점진적 알고리즘을 이용한 웹 문서 클러스터링 시스템의 설계 및 구현

황 태 호*, 손 기 락
한국외국어대학교 컴퓨터공학과 대학원

Design and implementation of web document clustering system using on incremental algorithm

Tae-Ho Hwang, Ki-Rack Sohn
Dept. of Computer Science & Engineering, Hankuk University of Foreign Studies

클러스터 분석은 관측의 대상이 되는 집합에 맞는 분류 구조를 생성하는 데 이용되는 통계학적인 기술이다 정보검색 응용에서 전형적으로 발견되는 높은 차원을 가진 많은 데이터 집합을 클러스터하기 위하여, 많은 공간과 시간이 필요하다. SLINK 알고리즘은 $O(n^2)$의 시간과 O(n)의 공간의 성능을 갖으며 점진성을 반영할 수 있는 알고리즘이다. SLINK 알고리즘을 이용하여 검색 엔진의 검색 결과에 온라인으로 클러스터 분류를 수행하는 시스템을 구현하였다. 구현 된 시스템은 상대적으로 높은 정확도와 각 클러스터를 저장하고 표현하는데 있어서의 장점을 제공하며, 상대적으로 느린 수행 속도는 온라인으로 문서들이 다운로드되는 속도가 느리므로 문제가 되지 않음을 알 수 있었다.

1 서론

일반적인 문서 검색 시스템은 사용자가 관련된 문서를 찾도록 계층별로 나누어진 문서들의 긴 리스트를 반환한다. 최근의 웹 검색 엔진 (e g Excite, AltaVista)은 대부분 이러한 패러다임을 사용하며 극히 낮은 정확도(precision)를 갖는다 웹 검색 엔진이 나열한 리스트들의 낮은 정확도는 사용자가 원하는 정보를 찾는 것을 어렵게 하기 때문에 리스트들의 정확도를 높이기 위해 여러 가지 방법이 시도되었으며 문서의 클러스터링은 그러한 시도 중에 하나이다

대부분의 문서 클러스터링 알고리즘들은 전체 문서들의 집합에 대해 off-line 에서 작업을 한다 그러나, 웹 검색 엔진에서 이러한 문서 클러스터링 알고리즘들을 각각의 질의에 적용하기에는 문서의 범위가 너무 방대하고 유동적이다 따라서 하나의 질의에 해당하는 문서 집합들을 대상으로 클러스터링 알고리즘을 적용하는 것을 생각해 볼 수 있고, 이러한 작업은 웹 검색 엔진과는 분리된 다른 기계 또는 클라이언트에서 수행되어야 한다.

웹 검색 엔진의 사용자들은 질의에 대해 검색 엔진으로부터 링크와 "Snippet"이라고 하는 작은 문서를 얻는다. 이 링크에 해당하는 완전한 웹 문서를 얻는 대신에 이 웹 문서를 요약한 짧은 Snippet 을 대상으로 클러스터링 과정을 수행할 수 있으며, 이것은 각 링크에 해당하는 완전한 웹 문서를 클러스터링 작업을 수행하는 클라이언트로 가져오는 시간을 없앨 수 있다

웹 문서들을 대상으로 클러스터링을 수행하는 모델에 있어서 고려해야 할 몇 가지 특성들은 다음과 같다[9].

(1) **관련성** 전혀 무관한 여러 문서들로부터 사용자의 질의에 대한 관련성을 가진 문서들을 웹 서치 엔진으로부터 가져온다. 따라서 클러스터링의 대상이 되는 각 문서들은 서로간에 어느 정도의 관련성을 모두 가지고 있다

(2) **속도** 문서는 네트웍을 통해 전송되고 문서들이 다운로드 되는 속도는 매우 느리다 따라서, 병렬적인 작업이 필요하며 클러스터링에 필요한 시간은 가능한 짧아야 한다

(3) **감내성** 어느 정도의 허용 시간 내에 다운로드한 문서들을 대상으로 클러스터링 과정을 수행하여야 한다.

(4) **점진성** 속도와 감내성 및 전체적인 성능 향상을 위해 하나의 데이터를 웹으로부터 받는 즉시, 클러스터링을 포함한 모든 처리 과정을 수행할 수 있어야 한다

(5) **중첩** : 문서의 특성상 한 문서는 여러 개의 주제를 언급 할 수 있기 때문에 각 클러스터에 중첩되게 문서들을 포함시킬 경우 정확도를 높일 수 있다

(6) **요약** : 사용자가 각 클러스터가 포함하는 문서들의 특성을 알 수 있어야 한다 따라서 각 클러스터에 대한 요약된 설명이 필요하다

(7) **사용자의 결정** 잘 정의된 클러스터라고 할지라도 사용자의 요구 및 의도에 따라 클러스터의 레벨 및 범위는 달라 질 수 있으며 정확도 역시 달라질 수 있다

위에서 나열한 웹 문서 클러스터링 시스템을 위한 특성 및 요구 사항들을 만족하는 몇 가지 알고리즘을 2 절에서 살펴보고 3 절에서는 시스템에서 사용한 SLINK 알고리즘을 살피본다 4 절에서 실제 시스템의 구성과 작동에 대하여 살펴보고 5 절에서 논문에서 구현한 시스템과 다른 시스템의 성능을 비교한다

2 관련연구

기존의 정보 검색 연구에서 사용된 몇 가지의 대표적인 클러스터링 알고리즘을 살펴보고 웹 문서 클러스터링을 위한 각 알고리즘의 장단점을 비교해 볼 필요가 있다.

Hierarchical Agglomerative Clustering[2]은 여러 논문에서 결과의 성능에 대하여 충분한 분석이 이루어 졌으며, 문서 클러스터링에서 가장 널리 사용되는 알고리즘이나 일반적으로 수행 속도가 매우 느리다 단일 연결과 집단평균연결 방법의 경우 $O(n^2)$의 시간이 걸리며 완전연결방법을 사용할 때는 $O(n^3)$의 시간이 필요하다[11]. 따라서, 실제 데이터들을 대상으로 이 알고리즘을 적용하기 위해서는 적절한 변형이 필요하다.. 여러 많은 논문에서 HAC 에 관한 몇 가지 불완전한 점들을 지적하고 이것들에 대한 보완 알고리즘들을 제안하였다.

온라인상의 클러스터링에 필요한 속도를 만족시키는 가장 좋은 것은 선형 시간 클러스터링 알고리즘이다 K-Means 알고리즘[1]의 경우 O(nkT) – k는 이상적인 클러스터의 개수, T는 과정의 반복 횟수 – 의 시간 복잡도를 가지며, HAC 와 달리 클러스터의 중첩을 허용한다 그러나, K-Means 알고리즘은 구형(spherical)의 분포를 기진 공간적인 데이터들을 대상으로 하였을 때 효율적으로 유사성을 비교하여 클러스터링을 수행한다 따라서, 각 웹 문서들의 가중치를 둔 벡터들을 K-Means 알고리즘이 사용하는 공간적인 도메인으로 어떻게 옮길 것인가에 관한 문제가 있다.

Buckshot[7]은 HAC 알고리즘과 K-Means 알고리즘을 적절히 사용한다 초기 클러스터의 중심 값을 결정하는 것에서 문서들의 Sample 들에 대해 HAC 알고리즘을 사용하여 각 클러스터의 유사성의 정도를 나타내고, 이를 공간 도메인에 적용하여 K-Means 알고리즘을 사용한다 물론, 문서들의 Sample 들을 추출하는 문제에 있어서 여러 문제점을 가지고 있만 Buckshot 의 주목할 김은 점진성이다

마지막으로 살피볼 알고리즘은 Suffix Tree Clustering[8]이다 STC 는 기존의 방식과는 다른 접근 방식으로 Suffix Tree 를 이용한 동일 어구(phrase)의 발생에 근거하여 클러스터링을 O(n)의 시간에 수행한다 STC 는 앞에서 살펴본 웹 문서 클러스터링 시스템의 특징을 모두 반영할 수 있으며, STC 를 이용한 웹 문서의 클러스터링 시스템[9]이 설계되었다. Suffix Tree 는 어떤 문자열 대한 suffix 들을 모두 포함하는 compact trie 이며, 클러스터링은 suffix tree 를 구성하고 베이스 노드를 구분하는

과정이다. 기존의 문서 클러스터링 알고리즘에 비해 STC는 상당히 빠른 수행속도를 보이며, 온라인상에서 웹 문서들을 효과적으로 클러스터링 할 수 있는 장점을 가지고 있다. 그러나, 클러스터링 단계에서 메모리상에 suffix tree를 유지하기 때문에 문서 집합이 커질 경우 문제가 발생한다.

3 SLINK

HAC의 단일연결 방법(single-link or nearest-neighbour)은 가장 전통적인 클러스터링 방법이며, 여러 문제점들을 보완한 SLINK[2]을 이용하여 웹 문서 클러스터링 시스템을 설계 하였다.

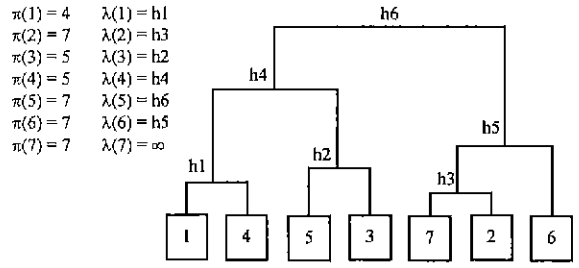
SLINK를 위한 dissimilarity coefficient(DC)의 정의는 다음과 같다.

$$d : P \times P \rightarrow R$$
$$P \text{ 는 문서 객체 집합, } d(a, a) = 0 \text{ for all } a \in P$$
$$d(D_i, D_j) = 1 - \frac{\sum_{k=1}^{L} (weight_{ik} \cdot weight_{jk})}{\sqrt{\sum_{k=1}^{L} weight_{ik}^2 \times \sum_{k=1}^{L} weight_{jk}^2}}$$

d는 가중치 벡터 모델에서 일반적으로 사용하는 코사인 상관 계수[11]이다. SLINK의 결과는 일반적으로 tree-diagram으로 표현하는데 익숙한 dendrogram이며 $c : [0, \infty) \rightarrow E(P)$로 정의할 수 있다. $E(P)$는 P에서 서로 유사한 관계를 가진 데이터들의 집합이며 SLINK를 이용한 클러스터링은 $d \rightarrow c$이다. SLINK 알고리즘에서 dendrogram은 포인터 표현인 두개의 인덱스로 표현되며 정의는 다음과 같다.

$$\pi : 1, \quad , N \rightarrow 1, \quad , N$$
$$\lambda : 1, \ldots, N \rightarrow 1, \quad , N \rightarrow [0, \infty]$$
$$\pi(N) = N, \lambda(N) = \infty, \pi(i) > i, \lambda(\pi(i)) > \lambda(i)$$
$$\text{for } i < N$$
$$\lambda(i) = \min \{ h \mid \exists j > i \text{ with } (i\ j) \in c(h) \}$$
$$\pi(i) = \max \{ j : (i, j) \in c(h) \}$$

<그림 1>은 포인터 표현을 이용하여 dendrogram을 나타낸 예제이다. λ는 연결된 문서 또는 클러스터 간의 최소 DC를 나타내고 π는 클러스터에 포함된 문서의 가장 큰 index를 갖는다.



<그림 1> The pointer representation

SLINK 알고리즘은 세 개의 배열을 사용하는데 Π, Λ는 λ_n과 π_n을 포함한다. n개의 문서로 구성된 클러스터 구조에서 하나의 새로운 문서, n+1이 추가 되었을 때 SLINK 알고리즘은 Π, Λ 그리고, n+1 번째의 object와 n개의 object 간의 DC를 갖는 일 차원 배열, M을 재귀 연산을 통해 새로운 클러스터 구조로 갱신한다. SLINK 알고리즘은 다음과 같다.

```
1 Set   Π(n+1) to n+1, Λ(n+1) to ∞
2 Set   M(i) to d(i, n+1) for i = 1,  , n
3 For   i increasing from 1 to n
    if Λ(i) ≥ M(i)
      set   M(Π(i)) to min { M(Π(i)), Λ(i) }
      set   Λ(i) to M(i)
      set   Π(i) to n+1
    if   Λ(i) < M(i)
      set   M(Π(i)) to min { M(Π(i)), M(i) }
4 For   i increasing from 1 to n
    if   Λ(i) ≥ Λ(M(i))
      set   Π(i) to n+1
```

SLINK 알고리즘의 공간적인 성능은 DC 얻는데 필요한 O(n)이며, HAC가 갖는 최소 시간인 $O(n^2)$을 만족한다.

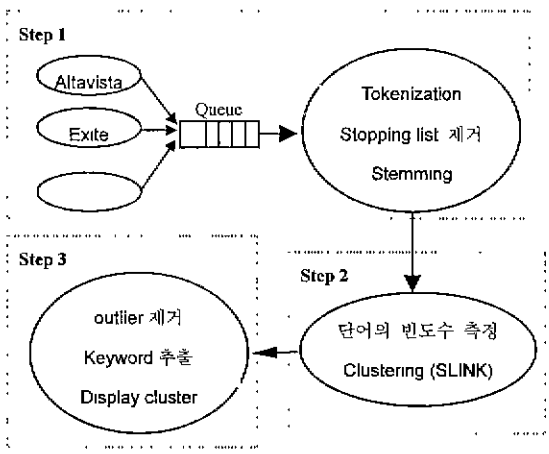
클러스터 결과를 시각적으로 표현하는 것에 있어서 포인터 표현은 전체적인 클러스터 구조를 이해하기 어렵다. 일반적으로 tree-diagram에 익숙한 dendrogram으로 클러스터를 표현하기 위하여 포인터 표현은 압축된 표현으로 변환하여 나타내어야 한다. 압축된 표현은 ν, τ의 두 함수로 구성되며 정의는 다음과 같다.

$$\nu(i) = \lambda(\tau(i))$$
$$\tau^1(\pi(\tau(i))) > i \quad \text{if } i < n,$$
$$\text{and}$$
$$\nu(j) \le \nu(i) \quad \text{if } i \le j \ \tau^1(\pi(\tau(i)))$$

포인터 표현에서 압축된 표현으로 전환하는데 필요한 시간은 $O(N^2)$이며 모두 4N + overhead의 공간적인 크기를 요구한다. 압축된 표현은 tree-diagram을 위한 수치적인 형식을 갖는다. tree-diagram을 위해 먼저 1, , N에서 baseline을 따라 문서의 번호를 삽입한다. i번째 위치의 번호는 $\tau(i)$가 되고 이것을 따라 높이를 나타내는 $\nu(i)$를 수직축에 나타낸다. 수직축이 모두 그려졌을 때 위치 번호가 증가하는 방향으로 다른 수직축과 만날 때까지 그린다. 이러한 방법으로 dendrogram을 나타내는 tree-diagram을 얻을 수 있다.

4 웹 문서 클러스터링 시스템

전체 시스템은 <그림 2>에서와 같이 세단계에 걸쳐 수행되며, JDK를 이용한 Java platform에서 구현되었다.



<그림 2> 웹 문서 클러스터링 시스템 구조

4.1 Step 1

첫 번째 단계는 전처리 단계로서 <그림 2>와 같이 각 웹 검색 엔진으로부터 질의의 결과를 가져와서 클러스터링에 필요한 과정들을 수행한다.

웹 검색 엔진으로부터의 결과는 HTML 문서 형태이다. 따라서, 첫 단계에서 HTML TAG로 구성된 트리를 파싱하여 엔트리를 추출하고 각 엔트리마다 필요한 데이터 부분 추출한다. 웹 검색 엔진으로부터 문서를 가져오는 것은 W4F[10]의 Java package를 이용한다. W4F는 매크로 형식의 정규식을 제공함으로서 사용자가 인터넷상의 HTML 문서에 대하여 원하는 문서의 특정 필드를 추출하여 가져오는 Java source code를 생성한다.

정보 검색에서 영어의 "the", "of", "and", "to"와 같이 발생 빈도가 높은 단어의 대부분은 색인 용어로 가치가 없다는 것이 인식되고 있다

이러한 단어들을 분류하여 불용어 목록(Stopping list)를 만든 후, 문서에 대해 어휘 분석의 과정에서 유한 오토마타를 이용하여 제거한다 시스템의 성능을 향상 시킬 수 있는 또 한가지의 방법은 스테밍 알고리즘을 적용하는 것이다 스테밍 알고리즘은 탐색 용어의 어형론적인 변형을 찾는 방법을 제공한다 그러나, 이러한 방법을 사용한다면 특정 언어에 의존하는 시스템이 될 수 있으며, 클러스터링의 전처리 과정에 큰 시간을 투자하게 되므로 전체 시스템의 성능을 저하 시킬 수 있다 따라서, 최소한의 범위 내에서 두 알고리즘을 적용하며, 토큰 분리와 불용어 목록의 제거는 같은 처리과정으로 수행한다. 시스템에서 200 개의 snippet 에서 모두 1500~1600 개의 서로 다른 단어가 사용 되었고, 전 처리 과정을 수행한 후, 1000 개 안팎의 단어들이 클러스터링의 대상으로서 남았다

4.2 Step 2

두 번째 단계에서 DC 측정을 위하여 각 entry 별 단어의 출현 빈도수를 계산하여 해쉬 테이블에 저장한다 snippet 은 상당히 요약되고 압축된 단어들의 리스트이기 때문에 각 단어의 출현 빈도수는 200 개의 snippet 을 기준으로 하였을 때, 결과 전체에서 1~10 정도로 낮았으며, 각 entry 내에서의 출현 빈도수도 역시 1~3 정도로 낮았다

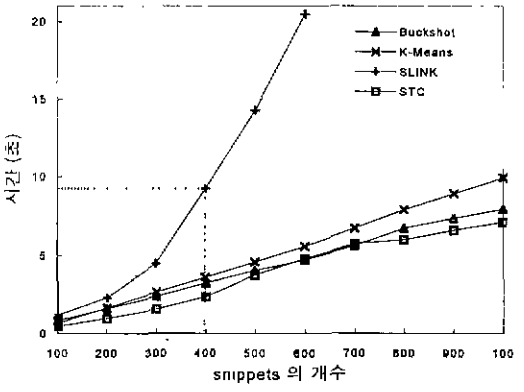
클러스터링은 앞에서 설명한 SLINK 알고리즘을 적용하였다 첫 단계와 두 번째 단계는 pipeline 으로서 병렬적인 작업으로 수행된다 이것은 SLINK 의 점진성 때문에 가능하며, 결국 SLINK 가 가진 느린 수행속도의 단점을 보완 할 수 있다

4.3 Step 3

마지막 단계에서 outlier 를 제거하고 각 클러스터들의 키워드들을 추출한 후 이를 출력하는 과정이다 이때, 최적화된 클러스터의 개수로 각 entry 를 출력하고 사용자가 원하는 레벨의 클러스터를 보여준다

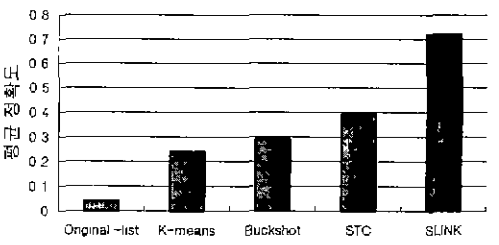
SLINK 의 결과인 dendrogram 의 특성은 클러스터 구조에서 outlier 의 구분을 매우 쉽게 한다 대부분의 outlier 는 dendrogram 의 상위 레벨에서 나타나게 되며, 결과를 출력하는 과정에서 간단한 처리과정을 통해 outlier 를 제거 했을 때, 클러스터의 간의 정확도는 크게 향상될 수 있다 또한 사용자가 원하는 레벨에서 클러스터 구조를 볼 수 있기 때문에 사용자에게 보다 정확한 결과를 보여줄 수 있다.

5 성능 비교



<그림 3> 수행 속도

<그림 3>은 Pentium 400 processor system 에서 snippet 의 개수를 100~1000 개에서 변화 시키며 다른 문서 클러스터링 알고리즘과의 수행속도를 비교한 그래프이다 SLINK 가 예상한대로 상당이 높은 수치를 기록했다 위에 수행 속도 비교는 off-line 에서 순수한 클러스터링의 시간을 비교한 결과이다 그러나 웹 문서는 네트웍을 통해 전송되고, SLINK 는 점진성을 가진 알고리즘으로서 문서를 받는 즉시 병렬적으로 클러스터링을 수행한다 따라서 네트웍을 통해 문서들이 전송되는 시간이 클러스터링을 위한 시간보다 매우 크다고 할 때, 실제 시스템의 수행속도는 검색 엔진에서의 처리 속도와 네트웍의 속도에 크게 의존한다 <그림 3>의 점선 부분은 빠른 인터넷 속도를 고려할 때 클러스터링에 필요한 시간과 네트워크를 통한 데이터의 전송 시간이 비슷해지는 구간이며, 사용자가 결과를 기대하는 최대한의 시간이다 다시 말해서, SLINK 를 이용한 클러스터링 시스템은 약 400 개 안팎의 문서 집합을 대상으로 했을 때 최고의 성능을 보여 줄 수 있다



<그림 4> 정확도 (Precison)

<그림 4>는 10 개의 클러스터를 기준으로 하였을 때, 웹 검색 엔진으로 받은 원래의 결과와 각 알고리즘이 나타낸 클러스터의 정확도를 비교한 그래프이다 SLINK 는 상당히 높은 정확도를 나타내었다 이것은 다른 알고리즘과는 달리 클러스터를 출력하는 과정에서 outlier 를 제거하기 때문에 큰 폭으로 상승된 정확도를 나타내었다

6 결론

성능비교에서 볼 수 있듯이 SLINK 는 상당히 느린 속도를 보여주었다. 그러나, SLINK 에서 주목할 점은 tree-diagram 으로 표현 가능한 dendrogram 의 제공과 이를 통한 높은 정확도, 그리고 점진성이다 점진성은 다른 알고리즘에 비하여 공간적인 성능을 크게 향상시킬 수 있으며, 네트웍을 통한 클러스터링 시스템에서 다른 클러스터링 시스템에 비하여 수행 속도에서 큰 차이를 보이지 않는다

SLINK 를 이용한 웹 문서 클러스터링 시스템은 앞에서 설명한 장점과 단점을 가지고 있으나, On-line 상에서 작은 규모의 문서 집합을 효과적으로 분류하는 최적의 시스템이다 인터넷 뉴스그룹 및 웹에서의 여러 문서들을 그 대상으로 확장할 수 있으며, 클러스터 구조를 저장하고 나타내는데 효과적이다.

참고문헌

[1] Rocchio J J Document retrieval systems – optimization and evaluation Ph.D Thesis, Havard University, 1966

[2] R Sibson. SLINK An optimally efficient algorithm for the single-link cluster method Computer journal.1973.

[3] Willet P "Similarity Coefficients and Weighting functions for Automatic Document Classification: an Empirical Comparision " International Classification, pages 10, 138-42. 1983.

[4] Lorr, M Cluster Analysis for social Scientists Techniques for Analyzing and Simplifying Complex Blocks of Data. SanFracisco,Jossey-Bass, 1983

[5] Willet P. Recent trends in hierarchical document clustering a critical review Information Processing and Management, pages 24, 577-97 1988.

[6] J. L. Fagan. Experiments in automatic phrase indexing for document retrieval a comparison of symtatic and non systactic methods Ph D Thesis, Cornel University, 1987.

[7] D. R. Cutting, D R Karger, J.O. Pedersen and J.W. Tukey. Scatter/Gather a cluster-based approach to browsing large document collections In Proceedings of the 15th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 318-29, 1992

[8] D. Gusfield. Algorithms on strings, trees and sequences . computer science and computational biology, chapter 6. Cambriged University Press, 1997

[9] O. Zamir, O Etzioni. Web Document Clustering A Feasibility Demonstration. In Proceedings of the 21th International ACM SIG IR Conference on Research and Development in Information Retrieval, http://www cs washington edu/research/clustering 1998

[10] A Sahuguet, F Azavant. WysiWyg Web Wrapper Factory(W4F) University of Pennsylvania, http://db cis upenn.edu/W4F 1998.

[11] W. B. Frakes, Richard Baeza-Yates. Information Retrieval, Data Structures & Algorithms, pages 419-42 Prentice Hall, 1995.