2000년 한국정보처리학회 추계 학술발표논문집 제 7 권 제 2 호

정보검색용 한국어 형태소분석기 구현

손소현*, 유병선**, 이택현**, 문병주*, 홍기채*, 정현수*
*한국전자통신연구원 정보통신기술경영연구소 지식정보센터, **T&I 시스템
e-mail : sohyun@info1.etri.re.kr

Korean Morphology Analysis Implementation for Information Retrieval

So-Hyun Son*, Byung-Sun Yu**, Tak-Hyun Lee**,
Byung-Ju Moon*, Gi-Che Hong*, Hyun-Su Jung*,
*ETRI Information Telecommunication Technology Management Institution
Knowledge Information Center, **T & I system

요 약

본 논문은 정보검색을 위한 형태소분석기를 소개한다. 검색엔진의 속도향상을 지향한다면 형태소분석 알고리즘과 참조하는 사전의 구조를 어떻게 구성하는가에 따라 처리속도에 상당한 변화를 기대할 수 있으며, 본 논문에서는 알고리즘으로 최장일치법을 이용하고, 사전내부구조로 AVL+Trie 구조를 이용하여 사전참조의 속도향상을 기대하였다

1. 서론

컴퓨터를 이용한 자연언어 처리분야는 사용자에게 편리한 인터페이스를 제공하기 위해 최근에 많이 연구되고 있는 분야이다. 단순히 원하는 품사단어를 추출하는 단계에서부터 시작하여 인간이 구사하는 일상언어에 대한 직접적인 이해와 언어생성을 통해 컴퓨터와의 양자간 대화 인터페이스 구현에 이르기까지 많은 자연언어 처리 응용시스템들이 있으며, 이들 응용시스템에서 형태소 [1] 분석 문제는 가장 기본적으로 선결되어야 하는 문제로 부각되고 있다.

2. 기존연구

현재 형태소분석기는 영문, 한글 구분없이 대부분의 응용시스템에서 실용화 단계에 도달하였으며, 문장구조변형이나 어형변화가 상대적으로 적게 일어나는 영문 언어처리연구가 한글 언어처리연구에 비해 약 20년정도 앞서 있는 편이다.

실용화 단계에 도달한 형태소분석기가 각각의 응용시스템에 적용된 형태를 분석해보면 그 목적에 따라 알고리즘과 참조사전 구조가 상당히 다양하다. 이에 따라 맞춤법 검사기, 정보검색, 번역시스템, 음성인식 시스템 등 각 응용목적에 적합하게 특화된 형태소분석 알고리즘 및 산출결과를 다른 시스템에서 공용으로 적용한다면 효율성 측면에서 성능감소를 감수해야 한다. 맞춤법 검사기용으로 개발된 형태소분석기는 정확성이 최우선으로 고려되어야 하고, 정보검색용으로 개발된 형태소분석기는 빠른 속도와 정확성이, 번역시스템용으로 개발된 형태소분석기는 속도보다는 번역대상 언어사이의 관용구 처리를 포함한 완벽한 변환매칭이 이루어져야 한다. 제안한 형태소분석기는 정보검색을 위한 보조시스템으로 사용된다.

검색엔진은 대용량의 데이터로부터 키(key)가 되는 중심 단어를 추출하여 고속으로 색인, 검색하는 기술이다. 효율적 정보검색엔진을 구축하기 위해서는 다양한 색인방법 중 속도, 재현률 [2], 정확률 [3]을 고려한 알고리즘을 선택해야 한다.

[1] 형태소 : 의미를 가진 가장 작은 말의 단위

[2] 재현률 : 의도한 검색질의 결과 개수에 비해 검색질의 결과가 나오는 개수에 대한 상대적 비율

[3] 정확률 : 의도한 검색질의 결과 개수에 비해 검색질의 결과내에서 의도한 검색질의 결과가 나온 개수에 대한 상대적 비율

색인방법으로는 N-gram 기반방법, 형태소분석방법, 구문분석 방법, 의미분석 방법이 있다. N-gram 기반방법은 단어의 뜻과 형태와는 상관없이 N 개의 음절 단위로 색인을 해나가는 방식으로 간단한 알고리즘을 사용하므로 속도가 상당히 빠르고, 신조어 등에 대한 특별한 고려를 하지 않아도 된다는 장점이 있다. 하지만, 색인을 위해 많은 공간을 필요로 하고, 정확률이 떨어진다는 단점이 있다. 형태소분석방법은 조사 혹은 어미 등의 어절변형이 일어나기 쉬운 기능어들을 분리하여 중심어들로만 색인어를 형성할 수 있도록 형태소 단위로 분석하는 방법이다. 분석알고리즘의 수행 중 속도가 약간 지연되지만, N-gram 기반색인방법에 비해 정확률이 높은 편이다. 구문분석방법 및 의미분석방법은 형태소분석 방법보다 한단계 발전된 언어분석 방법으로 형태소분석 방법에서 발생하는 중의성을 문장의 구문 및 의미를 분석하여 해결하려는 방법이다. 현재 한글 구문분석 및 의미분석방식은 연구단계에 머물어 있는 실정이다.

본 시스템에서는 기본적인 색인방법으로 N-gram 기반 색인방법을 이용하고, 다양한 검색 인터페이스 (단어중심의 클러스터 자동생성, 문서요약문 자동생성, 자연어 질의처리로 검색핵심어 자동추출 등)로 검색 편의성을 도모하기 위해 문서 및 문장에서 핵심어를 추출하는 형태소 분석방법을 이용한다.

현재 활용되고 있는 형태소분석 알고리즘은 대체로 속도면에서 구문분석, 의미분석방법보다 빠르며, N-gram 방식보다 정확한 검색결과를 기대할 수 있다.

대표적인 알고리즘으로는 최장일치법, 최단일치법, Tabular 파싱법 , Head-Tail 구분법, 음절 단위 분석법, CYK , Two-level 형태론, 음절단위 분석법 등이 있다.

검색엔진의 속도향상을 지향한다면 형태소분석 알고리즘과 참조하는 사전의 구조를 어떻게 구성하는가에 따라 처리속도에 상당한 변화를 기대할 수 있으며, 본 논문에서는 알고리즘으로 최장일치법을 이용하고, 사전내부구조로 AVL+Trie 구조를 이용하여 사전참조의 속도향상을 기대하였다.

3. 형태소 분석기 구성

본 형태소분석기는 다양한 정보검색 인터페이스 제공을 위한 입력으로 문서에서 핵심이 되는 명사들을 추출한다. 핵심 명사범주를 일반명사에 두지않고, 관심분야 전문기술 정보용어를 기반으로 구성함으로써 전문정보를 체계적이고 효율적으로 제공할 수 있는 기반을 마련한다. 전체 구성은 크게 7 단계로 이루어진다.

3.1 어절단위 단어추출

문서에서 어절단위로 단어를 추출한다. 스페이스(space), 콤마(comma), 기타 특수기호 등을 기준으로 단어를 구분한다. 단, '-'는 "IMT-2000"과 같은 핵심단어의 일부음절에 포함되므로 기준기호에서 제외한다.

단어의 추출은 문서단위와 문장단위로 처리함으로써 근접한 단어들 사이의 연관관계 도출 및 핵심문장 도출에 활용한다.

3.2 compact 불용어 사전참조

명사가 아닌 단어들(동사, 형용사, 부사, 감탄사 등) 중에서 출현 빈도율이 높은 단어들은 compact 불용어 사전으로 구성하고, 명사사전을 참조하기 전에 먼저 참조한다.

높은 출현 빈도율을 가지는 명사단어와 비교하여 불용어의 출현 빈도율이 높게 나왔다면 명사사전 참조 전에 출현 빈도율이 높은 불용어부터 참조함으로써 속도향상을 도모할 수 있다. 따라서, compact 불용어 사전의 크기는 작고, 참조 알고리즘도 간단한 완전일치법을 사용한다. 출현빈도가 높은 불용어선택을 위한 Threshold 값 결정은 아래 식을 바탕으로 정한다.

$$\frac{Max(freq(n \in N))}{set(freq(d \in D))} <= 1 \quad \text{...(수식.1)}$$

n: 명사, N:명사사전 ,d:불용어, D:불용어사전,

freq:출현빈도수, Max: 최고빈도값, set: 결과집합

3.3 명사사전참조

Compact 불용어가 아닌 단어들에 대해 명사사전을 참조하여 명사여부를 체크한다. 명사사전 참조는 최장일치방법을 사용한다. 최장일치방법은 찾고자 하는 단어와 사전에 저장된 단어들 사이의 비교결과, 사전에 저장된 단어가 여럿 있다면 이들 중 가장 긴 단어를 선택하는 방법이다. 최단일치방법과 비교하여 복합명사 추출시 추출 단어 갯수가 작은 최장일치방법은 복합명사를 포함하는 문장에서 복합명사를 추출하는데 효과적이다. 예를 들어, "한일기계번역시스템에서의 형태소해석 프로그램의 중간 버퍼를 표준안에 맞추어 ..."라는 문장이 있고, 각 복합명사들이 명사사전에 모두 등록되어 있다면 최장일치방법에 의한 결과로 "한일기계번역시스템", "형태소해석", "프로그램", "버퍼", "표준안"이 추출된다. 반면 , 최단일치방법은 "한일", "기계", "번역", "시스템", "형태소", "해석", "프로그램", "버퍼", "표준안"이 추출된다.

최단일치방법은 추출과정에서 "형태소해석"을 "형태소"와 "해석"으로 구분하지 않고, 최단일치로 찾기 때문에 경우에 따라서는 "형태"와 "소해석"으로 구분하는 등 더 많은 오류가 포함될 수 있다.

3.4 남은 음절체크

한 어절에서 최장일치 방법으로 명사사전에 등록된 명사를 추출하고 남은 음절들에 대해 명사 혹은 의존형태소 여부를 명사사전과 의존형태소사전을 참조하여 체크한다. 예를 들어, "한일기계번역시스템에서의"라는 어절에서 명사사전에 "한일기계번역", "한일",

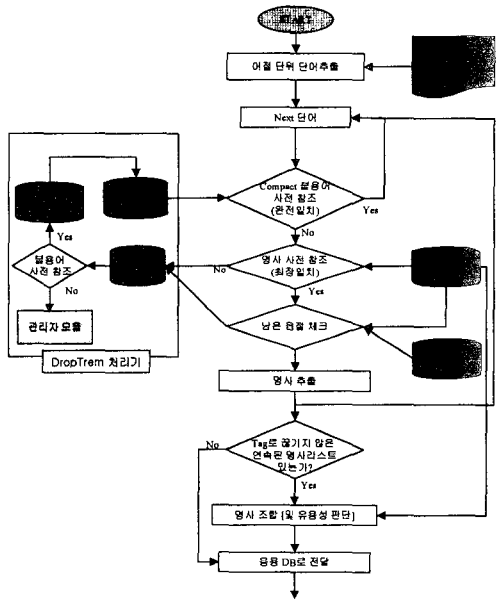
"기계번역"이 저장되어 있어, 최장일치방법으로 "한일 기계번역"만 추출되면, 남은 음절체크를 통해 "시스템"도 명사로 추출한다. 다음 "의"도 의존형태소임을 확인한다. 만일, 이 단계(4 단계)에서 기존의 명사추출(3 단계)이 잘못되었다고 판단되면 3 단계 추출부터 다시 수행한다.

3.5 불용어처리기로 전달

명사사전에 등록되지 않은 단어는 불용어처리기로 이동한다. 이동된 단어는 불용어사전의 단어수록 여부를 체크하여 사전에 수록되지 않은 단어들의 품사판정권한을 관리자에게 부여하고, 해당 품사사전에 미리 수록된 단어는 품사단어에 대한 빈도수만 증가시킨다. 관리자 처리모듈로 넘어간 단어는 관리자가 품사를 결정하여 해당 품사사전에 삽입한다.

3.6 명사조합

앞서 추출된 compact 불용어 및 의존형태소, 불용어로 이동한 단어들은 태그를 붙여, 태그를 포함하지 않는 연속된 명사 단어 리스트는 비록 어절단위 단어추출단계에서 분리되어 추출되더라도 하나의 복합명사일 가능성이 높으므로 명사조합을 한다. 조합된 명사가 의미있는 복합명사인지 체크하여 결과 명사리스트에 추가한다.



(그림. 1) 한글 형태소분석기 구성

3.7 응용 DB 로 전달

각 문서의 각 문장(sentence)당 출현한 명사리스트와 출현빈도수 정보를 작성하여 단어 클러스터 처리모듈 및 자동 요약생성 처리모듈, 자연어 질의처리 모듈로 전달한다.

4. 사전구조

4.1 기존 : AVL 구조

탐색트리가 좋은 성능을 유지하려면 한쪽 방향으로 기울어지지 않도록 저장해야 한다. AVL 트리는 균형이진 탐색트리로써 최악의 경우에도 비교 횟수가 log2(n)을 넘지 않는다. 또한, 삽입■삭제시 회전을 통한 국소적인 구조변화가 일어나며, 분리(split)이나 통합(mergy) 등의 연산을 통한 구조변화를 필요로 하지 않는다. 특히, 데이터량이 작은 색인일 경우 메인 메모리상에서 빠른 검색을 기대할 수 있다. 그러나, 색인의 양이 많다면 다른 구조들보다 삽입, 삭제시간이 많이 필요하고, 검색속도가 느리다는 단점이 있다.

4.2 기존 : Trie 구조

일종의 Tree 구조인 Trie 구조[1,2,4,5]전체를 노드에 저장하지 않고 키의 일부요소를 노드에 저장함으로써 키의 길이가 가변적이더라도 항상 일정한 검색속도를 유지할 수 있는 장점을 가진 저장구조이다.
Trie 를 이용한 문서검색의 경우 노드를 문자단위로 구성하기 때문에 질의에 대한 비교횟수는 색인량이 늘어나도 항상 문자갯수로 일정하게 유지된다. 또한, 삽입■삭제 시에는 해당 링크를 따라 단말노드를 찾고 삽입■삭제하면 되므로 B*,B+ Tree 처럼 분리나 통합연산도 필요없다.
한글의 경우, Trie 구조는 문자단위에 따라 크게 2 개의 구조[자소별 Trie, 음절별 Trie]로 나눌 수 있다. 자소별 Trie 는 저장, 검색의 기본 단위가 한글의 자소이며, 하나의 단어를 저장하는데 각 음절을 3 개의 자소로 분리하여 저장하므로 노드수가 음절단위방법의 3 배가 필요하다. 따라서, 이론적인 검색시간은 음절단위 방법의 3 배이다. 그러나, 실제 실험을 통한 효율성 측정을 통해 1.4 배의 검색시간이 소요되는 것으로 나타났다[3]. 음절단위 방법은 저장과 검색의 단위가 한글 음절이다. 임의의 단어를 검색하는데 걸리는 시간은 단어의 음절수에 비례하므로, 한글 형태소 사전을 구성하는 단어들의 평균 음절 수가 2.77 음절로 2 음절과 3 음절이 대부분(81%)을 차지하는 것[3]을 감안하면 사전참조횟수가 상당히 작다. 또한 가장 긴 단어 길이는 12 음절로 최악의 경우 12 번에 비례하는 비교로 검색이 가능하다.

4.3 AVL+Trie 구조

AVL+Trie 는 기존의 Trie 내부구조로 배열이나 리스트 구조 대신 AVL 구조를 사용하는 방법이다.

자소단위 Trie 구성시 한 음절은 초성, 중성, 종성으로 구성되며, 각 자소를 표현하기 위해 필요한 노드 수는 (수식. 2)에서 보는 바와 같이 최대 27 개이다.

한글음절 = 초성(19) + 중성(21) + 종성(27) ... (수식.2)

자소단위방법에 AVL 을 사용하여 한 음절을 검색한다면 (수식.3)에 따라 worst-case 의 경우 한 음절에 대해 초성(6.33) + 중성(6.51)+ 종성(7.00)=(19.84)번의 검색이 이루어진다.

$$1.44 * \log_2(N+2) : \text{worst case}$$

$$\log_2(N+1) : \text{normal case} \quad \ldots(\text{수식.3})$$

(19.84)*12 (최대음절길이를 12 라고 가정하면) = (238) 번의 비교를 통해 참조 가능하다. Normal-case 의 경우는 초성(4.32)+중성(4.46)+종성(4.81) 총 (13.59)번과 초성(4.32)+중성(4.46)+종성(1)의 9.78 번의 비율적 조합으로 형성된다. 말뭉치의 특성에 따라 달라지겠지만, 열린음절과 닫힌음절의 비율을 반반으로 보았을 때, 평균 11.19 번의 비교로 한 음절을 찾을 수 있다. 여기에 평균 단어의 음절길이(2.77)를 적용하면 11.19*2.77= 30.98 번의 비교를 통해 한 단어를 검색할 수 있다. 작은 양의 데이터에 대해 검색속도가 우수한 AVL Tree 를 자소단위로 노드구성을 함으로써 속도향상을 도모할 수 있다. 아래에 실제 실험을 통한 AVL 성능[6]을 보면 이론보다는 좀더 빠른 것을 알수 있다.

n	C(n)	E[h(n)]	R(n)
5	2.2	3.0	0.213
10	2.907	4.0	0.318
50	4.930	6.947	0.427
100	5.889	7.999	0.444
500	8.192	10.923	0.461
1000	9.202	11.998	0.463
5000	11.555	14.936	0.465
10000	12.568	15.996	0.465

(표.1) AVL 의 성능

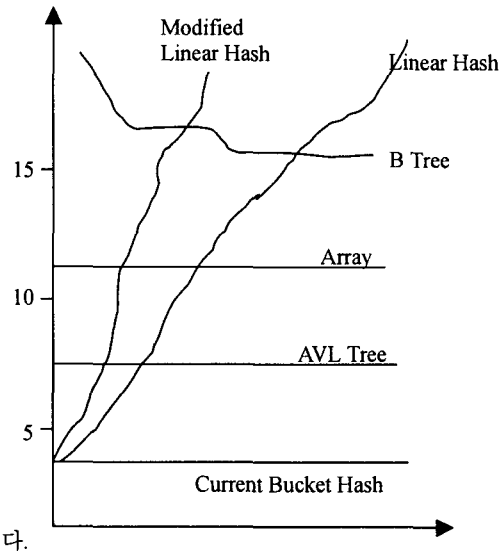
n = number of nodes in the tree
C(n) = average number of comparisons to find a key
E[h(n)] = expected height of tree
R(n) = average number of rotations per insertion/deletion

반면, 음절단위 Trie 를 구성한다면 한글 음절은 열린음절 19*21 개, 닫힌 음절 19*21*27 개로 총 11,172 개로 구성된다[3]. 음절단위 Trie 구성시 한글 음절은 최대 11,172 개의 음절이 생성된다[3].

11.172 = 열린음절(19*21)+닫힌음절(19*21*27).(수식.4)

AVL 을 사용하면 최대 11,172 개의 노드구성에 대해, worst-case 의 경우 한 음절에 대해 (19.36)번의 검색을 하므로, (19.36)*12 (최대음절단어) = 232.38 번의 비교를 통해 참조가 가능하다. Normal-case 의 경우 한 음절에 대한 평균 비교횟수는 (13.45)번이다. 여기에 평균 단어의 음절길이(2.77)를 적용하면 (13.45)*2.77 = (37.25) 번의 비교검색이 가능하다.

이상과 같이 음절단위방법이 자소단위방법에 비해 worstcase 에서 약간 더 나은 성능을 보여 복합명사를 많이 포함한 한글 사전의 경우 효과적이다. 그러나, 속도에 있어서 두 방법 모두 근사한 성능을 보였는데, 처리하는 노드수가 작을 경우 (그림.2)[7]에서와 같이 AVL Tree 가 다른 구조들보다 속도가 우월함을 알 수 있다.



(그림.2)각 구조별 검색속도

AVL+Trie 구조를 이용할 경우 또다른 특징으로 삽입/삭제시 국소적으로 회전(rotation)처리만 사용하기 때문에 삽입/삭제에서도 빠른 성능을 기대할 수 있다.

5. 결론

여러 품사 사전을 이용하여 빠른 속도의 검색성능을 위한 한글 형태소분석기를 구성하였다. 특히 AVL+Trie 구조를 사용하여 다른 구조들보다 빠른 검색속도를 지원하였으며, 사전에 단어를 삽입,삭제처리를 하는데도 속도의 향상이 있다.본 형태소분석기를 통해 추출된 명사 Term 들은 단어 클러스터 뷰어, 자동 문서요약기, 자연어 질의처리 등의 여러 가지 응용분야에 활용될 수 있으며, 앞으로는 전문 색인용으로 형태소분석기술 뿐 아니라 구문분석 기술을 연구해 볼 계획이다.

참고문헌

[1] 장동현, 맹성현,"효율적인 색인어 추출을 위한 복합명사 분석 방법", 충남대
[2] 강승식,"음절정보와 복수어단위 정보를 이용한 한국어 형태소 분석, 서울 대학교 공학박사 학위 논문, 1993
[3] 김철수, 배우정, 이용석,"이중 트라이를 이용한 한국어 단어 검색", 전북대학교 전자계산학과
[4] E.Fredkin, "Trie Memory", Comm.ACM, 3 pp490-500,1960
[5] J.I.Aoe, K.Morimoto, "An Efficient Implementation of Trie Structures" Ssep, pp.695-721.1992
[6] "CS 660: Combinatorial Algorithms AVL TREES", San Diego State University http://www.eli.sdsu.edu/ courses /fall95/cs660/notes/AVL/AVL.html
[7] Tobin J Lehman, Michael J Carey, "Query Processing in Main Memory Database Management Systems" ACM, 1986