K-Means 알고리즘을 이용한 계층적 클러스터링에서 클러스터 계층 깊이와 초기값 선정*

Selection of Cluster Hierarchy Depth and Initial Centroids in Hierarchical Clustering using K-Means Algorithm

이 신 원(Shin-Won Lee)**
안 동 언(Dong-Un An)***
정 성 종(Sung-Jong Chong)****

초 록

정보통신의 기술이 발달하면서 정보의 양이 많아지고 사용자의 질의에 대한 검색 결과 리스트도 많이 추출되므로 빠르고 고품질의 문서 클러스터링 알고리즘이 중요한 역할을 하고 있다. 많은 논문들이 계층적 클러스터링 방법을 이용하여 좋은 성능을 보이지만 시간이 많이 소요된다. 반면 K-means 알고리즘은 시간 복잡도를 줄일 수 있는 방법이다. 본 논문에서는 계층적 클러스터링 시스템인 콘도르(Condor) 시스템에서 간단하고 고품질이며 효율적으로 정보 검색 할 수 있도록 구현하였다. 이 시스템은 K-Means Algorithm을 이용하였으며 클러스터 계층 깊이와 초기값을 조절하여 88%의 정확율을 보였다.

ABSTRACT

Fast and high-quality document clustering algorithms play an important role in providing data exploration by organizing large amounts of information into a small number of meaningful clusters. Many papers have shown that the hierarchical clustering method takes good-performance, but is limited because of its quadratic time complexity. In contrast, with a large number of variables, K-means has a time complexity that is linear in the number of documents, but is thought to produce inferior clusters. In this paper, Condor system using K-Means algorithm Compares with regular method that the initial centroids have been established in advance, our method performance has been improved a lot.

키워드: 문서 클러스터링, K-Means 알고리즘, 클러스터 계층 깊이, 클러스터 초기값, 계층적 클러스터링, 클러스터 중심
document clustering, K-Means algorithm, cluster hierarchy depth, cluster initial value, hierarchical clustering, cluster centroid

---

* 본 연구는 한국과학재단 목적기초연구(R01-2003-000-11588-0) 지원으로 수행되었음.
** 전북대학교 전자정보공학부 박사과정(swlee9237@chonbuk.ac.kr)
*** 전북대학교 전자정보공학부 교수(duan@chonbuk.ac.kr)
**** 전북대학교 전자정보공학부 교수(sjchung@chonbuk.ac.kr)
■ 논문접수일자 : 2004년 11월 17일
■ 게재확정일자 : 2004년 12월 18일

1. 서 론

정보검색의 방법 중 하나인 클러스터 기반 정보검색은 서로 관련있는 문서들을 클러스터로 형성하고, 사용자 질의에 대해서 클러스터의 관련도에 따라 관련이 높은 클러스터에 있는 모든 문서를 검색 결과로 제시하는 것이다. 그리고 벡터공간모델에 기반한 검색은 질의에 나타난 키워드들이 문서에서 어느 정도의 가중치를 가지고 존재하는가를 기준으로 문서들에 우선 순위를 부여한다. 벡터공간 검색은 질의에 나타난 단어를 포함하는 문서는 확실히 검색한다는 것을 보장할 수 있다(이경순 2001).

문서 클러스터링은 대용량의 문서 집합을 주제에 따라 분류하는 것으로 정보 검색 분야에서 문헌 구조를 분석하거나, 검색 효과와 성능을 높이기 위해 이용되고 있다. 문서 클러스터링 방법은 다양한 이론과 방법이 이미 제기되었고 어떤 알고리즘을 선택하느냐에 따라 효율성이 결정된다.

많은 정보 검색 시스템들은 질의어에 대한 결과를 리스트 형태로 제시하기 때문에 적합한 문서를 찾기 위하여 사용자가 리스트를 세밀하게 살펴봐야 한다. 이러한 불편한 점을 개선하기 위하여 시스템이 자동으로 문서를 분류하여 계층적 구조로 제시해 주는 방법에 대한 연구가 필요하다. 계층적 클러스터링은 문서간의 유사도를 통해 단계적으로 계층 구조로 만들어 저장한 것으로 검색 엔진의 계층적 구조를 하향 탐색하여 검색 성능을 높일 수 있다. 또, 사용자에게 검색된 문서 구조를 계층적으로 보여주어 평탄한 구조보다 검색 결과를 직관적으로 이해하기 쉽게 보여준다. 본 논문에서는 콘도로(Condor) 시스템이 사용한 K-Means 클러스터링에서 초기 클러스터링 중심값을 변화시켜 최적의 검색성능을 보이는 초기 중심값과 분류 계층 깊이 사이의 관계를 살펴보고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구로 다양한 문서 클러스터링 기법을 소개하고 3장에서는 정보 검색 시스템인 콘도르 시스템과 사용된 K-Means 알고리즘에 대하여 기술하고 4장에서는 실험 결과를 보여준다. 마지막으로 5장에서는 결론을 맺는다.

2. 관련 연구

문서 클러스터링은 정보 검색의 효율성과 유효성을 증대시키기 위한 목적으로 사용한다. 대표적인 문서 클러스터링의 방법론은 클러스터링의 결과로 생성되는 그룹의 구조에 따라서 계층적 클러스터링(hierarchical clustering method)과 비계층적 클러스터링(non-hierarchical clustering method)으로 나눌 수 있는데 각각의 방법론에 따라 여러 가지 구현 알고리즘이 있다(Qin 1999).

비계층적 클러스터링 기법은 클러스터의 계층을 고려하지 않고, 각 문서를 평면적으로 클러스터링 하는 방법으로 일반적으로 미리 몇 개의 클러스터로 나누어 질 것이라고 예상하고 클러스터의 개수를 제공해야 한다. 이런 기법 중에는 입력되는 문서의 순서에 따라 클러스터링 결과가 달라지는 단일 처리 방법(single pass method)과 이의 단점을 보완한 재배치 방법(reallocation method) 그리고 대용량에

대한 탐색적인 기법으로 사전적인 정보 없이 의미 있는 자료구조를 얻으며 모든 형태의 데이터에 적용 가능한 K-Means 알고리즘이 있다. 이 기법은 처리속도가 빠른 반면 가중치와 비유사성 거리정의와 통계 정보를 반영하지 못하고 클러스터링 오차가 크다는 단점이 있다(Patrice, Marc 1999; Tapas 2000).

계층적 클러스터링은 문서간의 유사도 정보를 토대로 단계적으로 계층적인 클러스터를 형성하는 방법으로 응집 알고리즘(agglomerative method)과 분할 알고리즘(divisive method)이 있다(Ramon and Mollineda 2000). 계층적 응집 알고리즘에는 단일 링크 방법(single link method), 완전 링크 방법(complete link method), 그룹 평균 연결 방법(group average link method)등이 있다(Michael, George, and Vipin 2000). 이 기법은 클러스터링 오차가 적은 반면 대용량에 대한 처리 속도가 느리다.

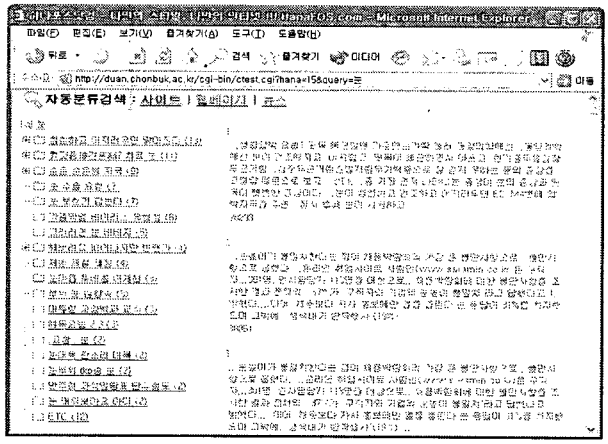
일반적으로 대규모 웹 문서를 처리하는 클러스터링 알고리즘은 수백만 건을 처리하는 정보검색 시스템에 처리의 과부하를 주는 것을 피하고 메모리를 적게 사용해야 할 필요가 있다. 인공지능 분야에서 개발된 기존의 클러스터링 알고리즘들은 고차원의 대규모 데이터 집합으로 문서들을 벡터로 표현할 때 매우 고차원적이며 드문드문 데이터가 나타나서 특성이 있어서 많은 메모리를 요구한다. 따라서 문서에서 중요한 내용은 포함하면서 중요하지 않은 부분을 제외시킨다면 클러스터링의 성능에 크게 영향을 미치지 않으면서 메모리의 요구사항을 줄일 수 있다는 장점을 가질 수 있다.

대규모 웹 문서를 처리하기 위해 정보검색 시스템에서는 처리 속도가 빠르며, 구현과 계산 복잡도를 요구하는 알고리즘을 사용하여 사용자의 검색 요구에 빠르게 응답해야 할 필요성이 있다. K-Means 알고리즘은 비계층적이며 재배치 기법을 사용하는 방법으로서 다양한 응용에서 사용하는 기술이며, Khaled는 Min-Max 기법을 이용하여 클러스터의 거리 계산하는 시간을 줄이려고 했고(Khaled, Sanjak and Vineet 1998), Patrice는 질의 크기에 따라 클러스터 양을 선택하는 기법을 썼다(Patrice, Marc 1999). 본 논문에서는 대규모 웹 문서를 처리하는 데 효율적인 K-Means 알고리즘을 사용하고 클러스터링의 오차를 줄이기 위해서 클러스터 계층 깊이와 초기값을 변화시켜가면서 실험하였고 계층적 클러스터링 구조로 결과를 나타낸다.

3. 콘도르 시스템과 클러스터링 알고리즘

3.1 콘도르(Condor) 시스템

콘도르는 전북대학교 지능공학연구실, 카네기멜론 대학교(CMU : Carnegie Mellon University)의 언어기술연구소(LTI:Language Technology Institute), 그리고 (주)서치라인이 공동 개발한 검색엔진이다.(박순철, 안동언 2003). 콘도르는 웹 검색 분야의 처리 대상규모, 질의 처리, 랭킹방식 및 서비스 형태가 달라서 각 분야에 적합한 구조를 따로 설계하여 별도로 구현되어 있다. <그림 1>은 콘도르 시스템의 검색 결과이다.

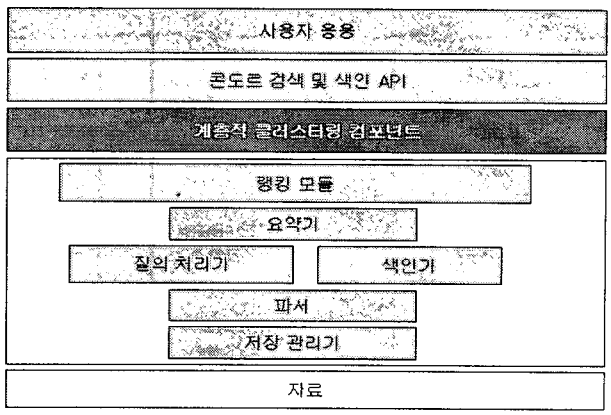


〈그림 1〉 콘도르 시스템의 검색 결과

콘도르는 대용량의 문서를 검색 및 색인 하는 정보 검색 엔진으로 크게 색인(Index) 모듈, 검색 엔진(Search engine) 모듈, 사용자 인터페이스 모듈 등 세 부분으로 나누어져 있다. 콘도르 시스템에서 계층적 클러스터링 부분은 Data를 색인, 질의 처리, 요약 등을 처리하는 엔진 컴포넌트와 API, 사용자 인터페이스 사이에 위치하여 색인이 끝난 뒤 전처리 과정을 담당하게 된다.

〈그림 2〉는 시스템의 소프트웨어 구조이다. 최근의 정보검색 시스템은 벡터 계산시 소요되는 시스템 부하와 메모리 요구사항, 또한 실시간 처리를 고려하여 원문 전체를 사용하지 않고 자동으로 문서를 요약한 후 압축 문서를 사용하는 경향이다. 요약문서 작성은 전체 문서에서 문장별로 중요도를 계산하여 중요도가 높은 문장들을 뽑아내는 방법을 이용한다.

본 논문에서 구현한 클러스터링 모듈을 포



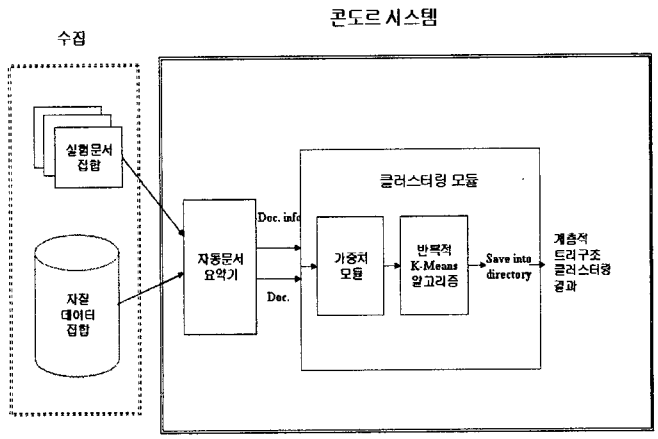
〈그림 2〉 콘도르 시스템 소프트웨어 구조도

함한 시스템이 〈그림 3〉과 같다. 전체 시스템은 크게 자동문서 요약 모듈과 문서 클러스터링 모듈로 구성되어 있다. 자동 문서 요약 모듈은 HTML 태그 제거기, 자질 추출 모듈로 구성되어 있다. 실험 문서 집합은 HTML 태그로 구성되어 있기 때문에 HTML 태그 제거기를 사용하여 각 문서들을 구분하는 시작태그만을 제외하고 HTML 태그를 제거한다. 자질 추출 모듈은 각 문서의 특징을 나타낼 수 있는 자질을 문서로부터 추출하는 부분으로써, 여기에서 자질은 문서 번호, 문서 길이(불용어를 제외한 문서 길이), 전체 문서 수, 평균 문서 길이 등 문서 정보(Document Information)와 문서 내 색인어 위치(Term location), 색인어 번호(Term id), 문서 내 색인어의 빈도수(Term Frequency)로 구성되어 있다. 각 문서의 특성 정보를 가지고 있는 자질 정보들은 자질 데이터 저장소에 저장된다. 실험 문서 전체 집합과 자질 데이터 저장소에 저장된 문서 정보들은 자동 문서 요약기(Automatic Document Summarizer)를 통하여 문서 내에서 가장 중요한 문장을 선택하여 요약문을 생성한다. 본 논문에서는 클러스터링 모듈을 중점적으로 다루고 있다. 다음 절에서 가중치와 K-Means 알고리즘에 대해서 알아본다.

3. 2 K-means 알고리즘을 이용한 문서 클러스터링

K-means 알고리즘을 사용한 이유는 다른 계층적 클러스터링 알고리즘에 비해 정확성은 떨어지지만 구현이 간단하고 처리 속도가 매우 빨라 사용자의 질의에 따라 실시간으로 많은 양의 문서를 클러스터링 해야 하는 웹 검색엔진에 어울리기 때문이다. 그리고 계층적 클러스터링을 구현하기 위해서 K-means 알고리즘의 초기값 선정을 변형시켰고 계층 깊이에 대해서도 다르게 구현하였다.

K-Means 알고리즘은 문서와 클러스터의 중심값과의 유사도를 측정하여 문서를 적합한 클러스터에 재배치하는 기법이다. 클러스터에 영향을 미치는 요소는 초기 클러스터 중심값과



〈그림 3〉 클러스터링 모듈

새로 생성된 클러스터 중심 결정이다.

본 논문의 클러스터 중심값(centroid)은 클러스터에 속하는 문서들의 평균 벡터값을 이용한다. 초기의 클러스터를 형성하고 〈그림 4〉와 같이 이를 계속적으로 정련하는 과정을 통해 최종의 클러스터를 형성한다. 이 기법은 초기 클러스터의 선택에 따라서 클러스터의 결과가 달라지며 특히 초기 클러스터 중심을 어떻게 선택하는가에 따라서 빠른 시간에 최적의 클러스터링 결과가 나오는 경우와 그렇지 않은 경우가 존재한다.

문서 클러스터링 모듈은 자동 문서 요약기로 부터 생성된 요약문(Summarized Document)과 문서 정보(Term ID, TF, DF)를 이용하여 문서간의 유사도를 기반으로 클러스터링을 수행하며, 각 문서내의 색인어의 가중치를 부여하는 weighting 부분과 클러스터링을 담당하는 K-Means 알고리즘 담당 모듈로 구성되어 있다.

클러스터링에 영향을 미치는 또 다른 요소는 클러스터링 과정에서 발생하는 새로운 클러스터 중심(Cluster Centroid)을 결정하는 것이다. K-Means 알고리즘에서는 클러스터에 속하는 문서들의 색인어와 가중치만을 단순히 하나의 클러스터 벡터로 병합하였으며 새로운 클러스터 중심은 식(1)과 같다.

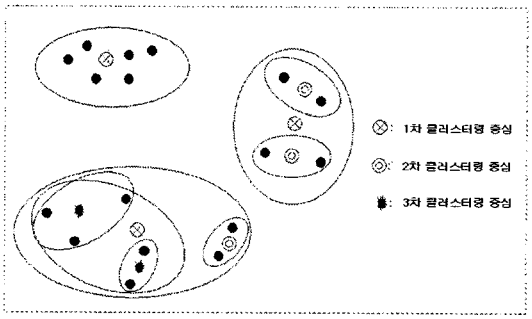
$$\overline{c_j} = \frac{1}{|c_j|} \sum_{i=1}^{|c_j|} \overline{d_i} \qquad (1)$$

K-Means 알고리즘에서는 클러스터 중심을 L-차원의 공간에서 벡터 $(x_{i1}, x_{i2}, \ldots, x_{iL})$로 표현하였을 때 클러스터에 속하는 문서를 대표하는 색인어와 가중치만을 단순히 하나의 클러스터 벡터로 머지(merge)한 것이다.

본 논문에서 사용한 K-Means 알고리즘에서는 초기 클러스터 중심을 결정할 때 문서를 3개(m=3)로 선택하여 중복된 색인어를 제외하고 병합한 후 초기 클러스터 중심 벡터로 설정하였다. 변형한 초기 클러스터 중심은 식(2)와 같다.

$$c_i^{initial} = \sum_{j}^{m=3} d_j \qquad (2)$$

클러스터링을 수행하는 과정에서 문서와 클러스터들간의 거리는 하나의 문서만을 초기 클러스터로 설정할 때 문서 클러스터와 클러스터간 거리는 전체적으로 커지며, 최단 거리 문서



〈그림 4〉 클러스터 중심 생성 과정

-클러스터 거리 판별에 영향을 주게 된다.

새로운 클러스터 중심 벡터는 클러스터에 포함된 모든 문서들이 갖는 색인어의 가중치의 평균으로 계산한다. 클러스터 중심 ci와 문서 dj가 병합되어서 생성된 클러스터 중심은 식 (3)과 같이 계산한다.

$$c_i^{new} = \frac{m_i \cdot C_i + m_j \cdot d_j}{m_i + m_j} \quad (3)$$

c_i : i^{th} 클러스터 벡터

d_j : j^{th} 클러스터에 할당된 문서 벡터

m_i : i^{th} 클러스터의 크기

m_{ij} : i^{th} 클러스터에 할당된 j^{th} 문서의 크기

c_i^{new} : i^{th} 새로운 클러스터 중심 벡터

생성된 클러스터 중심은 클러스터에 속하는 문서들이 클러스터 중심을 형성하는 과정에서 문서에 표현되어 있는 색인어들의 가중치들로 자신들의 특성을 반영하며 서로 이웃한 문서들에게 영향을 미치게 되어 문서간의 문맥을 고려한 클러스터링 효과를 얻을 수 있게 된다.

사용한 K-means 알고리즘은 〈표 1〉과 같다.

〈표 1〉 사용한 K-Means 알고리즘

1. K값 클러스터 개수를 구한다.
2. K개의 초기 중심값을 구한다.

   $C_i^{initial} = \sum_{j=1}^{3} d_j$

   $C_i^{initial}$: I번째 클러스터 벡터

   d_j : j번째 문서 벡터, j=rand()%100
3. 각 문서(d)들과 중심값(c) 사이의 거리를 구한다.

   $dist(\overline{d_i}, \overline{c_j}) = \sqrt{\sum_{k=1} (d_{ki} - c_{kj})^2}$,

   $i = 1, 2, \ldots n$ n: 전체문서개수

   $j = 1, 2, \ldots K$ k: $centroid$의개수
4. 문서를 가장 짧은 거리의 중심값에 할당한다.

   arg min $dist(\overline{d_i}, \overline{c_j})$

   . $i = 1, \ldots n$, $j = 1, \ldots k$

   $d_i \in G_c$, if $dist(\overline{d_i}, \overline{c_j}) < dist(\overline{d_i}, \overline{c_l})$

   (for all $l = 1, 2, \ldots, k$ $l \neq j$)
5. 새로운 클러스터 중심값을 재계산 한다.

   $c_i^{new} = \frac{m_i \cdot C_i + m_{ij} \cdot d_{ij}}{m_i + m_{ij}}$

   c_i : i^{th} 클러스터 벡터

   d_{ij} : i^{th} 클러스터에 할당된 j^{th} 문서의 벡터

   m_i : i^{th} 클러스터의 크기

   m_{ij} : i^{th} 클러스터에 할당된 j^{th} 문서의 크기

   c_i^{new} : i^{th} 새로운 클러스터 중심 벡터
6. 이전의 중심값과 새로운 중심값을 비교하여 벡터간 차이가 거의 없을 때까지 반복한다.

   $If \max \delta(\overline{c_j^{old}}, \overline{c_j^{new}}) < \theta$ $then return$

   $else$ $goto 3$
7. 클러스터 내 문서의 유사도가 한계치보다 적으면 클러스터 안에서 다시 클러스터링 한다.
8. 클러스터를 트리에 저장한다.

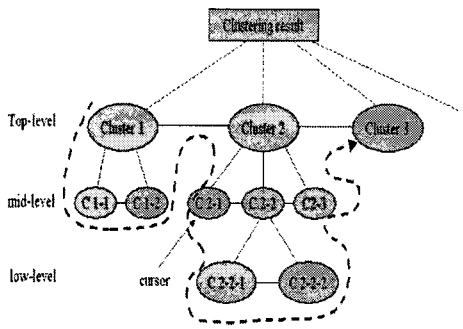
3. 3 클러스터링 결과 계층 구조

총 클러스터의 계층은 문서 집합 상황에 따라 최고 3단계까지 세부 분류를 한다. 그렇지만 소속 문서들의 상황에 따라 어떤 클러스터는 2, 3단계까지 세부 분류가 될 수도 있고 어떤 클러스터는 1단계에서 분류가 멈출 수도 있다. 각각의 패스(pass)와 노드(node)가 임의의 숫자의 자식 노드(child node) 개수를 가진 트리(tree) 구조를 접근하는 것과 같은 접근 방식으로 하위분류결과(자식노드)가 있으면 그 하위결과를 접근하고, 그 다음에는 같은 단계(level)의 옆 클러스터(형제노드)로 옮겨가고 하는 깊이우선(depth-first) 재귀적 접근(recursive traverse)을 바탕으로 한다. 〈그림 5〉는 클러스터링의 결과 구조를 나타낸다.

계층적 클러스터링에 접근하기 위한 알고리즘은 〈그림 6〉과 같다.

4. 실험 및 성능 평가

본 논문에서 사용한 실험 데이터는 2002년 5월부터 2003년 9월까지의 조선일보, 중앙일보, 동아일보 등의 뉴스와 10월부터 11월의 한겨레신문, 한국일보, 문화일보 등의 뉴스를 매 10분 간격으로 수집한 후 색인하여 실험에 사용하였다.



〈그림 5〉 클러스터링 결과 구조

4. 1 실 험

본 시스템에서는 명사 이외에 형용사, 동사 등 시소러스를 이용하여 최적의 자질 단어를 추출하였다. 각각의 클러스터에서 최대 3개 키워드(keyword)를 사용하여 실험하였다. 색인 단어의 가중치는 식(4)와 같이 계산하였다.

$$weight = \frac{tf}{tf + 2} \times \frac{df + 2}{df} \qquad (4)$$

tf : term frequency

df : document frequency

문서 가중치의 영향을 줄이기 위해 용어 빈도수(term frequency)를 $tf/(tf+2)$로 하고 문서 가중치를 향상 시켰다. 〈그림 7〉은 자질 단어 선택 알고리즘이다.

```
access_current (cid)
{
    get_fist_cluster_this level (cursor, cid)
    if (first_child_cid = have_child (cursor, cid))
      access_current (first_child_cid)
    while (next exist)
    {
        get_next_cluster_this_level (cursor, cid)
        if (first_child_cid = have_child (cursor, cid)
          access_current (first_child_cid)
    }
}
```

〈그림 6〉 클러스터링에 접근하는 알고리즘

1) Output noun using thesaurus

$if\ (term_{i-C_j} \in noun)\ then\ (term_i \rightarrow TermS_{C_j})$

$TermS_{C_j}$: *jth cluster's index words collection*

$term_{i-C_j}$: *ith index word of jth cluster*

i : *the number of special cluster's index words*

j : *the number of clusters*

2) Remove the words are used by superstratum directory

$if\ (TermS_{C_{j+1}} \notin TermS_{C_j})\ then\ TermS_{C_{j+1}}$

3) Select 3 words with the highest weight as the cluster feature words

〈그림 7〉 자질 단어 선택 알고리즘

본 논문에서는 최적의 초기 클러스터 수와 클러스터링 깊이를 실험하였다. 초기 클러스터의 수(k)를 검색된 문서 수(n)에 따라 $\ln(n), n/4, 0.5\sqrt{n}$ 로 변화시켰다.

4. 2 평 가

본 논문에서는 질의어에 대한 검색된 문서의 수(n)의 최대값을 700으로 제한하고, 계층 깊이를 3으로 하여 검색된 문서 수(n)에 따라 유동적으로 초기 클러스터의 개수(k)를 변화시켜 최적의 상태를 알아내기 위하여 실험을 하였다. 〈표 2〉와 같이 검색된 문서의 수를 200, 400, 700으로 변화시키고 초기 클러스터의 수도 $\ln(n), n/4, 0.5\sqrt{n}$ 로 변화시켰을 때 문서의 수를 200으로 제한한 경우는 초기 클러스터 수가 $\ln(n)$인 경우 성능이 좋게 나타남을 알 수 있다.

질의어에 대한 검색된 문서의 수를 100개로 제한한 실험에서는 초기 클러스터의 개수를 10, 분류 계층 깊이를 3으로 하였을 때 가장 좋은 결과를 나타냈다. 그래서 동음이의어를 질의어에 사용하여 정확한 클러스터 수를 조사하였다.

〈표 3〉은 초기 클러스터 수가 15인 경우에 계층 깊이가 2인 경우와 3인 경우에 대해서 동음이의어를 대상으로 실험한 결과이다. 계층 깊이가 2인 경우보다 3인 경우가 더 정확하게

〈표 2〉 문서수에 따른 초기 클러스터 수

	3-depth average precision (%)		
검색된 관련 문서 수(n)	$\ln(n)$	$n/4$	$0.5\sqrt{n}$
$0 \pi n \leq 200$	56.3	55.6	52.4
$200 \pi n \leq 400$	54.0	61.2	59.2
$400 \pi n \leq 700$	53.9	56.7	58.1

〈표 3〉 실험 결과

질의어	TCN CCN	D = 2 k = 15	D=3 k=15
유 산	TCN	32	33
	CCN	20	23
장 수	TCN	28	33
	CCN	13	24
조 선	TCN	30	31
	CCN	26	26
화 장	TCN	32	35
	CCN	27	29

TCN: 전체 클러스터 수
CCN: 정확한 클러스터 수
k:초기 클러스터 수
D: 분류 계층 깊이

클러스터링 되는 것을 알 수 있다.

〈표 4〉는 초기 클러스터와 깊이를 각각 다르게 했을 때의 결과이다. 초기 클러스터 수가 10인 경우를 보면 계층 깊이가 3인 경우가 가장 좋음을 알 수 있다. 클러스터 수가 20인 경우를 제외하고는 계층 깊이가 3인 경우가 가장 좋음을 알 수 있다.

〈그림 8〉은 초기 클러스터 수와 계층 깊이와의 관계를 그래프로 나타내었다. 그림에서 볼 수 있듯이 계층 깊이 3인 경우 가장 좋은 성능을 보임을 알 수 있다. 실험 결과 초기 클러스터 수가 10이고 계층 깊이 3인 경우 정확율이 88%를 넘는 것을 알 수 있다.

〈그림 9〉는 초기 클러스터 수가 10이고 계층 깊이가 2인 경우 실험 결과이다.

〈그림 10〉은 초기 클러스터 수가 10이고 계층 깊이가 3인 경우 실험 결과이다.

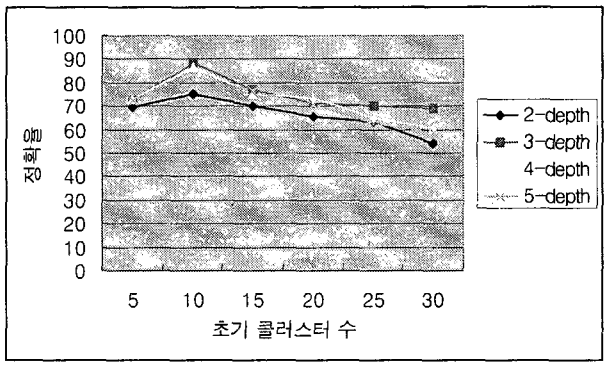
계층 깊이가 2인 경우보다 계층 깊이가 3인 경우 더 자세히 분류되었음을 보여준다.

5. 결 론

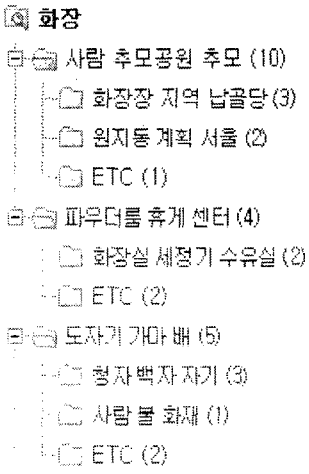
많은 정보검색 시스템들이 개발되었고 새로운 시도들이 이루어지고 있다. 사용자에게 검색된 정보를 리스트 형태가 아닌 계층적 구조

〈표 4〉 초기 클러스터수와 깊이

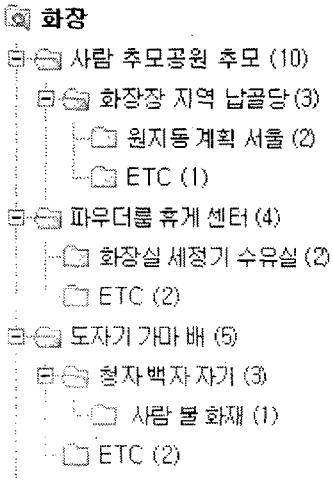
초기 클러스터 수	정확율(%)			
	2-depth	3-depth	4-depth	5-depth
5	69.73	72.54	73.12	72.14
10	75.42	88.74	87.04	82.48
15	70.00	77.29	75.24	76.68
20	65.32	71.38	73.38	70.49
25	63.49	70.12	64.67	62.38
30	54.21	68.89	60.43	58.49



〈그림 8〉 초기 클러스터 수와 계층 깊이



〈그림 9〉 계층 깊이 2단계인 경우



〈그림 10〉 계층 깊이 3단계인 경우

로 보여 주는 계층적 클러스터링은 사용자들이 직관적으로 정보를 검색하는데 많은 도움이 될 것이다. 가능하다면 첫 페이지에 사용자가 찾고자하는 정보를 바로 찾을 수 있으면 가장 좋은 시스템이라 할 수 있을 것이다. 본 논문에서는 대용량의 웹 문서를 빠른 시간에 검색하기 위해서 K-Means 알고리즘을 이용하고 클러스터링 결과를 계층적 구조로 보여주고 초기 클러스터링의 중심값의 개수 설정이 모호하다는 단점을 개선하기 위하여 검색된 문서수와 초기 클러스터링 중심값 개수 사이의 함수를 이용하여 유동적으로 바꿀 것을 제안하였다.

검색된 관련 문서 수에 따라서 계층 깊이 3인 경우 초기 클러스터링 중심값의 개수는 변화시켜 가면서 실험을 하였다. 문서 수와 초기 클러스터 수에 따라서 정확율이 달라지는 것을 알아 보았다.

문서 수를 100개로 제한하고 초기 클러스터

수가 15인 경우 동음이의어에 대해서 계층 깊이 2인 경우와 3인 경우를 실험해 보았는데 계층 깊이 3인 경우가 더 좋은 결과를 보임을 알 수 있었다.

또한 초기 클러스터 수를 변화시키고 계층 깊이도 다르게 한 경우를 실험해 보았는데 초기 클러스터 수가 10이고 계층 깊이 3인 경우가 정확율이 가장 좋음을 알 수 있었다.

향후 연구로는 다양한 분류 깊이에서의 최적의 초기 클러스터링 개수를 찾아야 하겠고, 계층적 구조로 제시한 검색결과에서 나오는 기타(ETC)항목의 효과적인 처리방법, 검색 속도의 향상 등이 연구되어야 한다.

참 고 문 헌

김해남, 이신원, 안동언, 정성종. 2004. 계층적 클러스터링에서 분류 계층 깊이에 관한 연구. 『한국정보처리학회 춘계학술발표대회 논문집』, 2004년 5월 14-15일[서울: 중앙대학교].

박순철, 안동언. 2003. 콘도르 정보 검색 시스템. 『한국산업정보학회지』, 8(4): 31-37.

오형진. 2002. 『클러스터 중심 결정 방법을 개선한 K-Means Algorithm의 구현』. 석사학위논문, 전북대학교 대학원, 컴퓨터공학과.

오형진, 고지현, 안동언, 박순철. 2003. 색인어 가중치 부여 방법에 따른 K-Means 문서 클러스터링의 LSI 분석. 『한국정보처리학회지』, 10-B(7): 735-742.

이경순. 2001. 『정보검색에서 벡터공간 검색과 클러스터 분석을 통한 문서 순위 결정 모델』. 박사학위논문, 한국과학기술원.

이상선, 이신원, 안동언, 정성종. 2004. 계층적 클러스터링에서 분류 대표어 선정에 관한 연구. 『한국정보처리학회 춘계학술발표대회 논문집』, 2004년 5월 14-15일[서울: 중앙대학교].

Baeza-Yates, Rebeiro-Neto. 1999. 『Modern Information Retrieval』. Addison-Wesley.

Khaled Alsabti, Sanjay Ranka, Vineet Singh. 1998. "An Efficient K-Means Clustering Algorithm". IPPS/SPDP Workshop on High Performance Data Mining. 〈http://www.cit.gu.edu.au/~s2130677/teaching/KDD.d/readings.d/alsabti98efficient.pdf〉.

Michael Steinbach, George Karypis, Vipin Kumar. 2000. "A Comparison of Document Clustering Techniques". Technical Report #00_034, Department of Computer Science and Engineering, University of Minnesota.

Patrice Bellot, Marc El-Beze. 1999. "A Clustering Method for Information Retrieval". Technical Report IR-0199.

Qin He. "A Review of Clustering Algorithms as Applied in IR". UIUCLIS1999/6+IRG.

Ramon A., Mollineda, Enrique Vidal. 2000. "A relative approach to hierarchical clustering". in Proceeding of ACM symposium of Computational geometry, Hongkong, June 12-14.

Tapas Kanung. 2000. "The Analysis of a Simple k-Means Clustering Algorithms". in Proceedings of ACM symposium on Computational geometry, Hongkong, June 12-14.