이라 함은 일반적으로 문장 구조를 연구하는 분야를 의미하는 말로 문장내의 언어요소, 즉 단어 구(句), 절(節)간의 관계를 규명하는 분야이다. 그러나 통사론에 대한 좀더 포괄적인 의미를 갖는 기호학 정의는 언어를 접근하는 방법론에 관한 또 다른 시각을 제공하는데, Morris는 지표(index), 도상(icon), 상징(symbol)의 복합체인 기호(sign)에 대한 세 가지 연구 영역 중 하나인 통사론(syntax)을 '기호들간의 상호 관계를 규명하는 분야'로 정의하고 있다[1]. 이

* 正會員, 漢陽大學校 컴퓨터工學科

(Computer Science & Engineering, Hanyang University)

接受日字:2002年7月18日, 수정완료일:2002年11月7日

때 기호가 단어라면 언어학에서 일반적으로 이야기하는 문장 구조에 관한 통사론이 될 것이고, 기호가 형태소라면 단어 구조에 관한 통사론이 될 것이다. 이에 Selkirk는 문장 구조에 관한 통사론을 문장통사론(S-Syntax), 단어 구조에 관한 통사론을 단어통사론(W-Syntax) 명명하였다[2]. 단어형성에 관하여 구조주의 학자들은 형태부 영역 내에서만 해결하려한 반면[3], 형태부의 완전 독립을 선언하면서 시작된 생성형태론에서의 단어통사론적 접근은 인접 영역의 연관성 속에서 해결을 시도하였으며[4], 형태부에 본격적으로 문장통사부의 장치들을 도입하였다[2]. 물론 문장 형성과정과 단어 형성과정은 그 성격이 근본적으로 다르므로 단어 형성과정을 핵계층이론(X-bar theory)에서 사용되는 문맥자유문법을 사용했을지라도 구금지제약(No Phrase Constraint)이나 어기(base)에 하나의 접사 첨가만을 허용하는 식으로 문장통사부의 장치들과는 다른 모습을 취하고 있다. 특히 단어구조 독자성 조건(The Word Structure Autonomy Condition)을 통하여 "어떤 통사규칙도 형태론적 자질과 범주를 참조할 수 없다"는 강어휘론적가설(Strong Lexicalist Hypothesis)을 철저히 따르면서 형태부의 독자성을 강조하고 있다. 또한 파생 및 굴절 요소들을 구분할 수 있는 분명한 근거가 없음을 지적하고 파생뿐만 아니라 굴절까지도 모두 형태부에서만 이루어진다고 주장하기에 이르렀다[2]. 이후 이러한 주장은 많은 비판을 받았고 1980년대 들어서는 [5]에 근거를 바탕으로 굴절은 통사부에서, 파생은 형태부에서 이루어질 것이라는 약어휘론적가설(Weak Lexicalist Hypothesis)을 따르는 시각과[6,7], 단어 형성과정 전체가 문장 형성과정과 동일한 단일한 모습의 통사적 장치에 의해 형성되어야 한다는 형태부의 존재를 부정하는 시각이 나타났다[8-10]. 이러한 형태부 모습의 변화에 따라 전산언어학분야에서 언어학자들의 이론의 검증을 위한 단어통사론적 시각에 따른 계산모형이 등장하기 시작하였다[11-14].

최근 한국어 순수 언어학 분야에 있어서도 단어통사론적 시각에 따른 종합적인 연구가 이루어 졌는데[15,16], [16]에서는 관용어, 어휘적 단어, 통사적 단어, 음운적 단어로 포괄적인 단어의 유형을 제시하고 형태부, 통사부, 음운부 모두에서 단어가 형성될 수 있음을 보이고 있다. 차이가 있어 보이긴 하지만 이러한 시각은 어형성부에서 유형별로 단어가 형성되어 형태부, 통사부, 음운부에 각각 삽입된다는 평행형태론(parallel morphology)[17]과 같은 맥락이라 볼 수 있겠다. 이러한 연구에도 불구하고 이를 검증할 수 있는 적합한 계산 모형에 관한 연구는 거의 이루어지지 않고 있는데, 이는 물론 한국어의 단어 형성이 우변첨가 양상이 지배적이고 문맥자유문법 수준의 언어적 현상이 생산적이지 않은데 비해 규칙 표현이 어렵고 시간복잡도(time complexity)가 높을 뿐만 아니라 형태소 분석 단계에서 발생하는 구조적 중의성으로 인해 효율성이 떨어진다는 문제 때문이었다. 이러한 이유로 대부분의 계산모형은 우변 첨가 양상만을 반영한 선형적인 것들로 단어내부 구조분석보다 형태소 분리(segmentation or isolation) 문제에만 관심을 두고 있다. 순수 언어학적인 동기를 배제할 경우 선형적인 모형 외에 다른 연구가 없을 수 밖에 없는 더 근본적인 이유로 단어의 계층적 내부 분석 방법론에 대한 응용 수준에서의 수요 부족을 들 수 있다. 다시 말해, 맞춤법 검사기에서는 어절의 최종 합법성 여부만 알면 될 것이고, 정보검색에서는 색인어를 추출하기만 하면 되는 것과 같이 대부분의 응용에서는 단어 내부구조를 파악하는 것 따위는 필요 없었기 때문이다.

본 논문에서는 단어내부 구조 분석에 대한 계산론적 필요성과 언어학적 동기에 대해 간략히 설명한 후 단어통사적 한국어 형태소 분석을 위한 자료구조와 알고리즘을 제안한다. 이를 위해 70년대 이후의 단어통사론에 관한 시각을 개관(槪觀)하여 한국어 단어통사론적 시각에 따른 적합한 계산 모형을 제시하고, 언어이론 학자들의 이론 검증과 확인에 사용할 수 있는 도구로 활용할 수 있게 하며 좀더 나아가 언어학적 원리에 근접한 지식을 필요로 하는 자연언어처리 응용에 활용할 수 있도록 한다. 논문에 제시한 몇몇 단어통사론적 현상에 관한 예는 실제 이론가들에 대한 공통적 시각이 아닐 수도 있다. 가능하면 단어통사론의 모든 관점을 포함하려 노력하였으나 기본적으로 파생과 굴절이 모두 형태부에서 이루어진다는 관점을 유지하며 이에 관한 단어통사규칙을 사용하고 있다. 그러나 문장통사부에서 굴절이 이루어진다는 시각에서의 문장 분석도 가능하도록 하기 위해 굴절부의 계층구조는 출력하지 않는 분석 결과도 가능하도록 하였다.

또한 제안하는 알고리즘은 자질 연산을 기반으로 하는 형태소 분석 시스템에서의 효율적인 파싱을 위한 것이다. 일반적으로 자질 연산을 기반으로 하는 시스템에서는 자질 연산만으로 분석이 이루어지지만, 먼저 연

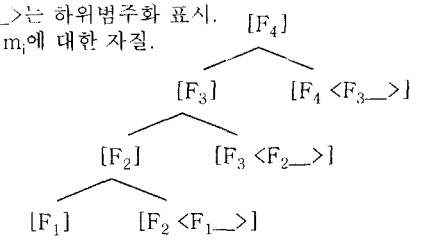
산 대상의 범위를 축소한 후 자질 연산을 수행하는 것이 더 효율적이다[18]. 즉, 효율적인 자질 기반 형태소 분석을 위해서는 연산 대상의 범위를 축소하기 위한 안내자 역할의 품사 파싱과 세부 자질에 대한 연산이 교호적(交互的: interleaving)으로 이루어져야 한다. 본 논문에서는 교호적 자질 연산 형태소 분석 시스템에서 연산 대상의 범위를 축소하기 위한 파싱 알고리즘을 제시한다.

Ⅱ. 단어의 통사적 현상과 형태부

핵(head)과 자질(feature), 자질의 삼투(percolation), 등 자질 연산을 기반으로 하는 핵심어주도 구구조문법(HPSG)과 같은 형식화를 이용하여 문장의 구조분석을 행할 경우, 가장 간단한 형태부의 모습은 모든 단어들이 등재된 사전과 어휘삽입규칙(Lexical Insertion Rule)을 갖춘 것이다. 이때 사전에 각각의 단어는 완전한 자질을 가지고 있어야만 한다. 그러나 이미 파생과 굴절, 복합이 일어난 모든 단어를 사전에 준비하고 있는 것을 불가능할 것이다. 따라서 형태부는 자질을 갖는 형태소가 등재된 사전과 이 형태소들을 이용해 단어를 형성할 수 있도록 하는 파생과 굴절, 복합을 위한 형태소 결합 규칙(혹은 분해 규칙)을 갖는 모습이어야만 한다. 전산언어학 분야의 대부분 응용에서는 형태소 결합 규칙의 형태는 이웃하는 형태소들의 결합 가부만을 판단하는 좌우접속표의 형식으로 구현된다. 이러한 결합 규칙은 형태소가 결합하여 새로운 범주를 형성할 수 있는 정규문법(regular grammar)과는 약간의 차이가 있긴 하지만, 1) 세부 품사가 형태소의 기본적 자질에 대한 암묵적 표시[1]라고 하고, 2) 단어 형성에서 어기의 최우측(right-most) 요소가 상위 요소 형성에 항상 핵의 역할을 한다[2]고 가정한다면 선형 모형에서도 (1.a)에서와 같은 자질 연산을 통하여 전체 단어의 자질을 문장 통사부에 넘겨 줄 수 있을 것이다.

(1) a. 형태소 결합 : $m_1+m_2+m_3+m_4$

<F_i—>는 하위범주화 표시.
F_i는 m_i에 대한 자질.



b. 부/xp + 자유/ncn + 스럽/xsm + (다)[3]

또한 1)과 2)의 두 가지 이유에서 최우측 요소 '[F_4 <F_3—>]'만 보고서도 전체 단어의 범주 및 자질을 알 수 있으므로 단어 내부 정보에 관심을 기울일 필요가 없어 보인다. 즉, (1.b)의 형태소 분리 결과에서 '-스럽'의 세부 품사 xsm(형용사파생접미사)만 참조하고서도 전체 단어는 형용사라는 사실과 '-스럽' 앞에 있는 어기는 명사라는 사실을 단어내부로부터의 자질 연산없이도 기계적으로 알 수 있을 것이다. 이러한 생각은 [21]에서 근간으로 채택하고 있는 개념이라 볼 수 있고, 우변첨가양상이 지배적인 한국어에서 응용 수준에 따라, 혹은 사전과 분석 알고리즘에서의 형태소 기준의 세밀화 정도에 따라서는 이렇게 단어 내부 구조적 정보가 필요없이 형태소만 분리하고 문장 통사부에서는 최우측 형태소 자질만 참조하는 방식은 근사적으로 옳다고 할 수 있다.

그러나 형태소 분석의 결과가 구문 분석으로 입력되고 구문 문석의 결과가 의미 분석에 활용되어야 하는 자연언어처리 전체 과정을 고려하고, 응용 수준이 형태소 분석 단계를 벗어났을 때, 위의 가정은 정확하지 않다. (2.a, b)에서 '발령받-'의 시제는 시간을 나타내는 명사 '어제'와 '내일'에 문장 통사부 수준에서 원거리 의존적(long-distance dependency)이다. 그런데 앞서 설명한 형태소는 분리만 하고 최우측 형태소 '-부'의 자질만 참조하는 방식으로는 이와 같은 의존성을 반영할 수 없어 (2.a, b)와 같이 비합법적인 문장을 배제할 수 없다. 즉, '-부'의 자질에는 앞에 오는 어기가 시간 명사라는 사실은 알 수 있지만 과거/현재/미래 어느 것인지는 관한 정보는 가지고 있지 않기 때문이다. 또한 (2.c)의 경우에도 '-조차'라는 조사에 의해 가려져 있어 최우측 요소만 참조하면 존칭의 자질을 상위로 전달할

1) 예를 들어, 일부 명사 뒤에 붙어 그러한 사람임을 나타내는 '-자'라는 접미사의 세부 품사 xsn(명사파생잡미사)은 이 접미사 앞에 불가산 명사를 요구하는 하위정보화 표시

$$\left[\text{SUBCAT}\left[\begin{array}{l}\text{MAJ}\quad\text{N}\\ \text{COUNT}\quad -\end{array}\right]\right]$$

와 접미사가 붙은 후 가산 명사를 만들어 낸다는 핵자질

$$\left[\begin{array}{l}\text{MAJ}\quad\text{N}\\ \text{COUNT}\quad +\end{array}\right]$$

가 묵시적으로 담겨있다고 볼 수 있겠다.

2) Williams의 우변핵심어규칙(Righthand Head Rule)과 동일[19].

3) 형태소 태그는 KAIST 국어정보베이스 태그[20] 참조

수 없게 된다. 이와 유사한 현상은 굴절에서도 발견된다. 동사뒤에는 현재서술형 굴절어미 '-ㄴ다'가 오고 형용사뒤에는 '-다'가 올 수 있다. 물론 핵을 무엇으로 볼 것인가에 따라 문제가 조금 달라지지만 굴절요소를 핵으로 보고, 어간과 현재서술형 굴절어미 둘 사이에 선어말어미가 있을 경우, 내부로부터의 비핵자질이 상위로 전달되지 않으면 어떤 형태의 현재서술형 굴절어미가 오는지를 제약할 방법이 없다. 따라서 통사부 수준에서 HPSG의 결속승계원리(Binding Inheritance Principle)류와 같은 원리를 통한 원거리 의존성을 반영할 수 있으려면, 형태부에서 손실없이 정보의 충분한 전달을 위해 형태부에서 단어내부에 대한 완전한 자질연산이 필요하다. 이러한 요구사항은 분석 단계가 의미까지로 확장된다면 더욱 절실해 진다.

(2) a. *그는 어제부로 총무부에 발령받을 것이다.
b. *그는 내일부로 총무부에 발령받았다.
c. *아버님께서조차 그 음식을 먹었다.

(3) 단어통사규칙 1

모든 규칙은 $A \rightarrow Ba$ 혹은 $A \rightarrow a$의 형태를 갖고, $a \in V_T$이고 $A, B \in V_N$이다. V_T는 종단기호(terminal symbol)인 형태소 태그들의 집합이고 V_N은 비종단기호(nonterminal symbol)인 형태통사(morphosyntactic) 태그들의 집합이며, B는 어기이고 a는 핵(head)이다.

이제 형태부의 모습은 형태소 사전과, (3)과 같이 항상 어기의 최우측 요소에 의해 상위 범주가 결정되는 우선형정규문법(right-linear regular grammar)이면 될 것 같아 보인다. 이 정도 수준에서는 형태소의 선형결합만 정확하게 파악된다면 체계적이지 못한 모습이긴 하지만 핵자질만 단어의 최우측 형태소에서 참조하고 비핵자질 중 상위로 전달되어야 할 자질은 나머지 형태소에서 참조하는 방식도 가능할 것이다. 그렇다면 한국어 단어형성에서는 항상 어기의 최우측 요소가 핵역할을 하는지 확인해 봐야 한다. (4)에서 보듯이 한자어계 접두사 중 핵으로 작용하여 어기의 범주를 변경시키는 '불(不), 대(對/大), 무(無), 비(非), 반(反)' 등과 같은 서술성 접두사가 있다[22].

(4) a. 탈북(脫北)하다, *북(北)하다, 부(不)도덕하다, ?도덕하다
b. 대(對)북한 전략, 대북한 경제지원, *대북한을, *대북한에서, *대북한으로

(4.a)에서의 '탈-'과 '부-'는 '북'이라는 비서술성 일반명사를 동작성 명사로, '도덕'이라는 추상명사를 상태성 명사로 변경시킨다. 또한 (4.b)에서는 '대-'라는 한자계 서술성 접두사가 '북한'이라는 일반명사를 관형어화하여 독립적으로는 문법적 역할을 못하고 반드시 피수식 명사가 오도록 만들었다. 이러한 경우 어기 왼쪽에 접사가 붙고 이 접사가 핵역할을 하므로 $A \rightarrow aB$ 형태의 좌선형정규문법(left-linear regular grammar)이 필요하다. 따라서 최종적인 형태부의 문법 규칙은 (5)과 같이 좌선형, 우선형이 동시에 존재하는 제한된 형태의 문맥자유문법이어야만 할 것이다.

(5) 단어통사 규칙 2

모든 규칙은 $A \rightarrow Ba$, $A \rightarrow aB$, $A \rightarrow a$의 형태를 갖고, $a \in V_T$이고 $A, B \in V_N$이다. V_T는 종단기호인 형태소 태그들의 집합이고 V_N은 비종단기호인 형태통사 태그들의 집합이며, B는 어기이고 a는 핵이다.

이러한 접두사가 비록 굴절접사나 접미사에 비해 생산적이지는 못하나 다른 접사에 비해 잠재어(potential word)나 신조어 생산에 있어서만은 매우 활발히 참여하고 있으므로 형태소 분석기가 어느 정도의 시대적 유연성을 갖고 견고한 분석을 위해서는 이와 같은 접미사의 범주 설정과 분석은 필수적이라고 하겠다.

좌우에 모두 핵이 존재할 수 있는 또 다른 경우는 합성현상이다. 비록 구체적이지는 않을지라도 여과(filter) 장치를 도입하여[4] 어느 정도 선에서 생성을 제한하여야 한다는 주장도 있긴 하지만, 많은 이론가들이 단어 길이에 대한 원칙적인 상한선이 없으며 최소한 합성현상에 있어서만은 재귀성(recursiveness)과 자기내포성(self-embedding, self-feeding)을 갖는다고 말하고 있다[2,23,24]. 합성단어의 구조 파악을 위한 것이 풍부한 표현력을 갖는 문맥자유문법 수준의 규칙을 사용하는 가장 중요한 이유라고 할 수 있다. 특히 (6.a)와 같이 명사의 복합에 있어서는 매우 생산적인데, 논항구조를 갖는 어휘항목, 즉 서술성 명사가 올 때, 나머지 명사를 제한하는 형태로 복합명사의 단어 내부에 관한 구조를 갖게 된다. 그런데 [25]에서는 다른 이론가들과 반대로 뜻(The friend is a girl)에서 형태(a girlfriend)를 얻어냄으로써 단어도 문장과 같이 변형 규칙에 의해 기저에서 생성됨을 주장하였다. 이 주장은 매우 자의적인 것으로 통용되는 이론은 아닌 것 같지만, 영어에서와는 달리 최소한 한국어에서는 문장형성 과정과 복합명사

의 형성 과정은 분명히 구분된다. 즉 한국어 문장에서 (6.b)와 같이 문장 구성성분의 일부 순서를 서로 바꾸어도 강조의 의미외에는 의미변화가 없느나, (6.c)와 같이 복합명사에서 순서를 변경한다면 전혀 다른 의미가 되어 버린다. 즉, '메일확인'은 논항구조를 갖는 단어 '확인'의 논항으로 무엇을 '확인'하는지가 단어 내부에 나타나 있지만, '확인메일'에서는 '확인'은 '메일'의 부가어 역할을 할 뿐 '확인'하는 대상이 나타나있지 않다. 역으로 이야기하면 복합명사의 내부 구조도 문장 구조와 유사하게 논항 구조를 갖지만 복합명사 내부에서는 어순이 철저히 지켜져야만 한다. 이것은 명사의 복합이 문장 통사부에서 발생하는 것이 아니라 형태부에서 발생하는 것이거나, 최소한 문장 통사부가 아닌 다른 위치에서 발생한다는 반증이라 할 수 있겠다.

(6) a. [[[자동차 [보험 [계약]]] [자]] [[만기 예고] [안내 [문]]]]
b. 메일을 확인하다. 확인하다, 메일을.
c. 메일확인, 확인메일

이상에서 단어 내부로부터의 계층구조 파악이 필요하고 문맥자유문법 수준의 규칙을 사용해야 하는 이유에 대해서 설명하였고, 다음과 같이 요약할 수 있다.

(7) a. 형태부에서의 정보손실의 최소화
b. 어기의 범주를 바꾸는 서술형 접두사의 처리
c. 합성어의 처리

이제 형태부의 모습은 단어통사 규칙과, 이 규칙으로 설명할 수 없는 특이성(idiosyncrasy)을 갖는 형태소들을 저장한 사전으로 결론 지을 수 있다. 이 때의 단어통사 규칙은 문장통사부에서의 규칙과는 별도로 형태부에만 존재하는 규칙으로 두 가지로 구성되어 있다. 하나는 형태소 분리와 동시에 형태/음운론적 변형(alternation)을 다루는 철자규칙(spelling rule)으로 2단계형태론의 형식화를 사용하고 있으며, 다른 하나는 철자규칙에 의해 어휘형(Lexical Form : LF)으로 복원되어 분리된 형태소들의 통사적 관계를 파악하는 형태통사 규칙(morphosyntactic rule)이 있다. 다음 절에서 설명되겠지만 주의해야 할 사항은 두 규칙이 적용되는 단계는 서로 분리된 개별의 단계가 아니라는 점이다. 또한 2단계형태론의 철자규칙에서 형태/음운론적 변화가 없는 형태소의 단순 분리는 TRIE 구조의 사전이 그 역할을 동시에 수행하게 된다.

Ⅲ. 형태소 분석

1. 개요

단어통사의 문법규칙의 범위는 보는 시각에 따라 다양하다. 이 논문에서는 순수 언어학 이론에서의 1접사 1규칙 가설(One Affix, One Rule Hypothesis)[26], 2항분지가설(Binary Branching Hypothesis)[27], 우변핵심어규칙(Righthand Head Rule)[19], 등과 같이 문법규칙의 표현력을 제한하는 원리나 가설들이 존재한다. 본 논문에서 채택하고 있는 문법 표현력의 범위는 촘스키 정규형(Chomsky Normal Form)보다 문법적 표현력이 작은 문법 (5)를 사용한다. 또한 사용되는 형태소 파싱법은 문맥자유문법을 파싱할 수 있는 GLR[28,29]에 기반한 방법으로, 사용되는 규칙이 정규문법에 가까우면 가까울수록 평균적으로 선형적 시간 복잡도에 가까와진다. 본 논문에서 사용하고 있는 문맥자유문법은 표현력이 극히 제한된 정규문법에 가까운 규칙이므로 GLR법을 사용한 방법에 있어서 근사적으로 선형적 시간 복잡도에 근접할 것으로 기대한다. 그러나 GLR법은 입력 단위가 분명한 구문 분석을 위해 고안된 방법이므로 본 논문에서는 이 방법을 형태소 분석에 적용하기 위해서 형태소 경계를 찾아내고 형태/음운론적 변형을 다루기 위한 철자규칙을 함께 고려한 변형된 GLR법을 제안한다. 먼저 형태/음운론적 변형 및 형태소 분리문제를 다루기 위해 2단계형태론의 형식화와 이에 대한 저수준 표현인 유한상태변환기를 이용한다. 그리고 계층적 형태소 결합의 합법성 검사를 위해 별도의 문법규칙을 사용하므로, 초기 2단계형태론[30]과는 달리 여러 개의 부사전을 갖출 필요가 없어 어휘의 중복이 없으며, 어휘 항목에는 연속범주(continuation class)에 대한 정보를 유지할 필요 없이 품사만을 둘 수 있다. Tabular 파싱 방법[31]과 달리 분석을 위한 스택 자료구조와 분석 결과의 저장을 위한 별도의 자료구조가 유지된다. 또한 2단계형태론[30]이나 개선된 CYK 방법[32]과 동일하게 다음 형태소 분석 결과는 이전 형태소 분석 결과에 의존적인 전파(propagation)방식이므로 불필요한 계산을 방지할 수 있을 뿐만 아니라, 분석 결과를 저장하기 위해 Tabular 파싱에 기반한 방법[31,32]에서의 사용되지 않는 공간을 초기에 할당함으로써 발생하는 메모리 낭비가 없다. 이러한 특성에 추가하여 GLR 방법에서는 애매성에 대한 효율적인 분석을 위해 LR 방식에서의 선형적

인 스택을 사용하지 않고 그래프 구조화 스택(graph-structured stack)을 사용하며 분석 결과의 효율적인 저장을 위해 압축 공유 숲 트리(packed shared forest tree)를 사용한다. 그래프구조화 스택은 분석 과정에서 발생할 수 있는 메모리의 낭비를 효과적으로 줄일 수 있을 뿐만 아니라 형태소 분석 내부의 한 단계에서 발생된 애매성이 내부의 다음 단계에 영향을 미치지 않게 하여 성능을 획기적으로 높일 수 있는 중요한 자료구조이다. 또한 앞축 공유 숲 트리는 스택과 유사한 이유에서 형태소 분석에서의 다중의 애매한 결과가 이 결과를 활용하는 문장분석에 미치는 영향을 최소화하기 위한 중요한 자료구조이다. GLR 방법은 계산을 통하여 다음 유도를 결정하지 않고, 추가적인 계산없이 파싱 테이블의 참조를 통하여 하나의 품사 입력에 대해 다음 유도를 정할 수 있는 결정론적(deterministic) 방법이다. 본 논문에서 사용하는 분석 방법의 기본 개념은 GLR법과 유사하므로 방법에 관한 형식적인 기술은 생략하고, 형태소의 분리와 형태/음운론적 변형, 그리고 형태통사 파싱을 중심으로 설명할 것이다.

2. 2단계형태규칙

먼저 2단계형태론에서와 마찬가지로 형태론적 변형과 분리문제를 다루기 위해 유한상태변환기(finite state transducer)와 TRIE 사전을사용한다. 유한상태변환기를 통해 음소별 변환 과정을 거치면서 동시에 TRIE 사전의 음소 노드(node)를 통과한다. 고수준 형식화의 시각이 아닌 유한상태변환기의 저수준 시각에서 2단계형태론[14,30]의 진행과정과 다른 점은 모든 가능한 해석을 찾기 위해 깊이-우선 탐색(depth-first search)을 하는 것이 아니라 넓이-우선 탐색(breadth-first search)을 행한다는 것이다. 이해를 돕기 위해 'ㄹ'불규칙, '으'삽입 현상, 그리고 '여'불규칙과 축약현상이 동시에 발생한 현상의 처리를 위한 2단계규칙을 예로 들고 형태통사 규칙으로 문맥자유규칙 일부만을 제시하여 이를 토대로 형태소 분석 진행 과정을 기술한다.

(8) a. 살 + 시는 ↔ 사시는

LF: ㅅ ㅏ ㄹ + ㅅ ㅣ + ㄴ ㅡ ㄴ
SF: ㅅ ㅏ ∅ ∅ ㅅ ㅣ ∅ ㄴ ㅡ ㄴ

b. 먹 + ㄹ까 ↔ 먹을까

LF: ㅁ ㅓ ㄱ + ∅ ㄹ ㄲ ㅏ
SF: ㅁ ㅓ ㄱ ㅇ ㅡ ㄹ ㄲ ㅏ

c. 검색 + 하 + 았 ↔ 검색했

LF: ㄱ ㅓ ㅁ ㅅ ㅐ ㄱ + ㅎ ㅏ + ㅇ ㅏ ㅆ
SF: ㄱ ㅓ ㅁ ㅅ ㅐ ㄱ ∅ ㅎ ㅐ ∅ ∅ ∅ ㅆ

(9) a. $ㄹ_c$:∅ ↔ __ +:∅ {ONB | $ㅇ_o$ $ㅡ_n$}
b. +:$ㅇ_o$ ↔ CR __ ∅:$ㅡ_o$ {ONR | $ㅇ_o$ $ㅡ_n$}
c. $ㅏ_n$:$ㅐ_n$ ↔ $ㅎ_o$ __ +:∅ $ㅇ_o$:∅ $ㅏ_n$:∅
where ONB = {$ㄴ_o$, $ㅂ_o$, $ㅅ_o$}
ONR = {$ㄴ_o$, $ㄹ_o$, $ㅁ_o$, $ㅅ_o$}
CR = 'ㄹ'을 제외한 종성 집합

(8.a, b, c)는 각각 'ㄹ'불규칙 현상과 매개모음 '으' 삽입 현상, 그리고 '여'불규칙 및 축약 현상이 동시에 발생한 예를 표층형(SF : Surface Form)과 어휘형으로 대응시켜 놓은 것이다. 이 세 가지 형태/음운론적 변형은 (9)와 같은 2단계규칙으로 표현할 수 있다. (9)의 자음에 붙어있는 아래첨자는 초성(o : onset)과 종성(c : coda)을 구분하기 위한 것이다. 중성(n : necleus)에 붙어있는 아래첨자는 다른 기호와의 시각적 구분을 위한 것일 뿐, 반드시 필요한 것은 아니다[4]. 분석시 (9.a)는 초성 'ㄴ', 'ㅂ', 'ㅅ'이나 '오' 앞에 종성이 없을 경우에는 종성 'ㄹ'과 경계구분 문자(diacritics) '+'를 삽입한다는 의미이며, (9.b)는 좌측에 'ㄹ'을 제외한 종성이 오고 우측으로는 초성 'ㄴ', 'ㄹ', 'ㅁ', 'ㅅ'이나 '오'가 올 경우 초성 'ㅇ'를 '+'로 바꾸고 'ㅡ'를 삭제하라는 의미이다. 또한 (9.c)는 표층형으로 'ㅎㅐ'가 오면 'ㅐ'를 'ㅏ'로 바꾸고 'ㅇㅏ'를 삽입하라는 의미이다. 그러나 이와 같이 너무 일반적인 변형규칙은 과도한 규칙 적용을 유발한다. 즉, 첫 번째 규칙은 '사시는'의 분석에서 기본 대응 규칙을 제외하고도 '살+시는', '사실+는'과 '살+실+는'에 대한 모든 변환을 시도해야만 한다. 이 규칙은 받침이 없는 음절 뒤에 초성 'ㄴ', 'ㅂ', 'ㅅ'이 오거나 '오'로 시작되는 음절의 모든 부분에서 적용된다. 이 문제는 2단계규칙 기반 방법에서만 발생하는 것이 아니라 형태/음운론적 변형 문제의 처리를 오토마타에 의존하는 기존의 모든 방법론에서 동일하게 겪는 문제점이다. 본 논문에서는 과도한 규칙 적용 문제의 최소화를 위해, 한국어의 형태론적 변형은 형태소 경계에서 발생하고 변형 위치를 기준하여 좌우에 올 수 있는 형태소들의 품사가 제한되어 있다는 사실을 반영한 변형된 2단계 규칙을 제안한다.

기존의 대부분 형태소 분석 방법론에서는 형태론적 변형에 대한 처리와 결합의 합법성 검사를 독립적으로

4) 나머지 형식화에 관한 기호는 [12, 14] 참조

표 1. 기능성 구분 문자.
Table 1. Functional diacritics.

구분문자	좌문맥문자	우문맥문자
'0'	없음	없음
'1'	용언류	어미류
'2'	용언화접사류	어미류
'3'	체언류	조사류

수행하기 때문에 변형 현상에 따른 품사 의존성을 반영할 수 없어 추가적인 연산이 발생할 수 있다. 예를 들어, '사는'이라는 어절에 'ㄹ' 불규칙이 적용되어 '살+는'이라는 어휘형을 생성했을 경우 '살/pv+는/et'뿐만 아니라 '살/ncn+는/jc'에 대해서도 합법성 여부를 조사하게 된다. 그러나 불규칙 현상은 용언과 어미의 결합에서만 발생하는 언어 현상이므로 '살'이라는 형태소가 명사로 분석될 가능성은 변형 규칙 적용시에 미리 제거될 수 있다. 즉, 한국어에서 형태론적 변형 규칙은 규칙의 적용과 동시에 형태소의 품사 범위를 줄여주는 여과 역할을 할 수 있다. 따라서 형태론적 변형 규칙의 적용과 동시에 여과 역할을 수행할 수 있도록 하기 위해 <표 1>과 같이 형태소 구분 문자 '+'를 대신하는 기능성 구분 문자(functional diacritics)를 제안한다. 어절의 음소들이 유한상태변환기를 통과하는 중에 이 기능성 구분 문자를 만나면 이전 기능성 문자로부터 통과되어온 문자열의 품사를 참조하고, 각 기능성 문자에 상응하는 문맥제약을 통하여 불필요한 품사를 여과시킨다. 즉, <표 1>의 '1'을 만나면 좌측의 최우측 형태소의 품사는 동사/형용사류이어야 하고, 우측의 최좌측 형태소의 품사는 어미류이어야 한다는 의미이다. '2'의 용언화접사류는 어간이 형태/음운 변화를 거치는 것이 아니라 명사에 붙어 용언화시키는 접사류가 불규칙이나 축약 등과 같은 형태/음운 변화를 거치는 것들로, '-스럽', '-답', '-롭', '-하', '-되', '-시키'와 일반적으로 서술격조사라 하는 '-이'가 여기에 해당한다[5]. (9.a, b)는 분류 1에 해당하고, (9.c)는 분류 2에 해당하므로 이를 적용하여 (9)의 2단계규칙은 (10)과 같이 표현할 수 있다.

(10) a. ㄹ$_c$:∅ ↔ __ 1:∅ {ONB | ㅇ$_o$ ㄴ$_n$}

b. 1:ㅇ$_o$ ↔ CR __ ∅:ㅡ$_o$ {ONR | ㅇ$_o$ ㄴ$_n$}

c. ㅏ$_n$:ㅐ$_n$ ↔ ㅎ$_o$ __ 2:∅ ㅇ$_o$:∅ ㅏ$_n$:∅

(10.a)의 2단계규칙에서 어휘형에 기능성 구분 문자가 포함되어 있는 대응쌍(correspondence pair)을 기준으로 할 경우, 좌문맥은 'ㄹ$_c$:∅'을 의미하고 우문맥은 '{ONB | ㅇ$_o$ ㄴ$_n$}'을 의미한다. 분석시 좌문맥 매칭 후 기능성구분 문자를 만나면 사전에서 좌문맥 문자열의 품사정보를 구분문자의 좌문맥제약을 통하여 여과시킨다. 여과된 이후 규칙적용 과정에서 자신, 혹은 또 다른 규칙에 의해 또 다시 기능성 구분 문자를 만나면 먼저 만난 기능성 구분 문자의 우문맥제약과 현재 기능성 구문 분자의 좌문맥제약을 통해 문자열의 품사정보를 사전으로부터 여과한다.

이 외에도 불필요한 규칙 적용을 최소화하기 위한 방법으로는 문자 집합을 세분화하여 2단계규칙 자체를 구체화하는 방법이 있다. 문자 집합이 세분화되면 세분화될수록 좌우 문맥 문자열은 사전의 형태소에 가까워질 것이고 매우 제한된 형태소에만 적용될 수 있을 것이다. 또 다른 방법은 기능성 구분 문자를 세분화하여 규칙 적용의 수를 줄일 수도 있다. 일부 형태론적 변형은 구체적인 품사와 연관되어 있기 때문에 기능성 구분 문자를 <표 1>에서보다 더 세분화한다면 'ㅎ' 불규칙과 같은 현상이 동사와 어미의 결합에 적용되는 것 또한 막을 수 있을 것이다.

(10)과 같은 고수준의 형식화는 천이 테이블(transition table) 형태로 된 저수준의 유한상태변환기로 변환되며[33], 분석시 입력되는 음소열은 유한상태변화기를 통과하면서 어휘형 음소열이 생성된다. 어휘형 음소 하나가 생성될 때마다 TRIE 사전의 한 노드를 천이하게 되며, 형태소 정보가 담긴 노드에 이르게 되면 하나의 형태소가 완성되어 단어의 구조 분석 요소로 간주되고, 다음절에서 설명하게 될 과정에 적용을 받게 된다.

3. 단어통사규칙

2단계형태규칙을 이용하여 형태론적 변화를 거치는 동시에 문맥자유문법을 이용하여 합법성 검사를 행한다. <그림 1>은 제안하는 방법의 설명을 위해 구성한 간단한 단어통사규칙의 예로, 이 규칙에서는 KAIST 국어정보베이스[20]의 중분류에 해당하는 형태소 태그 일부를 사용하였다. 태그 중 대문자로 시작하는 것은 형태구(morphological phrase) 태그이고, 형태구 태그는 하나의 대문자로 이루어진 어휘수준의 형태구 태그와

5) 일반적으로 용언화 접미사라 부르는 '-답', '-하'와 같은 것들은 문장통사적 특성을 지닌 것으로 많은 학자들에 의해 밝혀졌고, 일부는 본동사로 인정되고 있다. 그러나 본 논문에서는 일단 학교문법을 기준하여 접미사로 분류한다.

1. Word → N
2. | Nj
3. | Ve
4. Nj → Nj jc
5. | N jc
6. | Vep et
7. N → N nc
8. | nc N
9. | nc
10. Ve → Vep et
11. | Vep ef
12. Vep → Vep ep
13. | pv
14. | N jcp

그림 1. 단어통사규칙.
Fig. 1. Word-syntax rules.

state	action								GOTO				
	jc	jcp	et	ef	ep	nc	pv	$	Vep	N	Nj	Ve	W
0						s6	s7		5	4	3	2	1
1								acc					
2								r3					
3	s8							r2					
4	s10	s11				s9		r1					
5			s12	s13	s14								
6	r9	r9				r9,s6		r9		15			
7			r13	r13	r13								
8	r4							r4					
9	r7	r7				r7		r7					
10	r5							r5					
11			r14	r14	r14								
12	r6							r6,r10					
13								r11					
14			r12	r12	r12								
15	r8	r8				r8		r8					

그림 2. 파싱 테이블
Fig. 2. Parsing Table.

하나의 대문자와 소문자들로 이루어진 어절수준의 형태구 태그로 나누어진다. 이 둘 간의 구분은 통사적접사의 포함 유무에 따른다.

형태소 분석을 위해 먼저 LR 파싱에서 사용되는 파싱테이블 생성법[34]을 이용하여 <그림 2>와 같이 파싱테이블을 생성한다. <그림 1>의 규칙에서는 두 가지 종류의 구조적 애매성이 존재하게 된다. 즉, 'Vep et'에 대해 'Nj'와 'Ve'로 해석될 수 있으며, 'nc nc'의 배열에 대해 'N nc'와 'nc N'의 해석이 가능하다. 물론 이 둘은 태그를 세분화하거나 자질 제약을 사용할 경우 어느 정도 제거될 수 있는 규칙이다. 그러나 문맥자유문법으로 기술된 규칙은 언어적 본질로 인하여 애매성은 존재할 수밖에 없을 것이다. 이러한 애매성을 갖는 문법을 파싱테이블로 만들었을 경우 <그림 2>에서 보듯이 하나의 테이블 항목에 두 가지 이상의 액션이 존재하는 충돌(conflict)이 발생하게 된다. GLR에서는 이러한 충돌이 발생한 경우 넓이-우선 탐색 통하여 모든 해를 찾도록 한다.

4. 형태소 분석

2단계규칙과 파싱테이블을 이용하여 어절을 분석하는 과정에서 입력된 표층형 어절의 음소는 2단계규칙의 저수준 표현인 유한상태변환기의 아크들을 통과함과 동시에 어휘형 음소들을 생성하며, 생성된 어휘형 음소들은 TRIE 사전의 노드를 통과한다. 이 때 어휘형으로 기능성 경계구분문자가 생성되면 사전으로부터 품사를 참조하고 파싱테이블로부터 다음 취할 액션을 결정하게 된다.

사용되는 주요 자료구조 G와 T는 다음과 같다.

그래프 구조화 스택(graph-structured stack : G) : 분석 결과의 효율적인 저장을 위해 사용되는 스택은 방향성 비순환 그래프(directed acyclic graph)로 정점 $V(v_i)$와 에지 $E(e_{ij})$들로 이루어져 있다. V는 상태정점(state vertex)과 기호정점(symbol vertex)으로 구분되며, 상태정점은 시작 정점 v_0으로부터 경로를 이룰 때 짝수 번째 위치하며 원으로 표시된다. 기호정점은 시작정점으로부터 홀수 번째 위치하며 사각형으로 표시된다. 상태정점 v_i에서 i는 파싱테이블에서의 파싱 상태 번호이고 원 안에 표시된다. 기호정점 v_i에서 i는 압축 공유 숲 트리에서 원소 번호이고 사각형 안에 표시된다. 그리고 액션 연산이 행해지는 대상인 상태정점을 활성상태정점(active state vertex)이라고 한다.

압축 공유 숲 트리(packed shared forest tree : T) : 분석 결과가 저장되는 선형적인 자료구조로 각 원소 t_i는 분석 결과에 대한 트리 구조 표현에서 각 노드에 해당한다. 분석 결과의 트리 구조 표현에서, 단말 노드에 해당하는 원소들은 꺾은 괄호 내에 형태소 태그와 형태소들이 들어가 있는 형태이고, 나머지 원소들은 꺾은 괄호 내에 형태구문 태그와 현재 비단말 노드와 직접 연결되는 노드들에 대한 T의 색인 정보가 괄호로 묶인 형태로 들어간다. 이 때 한 비단말 노드가 두 가지 이상의 해석을 갖는다면 두 개 이상의 괄호 묶음이 들어가게 된다. 괄호로 묶인 하부 노드들에 대한 집합이 들어간다.

분석에 사용되는 주요 연산은 '천이', '여과', '액션'으로 세 가지이며, '액션' 연산은 '쉬프트', '리듀스', '분할',

'병합', 네 가지로 구성되어 있다.

천이(transition) : 공백문자(blank character : ∅)를 포함한 표층형 음소 하나가 입력될 때마다 유한상태변환기의 아크 하나를 통과하고 position의 값이 하나씩 증가한다. 천이되고 있는 유한상태변환기의 모든 경로들 중 어느 하나가 기능성 구분문자를 만날 때까지 유한상태변환기의 아크들을 통과한다. 통과하는 과정에서 생성되는 어휘형 문자열들은 그 문자열의 시작 음소를 루트로 하는 TRIE 사전 노드 경로를 병행하여 통과한다. 이 때 유한상태변환기에는 없는 기본 기능성 구분문자(default functional diacritics) '0'은 TRIE 사전에서 한 어휘의 끝이 그 역할을 수행한다. 사전의 경로를 통과하여 품사 정보를 참조했을 경우 다음 통과해야할 TRIE 노드는 2단계규칙에서 새로 만들어지는 음소를 새로운 루트로 하는 경로가 된다.

여과(filering) : 기능성 구분문자를 만나면 문맥 제약조건을 통해 좌변 문자열에 대한 사전 품사를 제거한다. 만약 각 경로에서 처음 구분 문자를 만났다면 좌변 문맥만 적용하고 두 번 이상 만났다면 좌변 문자열에 대해서 이전 구분 문자의 우변 문맥 제약을 먼저 적용한 후, 현재 구분 문자의 좌변 문맥 제약을 적용한다. 여과되고 남은 형태소 품사 정보와 해당 상태정점 정보를 이용하여 파싱테이블로부터 각 정점들에 대해 취해질 액션을 결정한다. 액션이 결정된 상태정점들은 활성상태정점 목록에 추가되어 액션 연산이 행해지기를 기다리게 된다.

액션(action) : 활성상태정점 목록에 있는 정점들에 대해 리듀스 연산을 우선적으로 행한다. 그 후 리듀스 연산이 수행된 정점 경로의 마지막 정점에 대해 파싱테이블로부터 또다시 다음 취해야 할 액션을 결정한다. 리듀스 연산이 행해져야 하는 정점이 더이상 존재하지 않는다면 쉬프트 연산을 수행한다. 쉬프트 연산이 수행된 상태정점은 활성상태정점 목록에서 제거되고 다시 천이를 수행한다. 이 때 행해야 할 모든 액션이 acc(accept)라면 분석 성공하고 종료하게 된다.

리듀스(reduce) : v_i에서 v_j로 이르는 경로에, v_j를 제외하고 v_j를 포함하여 $n\times2$ 개의 정점이 존재하고 k번 규칙의 우변의 기호의 개수를 n이라고 했을 때, v_j에서 액션 re$k$를 수행하면 v_j로부터 역으로 $n\times2$ 개의 정점을 제거하고 제거된 기호정점의 T에서의 원소 번호들과 k번 규칙에서 좌변 형태구문 태그를 내용으로 하는 새로운 원소 tl를 생성한다. 그리고 새로운 정점 v_l를 만들고 e_{il}로 연결한다. 그리고 상태번호 i와 tl의 태그로 GOTO 테이블을 참조하여 다음 상태 q를 결정하고, 상태정점 v_q을 만들고 e_{lq}로 연결한다. 만약 v_j에서 다른 액션과 공유되고 있는 경로가 존재한다면 공유되는 부분은 제거하지 않는다. 새로 생성된 상태정점에 대해 다시 액션을 결정한다.

쉬프트(shift) : 활성상태정점 목록에서 한 정점 v_i에 대한 형태소 m과 이에 대한 품사 c로부터의 액션 sh$j$를 수행하는 경우, T에 '[c 'm']'라는 원소 t_k가 다른 활성상태정점에 의해 만들어지지 않았다면 새로이 만들고, G에 기호정점 v_k와 상태정점 v_j를 만들어 e_{ik}와 e_{kj}로 연결한다. 모든 활성상태정점들에 대해 쉬프트 연산이 완료된 후, 병합 연산이 완료되면 정점들은 활성상태정점 목록에서 제거된다.

분할(split) : 파싱테이블에서 한 항목에 여러 개의 액션이 존재하거나, 형태론적 변형에 대한 복원이 두 가지 이상이거나, 혹은 형태소가 다품사일 경우 그래프 구조화 스택의 경로는 두 개 이상으로 분할된다. 그러나 다른 동일 상태정점에서 하나의 형태소에 대해 서로 다른 품사가 동일한 규칙에 의해 리듀스된다면 분할되지 않는다.

병합(merge) : 동일 상태로 쉬프트되는 정점들이 동일 상태정점과 직접 연결된다면 하나로 병합된다. 또한 동일 품사로 완성되는 리듀스 연산에 대해 연결 에지의 시작점이 동일 정점이면 하나의 t_i 원소를 생성하고 리듀스 연산 횟수만큼의 트리 노드 목록을 t_i에 넣고, 이 후로부터의 남은 리듀스 연산을 수행한다.

병합 과정은 애매성 있는 구조에 대해 부분적으로 공유될 수 있는 부분을 묶음으로써 공간과 시간에 대한 복잡도를 줄임과 동시에 애매성을 간결하게 표현하는 역할을 수행하는 중요한 과정이다. 분할 과정은 별도의 과정이 존재하지 않고 쉬프트와 리듀스 액션이 수행될 때 발생한다.

어절 '사시는'에 대한 입력 자료구조가 다음과 같을 때, <그림 3>부터 <그림 11>까지는 분석 과정의 예를 보인다.

입력어절: ㅅㅗ ㅏㄴ ㅅㅗ ㅣㄴ ㄴㅗ ㅡㄴ ㄴㄷ $
position: 0 1 2 3 4 5 6 7

<그림 3>의 단계 1에서는 유한상태변환기의 기본 규칙과 규칙 (10.a)를 천이한 '사'와 '살'에 대한 품사가 참조된 후, 기능성 구분문자의 문맥제약으로 '살(nc)'이 여과된 모습이다. 여과되지 않은 나머지에 대해서는 파

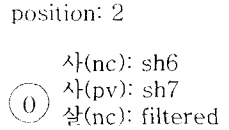


그림 3. 단계 1
Fig. 3. Step 1.

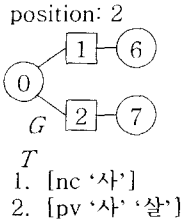


그림 4. 단계 2
Fig. 4. Step 2.

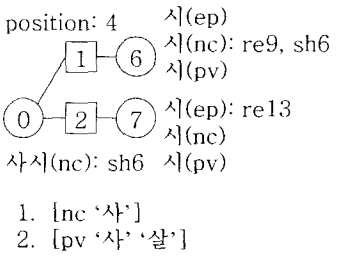


그림 5. 단계 3
Fig. 5. Step 3.

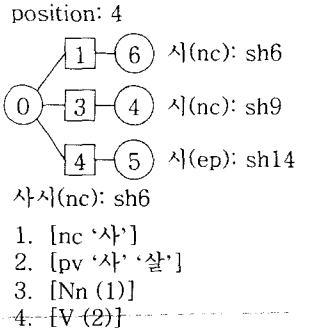


그림 6. 단계 4
Fig. 6. Step 4.

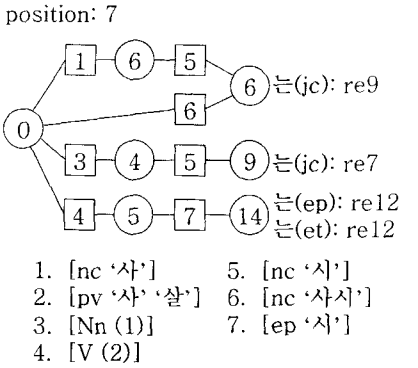


그림 7. 단계 5
Fig. 7. Step 5.

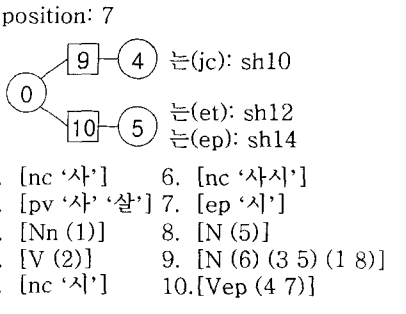
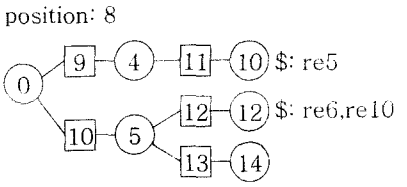


그림 8. 단계 6
Fig. 8. Step 6.

상태이블로부터 액션 sh6, sh7이 결정되고 활성상태정점 목록에 추가된다. 단계 2는 쉬프트와 병합 연산을 거친 후의 모습이다.

단계 3에서는 다시 '$ㅅ_0$'와 '$ㅣ_n$'의 천이를 거쳐 사전 품사를 참조하고, 파싱테이블로부터 상태정점 6에서는 'nc'에 대해 re9와 sh6이 결정되고 상태정점 7에서는 'ep'에 대해 re13이 결정된 후, 활성상태정점 목록에 추가된다. 이와 동시에 기본 규칙에 의해 TRIE 사전의 또 다른 경로를 천이해 온 '사시'라는 명사는 상태정점 0에 대해 액션 sh6가 결정되고, 활성상태정점 목록에 추가된다. 여기서 '시'는 (10.a)에 적용을 받아 어휘형 '실'을 생성하기도 하지만, 이 어휘는 접두사와 명사로만 사용되는 것으로 경계구분 문자 '1'을 만나 모두 여과되므로 그림에는 표시되지 않았다. 단계 3에서 상태정점 6에 대해 re9 연산을, 상태정점 7에 대해 re13 연산을 수행한 후, 각각에 대해 다시 액션을 결정하고 활성상태정점 목록에 추가되고 나면 단계 4와 같이 된다.

단계 4에서 활성상태정점 6과 활성상태정점 0의 sh6에 의해 병합된 새로운 상태정점 6을 만든다. 또한 상태정점 4, 5의 sh9, sh14를 수행한 후, 새로 생성된 상태정점 6, 9, 14에 대해 액션을 결정하여 활성상태정점에 추가하고 나면 단계 5와 같은 모습이 된다. 단계 5에서는 활성상태정점 6에서 기호정점 6으로 이어지는 경로의 re9와 활성상태정점 9의 re7이 동일 형태구문 태그 'N'을 완성하고, 동일 상태정점 0으로 시작하므로 하나의 트리 원소 '[N (6) (3 5)]'를 만든다. 상태정점 0과 형태구 N에 대해 GOTO 테이블을 참조하여 상태정점 4를 만들게 되며, 이 정점에 대한 액션 sh10을 결정하게 된다. 활성상태정점 6에서 기호정점 5에 이르는

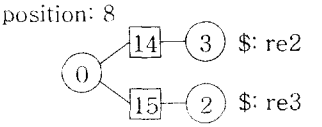


1. [nc '사']
2. [pv '사' '살']
3. [Nn (1)]
4. [V (2)]
5. [nc '시']
6. [nc '사시']
7. [ep '시']
8. [N (5)]
9. [N (6) (4 6) (1 8)]
10. [Vep (4 7)]
11. [jc '는']
12. [et '는']
13. [ep '는']

그림 9. 단계 7
Fig. 9. Step 7.

경로의 re9를 수행하고 나면 트리 원소 '[N (5)]'가 만들어지고 활성상태정점 6과 기호정점 5, 그리고 그들의 연결 에지들이 제거되고 상태정점 6과 에지로 연결되는 기호정점 8이 만들어진다. GOTO 테이블을 참조하여 기호정점 8과 에지로 연결되는 상태정점 15가 만들어진다. 그리고 상태정점 15에 대한 액션이 파싱테이블로부터 re8로 결정된다. 이에 대한 리듀스 연산을 수행할 경우 단계 5에서 활성상태정점 6과 기호정점 6로 이어지는 경로에 대한 리듀스 연산이 완성하는 형태구문 태그와 동일하고, 동일 상태정점 0으로 시작하므로 t_9과 병합된다. 나머지 상태정점 14의 re12를 수행한 후 액션을 결정한 모습이 단계 6이다.

단계 7은 단계 6에서 각 활성상태정점들에 대한 액션을 수행한 후 새로이 목록에 추가된 활성상태정점들에 대한 액션을 결정한 모습이다. 단계 7에서의 re5, re6은 동일 형태구문 태그 'Nj'로 완성되고 시작 상태정점이 동일하므로 병합되어 하나의 t_{14} '[Nj (9 11) (10 12)]'를 만들고 상태정점 0으로부터 기호정점 14를 만든다. 상태정점 0과 기호정점 14에 대해 GOTO 테이블을 참조하여 기호정점 14와 연결되는 상태정점 3을 만들어내고, 액션 re2가 결정된다. 단계7에서 re10까지 수행한 후, 액션을 결정하면 단계 8과 같이 된다. 단계 8에서도 re2와 re3는 동일 형태구문 태그 'W'를 완성하고 동일 시작 상태정점을 가지므로 t_{16}으로 병합된다. 상태정점 0으로부터 기호정점 16을 만들고 GOTO 테이블을 참조하여 상태정점 1을 만든다. 그 후 액션은 'acc'로 결정되고 단계 9와 같이 'acc'외에 더이상 수행할 액션이 존재하지 않으므로 형태구문 파싱은 완성된다.



1. [nc '사']
2. [pv '사' '살']
3. [Nn (1)]
4. [V (2)]
5. [nc '시']
6. [nc '사시']
7. [ep '시']
8. [N (5)]
9. [N (6) (3 5) (1 8)]
10. [Vep (4 7)]
11. [jc '는']
12. [et '는']
13. [ep '는']
14. [Nj (9 11) (10 12)]
15. [W (10 12)]

그림 10. 단계 8
Fig. 10. Step 8.

T

1. [nc '사']
2. [pv '사' '살']
3. [N (1)]
4. [Vep (2)]
5. [nc '시']
6. [nc '사시']
7. [ep '시']
8. [N (5)]
9. [N (6) (3 5) (1 8)]
10. [Vep (4 7)]
11. [jc '는']
12. [et '는']
13. [ep '는']
14. [Nj (9 11) (10 12)]
15. [Ve (10 12)]
16. [W (14) (15)]

그림 11. 단계 9
Fig. 11. Step 9.

<그림 12>는 단계 9에서 완성된 압축공유숲트리 T의 내용을 괄호 붙은 트리 형태로 출력한 모습이다. T와 같은 계층적 분석 결과는 다양한 형태로 활용할 수 있는데, 본 논문에서의 시스템에서는 두 가지 출력형태를 제공한다. 첫 번째 형태는 전체 트리 구조를 출력한 것으로 1~5는 '[W (14)]'에 의한 것이고 6과 7은 '[W (15)]'에 의한 것이다. 두 번째는 어절수준의 형태구문 태그는 출력하지 않고 형태소 태그를 포함하여 어휘수준 이하의 태그만을 출력하였다. 이 두 가지 출력형태는 구문 분석에서 구문 요소, 혹은 단어의 기준을 어떻게 둘 것인가에 따른 활용 예로 볼 수 있으며, 첫 번째는 어절 전체를 하나의 구문 요소로 삼은 것으로 파생과 굴절이 모두 어휘부에서 일어난다는 강어휘론적가설에 따른 것이다. 두 번째는 굴절은 통사부나 그 이후의 단계에서 이루어질 것이라는 약어휘론적가설에 따른 것이다. 이 두 종류의 출력형태는 단어통사규칙을 통사적접사의 포함 유무에 따라 어휘 수준의 형태구 태그와 어절 수준의 형태구 태그로 나누고 어휘수준에

input> 사시는
1. (Nj (N 사시/nc) 는/jc)
2. (Nj (N (N 사/nc) 시/nc) 는/jc)
3. (Nj (N 사/nc (N 시/nc)) 는/jc)
4. (Nj (Vep (Vep 사/pv) 시/ep) 는/et)
5. (Nj (Vep (Vep 살/pv) 시/ep) 는/et)
6. (Ve (Vep (Vep 사/pv) 시/ep) 는/et)
7. (Ve (Vep (Vep 살/pv) 시/ep) 는/et)

input> 사시는*
1. (N 사시/nc) + 는/jc
2. (N (N 사/nc) 시/nc) + 는/jc
3. (N 사/nc (N 시/nc)) + 는/jc
4. 사/nc + 시/ep + 는/et
5. 살/pv + 시/ep + 는/et

그림 12. 형태소 분석 결과
Fig. 12. Results of morphological analysis.

서의 단어 형성과 어절 수준의 단어 형성을 구분하여 만들었기 때문에 가능한 것이다.

Ⅳ. 결론 및 고찰

제안하는 형태소 분석 방법론은 구조분석의 측면에서 문장분석과 유사하지만 단위의 경계를 알 수 없다는 점에서 다르다. 따라서 분석을 위해서는 형태소의 경계를 알아냄과 동시에 구조분석이 이루어져야 한다. 본 논문에서는 통합된 하나의 TRIE 사전과 기능성 구분 문자를 사용하는 변형된 2단계형태론을 통하여 형태소의 경계를 알아냈다. 2단계형태론의 형식화에 대한 저수준의 표현인 유한상태변환기의 넓이-우선 탐색을 통하여 모든 형태소 경계를 찾아나감과 동시에 발견된 형태소는 GLR법으로 단어통사적 결합 합법성을 검사하였다.

본 논문에서 제안한 방법론에서는 분석 대상을 띄어쓰기 단위의 어절 범위로 한정하고 있다. 그러나 띄어쓰기는 관용어나 음운적 단어, 복합명사와 같은 단어에 대해 단어 경계에 대한 정확한 정보를 제공하지 못 할 수도 있다. 관용어나 음운적 단어에 대해서는 사전 등재와 간단한 규칙으로 해결 가능하고, 복합명사에 대해서도 명사의 나열을 하나의 단어로 묶는 방식으로 해결할 수도 있다. 그러나 시간을 나타내는 명사는 부사적 용법으로도 쓰일 수가 있어 단순한 명사의 나열만으로는 불충분하다. 이 부분에 대해서는 추가적인 연구가 필요할 것이다. 앞으로 남은 또 다른 과제는 하위범주화나 어휘자질에 대한 형식화와 자질연산을 도입하여 완전한 형식화를 제공하여야 할 것이다.

참 고 문 헌

[1] Charles W. Morris, *Foundations of the Theory of Signs*, Chicago : Chicago University Press, 1938.

[2] Elisabeth O. Selkirk, *The Syntax of Words*, MIT Press, 1982.

[3] 안상철, 형태론, 민음사, 1998

[4] Morris Halle, "Prolegomena to a theory of word formation," *Linguistic Inquiry*, vol. 4, pages 3~16, 1973.

[5] Noam Chomsky, "Remarks on normalization," In R. Jacobs and P. Rosenbaum (eds.) *Readings in English Transformational Grammar*, Waltham, MA : Blaisdell, pages 184-221, 1970.

[6] S. R. Anderson, "Where's morpholgy," *Linguistics Inquiry*, vol. 13, pages 15~44, 1982.

[7] N. Fabb, *Syntactic Affixation*, PhD Thesis, MIT, 1984.

[8] M. Baker, "The mirror principle and morphosyntatic explanation," *Linguistic Inquiry* vol. 16, pages 373~416, 1985.

[9] M. Baker, *Incorporation : a Theory of Grammatical Function Changing*, Chicago : Chicago University Press, 1988.

[10] Richard W. Sproat, *On Deriving the Lexicon*, PhD Thesis, MIT, 1985.

[11] David J. Weber, H. Andrew Black, Stephen R. McConnel, *AMPLE : A Tool for Exploring Morphology*, Occasional Publications in Academic Computing 15. Dallas, TX: Summer Institute of Linguistics, 1988.

[12] Graeme D. Ritchie, Graham J. Russell, Alan W. Black, Stephen G. Pulman, *Computational Morphology : Practical Mechanisms for the English Lexicon*, Cambridge, MA : MIT Press, 1992.

[13] Richard W. Sproat, *Morphology and Computation*, Cambridge, MA : MIT Press, 1992.

[14] Evan L. Antworth, *User's Guide to PC-KIMMO Version 2*, Dallas, TX : Summer Institute of Linguistics, 1995.

[15] 강진목, *현대국어의 단어 형성연구*, 박사학위논문, 전남대학교, 1994

[16] 시정곤, *국어의 단어형성 원리 - 수정판*, 한국문화사, 1998

[17] H. Borer, "On the morphological parallelism between compounds and constructs," *Yearbook of Morphology*, vol. 1, pages 45~66, 1988.

[18] John Andrew Carroll, *Practical Unification-based Parsing of Natural Language*, PhD Thesis, University of Cambridge, 1993.

[19] E. Williams, "On the notions 'lexically related' and 'head of a word'," *Linguistic Inquiry*, vol. 12, pages 245~274, 1981.

[20] 최기선, 남영준, 김진규, 한영균, 박석문, 김진수, 이춘택, 김덕봉, 김재훈, 최병진, "한국어정보베이스를 위한 형태 통사 태그 표준에 관한 연구", *인지과학*, 제 7권, 4호, 43~61, 1996

[21] 안동언, *기계번역을 위한 한국어 해석에서 형태소로부터 구문요소의 형성에 관한 연구*, 석사학위논문, 한국과학기술원, 1986

[22] 차준경, 강범모, "형태소 분석 말뭉치의 파생명사 처리", *제12회 한글 및 한국어 정보처리 학술발표논문집*, 2000

[23] 전상범, *형태론*, 한신문화사, 1995.

[24] Andrew Spencer, *Morphological Theory : an Introduction to Morphology in Generative Grammar*, Oxford : Blackwell, 1991.

[25] R. Lees, *The Grammar of English Nominalizations*, The Hague : Mouton, 1960.

[26] M. Aronoff, *Word Formation in Generative Grammar*, Cambridge, MA : MIT Press, 1976.

[27] S. Scalise, *Generative Morphology*, Dordrecht : Foris, 1984.

[28] Masaru Tomita, *Efficient Parsing for Natural Language*, Boston, MA : Kluwer, 1986.

[29] Masaru Tomita, *Generalize LR Parsing*, Kluwer, 1987.

[30] K. Koskenniemi, "Two-level model for morphological analysis," In *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, pages 683~685, Karlsruhe, 1983.

[31] 김성용, *Tabular parsing 방법과 접속 정보를 이용한 한국어 형태소 분석기*, 석사학위논문, 한국과학기술원, 1986

[32] 이은철, *CYK법에 기반한 한국어 형태소 분석에서의 개선기법*, 석사학위논문, 포항공과대학, 1992

[33] K. Koskenniemi, "Compilation of automata from morphological two-level rules," In *Proceedings of the 5th Scandinavian Conference of Computational Linguistics*, University of Helsinki, pages 143~149, 1986.

[34] Alfred V. Aho, Jeffrey D. Ullman, *The Theory of Parsing, Translation and Compiling, Volumn 1 : Parsing*, Prentice-Hall, Englewood Cliffs, N. J., 1972.

저 자 소 개

金 東 柱(正會員)

1996년 2월 : 한양대학교 전자계산학과 졸업(공학사). 1998년 2월 : 한양대학교 대학원 전자계산학과 졸업(공학석사). 1998년 3월~현재 : 한양대학교 대학원 컴퓨터공학과 박사과정 재학중. <주관심분야 : 한국어 형태소 및 구문 분석, 구문론, 기계번역, 정보검색>

金 漢 宇(正會員)

1975년 2월 : 한양대학교 전자공학과 졸업(공학사). 1978년 2월 : 한양대학교 대학원 전자공학과 졸업(공학석사). 1980년 : 일본 동경대학 정보공학과 연구원. 1981년~현재 : 한양대학교 컴퓨터공학과 교수. 1999년~현재 : 정보통신부 문자방송기술협회 이사. <주관심분야 : 맞춤법검사, 기계번역, 한국어정보처리>