DOI: 10.3745/KIPSTD.2008.15-D.4.485

유사한 인기도 추세를 갖는 웹 객체들의 클러스터링

노 웅 기†

요 약

인터넷이 광범위하게 활용됨에 따라 검색 키워드, 멀티미디어 객체, 웹 페이지, 블로그 등의 다양한 웹 객체들이 크게 증가하고 있다. 이러한 웹 객체들의 인기도는 시간에 따라 변화하며, 그러한 웹 객체 인기도의 시간적 패턴에 대한 마이닝이 여러 가지 웹 응용에 필요한 중요한 연구 과제가 되고 있다. 예를 들어, 검색 키워드에 대한 인기도 패턴의 분석은 앞으로 인기가 높아질 키워드를 미리 예측할 수 있게 하여 광고주들에게 키워드를 판매하기 위한 가격을 결정하는 데에 중요한 자료가 될 수 있다. 하지만, 웹 객체 인기도가 시간에 따라 변화하고 웹 객체의 개수가 매우 방대하다는 특성으로 인하여 웹 객체 인기도에 대한 분석은 매우 어려운 문제이다. 본 논문에서는 웹 객체 인기도의 시간적 패턴을 마이닝하기 위한 효율적인 알고리즘을 제안한다. 본 논문은 웹 객체 인기도를 시계열로 표현하고, 두 웹 객체 인기도 간의 유사성을 측정하기 위하여 gap 척도를 제안한다. gap 척도의 효율적인 계산을 위하여 FFT를 활용한 알고리즘을 제안하고, 밀도기반 클러스터링 알고리즘을 이용하여 유사한 인기도 추세를 갖는 웹 객체들의 클러스터를 생성한다. 본 논문에서는 웹 객체 인기도가 특정 분포를 따르거나 주기적이라고 가정하지 않는다. Google Trends 웹 사이트로부터 구한 검색 키워드 인기도를 이용한 실험을 통하여, 제안된 알고리즘이 실세계 응용에서 유용함을 보인다.

키워드 : 웹 객체, 인기도 추세, 시간적 패턴, gap 척도, 밀도기반 클러스터링

Clustering of Web Objects with Similar Popularity Trends

Woong-Kee Loh†

ABSTRACT

Huge amounts of various web items such as keywords, images, and web pages are being made widely available on the Web. The popularities of such web items continuously change over time, and mining temporal patterns in popularities of web items is an important problem that is useful for several web applications. For example, the temporal patterns in popularities of search keywords help web search enterprises predict future popular keywords, enabling them to make price decisions when marketing search keywords to advertisers. However, presence of millions of web items makes it difficult to scale up previous techniques for this problem. This paper proposes an efficient method for mining temporal patterns in popularities of web items. We treat the popularities of web items as time-series, and propose gapmeasure to quantify the similarity between the popularities of two web items. To reduce the computation overhead for this measure, an efficient method using the Fast Fourier Transform (FFT) is presented. We assume that the popularities of web items are not necessarily following any probabilistic distribution or periodic. For finding clusters of web items with similar popularity trends, we propose to use a density-based clustering algorithm based on the gap measure. Our experiments using the popularity trends of search keywords obtained from the Google Trends web site illustrate the scalability and usefulness of the proposed approach in real-world applications.

Key Words : Web Items, Popularity Trends, Temporal Patterns, Gap Measure, Density-Based Clustering

1. 서 론

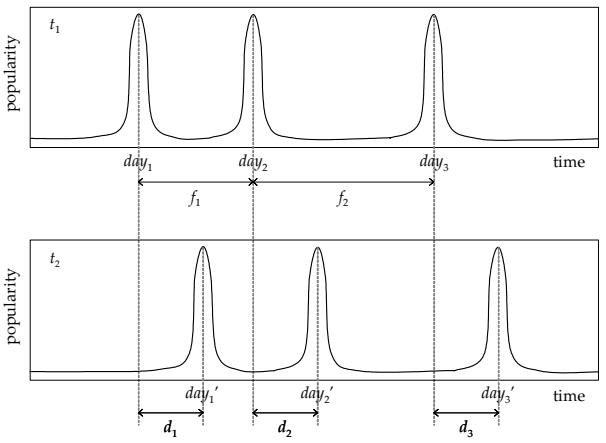
최근 인터넷이 광범위하게 활용됨에 따라 다양한 웹 객체들(web items)이 크게 증가하고 있다. 웹 객체들의 예로는 검색 키워드, 멀티미디어 객체, 웹 페이지, 블로그 등을 들 수 있다. 이러한 웹 객체들이 증가함에 따라 사용자와의 관련성(relevance) 및 인기도(popularity)를 측정하고 분석하기 위한 새로운 연구 주제들이 등장하였다. 웹 객체와 사용자와의 관련성은 웹 마이닝 분야에서 많이 연구되었고, 검색 엔진에서 그러한 관련성에 따라 웹 페이지 또는 다른 웹 객체들의 순위(ranking)를 결정하는 데에 응용되었다[12, 16, 17]. 웹 객체의 인기도는 사용자 액세스 횟수와 같은 사용자 요구량에 따라 결정된다. 이러한 웹 객체 인기도의 시간적 패턴에 대한 분석은 현재도 중요한 연구 과제의 하나이다. 유사한 인기도 패턴을 갖는 웹 객체들에 대한 분석은 인터넷 기업들을 비롯한 다양한 응용 분야에서 유용하다. 예를 들어, 검색 키워드 인기도의 시간적 패턴에 대하여 분석함으로써

※ 본 논문은 2006년 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행한 연구임.(KRF-2006-214-D00130).
† 정 회 원 : 성결대학교 멀티미디어학부 전임강사
논문접수 : 2007년 10월 30일
수 정 일 : 1차 2008년 5월 6일
심사완료 : 2008년 5월 6일

미래에 인기도가 높아질 키워드를 예측할 수 있다. 이는 검색 키워드의 가격을 결정하기 위한 중요 자료로 쓰일 수 있으며, 기업의 마케팅 정책에 영향을 줄 수 있다. 또한, 그러한 인기도 패턴 분석은 컴퓨팅 장비 및 디스크 저장 장치와 같은 기업 자원을 최적으로 운용할 수 있도록 활용할 수 있다.

본 논문에서는 웹 객체 인기도의 시간적 패턴에 대한 마이닝 문제를 다룬다. 이 문제를 공식적으로 정의하면, 대량의 웹 객체들의 시간적인 인기도 추세가 주어졌을 때, 유사한 인기도추세를 갖는 웹 객체들의 클러스터들을 찾는 문제이다. (그림 1)은 본 논문에서 다루고자 하는 문제를 보인 것이다. 그림에서, 두 웹 객체 t_1 및 t_2의 인기도 추세를 보이고 있다. 객체 t_1은 day_1, day_2, day_3에 매우 인기가 높고, t_2는 day_1', day_2', day_3'에 인기가 높다. 두 객체는 서로 다른 날에 인기가 높지만, t_1이 인기가 높아지고 임의 시간 후에 t_2가 인기가 높아짐을 알 수 있다. 따라서, 두 객체들의 인기도 추세 간에는 유사성이 존재한다고 판단하며, 본 논문에서 제안하는 알고리즘은 두 객체로 구성된 클러스터를 반환한다. 참고 문헌 [27]에서도 유사한 문제를 다루었으나, 오직 두 개의 데이터 스트림(data stream)에 대해서만 다룬 데에 반해, 본 논문에서는 대량의 웹 객체를 다룬다. 본 논문에서는 웹 객체의 인기도가 반드시 주기적(periodic)이지 않아도 된다고 가정한다. 즉, (그림 1)에서 $f_1 = f_2 = \cdots$ 라고 가정하지 않는다. 인기 지점(spike)의 개수에 대한 제한도 두지 않는다. 또한, 대응되는 인기 지점 간의 시간 차(lag)가 가변적이라고 가정한다. 즉, (그림 1)에서 $d_1 = d_2 = \cdots$라고 가정하지 않는다. 이러한 일련의 가정은 본 논문에서 다루고자 문제를 충분히 일반화하여 실제 세계의 상황을 좀더 가깝게 반영하기 위함이다. 현재까지 동일한 가정에 기반한 기존의 연구는 존재하지 않는다. 본 논문과 관련된 기존의 연구들에 대해서는 제 2 절에서 좀더 설명한다.

본 논문에서 다루고자 하는 문제는 다음과 같은 세가지이유로 해결이 쉽지 않다. (1) 웹 객체의 인기도는 0 이상의 임의의 정수 값을 가지나, 그 최대값은 웹 객체에 따라 편차가 매우 크고 예측하기 어렵다. (2) 웹 객체의 인기도는 시간에 따라 변화한다. (3) 웹 객체의 개수가 매우 많다 (수억 개 이상). 본 논문에서는, 문제를 해결하기 위하여 웹 객체의 인기도를 시계열(time-series)로 표현하고, 두 웹 객체 간의 인기도의 유사성을 판정하기 위하여 gap이라는 새로운 척도를 제안한다. gap 척도에 기반하여 밀도기반(density-based) 클러스터링 알고리즘의 하나인 DBSCAN 알고리즘 [8]을 이용하여 유사한 인기도 추세를 갖는 웹 객체들의 클러스터들을 생성한다. 밀도기반 클러스터링 알고리즘의 장점은 (1) 클러스터링을 위한 입력으로 클러스터의 개수 k를 필요로 하지 않고, (2) 타원형이 아닌 임의 모양의 클러스터를 생성하며, (3) 잡음(noise *or* outlier)을 잘 처리한다는 점이다. 본 논문의 제 5 절에서 Google Trends 웹 사이트로부터 얻어진 웹 검색 키워드의 인기도 추세를 이용한 실험을 통하여, 제안된 gap 척도를 이용한 클러스터링이 실제 응용에서 웹 객체의 인기도 패턴을 찾아내는 데에 유용함을 보인다.



(그림 1) 문제 설명: 유사한 인기도 추세를 갖는 두 웹 객체들.

본 논문은 다음과 같이 구성된다. 제 2 절에서는 기존의 관련 연구들을 개략적으로 기술한다. 제 3 절에서는 두 웹 객체들 간의 유사성을 판정하기 위한 gap 척도를 정의한다. 제 4 절에서는 gap 척도의 효율적인 계산을 위한 알고리즘과 유사한 웹 객체들을 클러스터링하기 위한 방법을 제시한다. 제 5 절에서는 Google Trends 데이터를 이용하여 앞 절에서 제안된 알고리즘을 통해 얻어진 실험 결과를 분석한다. 제 6 절에서는 본 논문의 공헌을 요약하고 앞으로의 연구 방향을 간략히 논의한다.

2. 관련 연구

시계열에 대한 유사성 검색을 위하여 많은 알고리즘들이 제안되었다. 기존의 알고리즘에서 시계열 간의 유사성 척도로 가장 많이 사용된 것은 유클리드 거리(Euclidean distance)이다[1, 9, 21, 22]. 그 이유는 유클리드 거리가 효율적인 인덱싱을 위하여 필요한 삼각 부등식(triangular inequality) 등의 주요한 특성들을 만족하기 때문이다. 시계열 간의 거리 척도로 널리 사용되는 다른 유사성 척도로 Dynamic Time Warping (DTW) 척도가 있다[13, 14, 15, 30]. DTW 척도는 유클리드 척도에 비하여 두 시계열 간의 비교 대상 점을 좀더 유연하게 찾아내는 거리 척도이다. DTW 척도는 삼각 부등식을 만족하지 않지만, 최근까지 효율적인 인덱싱을 위한 많은 연구가 진행되었다[13, 14, 26]. 또다른 유사성 척도로는 Lp-norm [31], Longest Common SubSequence(LCSS) [28] 등이 있다. 시계열 클러스터링 알고리즘 [18, 19]은 길이 n의 하나의 시계열을 Harr Wavelet 변환을 통하여 f-차원 ($f << n$) 객체로 변환하고 k-means 알고리즘 [20]을 통하여 클러스터링을 수행한다. 본 논문에서는 웹 객체의 인기도를 시계열로 표현하지만 기존의 시계열 매칭/클러스터링 알고리즘과는 다른 유사성 척도를 사용한다. 새로운 유사성 척도에 대해서는 제 3 절에서 자세하게 설명한다.

본 논문에서 다루는 인기도 추세 분석과 관련된 문제를 다룬 기존의 연구는 다음과 같다[5, 6, 27, 29]. 참고문헌 [5]에서는 검색 엔진에 주어지는 두 개의 검색 질의(search query)

의 인기도가 시간에 따라 유사하게 변한다면 두 질의를 의미적으로 유사하다고 정의하였다. 이러한 관점에 기반하여, 두 질의 간의 유사성 척도를 정의하고, MSN 검색 질의를 이용한 실험을 통하여 그 척도가 유사한 인기도의 질의를 검색하는 데에 유용함을 보였다. 하지만, 참고문헌 [5]에서는 검색 질의 유사도 간의 시간 차이를 고려하지 않았다. 즉, (그림 1)에서 항상 $d_1 = d_2 = \cdots = 0$임을 가정하였다. 참고문헌 [29]에서도 참고문헌 [5]와 같이 유사한 인기도의 검색 질의를 찾는 문제를 다루었다. 하지만, 주기적인 인기도 패턴에 대해서만 다루었다. 즉, (그림 1)에서 $f_1 = f_2 = \cdots$임을 가정하였다. 참고문헌 [29]의 알고리즘은 그러한 주기성에 기반하여 가장 효율적인 DFT 계수를 추출하기 때문에, 주기적이지 않은 데이터에 대해서는 효율성이 떨어질 가능성이 매우 높다. 참고문헌 [6]에서는 시계열 내의 주기성(periodicity)을 찾아내고 주기율(rate of periodicity *or* period)을 계산하기 위한 알고리즘을 제안하였다. 하지만, 이 알고리즘은 유사성 검색에 대해서는 다루지 않았다. 참고문헌 [27]에서는 본 논문에서와 같이 (그림 1)에서 $f_1 \neq f_2 \neq \cdots$ 그리고 $d_1 \neq d_2 \neq \cdots$임을 가정하였으나, 두 개의 데이터 스트림(data stream)에 대해서만 문제를 다루었다. 따라서, 이 알고리즘은 다량의 데이터를 다룰 때에 확장성(scalability) 문제가 발생할 수 있다.

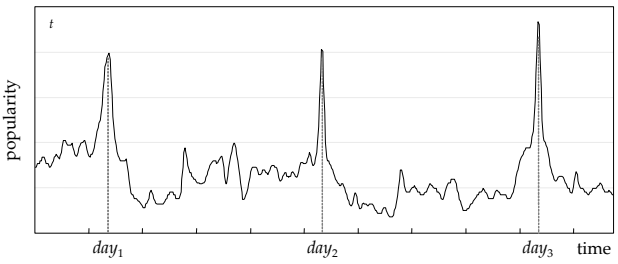
3. 웹 객체들 간의 유사성 척도

본 절에서는 두 웹 객체들 간의 새로운 유사성 척도인 gap 척도를 정의한다. *인기 객체*(*popular item*)란 단위 시간(time granule) 내에 다른 객체들에 비해 좀더 빈번하게 액세스되거나 사용된 웹 객체를 의미한다. 단위 시간은 사용자가 임의의 시간으로 정할 수 있으며, 한 시간, 하루, 또는 일주일이 될 수도 있다. 웹 객체의 인기도는 단위 시간 내에 임의의 사용자가 액세스하거나 사용한 횟수의 합이며, 웹 객체 인기도에 따라 인기 객체를 찾는다. 인기 객체를 공식적으로 정의하면, 미리 주어진 두 개의 한계 값(threshold) τ_1, τ_2에 대해 다음의 두 조건을 만족하는 웹 객체로 정의한다:

(1) 최대 인기도가 τ_1을 초과한다.
(2) 전체 단위 시간 동안의 인기도의 표준편차(standard deviation)가 τ_2를 초과한다.

조건 (2)의 이유는 인기도가 조건 (1)에서 주어진 τ_1을 항상 초과하는 객체들을 제거하기 위함이다. 그러한 객체들의 예로 a, and, it, of와 같이 사용자 검색 질의에 포함된 불용어(stop word) [3]를 들 수 있다. 본 논문에서는 객체의 인기도가 정규 분포(normal distribution)와 같은 특정 분포를 따른다는 가정을 하지 않는다.

본 논문에서는 하나의 웹 객체 인기도 추세를 시계열로 표현한다. 시계열의 길이 n은 인기도가 구해진 전체 기간을 나타낸다. 예를 들어, 단위 시간이 하루이고 객체 인기도를 1년간 구했다면 n = 365이다. 인기도 추세는 n-차원 벡터로



(그림 2) 객체 벡터의 예

도 표현된다. 즉, 웹 객체 t의 인기도 추세는 $\boldsymbol{t} = (t_0, \cdots, t_{n-1})$으로 나타낸다. 본 논문에서는 인기도 추세를 나타내는 시계열 또는 벡터를 *객체 벡터*(*item vector*)라 부른다. 또한, 의미가 분명하게 전달되는 부분에서는 '객체 벡터' 대신 '인기 객체' 또는 간단히 '객체'라는 용어를 사용한다. 그림 2는 Google Trends 사이트에서 얻어진 객체 벡터의 한 예를 보인 것이다. 가로 축은 단위 시간을 나타내고, 세로 축은 인기도를 나타낸다. 그림에서, 객체 t는 day_1, day_2, day_3에 인기가 높다.

본 절에서는 gap 척도를 정의하기 전에, 두 객체들 간의 유사성 척도 *sim*과 *dissim*을 먼저 정의한다. gap 척도는 이 두 가지 척도에 기반하여 정의한다. *sim* 척도는 (거의) 같은 시간에 인기가 높아지는 객체들을 찾기 위한 것이며 다음과 같이 정의한다:

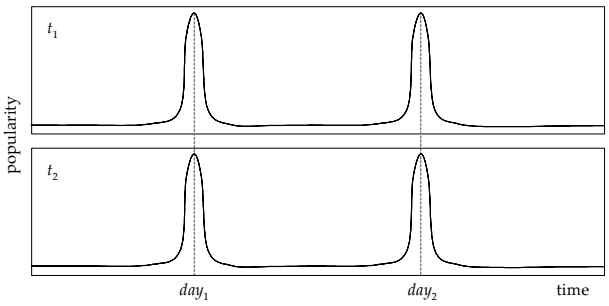
정의 1. 동일한 길이(차원) n의 두 객체 t_1과 t_2 간의 유사도(similarity)는 다음과 같이 정의한다:

$$sim(t_1, t_2) = \frac{< \boldsymbol{t}_1, \boldsymbol{t}_2 >}{\|\boldsymbol{t}_1\| \cdot \|\boldsymbol{t}_2\|} \quad (1)$$

여기에서, $\boldsymbol{t}_1$과 $\boldsymbol{t}_2$는 각각 객체 t_1과 t_2에 대한 객체 벡터이며, $<\boldsymbol{t}_1, \boldsymbol{t}_2>$는 두 객체 벡터 간의 내적(inner product)이고, $\|\boldsymbol{t}_1\|$과 $\|\boldsymbol{t}_2\|$ 각각 t_1과 t_2의 크기(norm)이다. □

정의 1과 같이 객체간 유사도를 정의한 이유는 다음과 같이 두 가지이다. (1) 정의 1은 벡터 크기에 무관하게 유사성 비교가 가능하며, 정보 검색 [3]과 같은 여러 분야에서 채택되었다. (2) 기존의 연구들에서 이 정의가 의미적으로 유사한 시계열을 찾는 데에 유용하다는 것을 보였다[5, 27]. 그림 3은 유사한 객체들의 예를 보인 것이다. 그림에서, 객체 t_1과 t_2는 같은 시간에 인기가 높아지므로 *sim* 척도에 의하여 유사하다고 판정된다.

기존에 시계열 간의 유사성 정의를 위하여 유클리드 거리와 DTW 거리가 가장 많이 사용되었으나, 그들은 다음과 같은 단점이 있다. 유클리드 거리는 시계열 내의 변동 추세(fluctuation trend)를 잡아내기 어렵다. 두 시계열이 유사한 변동 추세를 갖고 있더라도, 만약 그들간의 평균의 차가 크다면, 그들 간의 유클리드 거리가 커지므로 유사하다고 판정되지 않는다. 본 논문에서는 같은 시간에 인기가 높아지는 객체들을 찾고자 하는 것이며, 그들이 각각 얼마나 인기가



(그림 3) 유사한 객체들의 예

있는지는 중요하지 않다. 즉, 유사성 척도는 객체들의 변동 추세를 잡아내야 하고 절대적인 인기도 값은 무시하여야 한다. DTW 거리도 유클리드 거리와 유사한 문제를 갖고 있다.

정의 1의 객체간 유사도 정의에 따라, 두 객체 t_1, t_2 간의 sim 척도는 두 객체 벡터 $\boldsymbol{t}_1$, $\boldsymbol{t}_2$ 사이의 각도 θ ($0 \le \theta \le \pi$)의 코사인(cosine) 값과 같다. 따라서, sim 척도는 단조 감소(monotonic decreasing) 함수이며 [-1, 1] 영역 내의 값을 반환한다. sim 척도에 기반하여, 두 객체 t_1과 t_2 간의 비유사도(dissimilarity) 척도는 다음과 같이 두 객체 벡터 간의 각도 θ로 정의한다.

정의 2. 동일한 길이(차원) n의 두 객체 t_1과 t_2 간의 비유사도는 다음과 같이 정의한다:

$$dissim(t_1, t_2) = \arccos sim(t_1, t_2) \quad (2)$$

여기에서, arccos은 코사인의 역함수(inverse function)이다. □

$dissim$ 척도는 sim 척도와 같이 단조 감소 함수이며 [0, π] 영역 내의 값을 반환한다.

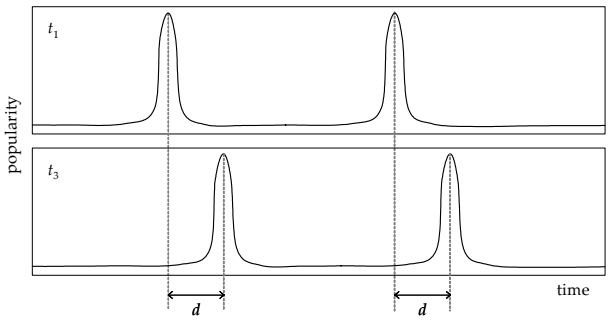
(그림 4)에서 객체 t_3는 객체 t_1과 유사하다고 판정되지 않는다. 즉, 객체 t_1과 t_3는 같은 시간에 인기가 높아지지 않으므로 그들간의 sim 척도는 매우 작은 유사도 값을 반환한다. 하지만, t_3는 t_1이 인기가 높아지고 (거의) 정확하게 d 단위 시간 후에 인기가 높아지므로, 두 객체 t_1, t_3 간에는 시간적인 패턴이 존재한다. 그림에서 단위 시간의 차이 d를 시간 차라고 부른다. 일반적으로, 두 객체 t_1과 t_3 간의 그러한 패턴을 찾기 위해서는, 시간 차 d를 미리 알 수 없으므로, 모든 가능한 (음, 양의) 시간 차에 대하여 조사해 보아야 한다. 즉, 객체 t_3의 모든 가능한 회전(rotation)에 대하여 객체 t_1 간의 유사도를 계산해 보아야 한다. 이러한 관점에 기반하여 두 객체 간의 gap 척도를 다음과 같이 정의한다.

정의 3. 임의의 두 객체 t_1과 t_2에 대하여 그들 간의 gap은 다음과 같이 정의한다:

$$gap(t_1, t_2) = \min \{ dissim(\boldsymbol{t}_1 + d, \boldsymbol{t}_2), 0 \le d < n \} \quad (3)$$

여기에서, $\boldsymbol{t}_1$과 $\boldsymbol{t}_2$는 객체 t_1과 t_2의 객체 벡터이며, $\boldsymbol{t}_1 + d$는 $\boldsymbol{t}_1$을 d만큼 회전한 벡터이다. n은 $\boldsymbol{t}_1$, $\boldsymbol{t}_2$의 길이(차원)이다. □

벡터 $\boldsymbol{t} + d$는 벡터 $\boldsymbol{t} = (t_0, t_1, \cdots, t_{n-1})$을 d만큼 회전한 벡터이므로, $\boldsymbol{t} + d = (t_d, t_{d+1}, \cdots, t_{n-1}, t_0, \cdots, t_{d-1})$이다. $dissim(t_1, t_2)$가 영역 [0, π] 내의 값을 가지므로, $gap(t_1, t_2)$도 동일한 영역 내의 값을 가진다.

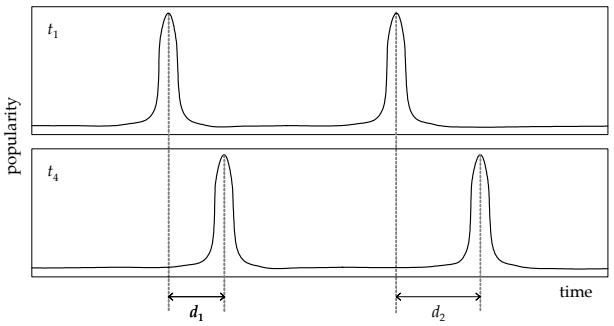


(그림 4) 일정한 시간 차 d를 갖는 유사 벡터들의 예

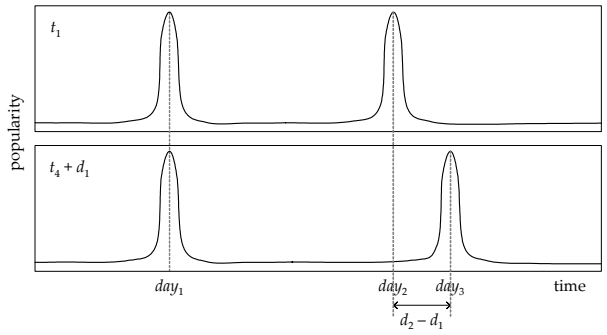
정의 3에 주어진 대로 두 객체 t_1과 t_2 간의 $gap(t_1, t_2)$를 계산하기 위하여, 단순한(naïve) 알고리즘은 t_1의 모든 가능한 회전에 대하여 $dissim$ 척도 값을 계산하고 그 중 최소값을 구할 것이다. 비유사도 값을 구하기 위하여 두 객체 벡터 간의 내적을 구해야 하므로, 단순한 알고리즘의 시간 복잡도는 $O(n^2)$이다. 단순한 알고리즘으로 m 개의 객체들 간에 gap 척도를 계산한다면 그 시간 복잡도는 $O(n^2m^2)$이 되고, 만약 m이 매우 크다면, 전체 gap을 계산하기 위한 비용도 매우 클 것이다. 유클리드 거리나 DTW 거리를 두 객체 간의 비유사도 척도로 사용할 때, 두 객체 간의 gap을 구하는 시간 복잡도를 줄이기 위한 방법은 존재하지 않는다. 제 4.1 절에서 sim과 $dissim$ 척도에 기반한 gap 척도를 효율적으로 계산하기 위한 알고리즘에 대해 설명한다.

(그림 5)에서 두 객체 t_1과 t_4 사이의 시간 차 d_1과 d_2가 서로 달라 두 객체 간의 gap 값이 커지므로, t_4는 t_1과 유사하다고 판정되지 않을 수 있다. 하지만, 시간 차가 다르더라도 만약 t_1이 인기가 높아진 얼마 후에 항상 t_4가 인기가 높아진다면 두 객체 t_1과 t_4 사이에는 인기도의 시간적인 유사성이 존재한다고 보아야 한다. 동시에, 두 시간 차 간의 차이 $d_L = |d_1 - d_2|$가 너무 큰 경우에 대해서는 별로 관심이 없을 수 있다. 즉, d_L이 작거나 거의 0에 가까운 경우를 선호하며, d_L이 작아질수록 두 객체 간의 gap도 작아지길 (유사성이 증가하길) 원한다.

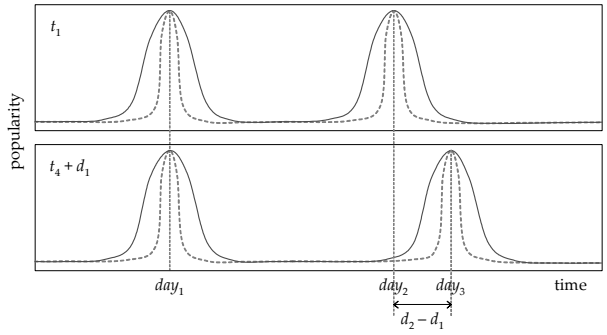
이러한 조건들을 만족시키기 위하여 객체 벡터의 블러링(blurring)을 이용한다. 각 객체 벡터를 먼저 블러링한 후에 그들 간의 gap을 구한다. 객체 벡터 $\boldsymbol{t}$를 블러링함에 따라 객체 t가 인기가 높아지는 시간 기간이 확장된다. (그림 6)은 객체 벡터의 블러링 전(a)과 후(b)의 예를 보인 것이다. (그림 6(b))에서 점선 그래프는 블러링 전을 나타내고, 실선 그래프는 블러링 후를 나타낸다. (그림 6(a))에서 (그림 5)의 객체 벡터 $\boldsymbol{t}_4$를 d_1만큼 회전하여 $\boldsymbol{t}_1$과 $\boldsymbol{t}_4 + d_1$이 모두 day_1에 인기가 높아지도록 하였다. 하지만, $\boldsymbol{t}_1$은 day_2에 다시 인기가 높아지고, $\boldsymbol{t}_4 + d_1$은 day_3에 인기가 높아지므로, day_1 외에 두 객체가 공통적으로 인기가 높아지는 기간이 존재하지 않는다. 그런 이유로 두 객체 간의 gap이 커지게 된다. 하지만, (그림 6(b))에서 두 객체 벡터가 인기가 높아지는 기간이 확



(그림 5) 가변 시간 차를 갖는 유사 벡터들의 예



(a) 블러링 전



(b) 블러링 후

(그림 6) 블러링 전과 후의 객체 벡터들

장됨에 따라 day_2와 day_3에서도 두 객체 벡터 $\boldsymbol{t}_1$과 $\boldsymbol{t}_4$가 공통적으로 인기가 높아지는 기간이 발생하여 두 객체 간의 유사성도 증가하게 된다. 블러링의 수준(degree)은 응용에 따라 조정할 수 있다.

블러링 기법은 디지털 이미지 처리(digital image processing) 분야에서 많은 이용되고 있다[10]. 이미지의 블러링에서는 각 픽셀 값이 그 주변 픽셀 값들의 가중(weighted) 평균 값으로 대치된다. 가우스 블러링(Gaussian blurring)은 객체 벡터의 블러링을 위한 좋은 기법 중의 하나이다. 객체 벡터 내의 주변 요소 값들에 대한 가중치(weight)가 가우스(정규) 분포를 다르며, 멀리 있는 주변 요소일수록 가중치도 작아지기 때문이다. 1-차원 가우스 블러링에서의 가중치는 다음과 같이 정의된다:

$$G(r) = \frac{1}{\sqrt{2\pi\sigma}} e^{-r^2/(2\sigma^2)} \tag{4}$$

여기에서, r은 원래 요소와 주변 요소 간의 거리이고, σ는 표준 편차, e는 자연 로그의 밑(base)이다. 본 논문에서는 r 값이 커짐에 따라 $G(r)$ 값이 급격히 감소하므로 r 값의 범위에 제한을 둔다.

필요한 경우에, 정의 3에서 두 객체 간의 시간 차 d의 범위를 설정할 수 있다. 객체 t_1이 인기가 높아진 후에 항상 객체 t_2가 인기가 높아진다고 하더라도, 만약 시간 차가 몇 달 내지 1년 정도로 너무 크다면 그러한 객체들 간에 존재하는 유사성에 대해서는 관심이 없을 수도 있다. 다른 유의할 점은 gap 척도를 벡터의 회전을 이용하여 정의함으로써 유사한 객체 벡터들 간의 시간적인 순서를 잃어버릴 가능성도 있다는 점이다. 이러한 문제를 해결하기 위하여 앞에서와 유사하게 시간 차 d의 절대값을 특정 w 이하로 한정할 수도 있다. 그렇다면, $-w \le d \le w$를 만족하는 시간 차 d에 대해서만 공식 (3)을 이용하여 두 객체 간의 gap을 구한다.

4. 효율적인 gap 척도 계산 및 클러스터링

앞 절에서 정의한 두 웹 객체 간의 인기도 유사성 척도인 gap 척도를 계산하기 위하여 단순한 알고리즘은 $O(n^2)$의 복잡도를 갖는다. 여기에서, n은 객체의 길이(차원)이며, 만약 오랜 기간 동안의 인기도를 비교한다면 n의 값이 매우 커질 수 있다. 동시에, 만약 인기도 유사성을 비교하기 위한 객체들의 개수 m이 커진다면, 유사한 인기도를 갖는 객체들을 찾기 위하여 gap 척도를 계산하는 비용 부담이 매우 커진다. 제 4.1 절에서는 효율적인 gap 척도 계산 알고리즘을 제시하고, 제 4.2 절에서는 유사한 인기도를 갖는 웹 객체들을 클러스터링하기 위한 방법을 설명한다.

4.1. 효율적인 gap 척도 계산

정의 2에 주어진 비유사도 척도를 사용하면, 객체 벡터에 대한 이산 푸리에 변환(Discrete Fourier Transform, DFT) [24, 25]을 이용하여 단순한 알고리즘에 비해 훨씬 효율적으로 $gap(t_1, t_2)$를 구할 수 있다. 객체 벡터 $\boldsymbol{t}$를 DFT 변환한 벡터를 $\boldsymbol{T}$라고 하자. 벡터 $\boldsymbol{t}$는 $\boldsymbol{T}$의 역 DFT 변환한 벡터이다. 본 논문에서는 두 벡터 $\boldsymbol{t}$와 $\boldsymbol{T}$를 *DFT 쌍*(*DFT pair*)이라고 부르며 $\boldsymbol{t} \Leftrightarrow \boldsymbol{T}$로 표기한다. 두 객체 벡터 $\boldsymbol{t}_1$과 $\boldsymbol{t}_2$ 간의 *순환 상관계수*(*circular correlation*) $CC(d)$를 다음 공식과 같이 정의한다:

$$CC(d) = \langle \boldsymbol{t}_1 + d, \boldsymbol{t}_2 \rangle \tag{5}$$

여기에서, $\langle \boldsymbol{t}_1 + d, \boldsymbol{t}_2 \rangle$는 $\boldsymbol{t}_1 + d$와 $\boldsymbol{t}_2$간의 내적을 의미한다. $CC(d)$는 시간 차 d의 함수이며, 상수 값을 반환한다. 여기에서, $\| \boldsymbol{t}_1 + d \| = \| \boldsymbol{t}_1 \|$ 이므로, $CC(d)$는 다음과 같이 다시 쓸 수 있다:

$$CC(d) = sim(\boldsymbol{t}_1 + d, \boldsymbol{t}_2) \cdot \| t_1 + d \| \| t_2 \| = sim(\boldsymbol{t}_1 + d, \boldsymbol{t}_2) \cdot \| \boldsymbol{t}_1 \| \| \boldsymbol{t}_2 \| \tag{6}$$

순환 상관계수 벡터(*circular correlation vector*) *CCV*는 각각의 시간 차 d ($0 \le d < n$)에 대한 n개의 $CC(d)$ 값으로 구성된 벡터로 정의한다. 즉, $CCV = (CC(0), CC(1), \cdots, CC(n-1))$이다. 벡터 *CCV*는 다음의 정리에 의하여 효율적으로 계산할 수 있다[24, 25].

정리 1. 두 객체 벡터 $\boldsymbol{t}_1$과 $\boldsymbol{t}_2$로 얻어진 순환 상관계수 벡터 *CCV*에 대해 다음의 공식이 성립한다:

$$CCV \Leftrightarrow \boldsymbol{T}_1 \bullet \boldsymbol{T}_2^* \tag{7}$$

여기에서, $\boldsymbol{T}_1$과 $\boldsymbol{T}_2$는 각각 $\boldsymbol{t}_1$과 $\boldsymbol{t}_2$의 DFT 변환이며, $\boldsymbol{T}_2^*$는 $\boldsymbol{T}_2$의 complex conjugate이다. □

정리 1과 *CCV* 벡터를 이용하여 두 객체 t_1과 t_2 간의 gap을 효율적으로 계산하기 위한 알고리즘은 다음과 같다. 먼저, 제안된 알고리즘은 객체 벡터 $\boldsymbol{t}_1$과 $\boldsymbol{t}_2$를 DFT 변환한 벡터 $\boldsymbol{T}_1$과 $\boldsymbol{T}_2$를 구한다. 이를 위하여 고속 푸리에변환(Fast Fourier Transform, FFT) 알고리즘이 사용되며, 시간 복잡도는 $O(n\log n)$이다. 다음에, 제안된 알고리즘은 $\boldsymbol{T}_1 \bullet \boldsymbol{T}_2^*$를 구하며, 이때 시간 복잡도는 $O(n)$이다. 벡터 $\boldsymbol{T}_1 \bullet \boldsymbol{T}_2^*$의 i-번째 ($0 \le i < n$) 요소는 $\boldsymbol{T}_1$과 $\boldsymbol{T}_2^*$ 내의 i-번째 요소끼리 곱하여 구하며, $\boldsymbol{T}_1$, $\boldsymbol{T}_2^*$, $\boldsymbol{T}_1 \bullet \boldsymbol{T}_2^*$ 내의 모든 요소 값들은 복소수(complex number)이다. 그리고 나서, 제안된 알고리즘은 $\boldsymbol{T}_1 \bullet \boldsymbol{T}_2^*$의 역 DFT 변환을 수행하여 *CCV* 벡터를 얻는다. 역 DFT 변환은 FFT 알고리즘을 통하여 수행할 수 있으며, 마찬가지로 시간 복잡도는 $O(n\log n)$이다. 공식 (6)에 보인 것과 같이, *CCV* 벡터 내의 하나의 요소 $CC(d)$ ($0 \le d < n$)는 $sim(\boldsymbol{t}_1 + d, \boldsymbol{t}_2)$와 $\|\boldsymbol{t}_1\| \|\boldsymbol{t}_2\|$를 곱한 값과 동일하다. 따라서, *CCV* 내의 각 요소에 대하여 $1/\|\boldsymbol{t}_1\| \|\boldsymbol{t}_2\|$를 곱하고 arccos을 계산하는 대신, 제안된 알고리즘은 다음 공식을 이용하여 t_1과 t_2간의 gap을 구한다:

$$gap(t_1, t_2) = \arccos\left(\frac{\max\{CC(d), 0 \le d < n\}}{\|\boldsymbol{t}_1\| \|\boldsymbol{t}_2\|} \right) \tag{8}$$

공식 (8)에서 최대 $CC(d)$를 찾는 이유는 함수 arccos이 단조 감소 함수이기 때문이다. 전체적으로, 제안된 알고리즘의 시간 복잡도는 $O(n\log n)$이다. (그림 7)은 객체 t_1, t_2 간의 gap을 구하는 과정을 요약한 것이다.

(1) 주어진 객체 벡터 $\boldsymbol{t}_1$과 $\boldsymbol{t}_2$를 DFT 변환한 벡터 $\boldsymbol{T}_1$과 $\boldsymbol{T}_2$를 구한다.
(2) $\boldsymbol{T}_1 \bullet \boldsymbol{T}_2^*$를 구한다.
(3) $\boldsymbol{T}_1 \bullet \boldsymbol{T}_2^*$에 대해 역 DFT 변환을 수행하여 *CCV* 벡터를 구한다.
(4) 최대 $CC(d)$ ($0 \le d < n$)를 찾는다.
(5) 최대 $CC(d)$와 공식 (8)을 이용하여 $gap(t_1, t_2)$를 구한다.

(그림 7) 두 객체 간의 gap을 계산하는 알고리즘.

FFT 알고리즘은 길이가 2의 정수 거듭제곱(integer power)인 시계열에 주로 사용된다. 하지만, 길이가 2의 정수 거듭제곱이 아닌 시계열에 대해서도 길이가 2의 정수 거듭제곱이 되도록 0을 첨부(padding)함으로써 FFT 알고리즘을 사용할 수 있다. 기본 DFT 알고리즘은 $O(n^2)$의 시간 복잡도를 갖는 반면, $O(n\log n)$의 시간 복잡도로 DFT 변환을 수행하는 FFT 알고리즘은 여러 가지가 있다[24, 25]. 그들 중 일부는 0을 첨부하지 않고도 길이가 2의 정수 거듭제곱이 아닌 시계열을 변환할 수 있는 것도 있다. 수년 동안의 인기 객체의 경우 n이 매우 클 수 있으므로, 제안된 알고리즘과 단순한 알고리즘 간에 gap 척도를 구하기 위한 성능의 차이는 매우 클 것이다. 제 5 절에서 gap 척도를 계산하기 위한 단순한 알고리즘과 제안된 알고리즘의 성능 실험 결과를 설명한다.

4.2 인기 객체 클러스터링

많은 수의 객체들 중에서 유사한 인기도 추세를 갖는 객체들을 모으기위해서는 gap 척도에 기반한 클러스터링을 수행하여야 한다. 클러스터링 문제는 전통적인 중요 데이터 마이닝 문제의 하나로, 현재까지도 수많은 알고리즘들이 제안되어 왔다[11]. 그리고, 대용량 객체에 대하여 클러스터링 속도를 향상시키고자 하는 연구도 진행되었다 [7, 23]. 그러한 알고리즘들은 각각 서로 다른 특성을 갖고 있으므로, 응용에 따라 적절한 클러스터링 알고리즘을 선택하여야 한다.

본 논문에서 유사한 인기도를 갖는 웹 객체들을 찾기 위하여 필요한 클러스터링 알고리즘의 특성들은 다음과 같다. (1) 유사한 인기도를 웹 객체 클러스터의 개수를 미리 알 수 없으므로, 그 값을 한정하여서는 안 된다. 클러스터의 개수 k를 입력으로 받아 k 개의 클러스터를 생성하는 기존의 클러스터링 알고리즘들 중의 대표적인 예로 k-means [20]를 들 수 있다. (2) 본 논문에서는 웹 객체들간의 유사성 척도만을 정의하고 있으므로, 클러스터링 대상 객체들의 (n-차원 공간 상의) 실제 위치에 기반한 클러스터링 알고리즘은 사용할 수 없다. 그러한 알고리즘의 예로는 BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)[32] 등을 들 수 있다. (3) 잡음을 잘 처리하여야 한다. 웹 객체들 중에는 다른 어떤 객체들과도 인기도 유사성이 다른 객체들이 많이 존재할 수 있다. 그러한 객체들은 잡음이라고 볼 수 있으며, 어떤 클러스터에도 포함되지 않을 수 있다.

본 논문에서는 밀도기반 클러스터링 알고리즘의 하나인 DBSCAN 알고리즘 [8]을 사용한다. 일반적으로, 밀도기반 클러스터링 알고리즘의 장점은 (1) 클러스터링을 위한 입력으로 클러스터의 개수 k를 필요로 하지 않고, (2) 타원 모양이 아닌 임의 모양의 클러스터를 생성하며, (3) 잡음을 잘 처리한다는 점이다. DBSCAN 알고리즘은 객체의 실제 위치를 이용하거나 또는 객체 간의 유사성만을 이용하는 두 가지 경우 모두를 잘 처리하는 알고리즘이다. 본 논문에서는 클러스터링 알고리즘으로 또다른 밀도기반 알고리즘인 OPTICS[2]도 고려해 보았다. 하지만, OPTICS와 DBSCAN의 구조적인 동일성으로 인하여, OPTICS는 DBSCAN과 거의 같은 시간 복잡도를 가진다[11]. 따라서, OPTICS를 이용함에 의한 장

점이 크지 않을 것으로 판단하였다.

5. 실 험

본 절에서는 제안된 gap 척도 계산 알고리즘의 성능을 평가한다. 또한, gap 척도를 이용한 클러스터링이 실제 응용에서 웹 객체의 인기도 패턴을 찾아내는 데에 유용함을 보인다. 제 5.1 절에서는 실험 데이터 및 설정에 대하여 설명하고, 제 5.2 절과 5.3 절에서는 성능 분석 및 클러스터 분석 결과를 기술한다.

5.1 실험 환경

본 실험에서는 Google Trends 웹 사이트(http://trends.google.com/)에서 얻어진 실세계 인기도 데이터를 이용한다. Google Trends 사이트에서는 특정 키워드가 Google 검색 엔진 상에서 얼마나 빈번하게 검색되었는지 정보를 그래프 형태로 제공한다. 본 논문에서는 먼저 Google Zeitgeist Archive 사이트(http://www.google.com/press/zeitgeist.html)에서 최근의 모든 인기 있는 검색 키워드들을 구하였다. 각 키워드는 하나 이상의 단어로 구성되어 있으며, 그러한 약 8,000개의 키워드로부터 2,560개의 단일 단어들을 추출하였다. 각 단어에 대하여 Google Trends 사이트로부터 인기도 추세 이미지 파일(.png 형식)을 구하였다. 이미지 파일은 580 × 260의 일정한크기를 가지며, 2004년 1월부터의 인기도 추세 그래프를 포함한다. 이 그래프를 이미지 분석하여 580개의 수치 인기도 값을 얻어냈다. 이때, 인기도 값이 580개에 미치지 않는 이미지는 버렸다. 다음에, FFT 변환을 쉽게 수행하기 위하여, 보간법(interpolation)[24]을 이용하여 인기도 값의 개수를 1,024로 확장하였다.

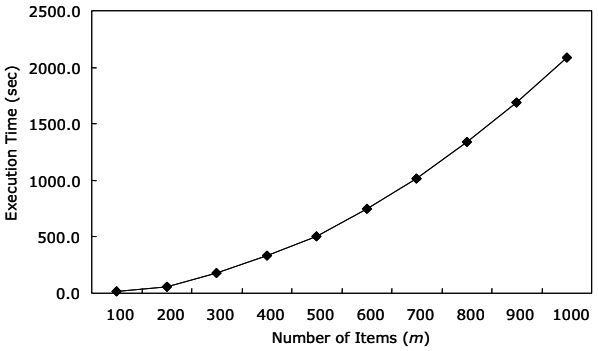
본 실험에서 사용된 하드웨어 플랫폼은 펜티엄 4 2.0GHz CPU, 1GB RAM, 80GB HDD를 장착한 PC이며, 소프트웨어 플랫폼은 Microsoft Windows XP Service Pack 2를 사용하였다. 제 3 절에서 설명한 τ_2 값을 변경하면서 m = 100, 200, …, 1000 개의 단어를 포함하도록 열 개의 데이터 셋을 생성하였다. 클러스터링을 위하여 DBSCAN 알고리즘[8]을 직접 구현하였다. DBSCAN 알고리즘의 입력 파라메터로 gap 한계 값은 ε = 0.1로, 클러스터의 최소 크기는 minpts = 2로 설정하였다.

5.2 성능 평가 결과

본 절에서는 두 가지 실험을 수행한다. 먼저, 두 웹 객체들 간의 gap 척도를 구하기 위한 두 가지 방법의 실행 시간(wall-clock execution time)을 비교한다. 두 가지 방법은 (a) gap 척도의 정의에 따른 단순한 방법, (b) (그림 7)에 주어진 효율적인 방법이다. 아래의 <표 1>은 비교 결과를 요약한 것으로, 두번째, 세번째 컬럼이 각각 방법 (a)와 (b)의 실행 시간을 초 단위로 보인 것이다. <표 1>의 네번째 컬럼에서 보인 것과 같이, 제안된 방법 (b)는 단순한 방법 (a)에 비하여 계산 횟수에 관계 없이 최대 45.98배까지 향상된 성능을 보였다.

〈표 1〉 첫 번째 실험 결과 : gap 척도를 계산하기 위한 실행 시간 비교

계산 횟수	방법 (a) (단순한 방법)	방법 (b) (제안된 알고리즘)	비율
2,000	23.44	0.51	45.88
4,000	46.84	1.02	45.83
6,000	70.50	1.54	45.72
8,000	97.67	2.12	45.98
10,000	121.71	2.66	45.82



(그림 8) 두번째 실험 결과: 웹 객체 개수(m)에 따른 클러스터링 성능 변화.

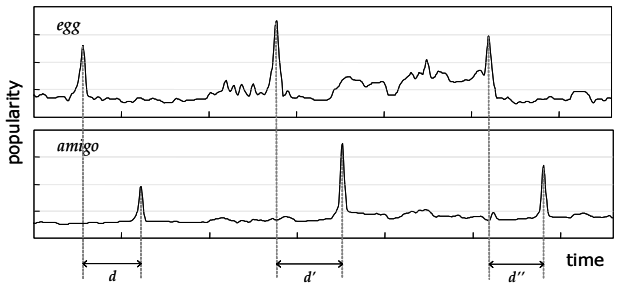
다음에, 클러스터링 성능을 비교하기 위하여, 웹 객체의 개수 m을 변경하면서 성능의 변동 추세를비교한다. 여기에서, ε = 0.1로 고정하였고, 실험 결과를 (그림 8)에 보였다. 그림에서 가로 축은 웹 객체의 개수 m, 세로 축은 실행 시간(초 단위)을 나타낸다. 그림에서 보는 바와 같이, 파라메터 m 값이 증가함에 따라 더 많은 실행 시간을 필요로 한다. DBSCAN 알고리즘의 시간 복잡도는 m의 함수이므로, 이러한 결과는 예측 가능한 것이다. 웹 객체 개수가 증가함에 따라 클러스터링 시간도 급격히 증가하므로, gap 척도를 효율적으로 빠르게 계산하는 것이 매우 중요하다.

5.3 클러스터 분석 결과

본 절에서는 Google Trends 웹 사이트에서 얻어진 키워드 인기도에 대해 DBSCAN 알고리즘을 수행하여 생성한 클러스터들을 분석한다. DBSCAN 알고리즘은 파라메터 m = 1,000, ε = 0.1에 대하여 실행하였다. 클러스터 내의 키워드들을 다음과 같은 세가지 패턴으로 요약된다.

5.3.1 예상된 패턴(expected pattern)

예상된 패턴은 상식적으로 서로 관련이 있을 것으로 여겨지는 키워드들의 클러스터들이다. 한 예로 'symantec'과 'virus' 키워드 쌍을 들 수 있다. Symantec은 백신 소프트웨어 개발 회사이므로, 컴퓨터 바이러스가 급속히 퍼짐에 따라 함께 많이 검색되는 키워드이다. 이러한 연관성은 두 키워드가 한 분야에 속하므로 충분히 예상 가능한 것이다. 이러한 예상 가능한 키워드들의 집합을 모두 찾아냄으로써 자원 활용의 극대



(그림 9) 주기적 패턴의 키워드들

화를 위해 활용할 수 있다. 예를 들어, 웹 검색 기업에서는 연관된 키워드들에 대한 검색 부담을 여러 노드로 분산함으로써 검색 엔진의 성능을 최대화할 수 있다. 예상된 패턴의 키워드들의 특징은 이들을 합쳐서 생성한 키워드도 검색이 빈번하다는 점이다. 예를 들어, 'symantec virus'도 매우 인기가 높은 검색 질의어이다. 예상된 패턴의 다른 예로는{a380, airbus}, {nostradamus, prophecies} 등이 있다.

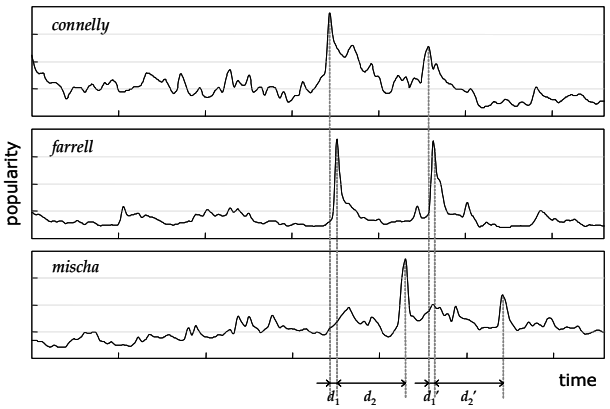
5.3.2 주기적 패턴(periodic pattern)

주기적 패턴은 특정 기간을 주기로 매우 인기가 높아지는 키워드들을 의미한다. 그 예로는 'amigo,' 'cupid,' 'egg,' 'fools,' 'oscars' 등이다. 클러스터링 알고리즘을 통하여 반환된 하나의 클러스터 내에는 거의 동일한 주기의 모든 키워드가 포함된다. (그림 9)에서 보인 것과 같이, 주기적 패턴의 키워드들은 특정 시점에 갑자기 인기가 높아짐을 알 수 있다. 예를 들어, 키워드 'egg'가 인기가 높아지는 시기는 3월 말부터 4월 초까지이며, 이 시기는 대체로 부활절(Easter)에 해당된다. 이러한 패턴은 검색 엔진이 키워드들의 주기적 인기도 정보를 분석하는 데에 유용하며, 그러한 정보는 검색 엔진의 성능을 최대화하고 광고주들에게 검색어를 판매하기 위한 가격을 결정하는 데에 활용도가 매우 높다.

5.3.3 비예측 패턴(unexpected pattern)

비예측 패턴은 상식적으로 상관관계가 쉽게 발견되지 않는 키워드들의 조합을 의미한다. 예를 들어, {connelly, farrel, mischa} 조합은 (그림 10)에서 보인 것과 같이 약간의 시간 차이를 두고 유사한 인기도를 갖는다. 하지만, 이들 키워드를 합쳐서 생성한 키워드는 별로 인기가 높지 않음을 알 수 있었다. 예를 들어, 'connelly farrell' 또는 'farrell mischa'와 같은 키워드는 인기도가 별로 높지 않았다. 이러한 비예측 패턴이 나타나는 것은 여러 가지 사회적, 문화적 원인이 있을 것이며, 그러한 원인의 분석은 본 논문의 범위를 넘는 것으로 판단된다. 본 실험을 통하여 얻어진 다른 비예측 패턴의 예로는 다음과 같은 것들이 있다.

- a380, airbus, cancun
- aniston, kingdom
- connelly, farrell, mischa, solar
- levy, lyonne, mcdermott, powerball



(그림 10) 비예측 패턴의 키워드들

- nostradamus, prophecies, spaceship
- pledge, spider
- portman, quake

6. 결 론

본 논문에서는 웹 객체 인기도의 시간적인 패턴에 대한 마이닝 문제를 다루었다. 이 문제를 해결하기 위하여 고려해야 할 몇 가지 중요 사항들을 제 1 절에서 제시하였다. 본 논문에서는 웹 객체의 인기도를 시계열로 표현하였고, 웹 객체 인기도간의 유사성을 측정하기 위한 gap 척도를 제안하였다. 또한, gap 척도를 기반으로 밀도기반 클러스터링 알고리즘의 하나인 DBSCAN을 이용하여 유사한 인기도 추세를 갖는 웹 객체들의 클러스터를 생성하였다. Google Trends 웹 사이트로부터 얻어진 검색 키워드의 인기도 추세를 이용한 실험을 통하여, 제안된 알고리즘이 실제 응용에서 웹 객체의 인기도 패턴을 찾아내는 데에 유용함을 보였다.

앞으로의 연구 방향은 다음과 같다. 첫째, 제안된 클러스터링 알고리즘을 사회적 네트워크 분석(social network analysis) [4]과 통합할 것이다. 웹 객체에 대한 액세스 패턴은 사용자들의 사회적인 네트워크에 많은 영향을 받을 수 있으며, 웹 객체 인기도 또한 유사한 영향을 받을 수 있다. 따라서, 사회적 네트워크를 분석함으로써 보다 정확한 클러스터링을 수행할 수 있을 것이다. 둘째, 인터넷 기업에서의 상용 목적에 따라 웹 객체 인기도 간의 척도를 보정하는 연구를 수행할 것이다. 예를 들어, 서버 관리 정책에 따라 제 3 절 마지막에서 설명한 w 값을 조정할 수 있을 것이다.

참 고 문 헌

[1] R. Agrawal, C. Faloutsos, and A. Swami, "Efficient Similarity Search in Sequence Databases," In *Proc. Int'l Conf. on Foundations and Data Organization and Algorithm (FODO)*, Chicago, Illinois, pp.69-84, Oct., 1993.

[2] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "OPTICS: Ordering Points To Identify the Clustering

Structure," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, Philadelphia, Pennsylvania, pp.49-60, June, 1999.

[3] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*, Addison Wesley, 1999.

[4] P. J. Carrington, J. Scott, and S Wasserman, *Models and Methods in Social Network Analysis*, Cambridge University Press, 2005.

[5] S. Chien and N. Immorlica, "Semantic Similarity between Search Engine Queries Using Temporal Correlation," In *Proc. Int'l Conf. on World Wide Web (WWW)*, Chiba, Japan, pp. 2-11, May, 2005.

[6] M. G. Elfeky, W. G. Aref, and A. K. Elmagarmid, "Periodicity Detection in Time Series Databases," *IEEE Trans. on Knowledge and Data Engineering (TKDE)*, Vol.17, No.7, pp.875-887, July, 2005.

[7] C. Elkan, "Using the Triangle Inequality to Accelerate *k*-Means," In *Proc. Int'l Conf. on Machine Learning (ICML)*, Washington, DC, pp. 147-153, Aug., 2003.

[8] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density -Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," In *Proc. Int'l Conf. on Knowledge Discovery and Data Mining (KDD)*, Portland, Oregon, pp.226-231, Aug., 1996.

[9] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast Subsequence Matching in Time-Series Databases," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, Minneapolis, Minnesota, pp.419-429, May, 1994.

[10] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Prentice Hall, 2nd Ed., 2002.

[11] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2nd Ed., 2005.

[12] M. R. Henzinger, "Web Information Retrieval - An Algorithmic Perspective," In *Proc. Annual European Symposium (ESA)*, Saarbrucken, Germany, pp.1-8, Sept., 2000.

[13] E. J. Keogh, "Exact Indexing of Dynamic Time Warping," In *Proc. Int'l Conf. on Very Large Data Bases (VLDB)*, Hong Kong, China, pp.406-417, Aug., 2002.

[14] E. J. Keogh and S. Kasetty, "On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration," In *Proc. Int'l Conf. on Knowledge Discovery and Data Mining*, ACM SIGKDD, Edmonton, Canada, pp.102-111, July, 2002.

[15] E. J. Keogh, L. Wei, X. Xi, S.-H. Lee, and M. Vlachos, "LB_Keogh Supports Exact Indexing of Shapes under Rotation Invariance with Arbitrary Representations and Distance Measures," In *Proc. Int'l Conf. on Very Large Data Bases(VLDB)*, Seoul, Korea, pp.882-893, Sept., 2006.

[16] R. Kosala and H. Blockeel, "Web Mining Research: A Survey," *SIGKDD Explorations*, Vol.2, No.1, pp.1-15, June, 2000.

[17] A. N. Langville and C. D. Meyer, "A Survey of Eigenvector Methods of Web Information Retrieval," *The SIAM Review*, Vol.47, No.1, pp.135-161, Jan., 2005.

[18] J. Lin, M. Vlachos, E. J. Keogh, and D. Gunopulos, "Iterative Incremental Clustering of Time Series," In *Proc. Int'l Conf. on Extending Database Technology (EDBT)*, Crete, Greece, pp.106-122, Mar., 2004.

[19] J. Lin et al., "An MPAA-Based Iterative Clustering Algorithm Augmented by Nearest Neighbors Search for Time-Series Data Streams," In *Proc. Pacific-Asia Conf. on Advances in Knowledge Discovery and Data Mining (PAKDD)*, Hanoi, Vietnam, pp.333-342, May, 2005.

[20] J. McQueen, "Some Methods for Classification and Analysis of Multivariate Observation," In *Proc. Berkeley Symp. on Mathematical Statistics and Probability*, Berkeley, California, pp.281-297, 1967.

[21] Y.-S. Moon, K.-Y. Whang, and W.-K. Loh, "Duality-Based Subsequence Matching in Time-Series Databases," In *Proc. Int'l Conf. on Data Engineering (ICDE)*, IEEE, Heidelberg, Germany, pp.263-272, Apr., 2001.

[22] Y.-S. Moon, K.-Y. Whang, and W.-S. Han, "General Match: A Subsequence Matching Method in Time-Series Databases Based on Generalized Windows," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, Madison, Wisconsin, pp. 382-393, June, 2002.

[23] M. Nanni, "Speeding-Up Hierarchical Agglomerative Clustering in Presence of Expensive Metrics," In *Proc. Pacific-Asia Conf. on Advances in Knowledge Discovery and Data Mining (PAKDD)*, Hanoi, Vietnam, pp.378-387, May, 2005.

[24] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, 2nd Ed., 1992.

[25] J. G. Proakis and D. K. Manolakis, *Digital Signal Processing*, Prentice Hall, 4th Ed., 2006.

[26] C. Ratanamahatana and E. J. Keogh, "Three Myths about Dynamic Time Warping Data Mining," In *Proc. SIAM International Data Mining Conference (SDM)*, Newport Beach, California, pp.506-510, Apr., 2005.

[27] Y. Sakurai, S. Papadimitriou, and C. Faloutsos, "AutoLag: Automatic Discovery of Lag Correlations in Stream Data," In *Proc. Int'l Conf. on Data Engineering(ICDE)*, Tokyo, Japan, pp.159-160, Apr., 2005.

[28] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. J. Keogh, "Indexing Multi-Dimensional Time-Series with Support for Multiple Distance Measures," In *Proc. Int'l Conf. on Knowledge Discovery and Data Mining*, ACM SIGKDD, Washington, D.C., pp. 216-225, Aug., 2003.

[29] M. Vlachos, C. Meek, Z. Vagena, and D. Gunopulos, "Identifying Similarities, Periodicities and Bursts for Online Search Queries," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, Paris, France, pp.131-142, June, 2004.

[30] B.-K. Yi, H. V. Jagadish, and C. Faloutsos, "Efficient Retrieval of Similar Time Sequences Under Time Warping," In

Proc. Int'l Conf. on Data Engineering (ICDE), Orlando, Florida, pp.201-208, Feb., 1998.

[31] B.-K. Yi and C. Faloutsos, "Fast Time Sequence Indexing for Arbitrary Lp Norms," In *Proc. Int'l Conf. on Very Large Data Bases(VLDB)*, Cairo, Egypt, pp.385-394, Sept., 2000.

[32] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An Efficient Data Clustering Method for Very Large Databases," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, Montreal, Canada, pp.103-114, June, 1996.

노 웅 기

e-mail : woong@sungkyul.ac.kr

1991년 2월 한국과학기술원(KAIST) 전산학과(학사)

1993년 2월 한국과학기술원(KAIST) 전산학과 멀티미디어 전공(석사)

2001년 2월 한국과학기술원(KAIST) 전산학과 데이터 마이닝 전공(박사)

2001년 2월~2003년 9월 ㈜티맥스소프트 미들웨어 개발 책임연구원

2003년 10월~2005년 3월 ㈜티맥스데이타 DBMS 엔진 개발 수석연구원

2005년 4월~2006년 5월 한국과학기술원 전산학과 초빙교수

2006년 6월~2007년 7월 Visiting Scholar, University of Minnesota, USA

2007년 8월~2008년 2월 NHN㈜ 대용량 데이터 분석 수석연구원

2008년 3월~현 재 성결대학교 멀티미디어학부 전임강사

관심분야 : 대용량 데이터 마이닝, 데이터 웨어하우징, 정보 검색, 멀티미디어 데이터베이스, 멀티미디어 정보 검색, 데이터베이스 시스템 엔진