음절단위 CYK 알고리즘에 기반한 형태소 분석기 및 품사태거

권오욱, 정유진, 김미영, 류동원, 이문기, 이종혁

포항공과대학교 컴퓨터공학과

Korean Morphological Analyzer and Part-Of-Speech Tagger Based on CYK Algorithm Using Syllable Information

Oh-Woog Kwon, Yujin Chung, Mi-Young Kim, Dong-Won Ryu, Moon-Ki Lee, Jong-Hyeok Lee

Dept. of Computer Science & Engineering, POSTECH

요약

본 논문에서는 포항공과대학교 지식 및 언어공학연구실에서 개발한 한국어 형태소 분석기 및 품사 태거에 대하여 설명한다. 먼저, 음운 축약 현상이 많은 한국어에 적합한 음절단위 CYK 알고리즘을 제안한다. 그리고, 복합명사 및 복합동사에 대한 처리와 실제 문서에서 빈번히 발생하는 띄어쓰기 오류 처리에 대한 방법론을 설명하고 미등록어에 대한 처리 방안을 제시한다. 품사 태거에서 사용된 방법론과 태그 집합간 매핑, 그리고 명사 추출기에 대해 기술한 후 마지막으로 MATEC'99 를 위한 준비과정에서 발생한 표준안과 우리 시스템 사이의 차이점을 나열 및 분석하고 간단히 MATEC'99 를 통해 얻은 실험 결과와 평가를 하고자 한다.

1. 서론

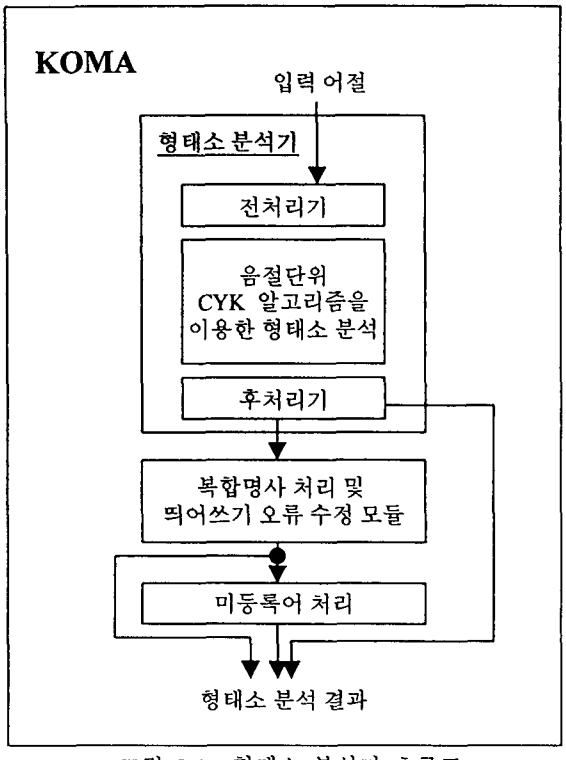
정보화시대에 대부분의 정보가 담겨져 있는 전자 문서들에서 필요한 정보를 어떠한 형태로도 추출하기 위해서는 당연히 자연언어처리가 필요하다. 앞으로는 자신의 언어를 전산적으로 잘 처리하는 문화만이 그 문화의 보존 및 발전이 효과적으로 이루어지리라고 생각한다. 이러한 이유에서 한국어 처리의 발전에 대한 열의로 우리는 MATEC'99 대회에 한국어 형태소 분석기, 품사태거와 명사추출 부분에 참가하였고, 본 논문에서는 우리가 참가한 모든 분야의 시스템들에 대해 설명하고 MATEC'99 대회를 위한 준비, 참가 소감 및 실험 결과에 대해서 간략히 언급하고자 한다.

2 장에서는 음운 현상이 빈번한 한국어에 적합한 음절 단위 CYK(Choke-Young-Kasami) 알고리즘에 기반한 한국어 형태소 분석기의 기본 모듈을 설명한다. 3 장에서는 한국어 형태소 분석에서 처리 곤란한 복합명사 및 띄어쓰기 오류 수정에 관한 방법과 미등록어 처리에 대한 방안을 제안한다. 4 장에서는 명사 추출기 및 통계기반 품사 태거에 대하여 설명한다. 5 장에서는 MATEC'99 대회를 위한 우리의 준비과정과 대회를 통해서 얻은 소감 및 성과와 실험 결과를 평가한다.

2. 음절단위 CYK 알고리즘에 기반한 형태소 분석기

<그림 2-1>은 음절 단위 CYK 알고리즘을 이용한 한국어 형태소 분석기인 KOMA(KOrean Morphological Analyzer)의 전체 시스템에 대한 처리 흐름도이다. KOMA 는 크게 3 가지 모듈들로 구성된다. 첫 번째 모듈은 형태소 분석 사전의 등록어만으로 처리할 수 있는 기본적인 형태소 분석기이다. 이 모듈에 의해서 분석되지 않는 입력 어절은 두 번째 모듈에 의해서 복합명사 처리 및 띄어쓰기 오류를 수정한다. 이 모듈에서는 단일 명사들이 한 어절에서 붙여 쓸 수 없다고 가정하여 복합명사도 일종의 띄어쓰기 오류로 파악하여 하나의 알고리즘에 의하여 복합명사 처리와 띄어쓰기 오류 수정을 동시에 가능하게 한다. 마지막 모듈에서는 사전에 등록되지 않은 형태소들에 의해 발생하는 미분석 어절을 처리한다[7][8].

다음 장에서 띄어쓰기 오류 수정과 미등록어 처리에 대한 설명하고, 이 장에서는 기본적인 형태소 분석기에



<그림 2-1> 형태소 분석기 흐름도

대하여 설명한다. KOMA 의 형태소 분석기는 크게 전처리기, 음절 단위 CYK 알고리즘을 이용한 한국어 형태소 분석 모듈과 이의 결과에 대한 불규칙 용언 처리를 하는 후처리기로 나누어진다.

2.1 전처리기

전처리기는 음절화 단계, 한자-한글 변환과 수사 오토마타로 3 단계로 이루어져 있다. 전처리기에서 하는 일은 한국어 형태소 분석에 들어가기 위한 준비 작업으로 형태소 분석 알고리즘을 효율적으로 작동하기 위한 단계이고 또한 형태소 분석 알고리즘에서 처리할 수 없는 처리를 미리 하는 단계이다. 이러한 전처리기에 의하여 음절 단위 CYK algorithm 을 이용하는 형태소 분석 엔진의 간단성을 유지할 수 있다.

2.1.1 음절화 단계

전처리기의 음절화 단계는 한국어 어절을 입력으로 하여 음절 단위의 구조로 변경하여 준다. 이 때, 본 논문에서의 음절에 대한 정의는 다음과 같다.

- 음절

① 한국어 한 음절을 음절이다.
② 연속되는 숫자 문자열은 한 음절이다.
③ 연속되는 알파벳 문자열은 한 음절이다.
④ 연속되는 한자 문자열은 한 음절이다.
⑤ 하나의 기호는 한 음절이다.

음절화 단계는 다음 단계에서 처리할 입력 단위를 나누어 줌으로써, 단계 별로 쉽게 처리 가능하게 하는데, 그 의의가 있다. 음절화 단계에서는 위에서 정의한 음절에 따라 전처리에서 처리하는 모듈이 달라진다. 숫자인 경우에는 숫자 수사 오토마타를 거쳐서 하나의 수사 음절로 인식한다. 그리고 문장기호/특수기호에 대한 음절에는 그 음절에 해당하는 품사 정보를 부여하여 형태소 분석 시, 그 기호가 문장기호 또는 특수기호로 사용 가능 타당성을 검사한다. 그리고, 이러한 문장기호는 형태소 분석의 애매성을 줄이는 효과를 보일 수도 있다.

2.1.2 한자-한글 변환

한자 문자열로 구성된 음절인 경우에는 한자-한글 변환을 하여 원 한자 문자열과 그에 해당하는 한글 독음을 추출한다. 이 때, 한국어 두음법칙에 따라 한자를 한글로 독음하게 된다. 두음법칙이 한자어 접두사를 단어의 처음으로 생각하지 않고 처리함으로 한자어 접두사가 필요하다. 또한 이러한 두음법칙은 고유명사와 단위성 의존명사에서는 예외이므로, 형태소 해석용 사전을 참조하여 고유명사이거나 단위성 의존명사인가를 검사하여 완벽한 독음이 되도록 한다.

2.1.3 수사 처리

수사 처리는 한글 및 한자어 수사에 대한 처리와 함께 품사 정보를 넘겨 준다. 이 때, 수사 처리는 오토마타를 이용해서 한다. 한글 음절과 숫자 음절을 오토마타의 입력으로 하여 수사인지 아닌지를 판명하고, 오토마타의 결과로 수사에 대한 형태소 품사 정보를 준다.

본 논문에서 수사 뿐 아니라, "하루, 이틀, ..., 열나흘, ..." 형식의 날짜 명사에 대한 처리를 위한 오토마타도 이 부분에서 처리하도록 하고 있다. 이런 날짜를 사전에 넣는 것은 불가능함으로 이와 같은 오토마타에 의한 처리로 사전 작업을 줄일 수 있다. 그리고, 이런 나열씩의 관형사 역시 이 부분에서 오토마타로 취급하여

사전 작업을 줄인다.

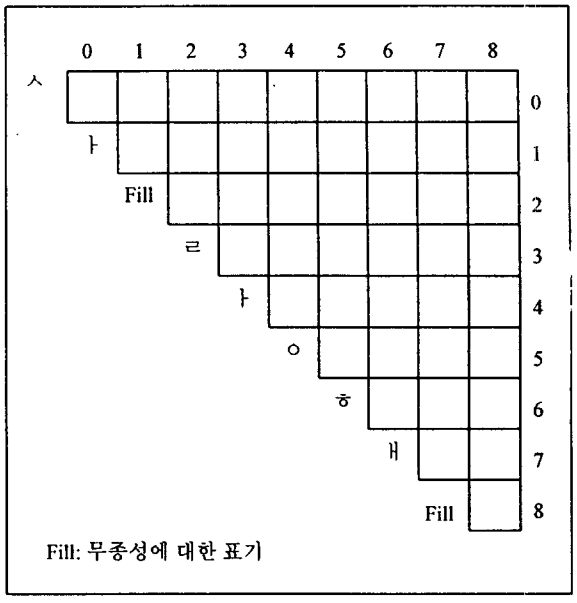
2.2 음절 단위 CYK 알고리즘을 이용한 형태소 해석

형태소 분석이란 문장을 구성하고 있는 최소의 의미 단위인 형태소를 추출하는 것을 말한다. 일반적으로 형태소 분석의 전산학적 관점에서 다음 3 가지 과정이 필요하다. 첫째, 형태소 단위로 분할하는 형태소 분할 과정, 둘째, 활용형으로부터 원형 복원 과정, 셋째, 분할된 형태소 간의 결합타당성을 검사하는 과정이다. 이러한 과정들을 통합적으로 수행하기 위한 한국어 형태소 분석 알고리즘이 필요하다. 본 논문에서는 기본 형태소 분석 알고리즘으로 CYK 알고리즘을 이용한 tabular parsing 방법을 이용한다. 그리고 형태소 해석 사전에 의한 형태소 분할의 타당성을 검사하고, 접속 정보를 이용하여 형태소 간의 결합 타당성을 검사하는 방법을 이용한다. 축약에 대한 원형 복원은 CYK 알고리즘에서 대부분 수행하고 불규칙 용언에 대한 복원은 주로 후처리기에서 처리한다.

2.2.1 기존 자소 단위 CYK 알고리즘의 문제점

기존의 자소 단위 CYK 알고리즘을 이용한 한국어 형태소 분석기는 다음과 같은 기본 동작으로 수행된다 [2][7]. <그림 2-2>에 있는 역 삼각형 테이블(table)을 CYK 테이블이라고 한다. 이 테이블에서 i 행 j 열의 셀(cell)을 T(i, j)라고 정의하자. 이 때, 각 셀 T(i, j)가 의미하는 것은 입력어절의 i 번째 자소에서 j 번째 자소까지의 문자열에 대한 형태소 분석 결과와 그에 따른 접속 정보를 가진다. 그리고, 각 음절은 초/중/종성 3 자소로 분리하여 테이블을 유지하기 때문에 한 음절을 위해서 $(3\times3)/2$ 셀이 필요로 하고 n 음절로 이루어진 입력 어절을 처리하기 위해서는 $(3n\times3n/2)$ 셀로 구성된 테이블이 필요하다. 그러므로, 형태소 분석이 끝난 후, T(0, 3n-1)에 최종 형태소 분석 결과를 가지게 된다.

자세한 CYK 알고리즘은 <그림 2-3>에 나타나있다. <그림 2-3>에서 Read_From_Dictionary(i, j)는 i 번째 자소부터 j 번째 자소까지의 문자열에 대한 형태소 품사 및 접속 정보를 사전으로부터 찾아 오는 함수이고, Connection_Check(sub1, sub2)는 sub1 의 형태소와 sub2 의 형태소 간에 어절 내에서 결합이 가능하면 1 을 아니면 0 을 리턴(return)하는 함수이다. 자소 단위 CYK 알고리



<그림 2-2> 자소 단위 CYK 테이블

```
Morph   T[3n][3n];   // CYK table 선언

Algorithm CYK_algorithm(res, n)
Morph   *res; // 형태소 분석 결과 및 접속 정보 저장구조
int   n; // 입력 어절 길이
{
  Morph   *sub1, *sub2;
  int   i, j;

  for(i=3n-1; i<=0; i--) {
    for(j=i; j < 3n – 1; j++) {
        sub1 = T[j+1][3n-1];
        if(sub1 != NULL)
            sub2 = Read_From_Dictionary(i, j);
        if(Connection_Check(sub1, sub2))
            T[i][3n-1] = sub1 + sub2;
    }
  }
  res = T[0][3n-1];
}
```

<그림 2-3> 자소 단위 CYK 알고리즘

즘에서 중간 결과들인 T(i, 3n-1)은 T(i, j)와 T(j+1, 3n-1) 간의 결합 검사에 의하여 생성되거나, 자소 i 에서 마지막 자소까지의 문자열을 사전 검사에 의해서 생성된다.

<그림 2-2>와 <그림 2-3>에서 보듯이 어절 "사랑해"를 입력으로 받았을 경우, "해"의 복원인 "하+어"로 보기 때문에 축약 현상이 자소 단위에서 발생하는 음운현상이기 보다는 음절단위에서 발생하는 음운현상으로 보

는 것이 타당하다. 또한 자소 단위로 CYK 테이블 구조를 표현한 경우, 축약의 복원에 대한 표현이 불가능하여 복원된 형태인 "사랑하여"를 다시 형태소 분석해야만 축약에 대한 복원이 가능하다.

기존의 자소 단위 CYK 알고리즘은 축약현상 및 불규칙현상을 처리하기에 어려움을 가지고 있다. 또한 축약이나 불규칙 현상이 있는 입력 어절에서 미등록어가 발생할 경우에 미등록어 추정에서 정확한 미등록어 처리를 하기 어렵다[1][7][8]. 본 논문에서는 기존 자소 단위 CYK 알고리즘의 문제점을 해결한 음절 단위의 CYK 알고리즘을 소개한다.

2.2.2 음절 단위 CYK 알고리즘

음절 단위 CYK 알고리즘과 자소 단위 CYK 알고리즘의 가장 큰 차이는 CYK 테이블의 각 셀에 대한 표현에 있다. 음절 단위 CYK 알고리즘에서의 테이블은 각 셀이 입력 어절의 각 음절에 대응하는 점이 차이가 있다. 또한 축약 음절을 정의에 의해서 달라 진다.

한국어에서 일반적으로 축약 현상은 두 음절이 줄어서 한 음절이 되는 현상을 말한다. 하지만, 경우에 따라 여러 음절이 축약된 경우도 있다. 본 논문에서는 우선 축약의 유형을 다음과 같이 분류한다.

1. 한 형태소 내의 음절들에 의한 축약: 예) "껜(→께는)" 조사 축약, "뵈다(→보이다)" 용언 축약, "그렇게(→그러하게)" 부사 축약
2. 각기 다른 형태소의 음절들에 의한 축약: 이 경우에 다시 2 가지 경우로 나눌 수 있다.
   2-1. 한 어절 내에서 결합 가능한 형태소들인 경우: 예) "그걸(→그것을)" 체언+조사 축약, "갔다(→가았다)" 용언+어미 축약
   2-2. 한 어절 내에서 결합 불가능한 형태소들인 경우: 예) "랍니까(→라고 합니까)" 어미+용언 축약, "재(→저 아이)" 관형사+체언 축약

2-2 축약 유형인 경우, 그 음절을 복원하더라도 일반적인 형태소 분석에서 결합이 가능하지 않기 때문에 이러한 축약 유형의 문제들은 사전에 복원 유형을 등록하여 해결하는 방법을 선택한다. 나머지 경우에 대해서는 복원 후, 접속 검사에 문제가 없어서 새로이 사전 등록없이 문제를 해결할 수 있다. 본 논문에서는 주로 축약 유형 1 과 2-1 에 대해서만 축약된 음절을 복원하여 형태소 분석을 한다.

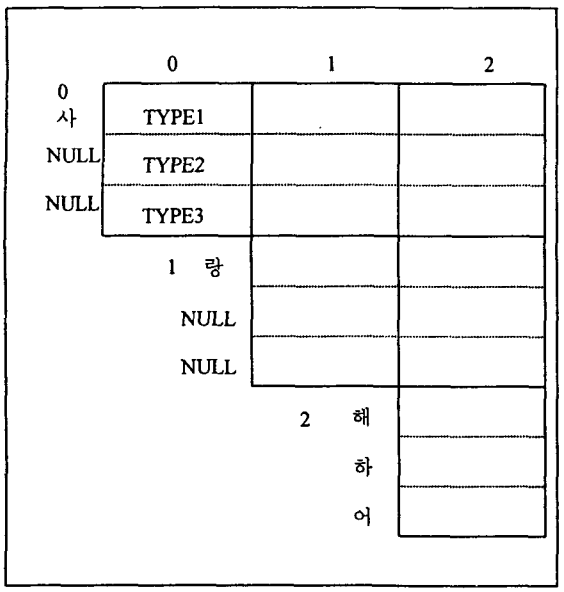
본 논문에서는 축약 음절이란 두 가지 이상의 형태소들의 음절들이 결합되어 하나의 음절된 음운현상이 발생한 음절을 말한다. 예를 들어, "그걸(→그것으로)"에서 축약 음절은 "걸"이다. 그 축약 음절이 2-1 유형일 경우, 어휘형태소의 마지막 몇 음절과 문법형태소 처음 몇 음절이 결합되는 경우이고 만약 유형 1 축약 음절일 경우는 어휘형태소나 문법형태소 중에 하나에서 처음이나 마지막 몇 음절이 하나의 음절로 결합되는 경우이다. 그러므로, 축약 음절이 있는 지점은 항상 두 개 이상의 형태소들이 분리되는 지점을 의미한다. 축약 음절에 의해 분리되는 음절도 두 개의 형태소들로 분리되는 복원 음절들로 표현 가능하다. 우리는 이러한 표현을 음절 축약 정보라고 부르고 다음과 같이 표기한다.

[축약 음절(Ei): 어휘형태소의 마지막 몇 음절($E1_i$), 문법형태소의 마지막 몇 음절($E2_i$)]

축약 유형 1 인 경우에는 한 형태소에 대한 복원이 가능하기 때문에 나머지 형태소에 대한 복원 정보를 "NULL"로 정의할 수 있다. 즉, "뵈"의 경우, [뵈, 보이, NULL]로 정의할 수 있다. 하지만, 빠른 실행과 정보의 저장 관점에서 사전에 축약형을 등록하여 처리하는 것이 더 효과적일 수 있다. 그러므로, 본 형태소 분석기에서는 축약 유형 1 인 경우는 사전에 등록하여 처리하고 있다. 그리고, 각 어휘형태소나 문법형태소에 그 음절이 나타나는 품사를 표현해서 오분석이 발생하지 않도록 할 수 있다. 이러한 제한 조건은 대부분의 축약음절이 용언과 어미에서 결합되는 경우가 많기 때문에, 제한 조건이 무시하면 용언과 어미의 결합으로 볼 수 있다.

축약 음절 정보를 사용하여 자소 단위 CYK 알고리즘의 문제점을 해결하기 위한 음절 단위 CYK 알고리즘에서 이용하는 테이블 자료 구조는 <그림 2-4>와 같이 표현할 수 있다. 자소 단위의 테이블과 달리 음절 단위의 테이블에서는 각 셀 T(i, j)는 다시 3 개의 서브셀 $T(i, j, 0 \le k \le 2)$로 이루어진다. 각 서브셀 $T(i, j, 0 \le k \le 2)$에 대한 정의는 다음과 같이 할 수 있다.

- T(i, j, 0): 입력 어절에서 j 번째 음절이 축약되지 않은 경우(즉 원래 입력된 음절 E_j), i 번째 음절부터 j 번째 음절까지의 형태소 분석 결과를 가진다. 이러한 서브 셀을 TYPE1 이라고 한다.
- T(i, j, 1): 입력 어절에서 j 번째 음절이 축약 음절인 경우, i 번째 음절부터 j 번째 음절의 어휘형태소의 마지막 복원 문자열 $E1_j$까지의 형태소 분석 결과를 가지고 TYPE2 라고 한다.



<그림 2-4> 음절 단위 CYK 테이블

- T(i, j, 2): 입력 어절에서 i 번째 음절이 축약 음절인 경우, i 번째 음절에 대한 문법형태소의 첫 복원 문자열 E2;부터 j 번째 음절까지의 형태소 분석 결과를 가지고 TYPE3 라고 한다.

예를 들어, "사랑해"라는 어절이 입력일 경우, T(2, 2, 0)에는 "해"에 대한 형태소 분석 결과가, T(2, 2, 1)에는 "하"에 대한 형태소 분석 결과, 그리고, T(2, 2, 2)에는 "어"에 대한 형태소 분석 결과가 저장된다. 또한 T(0, 2, 0)은 "사랑해", T(0, 2, 1)은 "사랑하"에 대한 결과가 저장된다. TYPE1 에 해당하는 T(0, 2, 0)에는 T(0, 2, 1)과 T(2, 2, 2)의 결과가 결합하여 저장된다. 축약 복원 정보에 의한 차이만을 고려하면 자소 단위 CYK 알고리즘과 유사한 음절 단위 CYK 알고리즘을 구현할 수 있다. <2-5>는 음절단위 CYK 알고리즘을 나타내고 있다.

음절 단위 CYK 알고리즘에서 Read_From_Dictionary 함수가 자소 단위 CYK 알고리즘과 달리 각각에 유형에 적합한 문자열에 대한 형태소 정보를 사전에서 가져온다.

축약 음절 정보는 지정사가 들어있는 3 가지 경우에는 복원 되는 형태소들의 수가 3 가지가 된다. 예를 들면, "그겁니까(→ 그것입니까)"에서는 축약 음절 "겁"이 "것+이+ㅂ"으로, "그걸까(→그것일까)"에서는 축약 음절 "걸"이 "것+이+ㄹ"로, "그건가(→그것인가)"에서는 축약 음절 "건"이 "것+이+ㄴ"으로 음절 복원하여야 한다. 즉 지정사가 복합된 음절 "입, 일, 인"이 두 가지 형태소 분리되기 때문에 축약 음절 정보로는 저장할 수 없다. 이러한 경우는 특별히 처리를 한다.

축약 음절 정보를 잘 사용하면, 불규칙 용언에 대한 처리도 가능하다. 예를 들어, "아름다운"을 분석하기 위해서 축약 음절 정보 [다, 답, NULL]과 [운, NULL, ㄴ]을 추가하여 "ㅂ" 불규칙에 대한 처리를 할 수 있다. 물론 위의 두 축약 음절 정보에 제한 조건을 주어야 한다. 하지만, 본 형태소 분석기에서는 이러한 처리로 불규칙 현상을 처리하지 않는다. 후처리기에서 보다 상세하게 설명한다.

```
Morph  T[n][n][3];  // CYK table 선언

Algorithm CYK_algorithm(res, n)
Morph  *res; // 형태소 분석 결과 및 접속 정보 저장구조
int  n; // 입력 어절 길이
{
  Morph  *sub1, *sub2;
  int  i, j;

  for(i=n-1; i<=0; i--) {

    // TYPE 3 에 대한 처리
    for(j=n-1; j >=i; j--) {
        sub1 = T[j+1][n-1][0]; // TYPE1 서브 셀
        if(sub1 != NULL)
            sub2 = Read_From_Dictionary(i, j, TYPE3);
        if(Connection_Check(sub1, sub2))
            T[i][n-1][2] = sub1 + sub2; // TYPE3 에 저장
     }

     for(j=i; j<n; j++) {

        // TYPE1 에 대한 처리
        sub1 = T[j+1][n-1][0]; // TYPE1 서브 셀
        if(sub1 != NULL)
            sub2 = Read_From_Dictionary(i, j, TYPE1);
        if(Connection_Check(sub1, sub2))
            T[i][n-1][0] = sub1 + sub2; // TYPE1 에 저장

        // TYPE2 에 대한 처리
        sub1 = T[j+1][n-1][2]; // TYPE3 서브셀
        if(sub1 != NULL)
            sub2 = Read_From_Dictionary(i, j, TYPE2);
        if(Connection_Check(sub1, sub2))
          T[i][n-1][0] = sub1 + sub2; // TYPE1 에 저장
      }
  }
  res = T[0][n-1][0];  // 최종 형태소 분석결과
}
```

<그림 2-5> 음절 단위 CYK 알고리즘

2.3 후처리기

앞에서 설명한 바와 같이, 형태소 분석기의 주 단계인 형태소 분석에서 형태소 분할과 형태소 결합 타당성을 검사한다. 활용형에 대한 원형 복원 작업은 한국어의 특성인 어미와 조사의 발달로 용언어간이 어미나 조사와 결합하여 변형이 일어날 수가 있다. 이러한 현상을 불규칙활용이라고 한다. 본 논문에서는 12 개의 불규칙 유형을 정의하였다. 이러한 불규칙 유형을 처리하는 방법은 크게 사전을 기반으로 하는 방법과 규칙을 기반하는 방법으로 나눌 수 있다. 사전을 기반으로 하는 방법은 용언의 불규칙 형태를 사전에 등록하였다가 접속 검사가 끝나고 난 후에 원형으로 복원하는 방법이다. 그리고, 규칙을 기반으로 하는 방법은 각 불규칙 형태를 어떤 접속정보의 용언이 어떤 어미와 결합 시에 일어나는가를 미리 규칙으로 만들어 두었다가 원형을 미리 복원하였다가, 접속 검사에서 그 복원이 제대로 이루어졌는가를 검사한다.

본 논문에서는 불규칙 유형에 따라 사전을 기반으로 하는 방법과 규칙을 기반으로 하는 방법으로 나누어 처리한다. 불규칙 형태에 따라 나누어지는데, 'ㅎ'불규칙과 'ㅂ'불규칙의 경우, 규칙 기반 방법과 사전 기반 방법을 모두 적용하여 처리한다. 다음은 12 가지 불규칙 유형과 더불어 처리 방법을 다루고 있다.

- 규칙 기반 방법 : '여' 불규칙, 'ㅎ' 불규칙, 'ㅂ' 불규칙
- 사전 기반 방법 : 'ㄷ' 불규칙, '르' 불규칙, 'ㅅ' 불규칙, 'ㅎ' 불규칙, 'ㅂ' 불규칙, '러' 불규칙, '르' 불규칙, '우' 불규칙, '으' 불규칙, '거라/너라' 불규칙

본 논문에서 규칙을 기반으로 하는 복원 방법은 축약 음절 정보를 이용하여 미리 복원하였다가 형태소 분석 시에 복원이 제대로 이루어 졌는가를 검사한다. 그리고, 사전 기반 방법은 사전에 불규칙 형태에 따라 형태소 해석을 한 후, 후처리에서 불규칙 형태에 따라 원형으로 복원을 한다.

후처리 단계에서 하는 일은 주로 사전 기반으로 하는 불규칙 형태에 대한 원형 복원 작업이다.

3. 띄어쓰기 오류 교정과 미등록어 처리

3.1 복합 명사 처리와 띄어쓰기 오류 교정

본 논문에서는 한국어에서 많이 틀리게 되는 띄어쓰기 오류에 대한 교정을 함으로써 한국어 형태소 분석기의 성능을 개선하고자 한다. 한국어 맞춤법 띄어쓰기는 매우 중요하면서도 현재 사회의 각계 각층에서 잘 이행하지 못하고 실정이고 자행자재로 사용하는 경우가 많다. 더욱이 문교부에서 제정한 교과서에서도 띄어쓰기 규칙이 일정하지 않으며 오류가 많다. 띄어쓰기의 경우, 출판된 출판물에서도 가장 빈번히 발생하는 문제이다. 그러므로 우리가 처리하고자 하는 실제 한국어 전산화 문서에서의 띄어쓰기 오류는 더욱 많을 것이다. 이러한 오류를 다소나마 교정하여 한국어 형태소 분석을 할 수 있다면 한국어 형태소 분석기의 분석률은 향상될 수 있다.

본 논문에서는 한국어에서 발생하는 띄어쓰기 유형을 3 가지로 구분한다.

- 어절 내 띄어쓰기 오류 : 한 어절 내에서 띄어쓰기 오류가 발생하는 경우의 오류이다. 대표적인 예로는 "그사람", "두가지"와 같이 관형사와 명사를 붙여서 쓰는 경우이다.
- 어절 간 띄어쓰기 오류 : 한 어절로 이루어져야 하는 어절을 여러 어절로 사용하면서 발생하는 띄어쓰기 오류이다. 이러한 예는 타이핑 오류나 띄어쓰기 맞춤법을 제대로 인지하지 못하여 발생한다. 예로는 "한국 에서는" 등이 있다.
- 복합적인 띄어쓰기 오류 : 위의 두 가지 띄어쓰기 오류 유형이 복합적으로 발생하는 경우이다. 주로 타이핑 오류에 의한 것이 많다. 예로는 "그사람 에게서는"와 같이 "그"와 "사람"은 띄어 써야 하고, "사람"과 "에게서는"은 붙여 사용하여야 한다.

띄어쓰기 오류 교정을 위하여, 위의 1 과 3 의 띄어쓰기 오류 유형에서 발생하는 어절 내 띄어쓰기 오류를 교정과, 2 와 3 의 띄어쓰기 오류 유형에서 발생하는 어절 간 띄어쓰기 오류 교정을 나눈다.

3.1.1 어절 내 띄어쓰기 오류 교정 및 복합 명사 처리

복합 명사를 띄어쓰기 오류 교정과 같이 처리하기 위해서, 우리는 한 어절에서 명사들이 붙여 쓸 수 있는 한국어 맞춤법 규칙을 무시하여 명사들이 붙여 쓸 경우, 띄어쓰기 오류로 생각한다. 이러한 가정에 의해서 한 어절 내에서의 복합 명사 분해 문제는 띄어쓰기 오류 교정의 부분 문제가 된다[9]. 역시 복합동사에 대해서도

"동사+어미+동사+어미" 형태의 나열을 띄어쓰기 오류 교정으로 처리한 후, 동사의 나열을 사항에 따라 복합 동사로 묶어서 출력하여 복합동사에 대한 처리를 동시에 할 수 있다.

어절 내 띄어쓰기 교정을 위해서는 우선 어절 내의 어느 음절 다음에서 띄어쓰기를 하여야 할 지를 판단하여야 한다. 물론 한 어절 내에 여러 번의 띄어쓰기를 하여야만 할 경우도 발생할 수 있다. 이러한 띄어쓰기를 해야 할 음절은 형태소 분석 단계에서 두 형태소 간의 접속 정보가 결합하지 않은 곳이 될 것이다. 이와 같이, 두 형태소 간의 결합하지 않는 곳은 입력 어절에 대하여 여러 가지 경우의 형태소 결과가 발생할 수 있으므로, 이들 중에서 가장 적절한 경우를 선정하여야 한다. 본 논문에서는 이를 위하여, 태깅(Tagging) 시스템에서 많이 사용하는 비터비 알고리즘을 사용하여 해결한다.

비터비 알고리즘은 다이나믹 프로그래밍(dynamic programming)의 응용이다. 비터비 알고리즘은 기본적으로 매 상태에서 다음 상태로 전이해 갈 때, 그 때까지의 가장 좋은 경로만을 기억하는 알고리즘이다. 보통 비터비 알고리즘에서 사용하는 경로 추정 계산법은 확률을 이용하여 품사 태깅이나 결정 구문 분석기에 사용하지만, 본 논구의 띄어쓰기 오류에 대한 언어학 확률 모델을 규정하기가 어렵기 때문에 한국어에서 발생하는 띄어쓰기가 많이 들어가는 품사 간의 가중치를 이용하여 이를 해결한다. 이러한 품사 간의 가중치를 구성하는 테이블을 문맥 가중치 테이블이라고 한다. 그리고 문맥 가중치 테이블의 값은 일반적인 비터비 알고리즘에서 이용하는 상태 전이 확률 값 대신에 이용한다. 일반적인 비터비 알고리즘에서 사용하는 상태 가중치는 어떤 형태소가 어떠한 품사로 많이 사용하는가를 확률로 나타내고 있다. 이러한 확률 값은 대량의 코퍼스를 통하여 구할 수 있고 또한 이러한 확률 값을 시스템에 적용하기 위해서 많은 저장 장소를 요구하기 때문에, 본 논문에서는 어떤 형태소의 길이가 어떤 품사로 많이 사용되는가에 대한 가중치 값으로 대체하여 사용한다.

본 논문에서 위에서 설명한 비터비 알고리즘을 한국어 형태소 분석을 위한 CYK 알고리즘에 결합하여 사용한다[9]. CYK 테이블에 사전 검색으로 각각의 표제어에 대한 형태소와 형태소 품사들을 본 시스템의 상태로 본다. 그리고 CYK 테이블의 행을 따라서 시간이 흘러간다고 정의하자. 이 때 우리는 띄어쓰기 오류 교정을 위한 최적의 경로를 구하기 위해 다음과 같이 정의한다.

- 상태 가중치 : 상태 a에 대한 상태 가중치를 W(a)라고 하자. 이 때, 상태 a는 $[M_i, C_j]$를 나타낼 수 있다. 즉 상태 a는 형태소 M_i가 C_j라는 형태소 품사를 가질 가중치를 나타낸다. 가중치는 M_i 형태소의 음절 길이가 C_j라는 품사가 될 수 있는 정도에 대한 휴리스틱한 가중치이다. 이 가중치는 값이 적을수록 형태소 M_i가 형태소 품사 C_j가 될 가능성이 높다.
- 상태 전이 가중치 : 한 상태 a($[M_i, C_j]$)에서 다음 상태 b($[M_k, C_m]$)로의 상태 전이 시의 가중치를 t(a, b)라고 하자. 이 때, 상태 a에서 상태 b로의 전이는 CYK 알고리즘에서 접속 결합 가능성을 검사하는 형태소 간의 전이를 의미한다. 즉 M_i의 형태소 품사 C_j 앞에 M_k의 형태소 품사 C_m가 나타날 수 있는가를 나타낸다. t(a, b)는 다음과 같이 정의된다.

  t(a, b) = 0.5, 품사 C_j와 품사 C_m가 접속정보 표에 의해 접속 가능한 경우,
  = W_{ba}, 품사 C_j와 품사 C_m이 접속정보 표에 의해 접속 불가능한 경우, 이 때, W_{ba}는 품사 C_j와 품사 C_m이 문맥 가중치에 의해 , 품사 Cj 앞에 띄어쓰기가 나타날 가능성에 의하여 정하여 진다. 이러한 가중치는 문맥 가중치 테이블에 나타난다.
- 상태 a까지의 경로 가중치를 P(a)라고 하고 상태 a에서 상태 b로 전이를 한다고 하자. 이 때, 상태 b의 경로 가중치 P(b)는 다음과 같이 얻을 수 있다.

  $$P(b) = P(a) + t(a, b) + W(b)$$

위와 같이 정의하였을 때, 상태 b로 올 수 있는 상태들의 집합을 Sb라고 하면, 다음을 만족하는 상태에서의 경로가 띄어쓰기 오류를 교정하기 위한 현재 상태 b까지의 가장 적절한 경로가 된다. 최적의 P(b)는 다음과 같이 얻어진다.

$$P(b) = MIN_{t \in Sb}\{P(t) + t(t, b)\} + W(b)$$

비터비 알고리즘의 수행 순서와 상태 전이 순서는 <그림 2-5>에 있는 CYK 알고리즘의 수행 순서와 같다. 그리고 현재 상태 b의 M_k가 T(i, j, 0)에 있는 형태소라면 이전의 최적 경로인 T(i+1, n-1, 0)에 있는 상태들과의 전이가 가능하고 그 최적 결과는 T(i, n-1, 0)에 기록한다. 최종적으로 띄어쓰기 오류 교정에 가장 적절한 결과는

T(0, n-1, 0)에 저장된다. 이 때, T(0, n-1, 0)에 있는 최종 결과에 대해서 임의의 최종 상태로의 전이를 통하여 우리가 원하는 하나의 결과를 얻게 된다. 이 결과에서 접속정보 표에 의해서 상태 전이가 안되고 문맥 가중치 표에 의해서 상태 전이가 된 부분에 띄어쓰기 문자인 스페이스를 삽입하면 우리가 원하는 새로운 어절들을 얻게 된다. 이 어절들을 한 번 더 형태소 해석하게 되면, 올바른 어절에 대한 형태소 해석 결과를 얻을 수 있다.

본 논문에서 사용되는 문맥 가중치 표는 접속 정보 표의 어절 내의 형태소 간의 접속 검사와는 달리 어절과 어절 사이의 나타날 수 있는 가능성을 검사하는데 사용한다. 이 때, 어절과 어절 간의 연결 가능성은 앞 어절의 문법형태소 품사와 뒤 어절의 어휘형태소 품사가 한국어 문장에서 얼마나 자주 연결되어서 나타나는가에 대한 가능성에 대한 역수 개념을 가중치로 나타내었다. 그러므로, 접속 정보 표와는 같은 구조이지만, 각 품사에 대한 연결 가능성에 대한 가중치가 들어 있는 것이 다르다. <그림 3-1>는 문맥 가중치 표의 구조를 나타내고 있다.

$K_i : [C_1, W_{i1}][C_2, W_{i2}] \ldots [C_m, W_{im}]\ \%$

K_i : 문맥 가중치 표에서 키가 되는 문법형태소 접속 정보
C_j : K_i의 앞 어절에서의 마지막 형태소가 될 수 있는 접속 정보 ($1 <= j <= n$)
W_{ij} : K_i와 C_j사의 문맥 가중치 ($1 <= j <= n$)

<그림 3-1> 문맥 가중치 표 구조

문맥 가중치 표에서 키가 되는 문법형태소의 접속정보 앞에 올 수 있는 접속정보(문법형태소나 실질형태소)에 대한 가중치는 한국어의 어절 간의 특성을 반영하여 휴리스틱 규칙으로 구성하였다. 이러한 규칙은 어절 내의 띄어쓰기 오류에 대해서 여러 어절이 한 어절로 구성되었을 때, 교정하는데 사용하기에 알맞게 가중치를 부여한다. 문맥 가중치에 대한 휴리스틱 적용 방법의 원칙은 다음과 같다.

규칙 1] 영어 및 한자와 명사, 영어, 한자인 경우
규칙 2] 명사와 명사의 경우
규칙 3] 종결어미와 기호의 경우
규칙 4] 명사와 기호의 경우
규칙 5] 관형사와 명사 경우
규칙 6] 명사 앞에 자주 오는 조사와 명사의 경우
규칙 7] 용언 앞에 자주 오는 어미와 용언의 경우
규칙 8] 보조 연결 어미와 용언의 경우
예외 규칙] 위와 같은 규칙으로만으로는 띄어쓰기 오류 처리기에서도 미등록어가 발생함으로 현재까지 처리된 결과를 보존하기 위해서 또는 한 음절짜리 어절로 너무 많이 분해하는 것을 방지하기 위해, 미등록어 품사를 두어서 처리하고 있다. 이러한 미등록어 품사는 복합명사에서 일부 명사만을 처리할 경우에 매우 유리하다.

위와 같은 원칙 순으로 우선 순위를 주어 각 자립어의 앞에 오는 접속 정보에 대한 상대 가중치를 휴리스틱하게 부여하였다. 위의 경우에서 [규칙 2]에서의 명사와 명사 사이의 가중치는 한국어 명사의 특성을 살려서 더욱 자세히 나누었다. 한국어 명사들은 2 음절, 3 음절, 4 음절, 5 음절, 1 음절, 6 음절 등의 순서로 분포한다. 그러나, 여기에서 1 음절의 수는 작지만, 거의 모든 1 음절 한국어 문자들이 명사가 될 수 있음으로 2, 3, 4 음절, 기타 음절, 1 음절 명사 순에 의하여 서로의 결합 가능성에 대한 상대 가중치를 부여하였다.

어절 내 띄어쓰기 오류 교정을 위한 비터비 알고리즘에 사용하는 상태 가중치는 앞의 명사 경우와 같이 각 품사에 많이 분포하는 음절 길이에 대하여 가중치를 부여하여 정하였다.

지금까지 살펴본 한 어절 내 띄어쓰기 오류 교정을 위한 비터비 알고리즘은 문맥 가중치 표와 상태 가중치 부여에 의하여 최소의 어절로 구분하고 최소의 형태소가 포함된 결과가 최종적인 결과로 추출한 가능성이 높아지게 구성되었다.

본 띄어쓰기 오류 처리기는 띄어쓰기 오류만이 아니라 복합명사에 대한 처리도 가능하게 하는 역할을 한다. 또한, 띄어쓰기 결과로 두 개의 동사가 연결되어 있을 경우, 복합 동사로 보아 하나의 복합동사로 형태소 분석을 한다. 그리고 미등록어를 포함하는 복합명사에 대해서 사전에 등록한 부분까지의 명사를 분석한 후, 미등록 명사에 대해서 다른 어절로 보아 분석 가능한 일부분이라도 분석 가능하게 한다.

3.1.2 어절 간의 띄어쓰기 오류 교정

CYK 테이블에서 비터비 알고리즘은 어절 내의 띄어쓰기 오류를 교정할 수 있으나, 여러 어절 간의 띄어쓰기 오류에 대한 교정을 하지 못한다. 어절 간의 띄어쓰기 오류 교정을 위해서 현재의 입력 어절이 미등록어을 포함해서 형태소 분석 결과로 아무런 결과를 추출하지 못한 경우일 때, 다음 입력 어절의 형태소 분석 결과가 역시 미등록어 때문에 아무런 형태소 해석 결과를 갖지 못할 경우, 이러한 어절들을 결합하여 한 어절로 만들어 앞에서 설명한 한 어절 내의 띄어쓰기 오류 교정을 위한 비터비 알고리즘을 통하여 여러 어절 간의 띄어쓰기 오류를 교정한다.

MATEC'99 대회에서는 띄어쓰기 오류가 포함된 테스트 어절이 없기 때문에, 문맥 가중치에서 복합 명사나 복합 동사를 생성할 수 있는 규칙을 제외하고 모든 규칙을 제거하여 오직 복합 명사와 복합 동사만을 처리하도록 하였고 어절 간의 띄어쓰기 오류 교정 또한 역시 하지 않았다.

3.2 미등록어 처리

형태소 분석에서 미등록어란 사전에 등록되어 있지 않은 단어를 말한다. 입력 어절에서 미등록어가 발생한다면, 대부분의 경우 형태소 분석이 되지 않고 또는 형태소 분석이 되더라도 그 어절에서는 잘못된 경우이다. 하지만 두 번째 형태의 미등록어에 대해서는 형태소 분석에서 처리를 하기가 어렵다. 본 논문에서 미등록어 처리는 첫 번째 미등록어와 같이 형태소 분석이 되지 않을 경우에 한해서 처리를 한다. 미등록어를 처리하고 나서 어느 분석이 가장 적절한가를 추정하는 방법을 미등록어 추정이라고 하는데, 본 논문에서는 이러한 미등록어 추정을 하는 것은 배제한다.

이미 잘 알려져 있듯이 미등록어의 경우, 대부분이 명사들이다. 미등록어 중에서 명사가 차지하는 비율이 분야에 따라 다르겠지만, 신문의 경우에는 98%가 훨씬 넘는 부분을 차지한다. 많은 미등록어 추정 시스템에서는 이러한 점을 이용하여 미등록어를 추정한다. 본 논문에서는 미등록어를 체언과 용언 두 분류로 정의한다. 조사와 어미와 선어말 어미의 경우에는 이미 많은 부분이 사전에 등록되어 있으므로, 실질적으로 불필요한 형태소 분석 결과를 줄이기 위해 정의하지 않았다. 그리고 조용보조어간이나 선어말 어미는 이미 정의가 한 데로 사전에 등록이 다 되어 있으므로 실제로 미등록어로 발생할 가능성이 없기 때문에 배제하였다. 그리고, 수사의 경우에도 수사 오토마타에 의해 모든 형태소가 분석이 가능함으로 배제하여도 된다.

- 체언 : 명사(고유명사, 보통명사, 의존명사), 대명사(인칭대명사, 지시대명사), 관형사, 부사, 감탄사
- 용언 : 동사, 형용사

본 미등록어 처리기에서는 위의 두 종류에 대한 미등록어를 그룹화하여 준다. 위의 두 종류의 미등록어 범주 중에서 거의 모든 경우가 체언일 것이고 또한 명사일 것이다. 체언 가운데서 부사, 관형사, 대명사의 가능성은 상당히 적을 것이다. 한국어에서 대명사의 수는 상당히 적고 대부분이 형태소 분석 사전에 이미 등록되어 있기 때문이다. 또한 감탄사 역시 문장의 첫 번째 어절이 아니거나 인용문이 아니라면 배제할 수 있다.

미등록어 처리기는 다음과 같은 방법에 의해서 구현되었다. 우선 CYK 알고리즘을 이용한 형태소 해석을 하여 <그림 2-4>의 CYK 테이블 T(0, n-1, 0)에 아무런 결과도 없다면, T(i, n-1, j) ($0 \le i \le n-1, 0 \le j \le 2$)에 대해서 i 번째 음절에서 n-1 번째 음절까지의 부분 형태소 해석 결과를 가지고 미등록어를 추정한다. 이들 결과들 중에서 위의 체언과 용언과 결합할 수 있는 부분 해석 결과를 이용하여, 0 번째 음절에서 i-1 번째 음절까지의 형태소에 대한 품사를 체언으로 할 것인가 용언으로 할 것인가를 결정한다. 또한 i-1 번째 음절이 2 음절 이상의 조사나 어미의 한 일 부분인 경우에 대해서 미등록어의 부분이 될 가능성이 없으므로 미등록어 추정을 무시한다. 그리고, 1 음절 조사나 어미에 대해서도 체언이나 용언으로 사용할 수 있는 음절에 대해서 미등록어에 속하도록 하는 휴리스틱을 사용한다.

그리고 용언과 체언에 대한 음절 정보를 이용하여 미등록어가 불필요하게 용언이나 체언으로 분류되는 것을 방지하도록 한다. 체언인 경우에는 상당의 미등록어가 고유명사이므로, 음절 정보를 이용하여 체언인지 아닌지를 판별하는 것은 위험하기 때문에, 음절 정보는 용언에 대해서만 사용한다. 어쩔 수 없이 부분 해석된 결과가 용언과 결합할 경우에만 체언에 대한 분류가 배제된다. 용언에 대해서는 각종 불규칙 형태를 고려해서 원형을 복원하고 또한 불규칙이 일어난 음절을 분리하여 처리한다. 이때 각 불규칙이 일어날 수 있는 음절 정보는 매우 유용하다[1].

MATEC'99 대회 참가를 위해서 미등록어 체언은 모두 보통명사로 미등록어 용언은 모두 동사로 하였다.

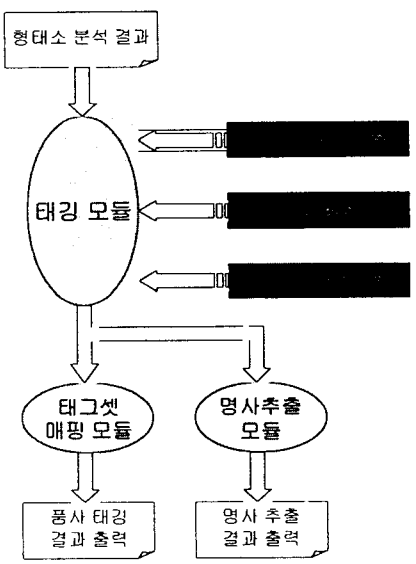
4. 품사 태거 및 명사추출기

4.1 전체 구조

품사태거 KLE-TAG[1]는 KOMA 의 형태소 분석결과를 받아 태깅을 수행한다. 명사추출기는 독립적으로 수행되는 프로그램이 아닌 KLE-TAG 에 포함된 하나의 모듈로서, 태깅이 완료된 후 태깅 결과로부터 명사 형태소만을 추출하여 출력시키는 간단한 형태로 이루어져 있다. 품사태깅과 명사추출을 동시에 수행하는 KLE-TAG 의 전체구조는 <그림 4-1>과 같다. KLE-TAG 는 기본적으로 통계 기반 태깅 방식을 사용하며 이를 언어 규칙들을 이용하여 보완시키는 형식으로 구성되어 있다.

4.2 제약 규칙 및 수정 규칙

언어 규칙은 태깅하기 전에 미리 적용되는 제약 규칙과 태깅 후 오류를 정정시키는 수정 규칙의 두 종류로 구성된다. 태깅 전 제약 규칙으로 긍정 언어 지식(positive linguistic information) 및 부정 언어 지식(negative linguistic information)들을 사용함으로써 태깅에 들어가기 전에 제약 규칙이 적용 가능한 어절들의 품사 모호성을 미리 감소시켜 주며, 또한 태깅이 끝난 후에는 수정 규칙들을 사용하여 태깅시 발생한 오류들을 정정함으로써 정확도를 향상시키도록 하고 있다[5][6].



<그림 4-1> KLE-TAG 및 명사추출기의 구조

언어 규칙들은 모두 자동 학습이 아닌 수작업으로 구축되었으며 <그림 4-2>에 규칙들의 유형이 제시되어 있다. 규칙들은 각 규칙 유형(TYPE)에 따라 참조하는 문맥의 조건이 다르게 설정되어 있으며 모두 14 개의 유형이 존재한다. 현재 KLE-TAG 에서는 총 82 개의 언어 규칙들을 사용하고 있다.

규칙 : <TYPE> <현재 태깅 결과> <문맥> <수정 결과>
TYPE 1 : 문맥 제약없이 무조건 수정
TYPE 2 : 앞 어절 마지막 형태소 태그가 X 일 때
TYPE 3 : 앞 어절 첫번째 형태소 태그가 X 일 때
TYPE 4 : 앞 어절 첫번째 형태소가 Y 일 때
• • •

<그림 4-2> 언어 규칙의 유형

4.3 통계 기반 품사 태깅

KLE-TAG 은 형태소 단위 품사 태깅 방식으로서 모호성 해소를 위한 통계 정보로 어휘 확률과 품사 bigram 을 이용한다. 태깅 시 어절간 품사 전이와 어절내 품사 전이를 구별하여 문맥 확률을 적용시키기 위해서 띄어쓰기를 고려한 품사 태깅 모델인 식 (4.1)의 모델을 사용하고 있다[3].

$$\phi(W) = \arg\max \prod_{i=1}^{n} P(t_i \mid t_{i-1}, k) P(w_i \mid t_i) \qquad (4.1)$$

$$\begin{cases} k = 0 : \text{if 어절내 전이일 때} \\ k = 1 : \text{if 어절간 전이일 때} \end{cases}$$

식 (4.1)에서 k 는 품사 전이가 어절내 전이인지 어절간 전이인지를 나타내는 변수이며, 그 값에 따라 다른 확률 분포가 사용된다. 그런데 어절간 품사 전이의 경우, ETRI 측이 제시한 품사 태깅 지침에서는 '-하다'류의 용언들을 '명사+하'로 분리하여 태깅하고 있기 때문에 아래와 같은 문장에서는 문제가 발생할 수 있다.

(예문) 수학을 공부하다
(태깅) 수학/nc+을/jc 공부/nc+하/xsv+다/ef

[1] 포항공대 지식 및 언어공학 연구실(KLE Lab.) 개발

위의 예에서 '공부하다'는 형태소 분석시 '공부+하+다'로 분리되어 <명사+동사파생접사+어미>로 인식되므로 앞어절 '수학을'과의 어절간 품사 전이를 적용할 때 <격조사-명사>의 bigram 값을 사용하게 된다. 그러나 실제로 '공부하다'는 어절 전체로는 용언에 해당하기 때문에 어절간 품사 전이의 적용시 <격조사-명사>보다는 <격조사-동사>의 bigram 값을 사용하는 것이 더 타당하다. 따라서 KLE-TAG 에서는 위와 같이 '명사+파생접사'로 분리된 어절들은 용언으로 인식하도록 하여 어절간 bigram 값을 계산하고 있다.

4.4 태그 매핑

KLE-TAG 에서 사용하는 태그는 총 50 개이며 ETRI 가 제시한 태그 기준에 일치시키기 위해 태깅 결과 출력부에서 태그 매핑을 수행한다. KLE-TAG 태그 집합에서 ETRI 태그 집합으로의 매핑은 *co*(지정사)를 제외한 나머지 모든 태그에서 n : 1 의 대응을 나타내므로 별다른 문제점이 발생하지 않는다. 그리고 *co* 의 경우에도 오직 '-이-'만이 지정사로 규정되어 있기 때문에 현재 KLE-TAG 에서는 동사파생접사로 분류되어 있는 '-이-'를 단순히 *co* 로 매핑시키는 작업만으로도 간단하게 해결될 수 있다.

5. MATEC'99 준비 및 실험 결과

5.1 준비 과정

ETRI 가 제시한 품사 부착 말뭉치 구축 지침과 KOMA 가 지향하고 있는 형태소 분석 지침은 상당 부분에서 차이가 있다. 그 차이점을 ETRI 측의 형태소 분석 지침 측면에서 나열하면 다음과 같이 크게 네 가지를 들 수 있다.

첫째, '-하다' 또는 '-되다'가 결합된 용언을 '명사 + 동사/형용사 파생 접사'로 분리시켜 태깅한다.

둘째, 수사의 처리가 보다 세분화되어 있다. 숫자와 한글이 섞여 있는 경우나, 기호 등이 포함되어 있을 때, 그리고 경, 조, 억, 만이 한글 수사 어휘에 포함되어 있을 때 기호와 단위를 구분하여 각각 별도로 태깅한다.

셋째, '명사+용언' 사이에 격조사가 생략되었다고 가정되면, 각각을 분리하여 태깅한다.

넷째, 어미에서 '하'가 탈락되었을 때, '하'를 복원하지 않고 어미만 분리한다.

위의 경우에서 첫째, 셋째, 넷째의 요구사항을 만족

<표 4-1> 태그 매핑

KLE-TAG (50 개)	ETRI (27 개)
CMCPA(동작성명사) CMCPS(상태성명사) CMCN(비서술성명사) CMP(고유명사)	nc(자립명사)
CMDU(단위성의존명사) CMDO(일반의존명사)	nb(의존명사)
CT(대명사)	np(대명사)
CSC(양수사), CSO(서수사) SGN(수관형사)	nn(수사)
K(감탄사)	ii(감탄사)
SGR(지시관형사) SGO(성상관형사)	mm(관형사)
SBJ(접속부사)	maj(접속부사)
SBR(지시부사) SBO(일반부사)	mag(일반부사)
YBDR(지시동사) YBDO(일반동사)	pv(동사)
YBHR(지시형용사) YBHO(일반형용사)	pa(형용사)
YA(보조용언)	px(보조용언)
-이-	co(지정사)
fjcs(주격조사), fjco(목적격) fjcc(보격), fjcac(공동격) fjcag(인용격), fjcl(독립격) fjcao(일반부사격)	jc(격조사)
fjcg(관형격조사)	jm(속격조사)
fjj(접속조사)	jj(접속조사)
fjb(보조사)	jx(보조사)
fmocc(대등적 연결어미) fmocs(종속적 연결어미) fmoca(보조적 연결어미) fmota(부사형 전성어미)	ec(연결어미)
fmb(선어말어미)	ep(선어말어미)
fmof(종결어미)	ef(종결어미)
fmotn(명사형 전성어미)	etn(명사형 전성어미)
fmotg(관형형 전성어미)	etm(관형형 전성어미)
fps(접미사)	xsn(명사파생접사)
fpd(동사파생접사)	xsv(동사파생접사)
fph(형용사파생접사)	xsm(형용사파생접사)
fpp(접두사)	xp(접두사)
F(외국문자)	f(외국어)
gf(마침표), gp(쉼표) gq(여닫는기호) gd(이음기호), go(기타기호)	s(기호)