블로그 서비스 시스템을 위한 효과적인 중복문서의 검출 기법

(An Efficient Method for Detecting Duplicated Documents in a Blog Service System)

이 상 철 † 이 순 행 † 김 상 욱 ††
(Sang-Chul Lee) (Soon-Haeng Lee) (Sang-Wook Kim)

요 약 블로그 서비스 시스템에 존재하는 중복문서는 블로그 검색의 서비스 질과 성능을 저하시키는 요인 중 하나이다. 기존의 웹 페이지 환경에서와는 달리, 블로그 서비스 시스템에서는 각 문서의 생성이 매번 보고되기 때문에 문서 생성 시점에 중복 판정이 가능하다. 본 논문에서는 이 점에 착안하여 문서를 저장하는 시점에 해당 문서의 중복 여부를 판정하는 새로운 중복문서 검출 기법을 제안한다. 제안된 기법을 통하여 검출된 중복문서는 검색 엔진을 위한 인덱싱에 반영시키지 않음으로써 중복문서가 검색 결과에 반영되는 문제를 원천적으로 방지할 수 있다. 또한, 효과적인 중복문서 검출을 위하여 3가지 인덱싱 기법을 제안하며, 실제 블로그 데이터를 이용하여 각 인덱싱 기법 중 가장 효율적인 기법을 보인다.

키워드 : 중복문서 검출, 블로그, 검색 엔진

Abstract Duplicate documents in blog service system are one of causes that deteriorate both of the quality and the performance of blog searches. Unlike the WWW environment, the creation of documents is reported every time in blog service system, which makes it possible to identify the original document from its duplicate documents. Based on this observation, this paper proposes a novel method for detecting duplication documents in blog service system. This method determines whether a document is original or not at the time it is stored in the blog service system. As a result, it solves the problem of duplicate documents retrieved in the search result by keeping those documents from being stored in the index for the blog search engine. This paper also proposes three indexing methods that preserve an accuracy of previous work, Min-hashing. We show most effective indexing method via extensive experiments using real-life blog data.

Key words : Duplicate document detection, Blog, Search engine

1. 서 론

중복문서(duplicate document)[1]란 기존의 다른 문서와 내용이 완전히 일치하거나 혹은 대부분의 내용이 일치하고 극히 일부 내용만 다른 문서를 말한다[1-3]. 다수의 블로거들이 다른 블로거의 좋은 문서를 자신의 블로그에 보관하기 위해 중복문서를 생성한다. 이때, 원본 문서와 동일한 내용의 문서를 생성하거나 일부만을 수정한 문서를 생성한다.

블로거들에 의해 생성된 중복문서들은 블로그 검색

· 이 논문은 2009년도 정부(교육과학기술부)의 재원으로 한국과학재단의 지원을 받아 수행된 연구임(R01-2008-000-20872-0)

† 비 회 원 : 한양대학교 전자컴퓨터통신공학
korly@hanyang.ac.kr
shlee@agape.hanyang.ac.kr
†† 종신회원 : 한양대학교 전자컴퓨터통신공학 교수
wook@hanyang.ac.kr

논문접수 : 2009년 10월 1일
심사완료 : 2009년 11월 2일

정보과학회논문지: 데이타베이스 제37권 제1호(2010.2)

1) 중복문서 검출과 유사문서 검출은 서로 다른 목적을 갖는다. 유사문서 검출은 유사한 주제이면서 서로 다른 내용을 갖는 문서들을 원하는 사용자에게 풍부한 정보를 제공해주는데 목적이 있는 반면, 중복문서 검출은 불필요한 중복 정보를 제거하여 사용자에게 꼭 필요한 정보만을 제공하는데 그 목적이 있다.

서비스[4]의 질을 저하시킨다. 중복문서들이 검색 결과에 모두 반영될 경우, 검색 서비스 사용자들은 거의 동일한 내용의 문서들로 이루어진 검색 결과를 열람하게 된다. 또한, 중복문서는 검색 서비스를 효율적으로 제공하기 위하여 필요한 검색 인덱스의 크기를 증가시킬 뿐만 아니라 이로 인하여 검색 시간을 지연시킨다. 이와 같은 문제점을 해결하기 위하여 블로그 서비스 시스템에서 중복문서를 효과적으로 검출하고, 검출된 문서들을 처리할 수 있는 방법이 필요하다.

최근 블로그에서 인용한 문장이 있는지 여부를 판정하는 qSign기법이 제안되었다[5]. qSign기법을 이용하여 중복된 문서의 존재 여부를 판정하기 위해 새로 생성된 문서의 모든 문장으로 각각 질의하여 한 문장이라도 인용한 문서들의 집합을 찾는다. 이때, 인용된 문장 수를 이용하여 새로 생성된 문서와의 중복 여부를 판정한다. 이 기법은 중복을 판정하기 위해 다수의 질의를 처리해야 함으로 성능 저하가 불가피하다.

두 문서에 대해 중복 여부를 효과적으로 판정하기 위하여 Min-hashing[2,3,6] 기법이 제안되었다. Min-hashing 기법은 웹 환경을 대상으로 하기 때문에 크롤 된 두 문서간의 중복을 판정한다. 그러나 블로그 서비스 시스템에서는 새로 작성된 문서에 대해 기존 모든 문서와의 중복 판정해야 한다. 따라서 Min-hashing기법 은 다수의 문서와 중복 여부를 판정해야 하는 성능상의 오버헤드를 가지고 있다.

본 논문에서는 문서의 생성시점을 알 수 있다는 블로그 서비스 시스템의 특징을 이용하여 중복문서를 검출하는 프레임워크를 제안한다. 제안하는 프레임워크는 기존 Min-hashing 기법을 기반하고 있으며, 효과적인 중복문서 검출을 위해 인덱싱 기법을 제안한다. 마지막으로 검출된 중복문서를 블로그 검색엔진에 반영하지 않는 처리 과정을 제안한다. 본 논문의 공헌은 다음과 같다.

1. 블로그 서비스 시스템의 특징 발견.
2. Min-hashing 기반의 중복문서 검출 프레임워크 제안.
3. 효과적인 검출을 위해 인덱싱 기법 제안.
4. 검출된 중복문서를 블로그 검색엔진에 반영하지 않는 처리 과정 제안.
5. 실험을 통해 제안한 3가지 인덱스 성능을 비교하여 가장 우수한 인덱스 기법을 보임.

본 논문의 구조는 다음과 같다. 제 2장에서는 관련 연구로 기존의 중복문서 판정 기법인 Min-hashing에 대해 설명하고, 제 3장에서는 제안된 중복문서 처리를 위한 프레임워크에 대해 상세하게 설명한다. 제 4장에서는 실험을 통해 제안하는 기법의 우수성을 평가한다. 제 5장에서는 본 논문을 요약하고 결론을 내린다.

2. Min-hashing

Min-hashing은 문서로부터 고정 개수의 일부 특징들만을 추출하여 두 문서의 유사한 정도를 근사 측정하는 기법이다[2,3,6]. 먼저, 문서로부터 연속적인 w개의 단어들로 구성된 시퀀스인 쉥글(shingle)을 추출한다. 문서 내 단어들의 개수가 n일 때, $n-w+1$개의 쉥글들이 추출된다. 또한, 추출된 쉥글은 Rabin's fingerprinting 함수[7]를 사용하여 고정된 크기의 정수 값으로 변환한다.

임의의 두 문서 A, B가 주어졌을 때, 두 문서의 중복 여부는 다음과 같이 판정된다. 먼저, 선형 순열[8] t개를 이용하여 각각의 문서에서 t개의 정수 값들을 추출해낸다. 임의의 두 문서를 비교할 경우 t개의 정수 값들을 순서대로 비교함으로써 정수 값들이 일치하는 정도를 계산한다. 또한, 두 문서간의 유사한 정도를 계산할 경우 시간과 공간 복잡도를 줄이기 위해 $t(=s \times k)$개의 정수 값들을 s개씩 k개의 그룹으로 나눈다. 각각의 그룹 내 포함된 s개의 정수 값들을 다시 Rabin's fingerprinting 함수[7]를 사용하여 최종적으로 고정 크기를 갖는 정수 값을 추출한다. 본 논문에서는 이렇게 추출된 k개의 정수 값들을 추출된 문서 특징들이라고 부르기로 한다. 문서 A와 B의 k개의 문서 특징들 중 z개 이상이 동일하면 두 문서는 중복으로 판정된다. 본 논문에서는 참고문헌 [6]에서 제시한 매개변수를 그대로 적용하여 각 문서로부터 6개의 문서 특징을 추출하고, 임의의 두 문서가 2개 이상 동일한 문서 특징을 가질 경우 중복문서로 판정한다.

3. 제안하는 프레임워크

3.1 중복문서 검출 시점

블로그 서비스 시스템에서는 웹과는 다르게 각 문서가 생성된 시점을 알 수 있다. 따라서 새로 생성된 문서가 데이터베이스와 검색 인덱스에 반영되기 전에 기존 문서들과의 중복 여부를 판정할 수 있다. 이것은 기존 웹 페이지 환경에서는 적용할 수 없는 새로운 중복문서 검출 시점이다. 본 논문에서는 문서가 새로 생성될 때 문서의 중복 여부를 판정하는 것을 문서 생성 시점 방식이라 부른다. 문서 생성 시점에 중복문서를 검출할 경우 블로그 검색 엔진의 성능에 영향을 주지 않으며, 중복문서가 검색 결과에 반영되는 것을 사전에 방지할 수 있게 된다. 또한, 문서가 생성된 시각이 데이터베이스에 함께 저장되므로 검출된 중복문서들 중에서 생성된 시각이 가장 앞선 것을 원본 문서로 볼 수 있다. 새로 생성된 문서가 중복으로 판정될 경우에는 해당 문서를 작성한 블로거가 소장할 수 있도록 데이터베이스에는 반영하되, 검색 서비스를 이용하는 사용자들을 위해서 검

색 인덱스에는 반영하지 않도록 한다.

3.2 제안하는 중복문서 처리 프레임워크

그림 1은 본 논문에서 제안하는 중복문서 처리 프레임워크를 나타낸 것이다. 중복문서 처리 프레임워크는 새롭게 생성된 문서로부터 특징을 추출하기 위한 특징 추출 과정, R*-트리[9]를 사용하여 추출된 특징들을 저장하는 인덱싱 및 질의 처리 과정, 그리고 검출된 중복문서 처리 과정으로 구성된다.

그림 1의 [1]은 중복문서 판정을 위한 특징추출 과정을 나타낸 것이다. 먼저, HTML 문서 형태로 되어 있는 문서로부터 HTML 태그를 분석하여 블로그 프레임을 제외한 코어 텍스트를 추출한다. 이렇게 추출된 코어 텍스트로부터 기존 Min-hashing 기법을 이용하여 순서를 가진 6개의 고정 크기 정수 값으로 변환한다. 따라서 기존 Min-hashing과 제안하는 프레임워크의 정확도는 동일하다.

문서 생성 시점에서 저장되는 문서와 이미 데이터베이스에 저장된 대용량의 문서들 간의 중복 여부를 빠르게 판정하기 위해서는 효과적인 인덱싱이 필요하다. 그림 1의 [2]는 본 논문에서 제안하는 중복 판정 인덱스의 구축 과정을 나타낸 것이다. 이 때 가능한 인덱싱 및 질의 처리 방식에 대해서는 제 3.3절에서 자세히 논의하기로 한다.

그림 1의 [3]은 본 논문에서 제안하는 검출된 중복문서를 처리하는 과정을 나타낸 것이다. 중복문서가 아닌 것으로 판정된 경우에는 검색 인덱스에 반영하고 데이터베이스에 저장한다. 또한, 이후의 중복문서 검출을 위하여 해당 문서의 15개의 인덱스 키들을 각각 대응되는 2차원 R*-트리에 반영시킨다. 반면, 중복문서로 판정된 경우에는 중복문서를 데이터베이스에는 저장하되, 검색 인덱스에 반영하지 않는다.

3.3 인덱싱 및 질의 처리

3.3.1 B6D1 기법

B6D1 기법(6-Btree 1-dimension)은 B-트리 6개를 사용한 기법으로 문서에서 추출된 특징 f_i ($1 \le i \le 6$)을 인덱스 키로 사용하여 6개의 B-트리에 저장한다. 이때, f_i ($1 \le i \le 6$)는 i 번째 B-트리에 저장한다. 질의할 문서의 6개의 f_i가 주어지면, 각각의 f_i로부터 생성된 질의 엔트리 $<f_i>$를 사용하여 6개의 B-트리를 검색한다. 임의의 두 문서가 중복문서로 판정되기 위해서는 2개 이상의 문서 특징이 동일해야 하므로, 6개의 B-트리를 검색된 결과로 반환된 문서 ID들 중 검색 결과 내 두 개 이상 존재하는 문서 ID를 찾아내는 후처리 과정이 필요하다. 따라서 i 번째 B-트리($1 \le i \le 5$)와 j 번째 B-트리($2 \le j \le 6$, $i \ne j$)에서 검색된 문서 ID들의 모든 쌍을 비교하여, 일치하는 문서 ID가 적어도 하나 이상 존재하면, 이후의 인덱스 검색을 중단하고 질의 처리에 사용된 문서를 중복문서로 판정한다.

3.3.2 R15D2 기법

R15D2 기법(15-R*-tree 2-dimension)은 2차원 R*-트리 15개를 사용한 인덱싱 기법으로 문서로부터 추출된 6개 특징 f_i ($1 \le i \le 6$)을 두 개씩 조합(= ${}_6C_2$)하여 2차원 R*-트리 15개에 각각 저장한다. 이때 인덱스 키는 $<f_i, f_j>$ ($1 \le i \le 5$, $i < j \le 6$)로 구성되며, 모든 조합의 순서대로 2차원 R*-트리 15개에 차례로 저장된다. 또한, 질의 처리 시 질의 점과 동일한 점이 검색될 때까지 15개의

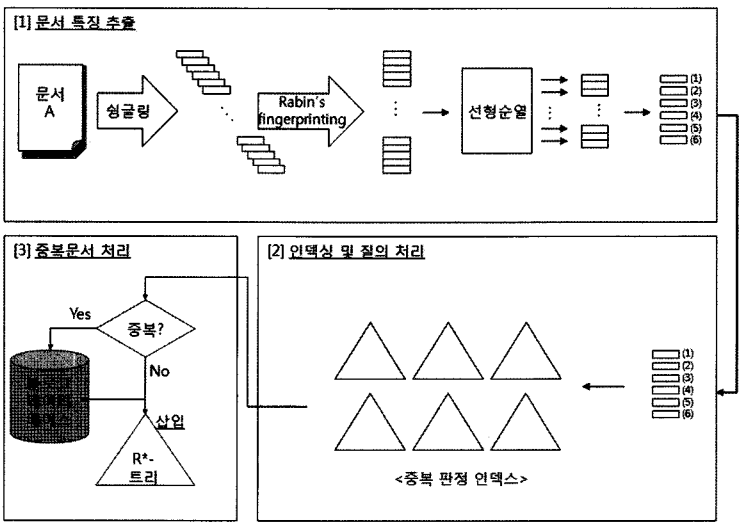


그림 1 제안하는 중복문서 핸들링 프레임워크

질의 점을 순서에 맞는 2차원 R*-트리를 이용해 검색한다.

3.3.3 R1D3 기법

R1D3 기법(1-R*-tree 3-dimension)은 3차원 R*-트리 1개를 사용한 기법으로 총 15개의 특징들의 조합과 조합의 순서를 인덱스 키로 사용한다. 인덱스 키는 $<Cid, f_i, f_j>$ $(1 \le i \le 5,\ i < j \le 6)$로 구성되며, 여기서 Cid는 15개 조합의 번호를 나타내고, fi과 fj는 문서에서 추출된 서로 다른 2개의 특징들을 나타낸다. 이러한 인덱스 구조를 기반으로 하는 질의 처리 과정은 다음과 같다. 먼저, 질의 처리에 사용될 문서로부터 추출된 6개의 f_i $(1 \le i \le 6)$를 2개씩 조합하여 조합의 순서와 함께 15개 질의 엔트리를 생성한다. 각 질의 엔트리에 대하여 3차원 R*-트리를 검색한다. 질의 엔트리가 검색되면, 이 후의 검색을 중단하고 질의 처리에 사용된 문서를 중복문서로 판정한다.

표 1은 앞서 제안한 3가지 인덱싱 및 질의처리 기법에 대해 최악의 경우 예상되는 페이지 접근 횟수를 나타낸 것이다. 인덱스 구축에 사용되는 문서의 개수를 D라고 할 때, 인덱스 검색 시 페이지 접근 횟수는 검색되는 인덱스 개수 $\times \log_{\frac{(block\ size)}{key\ size}} D$ 이다. 문서의 개수 D가 1,500,000일 때, 제안된 세 가지 기법의 예상되는 페이지 접근 횟수는 3차원 R*-트리 1개를 사용한 기법에서 약 55.0회가 발생되며, B-트리 6개를 사용한 기법에서는 약 14.9회, 그리고 R*-트리 15개를 사용한 기법에서는 약 42.6회의 페이지 접근이 발생된다.

표 1에서 볼 수 있듯이 3차원 R*-트리 1개를 사용한 기법이 B-트리 6개를 이용한 기법이나 2차원 R*-트리 15개를 사용한 기법에 비해 1.07배에서 2.58배 정도 더 많은 페이지 접근이 발생한다. B-트리 1개를 사용한 기법은 나머지 두 기법에 비해 평균적으로 페이지 접근 횟수는 적지만, 질의 결과에 대한 추가연산 비용이 많이 들기 때문에 대용량의 문서들을 처리해야 하는 환경에서는 적합하지 않다.

표 1 인덱싱 및 질의처리 기법에 따른 페이지 접근 횟수

기법	인덱스 검색 시 페이지 접근 횟수
B6D1	$1 \times log_{(block\ size\ /\ key\ size)} D + \alpha$ $= 6 \times log_{1024/4} D + \alpha$ $= 0.75 \times log_2 D + \alpha$
R15D2	$1 \times log_{(block\ size\ /\ key\ size)} D$ $= 15 \times log_{1024/8} D$ $\approx 2.14 \times log_2 D$
R1D3	$1 \times log_{(block\ size\ /\ key\ size)} D$ $= 15 \times log_{1024/12} 15D$ $\approx 2.34 \times (log_2 D + 3.9)$

4. 성능 평가

4.1 실험 데이터 및 환경

본 논문에서는 제안하는 기법의 질의 처리 성능을 평가하기 위하여 블로그 사이트 중 하나인 이글루스(Egloos)[4]로부터 1,525,465개 문서를 수집하였다. 그리고 데이터 셋의 크기 증가에 따른 질의 처리 성능 변화를 측정하기 위해, 수집한 문서들로부터 각각 300,000개, 600,000개, 900,000개, 1,200,000개, 1,500,000개의 총 5개의 문서 집합으로 구성하였다. 또한, 수집된 전체 문서 집합으로부터 1,000개 문서를 무작위로 선정하고, 해당 문서에서 임의의 위치에 있는 1~5개의 단어를 다른 단어로 변경하여 질의 문서들을 생성하였다.

본 실험에서는 제안된 세 가지 인덱싱 및 질의처리 방법에 대해서, 5개의 실험 데이터 집합을 사용하여 5개의 중복 판정 인덱스를 구축하였다. 그런 다음, 각각의 중복 판정 인덱스를 사용하여 1,000개의 질의 문서에 대한 질의 처리 시간을 측정한 후, 1000으로 나누어 한 개의 문서에 대해 평균 질의 처리 시간을 계산하였다. 또한, 세 가지 인덱싱 및 질의처리 방법에 대해 페이지 접근 횟수를 측정하였다. 질의 처리 성능 실험을 위하여 본 논문에서는 2G의 메모리 크기와 3 GHz 펜티엄 4 CPU를 가진 윈도우즈 XP 미디어 센터를 이용하여 수행하였다.

4.2 실험 결과

그림 2는 중복 판정 인덱스에 저장된 문서 개수가 증가함에 따른 질의 처리 시간의 변화를 나타낸 것이다. 이때 측정된 질의 처리시간은 검색된 중복문서를 인덱스에 반영하는 시간을 제외하고, 인덱스 검색 시간만을 측정한 결과이다. 또한, B6D1 기법은 질의 결과로 나온 후보들 중에서 동일한 후보들을 찾는 추가적인 연산 시간을 포함하였다. 그 결과, 실험 데이터가 1,500,000개 문서일 때, 3차원 R*-트리 1개를 사용한 경우 하나의 문서에 대한 평균 질의 처리 시간이 0.417초, B-트리 6개를 사용한 경우 0.244초, 2차원 R*-트리 15개를 사용

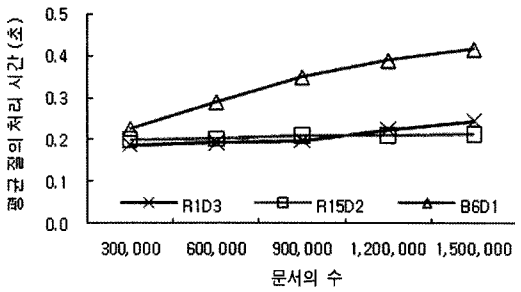


그림 2 제안하는 기법의 질의 처리 시간 비교

한 경우 0.214초가 소요되어 2차원 R*-트리 15개를 사용한 기법의 성능이 가장 우수하였음을 보였다. 그림 2에서 실험 데이터가 1,200,000개 문서일 때, 2차원 R*-트리 15개를 사용한 인덱싱 기법이 B-트리 6개를 사용한 인덱싱 기법보다 평균 질의 처리 시간이 빨라지는 것을 볼 수 있다. 이는 6개의 B-트리에서 질의 결과로 나온 후보들 중에서 일치하는 후보들을 찾는 추가적인 연산이 발생하기 때문이다.

그림 3은 질의처리 시 발생되는 평균 페이지 접근 횟수를 측정한 결과이다. 각 기법의 평균 페이지 접근 횟수는 제 3.3절에서 이론적으로 예상한 결과보다 적었다. 그 이유는 중복으로 판정되는 문서가 검색되는 순간 종료되기 때문이다. 예를 들어, B6D1기법에서 첫 번째 B-트리와 두 번째 B-트리만을 검색하여 중복인 문서를 찾았다면, 남은 3-6번째 B-트리는 검색하지 않는다. R1D3 기법은 실험 데이터 1,500,000개에서 평균 38.9회의 페이지 접근이 발생되었으며, R15D2 기법과 B6D1 기법은 각각 평균 29.9회와 평균 23.3회의 페이지 접근이 발생되었다. B6D1 기법이 R15D2 기법에 비해 적은 페이지 접근이 발생되는 반면, 그림 2의 질의 처리 시간은 B6D1 기법이 R15D2 기법에 비해 많이 소요되는 것은 B6D1에서 발생되는 후처리 연산의 오버헤드가 크다는 것을 설명해준다. 실험 결과를 통해 본 논문에서 제안한 세 가지 인덱싱 및 질의처리 기법들 중 R15D2 기법이 중복문서 검출에 가장 효과적인 인덱싱 및 질의처리 기법인 것으로 나타났다.

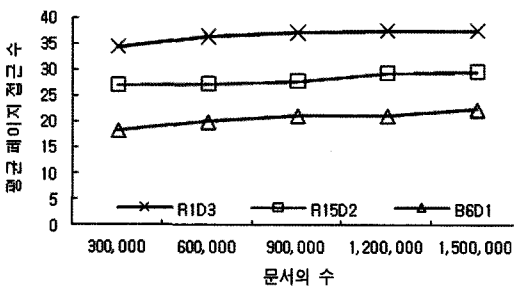


그림 3 제안하는 기법의 페이지 접근 횟수 비교

5. 결 론

중복문서 처리은 검색 서비스의 질적 향상을 위해 매우 중요하다. 블로그 서비스 시스템에서는 문서가 생성되는 시점에 중복 여부 판정이 가능하고, 문서의 생성시각을 이용해 중복된 문서들 중에서 원본문서를 판정할 수 있다. 따라서 본 논문에서는 이와 같은 블로그 서비스 시스템의 특성에 착안하여 문서 생성시점에서 중복 여부를 판정하고, 중복문서로 판정될 경우 해당 문서를 검색 인덱스에 삽입하지 않음으로써 원천적으로 검색결과에 중복문서를 반영하지 않는 효과적인 중복문서 처리 프레임워크를 제안하였다.

본 논문에서는 제안하는 기법의 확장성(scalability)을 검증하기 위해 약1,500,000개의 실제 블로그 데이터를 사용하여 문서의 증가에 따른 질의 처리 성능 변화를 측정하였다. 또한, 인덱싱 및 질의처리 기법의 변형들을 제안하고 질의 처리 성능을 비교함으로써 제안하는 인덱싱 및 질의처리 기법의 우수성을 입증하였다. 그 결과 제안하는 프레임워크에서 사용되는 2차원 R*-트리 15개를 사용한 인덱싱 기법이 다른 인덱싱 기법에 비해 약 0.5배에서 2배 정도 우수한 성능을 나타내는 것을 알 수 있었다.

참 고 문 헌

[1] A. Broder et al., "Syntactic Clustering of the Web," In *Proc. Int'l. Web Wide World Wide Web Conference, WWW*, pp.391-404, 1997.

[2] A. Broder, "Identifying and Filtering Near-Duplicate Documents," In *Proc. Int'l. Symp. on Combinatorial Pattern Matching, CPM*, pp.1-10, 2000.

[3] M. Henzinger, "Finding Near-Duplicate Web Pages: A Large-Scale Evaluation of Algorithms," In *Proc. ACM Int'l. Conf. on Information Retrieval, SIGIR*, pp.284-291, 2006.

[4] SK Communications, http://www.egloos.com.

[5] Jong Wook Kim, K. Selcuk Candan, and Junichi Tatemura, "Efficient Overlap and Content Reuse Detection in Blogs and Online News Articles," In *Proc. Int'l. World Wide Web Conference, WWW*, pp.81-90, 2009.

[6] A. Broder, "Identifying and Filtering Near-Duplicate Documents," In *Proc. Int'l. Symp. on Combinatorial Pattern Matching, CPM*, pp.1-10, 2000.

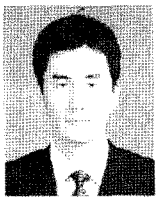
[7] M. Rabin, "Fingerprinting by Random Polynomials," Technical Report TR-CSE-03-01, Harvard University, 1981.

[8] A. Broder et al., "Min-Wise Independent Permutations," *Journal of Computer and System Sciences*, vol.60, no.3, pp.630-659, 2000.

[9] N. Beckmann et al., "The R*-tree: An Efficient and Robust Access Method for Points and Rectangles," In *Proc. ACM Int'l. Conf. on Management of Data, SIGMOD*, pp.322-331, 1990.

이 상 철

2005년 2월 한양대학교 소프트웨어학과 졸업(학사). 2007년 2월 한양대학교 전자컴퓨터통신공학과 졸업(석사). 2007년 3월~현재 한양대학교 전자컴퓨터통신공학과 재학(박사과정). 관심분야는 데이터베이스 시스템, 데이터 마이닝, 공간 데이터베이스/GIS, 이동 객체 데이터베이스/텔레매틱스, 사회 연결망 분석



이 순 행

2006년 2월 호서대학교 컴퓨터공과 졸업(학사). 2006년 8월~현재 한양대학교 전자컴퓨터통신공학과 졸업(석사). 관심분야는 데이터베이스 시스템, 데이터 마이닝, 사회 연결망 분석, 블로그 서비스 시스템, 그래프 마이닝, 데이터베이스 보안

김 상 욱

1989년 2월 서울대학교 컴퓨터공학과 졸업(학사). 1991년 2월 한국과학기술원 전산학과 졸업(석사). 1994년 2월 한국과학기술원 전산학과 졸업(박사). 1991년 7월~8월 미국 Stanford University, Computer Science Department 방문 연구원. 1994년 2월~1995년 2월 KAIST 정보전자연구소 전문연구원. 1999년 8월~2000년 8월 미국 IBM T.J. Watson Research Center Post-Doc. 1995년 3월~2000년 8월 강원대학교 컴퓨터정보통신공학부 부교수. 2003년 3월~현재 한양대학교 정보통신대학 정보통신학부 교수. 2009년 1월~현재 미국 Camegie Mellon University, Visiting Scholar. 관심분야는 데이터베이스 시스템, 저장 시스템, 트랜잭션 관리, 데이터 마이닝, 멀티미디어 정보 검색, 공간 데이터베이스/GIS, 주기억장치 데이터베이스, 이동 객체 데이터베이스/텔레매틱스, 사회 연결망 분석, 웹 데이터 분석