스킵리스트를 이용한 인터넷 토론 게시판 댓글 관리

Using Skip Lists for Managing Replying Comments Posted on Internet Discussion Boards

이윤정*, 김은경**, 조환규**, 우 균**
부산대학교 U-Port 정보기술 사업단*, 부산대학교 컴퓨터공학과**

Yun-Jung Lee(leeyj01@pusan.ac.kr)*, Eun-Kyung Kim(ekkim@pusan.ac.kr)**,
Hwan-Gue Cho(hgcho@pusan.ac.kr)**, Gyun Woo(woogyun@pusan.ac.kr)**

요약

최근 웹 블로그나 인터넷 게시판과 같은 가상 커뮤니티가 활발히 사용됨에 따라 댓글을 통해 자신의 의견을 적극적으로 나타내고자 하는 이용자들이 점점 증가하고 있는 추세다. 실제로 댓글 활동이 활발한 인터넷 토론 게시판에서 수천 개의 댓글이 달린 게시물도 어렵지 않게 찾아볼 수 있다. 대부분의 웹 블로그나 인터넷 게시판에서는 댓글이 작성된 시간에 따라 목록 형태로만 제공되고 있을 뿐 기본적인 검색 기능조차도 지원되지 않고 있다. 본 논문에서는 인터넷 토론 게시판의 댓글 분석을 통해 댓글 작성자의 분포가 거듭제곱 법칙을 따르는 것을 밝혔다. 그리고 이러한 댓글의 통계적 특성을 반영하는 스킵리스트 기반의 댓글 검색 구조를 제안한다. 제안 방법의 주안점 댓글 작성자들의 확률적 특성을 데이터 구조에 반영하는 것이다. 실험을 통해 제안 방법이 B-트리나 일반적인 스킵리스트의 이론적인 계산 복잡도인 $\log N$에 비해 더 빠른 검색을 수행할 수 있음을 보인다.

■ 중심어 : | 웹 블로그 | 인터넷 토론 게시판 | 댓글 | 스킵리스트 | 데이터 구조 | 거듭제곱 분포 |

Abstract

In recent years, the number of users who are actively express their opinions about Internet articles is more and more growing up, as the use of cyber community such as weblog or Internet discussion board increases. In fact, it is not difficult to find an article with hundreds of comments in famous Internet discussion boards. Most of the weblogs or Internet discussion boards present comments in the form of list and do not yet support even the basic operation such as searching comments. In this paper, we analysed large sets of comments in Internet discussion board named AGORA. It was found that from the result that the distribution of comment writers follows power-law. So we suppose a new search structure of comments using skip lists. The main idea of our approach is to reflect the probabilistic distribution properties of the commenters following the power-law to the data structure. Our empirical results show that the proposed method performs more efficient in searching the nodes with fewer number of comparison operations than $\log N$, which is the theoretical time complexity of general indexed structure such as B-trees or typical skip lists.

■ keyword : | Weblog | Internet Discussion Board | Comment | Skip List | Data Structure | Power-Law Distribution |

접수번호 : #100603-004
접수일자 : 2010년 06월 03일
심사완료일 : 2010년 07월 20일
교신저자 : 우 균, e-mail : woogyun@pusan.ac.kr

I. 서 론

블로그나 인터넷 게시판과 같은 인터넷 미디어들은 웹 2.0 기술을 기반으로 쌍방향 의사소통이 가능한 다양한 형태의 온라인 커뮤니티를 형성하여 새로운 여론 형성의 장이 되고 있다. 현재 많은 포털 사이트에서 사용자들이 자유롭게 이용할 수 있는 다양한 형태의 인터넷 게시판을 운영하고 있으며 개인이 쉽게 블로그 만들어 사용할 수 있도록 블로그 서비스를 제공하고 있다. 세계에서 가장 큰 규모의 블로그 검색 전문 사이트인 테크노라티의 2007년 블로고스피어 현황을 살펴보면 수집되는 블로그 수가 7천만 개, 하루에 12만 개씩 신규 블로그가 생성되고 있는 것으로 조사되었다[1].

현재 블로그나 인터넷 게시판 등이 기존 웹 페이지들과 가장 구별되는 특징은 양방향 커뮤니케이션이 가능하다는 것이다. 온라인 커뮤니티에서 개인의 의사 표현이나 의견 교환은 게시물을 만들어 등록하거나 다른 사람이 작성한 게시물에 댓글을 다는 형태로 나타난다. 인터넷 게시물이 개인의 관심사나 사회적으로 이슈가 되는 문제들을 반영한다면 그에 대한 대다수 사람들의 반응은 댓글로 나타난다고 볼 수 있다. 인터넷 게시물에 대한 독자들의 가장 적극적인 의사 표현은 댓글이라고 할 수 있다. 댓글은 누군가가 인터넷에 올린 원문에 대하여 짧게 답하여 올리는 글로 답글이나 덧글과 같은 용어로도 사용된다.

블로그나 인터넷 게시판 사용자들은 댓글을 통해 자신의 게시물을 읽는 독자들과 상호작용할 수 있으며, 대부분의 블로거들은 댓글을 통한 피드백이 게시물 작성의 중요한 동기가 된다고 인정하고 있다[2][3]. 또한 게시물 작성자뿐만 아니라 게시물을 읽는 독자들도 게시물에 대한 자신의 의견을 나타내거나 다른 사람들의 의견을 살피고 게시물에서 나타나지 않은 또 다른 정보들을 수집하기 위해 댓글을 이용하는 것으로 조사되었다[4]. 뉴스 블로그나 인터넷 토론 게시판과 같이 특정 게시물에 대해 독자들의 의견이 다양하게 나타나는 경우 댓글의 사용이 많아 그 역할이 더 크다고 할 수 있다.

한 게시물에 추가되는 댓글의 수는 게시물의 내용이나 사용자들의 관심 정도에 따라 수 개에서 많게는 수천 개 이상으로 다양하게 나타난다. 흔히 많은 댓글이 달린 게시물은 사용자의 관심을 많이 받은 게시물로 간주되어 블로그 이용자들에게 지속적으로 노출되는 경우가 많다. 그러나 이러한 댓글 중에는 실제로 게시물 내용에 관한 독자의 감상이나 자신의 의견을 표현한 일반적인 댓글도 있는 반면, 한 사람이 지나치게 많은 댓글을 작성하는 경우도 종종 포함되어 있다. 이러한 경우는 정상적인 댓글 활동으로 보기 어려우며 대부분 광고성 댓글이나 의도적으로 댓글 수를 늘리기 위한 행위로 볼 수 있다. 현재 블로그나 인터넷 게시판에서는 목록 형태로 댓글을 제공하고 있어 댓글이 많은 경우에는 사용자가 일일이 모든 내용을 읽고 확인하는 것은 어려운 일이다. 또한 앞서 언급한 비정상적인 댓글 작성자나 비정상적인 댓글은 실제로 읽어보기 전에는 파악하기 어렵다.

이와 같이 댓글이 다른 웹 페이지들과 구별되는 블로그나 인터넷 게시판과 같은 온라인 커뮤니티의 중요한 특징임에도 불구하고 댓글의 비중이나 댓글 작성자의 분포 등에 대한 연구는 부족한 실정이다. 인터넷 게시물의 댓글에 대한 기존 연구로 Mishne과 Glance의 연구를 들 수 있다[5]. Mishne과 Glance는 댓글 수에 따른 웹 블로그 게시물의 분포가 거듭제곱 법칙(power-law)을 따르며, 댓글이 많은 웹 블로그의 경우 그렇지 않은 것보다 사용자들의 관심이 높다고 제시하였다. 그러나 댓글 검색과 관리에 관한 연구는 보고된 바 없다.

본 논문에서는 블로그의 댓글 집합이 가지는 특성을 분석하고 이를 반영하는 효율적인 댓글 검색 구조를 제안하고자 한다. '아고라' 게시판에 등록된 댓글들의 분석을 통해 댓글 작성자의 분포가 거듭제곱 법칙을 따르는 것을 밝혔다[6]. 이것은 댓글 작성자의 댓글 참여 빈도에 심한 불균형이 존재하여 소수의 작성자가 많은 댓글을 생산함을 의미한다. 이 논문에서는 이러한 특성을 반영하여, 수정된 스킵리스트를 기반으로 하는 댓글 검색 구조를 제안한다. 제안 방법의 주안점은 두 가지이다. 하나는 댓글 작성자들의 확률적 특성을 고려하는 것이고, 다른 하나는 삽입과 삭제가 빈번한 댓글의 동적인 특성을 데이터 구조에 반영하는 것이다.

II. 관련 연구

블로그나 인터넷 게시판이 온라인에서 새로운 정보 교환과 의사소통의 중요한 매체가 됨에 따라 이에 대한 연구도 많이 이루어지고 있다. 블로그에 관한 연구는 주로 블로그 게시물의 태깅이나 스팸 게시물 검출 등이 대부분이며 댓글에 대한 연구는 부족한 실정이다. 인터넷 게시물의 댓글에 초점을 맞춘 연구로 Herring의 연구를 들 수 있다[7]. 이 연구에서는 블로그를 표준 웹 페이지와 비동기적 CMC(Asynchronous Computer Mediated Communication, 예: 이메일)의 중간적인 성격을 나타내는 것으로 간주하였다. 또한 Herring은 203개의 블로그를 대상으로 한 조사에서 블로그 게시물 당 평균 0.3개의 댓글이 작성되는 것으로 보고하였다. 그러나 Herring의 연구는 전체 블로그 공간으로 확장해 명확한 결론을 내기에 데이터 샘플이 너무 적고, 댓글 자체에 대한 분석은 수행하지 않았다.

Gumbrecht는 웹 블로그에서 댓글의 중요성에 대해 언급하였으며, Trevino는 댓글과 같은 피드백이 블로거들에게 게시물을 작성하는 중요한 동기를 제공한다고 언급하였다[2][3]. 또한 두 연구 모두 블로그가 가지는 상호작용성을 위해 댓글이 필수적임을 제시하였다.

Jure Leskovec은 동료 연구자들과 함께 45,000개 블로그와 2,200,000개 블로그 게시물을 분석하였다[8]. Leskovec의 연구에서는 게시물의 인기도가 -1.5 지수를 가지는 거듭제곱 법칙을 따른다고 제시하였다.

Mishne과 Glance는 상당히 큰 댓글 집합을 사용하여 블로그 공간에서에 차지하는 댓글의 비중을 추정하고, 블로그의 인기와 댓글 패턴간의 관계를 분석하였다[5]. 이 연구에서는 블로그 공간에서 댓글은 블로그 게시물 양의 약 30% 정도로 상당한 비중을 차지하고 있으며, 댓글의 양은 블로그나 게시물에 대한 관심 정도를 가리키는 지시자로 사용될 수 있다고 제시하였다. 또한 게시 물 당 댓글의 수가 [그림 1]과 같이 거듭제곱 법칙을 따른다는 것을 발표하였다.

최근에는 게시물에 달린 댓글을 이용하여 게시물의 내용을 요약하거나 대표 문장을 제시하는 방법에 대한 연구도 이루어지고 있다. Jean-Yves Delort는 블로그 게시물에 달린 댓글을 이용하여 게시물의 대표 문장을 제시하는 방법을 제안하였으며, Meishan Hu 등은 블로그 게시물에 달린 댓글이 그 게시물에 대한 독자의 이해에 영향을 줄 수 있다고 주장하고, 댓글에 숨겨진 정보를 이용하여 게시물을 요약하는 방법을 제안하였다[9][10].

또한 댓글 시각화에 관한 연구도 있었다. 이윤정 등은 게시물에 달린 댓글들을 내용에 따라 분류하고 이를 시각화하는 시스템을 제안하였다[11]. 이 시스템에서는 분류된 댓글들을 하나의 뷰로 시각화함으로써 게시물에 달린 댓글의 전체적인 개관을 파악할 수 있으며, 사용자가 원하는 댓글을 손쉽게 읽을 수 있도록 하고 있다.

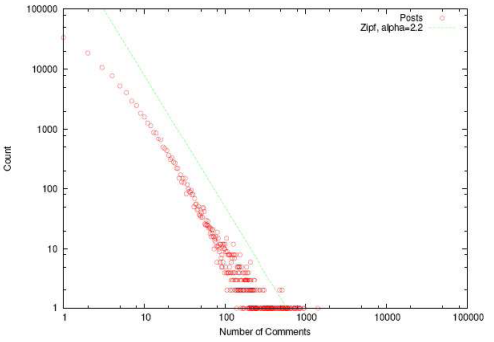


그림 1. 거듭제곱 법칙을 따르는 댓글 분포 (가로축: 댓글 수, 세로축: 게시물 수)[5].

III. 댓글의 통계적 특성

1. 거듭제곱 분포

거듭제곱 분포는 어떤 사건의 크기가 그 사건의 발생 빈도에 반비례함을 나타낸 것으로 빈익빈 부익부의 현상이 나타나는 분포라고 할 수 있다. 80%의 이탈리아 토지가 20%의 이탈리아인에게 집중되어 있다는 것과 같은 80:20 법칙으로 잘 알려진 파레토 법칙(Pareto principle)도 거듭제곱 분포의 특수한 경우다[12].

데이터 집합의 분포가 거듭제곱 법칙을 따른다는 것은 사건의 빈도와 크기가 심하게 반비례할 수 있다는

것을 의미한다. [그림 2]는 거듭제곱 분포를 나타내는 데이터 집합의 전형적인 형태를 보여준다.

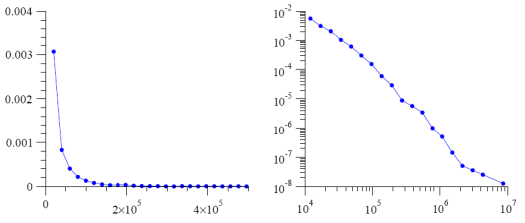


그림 2. 전형적인 거듭제곱 분포 그래프
(가로축 : 도시의 인구, 세로축 : 도시의 수)[12].

[그림 2]의 왼쪽 그래프는 인구가 10,000명 이상인 미국 도시의 인구 분포를 나타낸 그래프이고, 오른쪽 그래프는 같은 데이터를 로그 축으로 나타낸 것이다. 그림에서와 같이 거듭제곱 분포는 로그 축에서 오른쪽으로 기울어지는 직선 형태로 나타나며 직선의 기울기가 데이터의 분포 특성으로 간주된다. 거듭제곱 분포 함수는 식 (1)과 같이 정의될 수 있다.

$$p(x) = C \cdot x^{\alpha} \qquad (1)$$

여기서 C와 α는 상수다. C는 α가 결정되면 확률함수 $p(x)$의 합이 1이 되도록 정해진다. α는 Zipf의 계수라고 하며 $p(x)$를 로그 그래프로 나타내었을 때 직선의 기울기를 결정한다[12]. 거듭제곱 법칙을 따르는 분포는 지진의 크기, 논문 인용 횟수, 웹 페이지의 조회 수 등과 같은 여러 현상에서 찾아 볼 수 있다.

2. 댓글 작성자 분포

본 논문에서 인터넷 게시물의 댓글을 분석하기 위해 유명 포털 사이트인 '다음(DAUM)'에서 운영하는 '아고라(AGORA)' 게시판에서 게시물과 댓글을 수집하였다. 아고라는 포럼 형식의 게시판을 주제별로 정치, 경제, 문화 등 17개의 분야로 나누어 제공하고 있으며, 누구나 자유롭게 게시물을 읽거나 등록할 수 있도록 허용하고 있다. 또한 모든 게시물에는 댓글을 허용하고 있어 이용자들이 게시물에 대한 자신의 의견을 300자 이내로 표현할 수 있다.

보통 한 게시물에 한 사람이 하나의 댓글을 다는 것이 일반적이나 실제로는 한 사람이 같은 게시물에 여러 개의 댓글을 다는 경우도 흔히 볼 수 있다. 그러므로 댓글 목록에는 작성자가 많이 중복될 수 있다. 실제로 한 게시물에 댓글을 많이 쓴 작성자들의 댓글 중에는 토론이나 논쟁을 위한 것도 있으나 똑같은 내용을 반복적으로 작성하거나 게시물의 내용과 관계없는 광고나 욕설과 같은 스팸 댓글도 많이 포함되어 있다. 불필요한 댓글이 많을 경우 게시판 운영에도 많은 부담을 주게 되며 게시물이나 댓글을 읽는 다른 이용자들에게도 불편함과 좋지 않은 영향을 주게 된다. 따라서 댓글에 포함된 논쟁이나 스팸 댓글 등의 파악에 있어 작성자 정보는 중요한 열쇠가 될 수 있으며, 댓글 작성자들을 효율적으로 관리할 수 있는 방법이 요구된다.

본 논문에서는 아고라 게시물에 달린 댓글을 분석하여 댓글 작성자들의 분포를 조사하였다. 2009년 1월 1일부터 2009년 12월 31일까지 1년 동안 아고라의 '자유토론' 게시판에 등록된 게시물들의 댓글을 대상으로 각 게시물의 댓글 목록에서 작성자들의 댓글 작성 횟수를 분석하였다. [표 1]은 분석 게시물들과 댓글에 대한 통계 자료이다. 2009년에 아고라 '자유토론' 게시판에 등록된 게시물은 총 1,114,987개이나 삭제된 것은 제외하고 현재 읽기가 가능한 게시물은 670,800개이다. 이 중에서 댓글이 있는 게시물은 438,966개로 전체의 약 65% 정도이다. 게시물 당 평균 댓글 수는 약 5.11개이고, 최고 댓글 수는 7,316개로 나타났다. 평균 댓글 수에 비해 최고 댓글 수가 지나치게 높은 것은 Mishne과 Glance의 연구 결과[5]처럼 게시물 당 댓글 수의 분포가 거듭제곱 법칙을 따르는 것과 무관하지 않다.

표 1. 아고라 '자유토론' 게시판의 게시물 통계

게시물/댓글 분류		개수
게시물	전체	670,800 (100%)
	댓글 有	438,966 (65.4%)
	댓글 無	231,834 (34.6%)
댓글	전체	3,401,712
	평균	5.11
	최대	7,316

댓글 작성자들의 분포를 알아보기 위해 위의 게시물들 중에서 50개 이상의 댓글을 가진 9,690개 게시물을 대상으로 각 댓글 작성자들의 댓글 작성 빈도를 조사하였다. 작성자들의 분포는 [그림 3]에 나타나 있다.

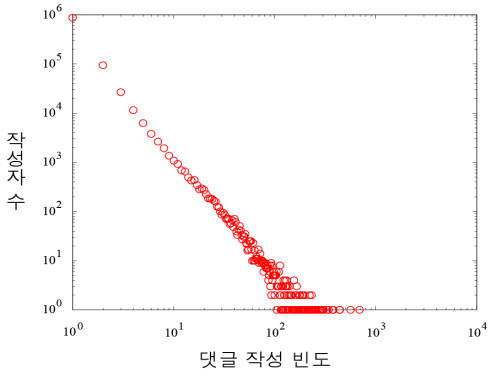


그림 3. 댓글 작성 빈도에 따른 댓글 작성자 분포

[그림 3]은 로그-로그 축으로 나타낸 것으로 가로축은 한 게시물에서 한 사람이 쓴 댓글의 수를 나타내고 세로축은 작성자의 수를 나타낸다. 그래프의 형태로 보아 댓글 작성 빈도에 따른 작성자들의 분포는 거듭제곱 법칙을 따름을 알 수 있다. 즉, 작성자들 간에 댓글 작성 횟수에 심한 불균형이 존재하며 지나치게 많은 댓글을 작성하는 작성자들이 적지 않다고 할 수 있다. 이것은 거듭제곱 분포가 가지는 중요한 특징이다.

실제로 댓글 활동이 활발한 작성자들이 쓴 댓글들 중에는 토론이나 논쟁을 위한 것도 있으나 한 게시물에 지나치게 많은 댓글을 작성한 경우는 대부분 똑같은 내용을 반복적으로 작성하거나 게시물의 내용과 관계없는 광고나 욕설과 같은 스팸 댓글이 많다. 불필요한 댓글이 많을 경우 게시판 운영에도 많은 부담을 주게 되며 이것을 효율적으로 관리할 수 있는 방법이 요구된다.

IV. 스킵리스트를 이용한 댓글 검색 구조

댓글이 지닌 특성은 다음과 같이 두 가지로 요약할 수 있다. 첫째, 댓글은 삽입과 삭제가 빈번한 동적인 데이터라는 것이다. 실제로 블로그나 인터넷 토론 게시판과 같이 댓글을 허용하는 사이트에서는 댓글 쓰기와 삭제가 빈번하게 일어난다. 둘째, 댓글 작성자의 분포가 거듭제곱 법칙을 따른다는 것이다.

본 논문에서는 댓글 관리를 위해 댓글 작성자를 기본키(primary key)로 사용하고자 한다. 왜냐하면 작성자는 댓글 내의 논쟁 관계 검출이나 스팸 댓글의 추출 등을 위해 중요한 자료가 될 수 있기 때문이다. 댓글을 통한 논쟁은 특정 작성자끼리 댓글을 주고받는 형식으로 이루어지고, 지나치게 많은 댓글을 다는 작성자들은 게시물의 내용과 관계없는 광고나 동일한 내용을 반복적으로 쓴 중복 댓글일 확률이 높다. 실제로 많은 사람들이 이용하는 웹 블로그나 인터넷 게시판에서 이러한 스팸 댓글 작성자들을 쉽게 찾아볼 수 있다.

따라서 본 논문에서는 댓글과 댓글 작성자의 효율적인 관리를 위해 스킵리스트를 사용한 데이터 구조를 제안한다. 스킵리스트는 William Pugh에 의해 처음으로 제안된 자료 구조로서 효과적인 탐색, 삽입, 삭제를 제공하는 리스트 기반의 자료구조로 다양한 응용분야에서 균형 트리의 대안으로 사용될 수 있다[13][14].

특히 스킵리스트는 구현이 간단하고 알고리즘 자체에 무작위성이 내재되어 있으므로 입력되는 자료의 크기, 순서에 큰 영향을 받지 않으며, 부분적인 수정만으로 데이터의 삽입과 삭제가 가능하다. 일반적으로 많이 사용되는 AVL 트리나 2-3 트리와 같은 다른 자료구조와 스킵리스트를 비교한 결과, 스킵 리스트는 AVL 트리보다 검색 시간이 약간 느린 것을 제외하고는 전반적으로 삽입, 삭제, 검색 시간이 빠른 것으로 나타났다. 특히 삽입과 삭제 연산에서 많은 차이를 보였다[15]. 따라서 스킵리스트는 댓글의 동적인 특성을 충분히 반영할 수 있다.

일반적인 스킵리스트의 구조는 [그림 4]와 같다.

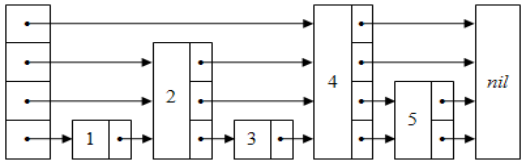


그림 4. 스킵리스트의 데이터 구조

[그림 4]에 나타낸 것처럼 스킵리스트에서 각각의 데이터 요소는 노드로 표현된다. 각 노드에는 레벨(높이; level)이 부여되는데 레벨은 그 노드로부터 다음 노드로 가는 포인터 수에 해당한다. 새로운 데이터 요소가 삽입될 때는, 그 데이터 요소에 대한 새로운 노드가 스킵리스트에 삽입된다. 이때 노드의 레벨은 미리 정해진 확률 p에 따라 무작위로 결정된다. 예를 들어 레벨 증가 확률이 $p=1/2$일 경우 레벨 1인 노드는 전체 노드의 50%, 레벨 2는 25%, 레벨 3은 12.5%와 같이 결정된다. 즉, 노드의 레벨이 확률 p로 높아지게 된다.

여기서 주목할 점은 스킵리스트에서 각 노드의 레벨이 확률 p에 따라 증가한다는 것이다. 즉, 높은 레벨을 가진 노드의 수가 더 낮은 레벨을 가지는 노드의 수에 비해 지수적으로 감소한다고 할 수 있다. 본 논문에서는 댓글과 댓글 작성자를 위한 데이터 구조로 스킵리스트를 활용하기 위해 작성자를 노드로 나타내고, 노드의 레벨을 작성자의 댓글 작성 횟수로 나타낸다. 이때 스킵리스트의 가장 낮은 레벨이 0부터 시작하므로 레벨 $k-1$인 노드는 k개의 댓글을 작성한 작성자로 생각할 수 있다.

댓글 작성자의 참여 빈도가 거듭제곱 분포를 나타내므로 전체 댓글 작성자 수가 N이고, Zipf의 계수가 -2라고 가정하면, k개의 댓글을 작성한 작성자의 수는 식 (2)와 같이 계산 된다.

$$N \times k^{-2} \tag{2}$$

작성자 데이터를 레벨 증가 확률이 0.5인 스킵리스트로 구성할 때, 레벨 k인 노드의 수는 $2^{-(k-1)}$이다. 식 (3)과 같이 k가 6보다 작거나 같을 경우 스킵리스트의 레벨 $k-1$의 노드 수가 확률 상으로 k회 댓글을 작성한 작성자 수보다 많으므로 스킵리스트의 각 레벨이 작성자의 댓글 작성 횟수를 충분히 수용할 수 있다.

$$2^{-(k-1)} > k^{-2}, \quad k \leq 6 \tag{3}$$

여기서는 댓글 작성 횟수가 6이하라고 가정하였으나, 댓글 작성 횟수가 7회 이상인 댓글 작성자들의 수는 일반적으로 그다지 많지 않고 이러한 경우에도 강제로 노드의 레벨을 증가시키면 되므로 스킵리스트를 구성하는 데 큰 문제가 되지 않는다. 이런 방법으로 지수적으로 감소하는 노드의 레벨과 댓글 작성자의 댓글 수를 대응시킴으로써 댓글 작성자와 댓글 수 사이의 거듭제곱 분포 특성을 스킵리스트에 반영할 수 있게 된다.

기존의 스킵리스트에서 각 노드는 하나의 키 값과 각 레벨에서 다음 노드를 가리키는 포인터 및 상/하 레벨을 연결시켜주는 포인터들로 구성되어 있다. 그러나 본 논문에서 구현해야 할 스킵리스트는 댓글 작성자뿐만 아니라 그 작성자가 쓴 댓글의 인덱스까지 함께 표현하여야 한다. 본 논문에서 제안하는 스킵리스트의 노드 구조는 [그림 5]와 같다.

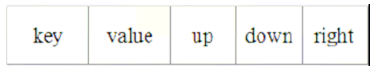


그림 5. 제안한 스킵리스트의 노드 구조

[그림 5]에서 key 필드는 작성자를 나타내고, value 필드는 작성자가 쓴 댓글 번호를 나타내며, up과 down은 같은 노드의 위쪽과 아래쪽 레벨을 가리키고, right는 각 레벨에서 다음 노드를 가리키는 포인터로 사용된다.

일반적인 블로그 사용에 있어 댓글에 관한 연산은 다음과 같은 것을 고려해 볼 수 있다.

- *Insert*(r_i) : 댓글 작성자가 게시물에 댓글을 작성하는 경우.
- *Delete*(r_i) : 댓글 작성자가 자신이 쓴 댓글 중의 하나를 삭제하는 경우.
- *Delete*(w_j) : 댓글 작성자 w_j가 쓴 모든 댓글을 삭제하는 경우.
- *Search*($R(w_j)$) : 댓글 작성자 w_j가 쓴 모든 댓글을 검색하는 경우.
- *Search*($|R(w_j)| \geq n$) :n회 이상 댓글을 작성한 작성자를 검색하는 경우.

제안 방법에서 위의 연산들이 어떻게 수행되는지를 설명하기 위해 12개의 댓글을 가진 임의의 게시물을 가정한다. 게시물의 댓글과 댓글 작성자는 [표 2]와 같다.

표 2. 댓글 목록

댓글의 순번 i	1	2	3	4	5	6	7	8	9	10	11	12
작성자 w_j	A	C	B	B	A	E	F	D	F	D	H	G

또한 설명을 간결하게 하기 위해 초기에 가질 수 있는 노드의 최대 레벨을 4로 제한하고, 레벨 증가확률은 0.5로 정의한다.

1. 댓글 삽입: Insert(r_i)

게시물에 새로운 댓글 r_i가 추가될 때, *Insert*(r_i) 연산이 발생한다. 댓글이 추가될 때 새로운 작성자가 댓글을 등록하는 경우와 기존의 작성자가 댓글을 추가하는 경우로 나누어 볼 수 있다. [표 2]에 있는 댓글 목록 중에서 1번에서 3번 댓글은 새로운 작성자에 의해 추가된 댓글이다. 이 경우 댓글 삽입 연산인 *Insert*(r_1)과 *Insert*(r_2), *Insert*(r_3) 연산 후의 스킵리스트는 [그림 6]과 같다.

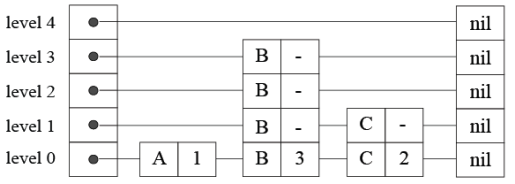
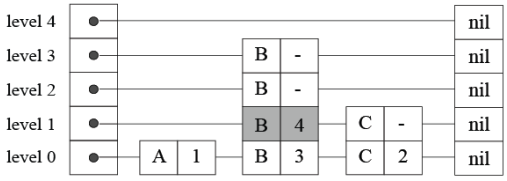


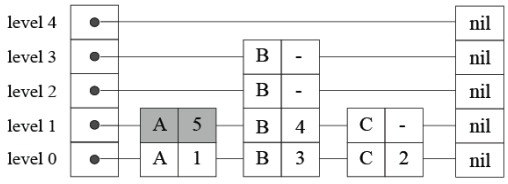
그림 6. 새로운 작성자가 쓴 댓글 삽입 – *SkipL*[1:3]

[그림 6]에서 볼 수 있듯이, 댓글 삽입 연산은 일반적인 스킵리스트의 노드 삽입과 같다. 새로운 댓글 r_i를 $i-1$개의 댓글이 삽입된 *SkipL*에 삽입하기 위해서 댓글 r_i의 작성자 w_j가 스킵리스트에 있는지 찾는다. 만일 w_j가 스킵리스트에 존재하지 않을 경우 새로운 노드를 삽입한다. 이때 노드의 레벨은 0에서부터 최대 레벨까지 중에서 확률 p로 무작위로 설정된다. 그리고 새 노드에서 모든 레벨의 key 필드에 w_j의 ID가 저장되고, 레벨 0의 value 필드에 댓글 r_i의 인덱스인 i가 저장되며 나머지 레벨의 value 필드에는 널(*null*)이 할당된다.

다음으로 이미 스킵리스트에 삽입되어 있는 작성자가 또 다른 댓글을 등록하는 경우를 고려할 수 있다. 기존 작성자가 새로운 댓글을 쓸 경우는 스킵리스트에 해당 key를 가지는 노드가 이미 존재하므로 검색 후 빈 value 필드에 새로 추가된 댓글의 인덱스를 넣으면 된다. [그림 7](a)는 작성자 'B'가 자신의 두 번째 댓글인 4번 댓글을 작성한 경우를 보여준다. [그림 7](a)에서는 'B'를 key로 하는 노드 중 비어있는 value에 해당 댓글의 인덱스인 'B'를 추가하였다. 이때는 노드 삽입이나 포인터 재조정이 발생하지 않는다.



(a) 작성자 'B'가 두 번째 댓글을 삽입한 경우 (비어 있는 value 필드가 있을 때)



(b) 작성자 'A'가 두 번째 댓글을 삽입한 경우 (비어 있는 value 필드가 없을 때)

그림 7. 기존 작성자가 쓴 댓글 삽입 – *SkipL*[1:5]

다음으로 작성자 'A'가 5번 댓글을 쓴 경우를 고려해 보자. 이 경우도 마찬가지로 스킵리스트에 이미 'A'를

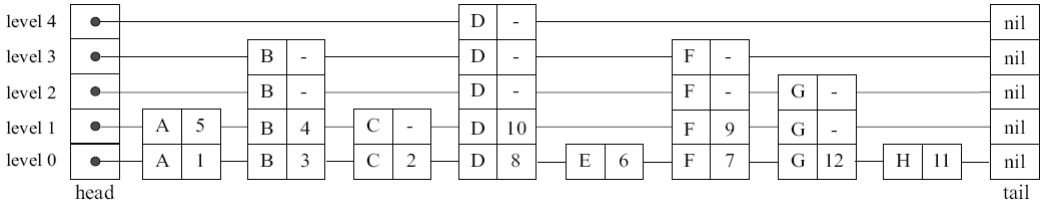


그림 8. 12개의 댓글이 삽입된 스킵리스트의 구조 – *SkipL*[1:12]

key로 하는 노드가 존재한다. 그러나 [그림 7](b)에서와 같이 'A'를 key로 가지는 노드는 초기 레벨이 0으로 더 이상 댓글을 저장할 비어있는 value 필드가 없다. 이럴 경우 새로운 노드를 하나 삽입하고 key와 value 필드에 각각 'A'와 '5'를 저장하고 포인터를 재조정한다. 포인터의 조정 방법은 일반적인 연결 리스트와 동일하다.

위와 같은 방법으로 [표 2]의 모든 댓글이 삽입된 후의 스킵리스트 *SkipL*[1:12]의 구조는 [그림 8]과 같다. 스킵리스트는 확률 기반의 자료구조이므로 실행할 때마다 각 노드의 레벨은 달라질 수 있다.

2. 댓글 삭제: *Delete*(r_i)와 *Delete*(w_j)

게시물에서 댓글이 삭제되는 경우에 스킵리스트에서도 노드 삭제가 발생한다. 제안 방법에서 댓글의 삭제는 한 작성자가 쓴 모든 댓글을 삭제하는 것과 하나의 댓글만 삭제하는 것, 두 가지 경우를 고려해 볼 수 있다.

우선 어떤 댓글 작성자가 쓴 모든 댓글을 삭제하는 경우인 *Delete*(w_j) 연산은 일반적인 스킵리스트의 노드 삭제 연산과 유사하다. *SkipL*[1:12]에서 [표 2]의 댓글 목록 중에서 작성자 'F'가 쓴 댓글을 모두 삭제하는 경우는 [그림 9]와 같다.

작성자 'F'는 7번과 9번인 2개의 댓글을 작성하였다. 'F'가 쓴 댓글을 모두 삭제하기 위해 스킵리스트에서 key 값이 'F'인 노드를 검색한다. [그림 9]에서 보듯이 해당 노드가 발견되면 검색을 멈추고 그 노드의 down 포인터를 따라가며 모든 노드를 삭제한 후 각 레벨에서 삭제된 노드 앞뒤에 위치한 노드의 포인터를 다시 설정한다.

다음은 댓글 하나를 삭제하는 Delete(r_j) 연산이다. [그림 10](a)는 [표 2]의 댓글 목록에서 9번째 댓글인 r_9

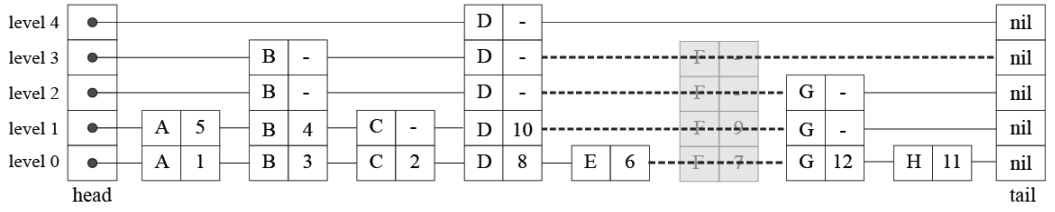
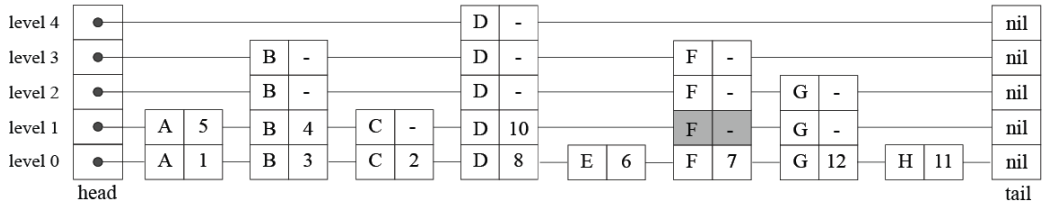
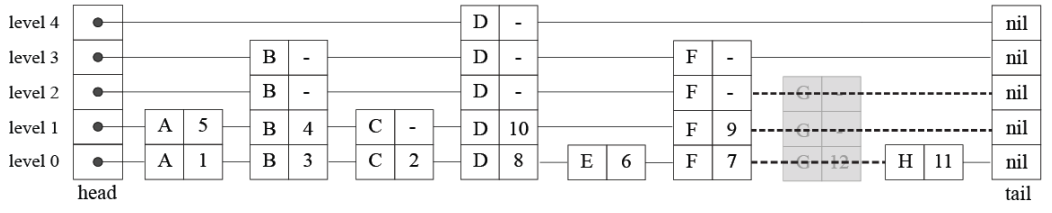


그림 9. 작성자 별 댓글 삭제



(a) 작성자 'F'가 쓴 9번 댓글 삭제 – *Delete*(r_9)



(b) 작성자 'G'가 쓴 12번 댓글 삭제 – *Delete*(r_{12})

그림 10. 댓글 삭제 – *Delete*(r_i)

를 삭제하는 *Delete*(r_9)를 나타낸 것이다. 먼저 삭제하려는 댓글의 작성자를 확인하고, 스킵리스트에서 해당 작성자를 key 값으로 가지는 노드를 검색한다. [그림 10](a)에서와 같이 r_9의 작성자 'F'를 key로 하는 노드는 level 3에서 찾아볼 수 있다.

작성자를 key로 하는 노드가 검색되면 그 노드의 down 포인터를 따라 해당 레벨에서부터 하위 레벨로 내려가면서 삭제하려는 댓글 인덱스를 value 값으로 가지는 노드를 찾는다. [그림 10](a)에서는 key 값이 'F'인 노드들 중에서 레벨 1의 노드가 댓글 9에 해당하는 노드다. 댓글 9에 해당하는 노드가 발견되면 이 노드의 value 값을 *null*로 초기화시킨다. 이러한 경우에서는 스킵리스트에서 실제 노드를 삭제할 필요가 없고 따라서 포인터를 재설정할 필요도 없다.

그러나 삭제하려는 댓글의 작성자가 하나의 댓글만을 쓴 경우는 댓글 삭제 시에 노드도 삭제해야 한다. [그림 10](b)에서처럼 [표 2]의 12번 댓글을 삭제하는 *Delete*(r_{12})의 연산의 경우 12번 댓글을 삭제하면 작성자 'G'가 쓴 댓글이 하나도 없으므로 스킵리스트에서 'G'를 key 값으로 갖는 노드는 더 이상 필요하지 않다. 따라서 각 레벨에서 해당 노드들을 모두 삭제한 후 앞뒤 노드의 포인터를 재설정해야 한다. 노드의 삭제는 앞서 설명한 바와 같이 일반적인 스킵리스트의 노드 삭제 과정과 동일하다.

3. 작성자 및 댓글 검색: *Search*($R(w_j)$)

일반적인 스킵리스트에서의 노드 검색은 스킵리스트에서 노드들이 가지는 최상위 레벨부터 시작하여 그 레벨에서 다음 노드를 가리키는 포인터를 따라 key 값을 비교하며 진행한다. 현재 레벨에서 찾고자 하는 key 값보다 크지 않은 key 값을 가지는 노드가 없을 때 까지 다음 노드로 진행한 후 하위 레벨로 방향을 바꾸어 검색을 진행한다. 가장 하위 레벨인 레벨 0에서 더 이상 검색을 진행할 수 없을 때의 노드나 그 다음 노드가 우리가 찾고자 하는 노드다. 그렇지 않을 경우에는 검색하고자 하는 key가 스킵리스트에 존재하지 않는 것이다.

제안 방법에서 작성자 별 댓글 검색 *Search*($R(w_j)$)의 동작을 살펴보면 [그림 11]과 같다. 작성자 'F'가 쓴 모든 댓글을 검색하려고 할 때, 우선 스킵리스트에서 작성자 'F'를 key 값으로 가지는 노드를 찾는다. [그림 11]에서와 같이 레벨 4에서부터 검색을 시작하여 포인터를 따라가면 레벨 3의 3번째 노드에서 key 값이 'F'인 노드를 찾을 수 있으며 이때부터 하위 레벨로 따라가며 *null*이 아닌 value 값을 찾는다. 검색 결과 9번과 7번 댓글을 검색할 수 있다.

본 논문에서 제안한 방법대로 구성된 스킵리스트에서는 작성자 별 댓글 검색을 할 경우 댓글의 작성 횟수가 많은 작성자 일수록 검색 속도가 빠르다. 왜냐하면 댓글 작성이 많은 작성자 일수록 해당 노드의 레벨이 높고, 전체 스킵리스트에서 레벨이 높은 노드는 상대적으로 수가 적기 때문에 적은 수의 노드 비교로도 원하는 작성자 노드를 검색할 수 있기 때문이다.

4. 댓글 작성 빈도 별 작성자 검색: *Search* (($|R(w_j)| \geq n$))

제안 방법으로 스킵리스트를 구성할 경우 노드의 레벨이 댓글 작성 횟수를 나타내기 때문에 작성 빈도별 작성자 검색이나 관리가 용이하다. 예를 들어 k회 이상 댓글을 쓴 작성자를 검색할 경우 스킵리스트의 $k-1$

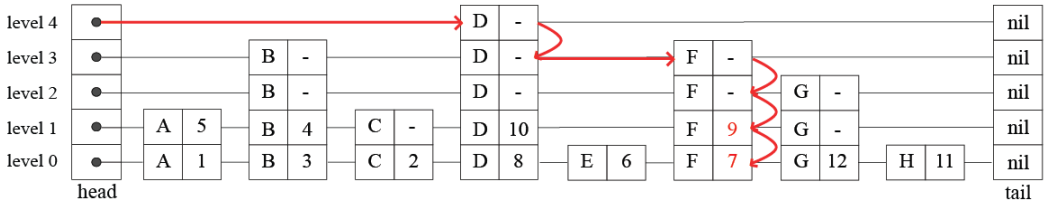


그림 11. 작성자 별 댓글 검색 – *Search*($R(w_j)$)

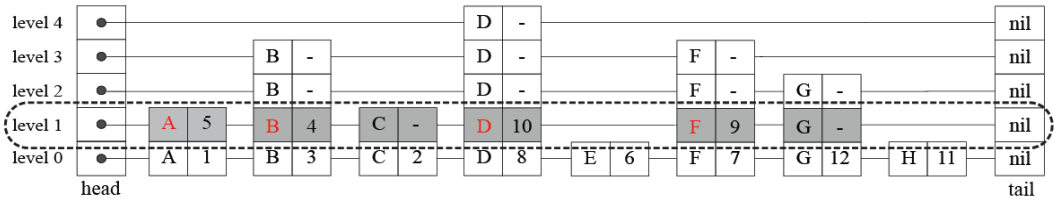


그림 12. 2회 이상 댓글을 쓴 작성자의 검색

레벨만을 검사하면 된다.

[그림 12]는 [표 2]의 댓글 목록에서 2회 이상 글을 쓴 작성자들의 검색을 보여준다. [그림 12]에서와 같이 레벨 1의 head에서부터 검색을 시작하여 right 포인터를 따라 검사가 진행된다. 이 때 각 노드의 value 값이 널(*null*)이면 실제 2회 이상의 댓글을 작성한 것이 아니므로 제외시킨다. 결과적으로 [그림 12]에서는 작성자 'A'와 'B', 'D', 'F'가 검색된다. 레벨이 높을수록 비교할 노드가 적으므로 댓글을 많이 쓴 작성자들은 적은 비교로도 쉽게 검색이 가능하다.

V. 실험 및 결과

본 논문에서 제안하는 수정된 스킵리스트를 사용한 댓글 검색 구조의 효율성을 보이기 위해 댓글 집합에 대한 검색 시간을 측정하였다. 실험은 세 가지 게시물 *UNIQ* 그리고 *A01*과 *A02*의 댓글에 대해 수행되었으며 각 게시물의 댓글 수와 작성자 수는 [표 3]과 같다.

표 3. 실험 게시물의 댓글 및 댓글 작성자 수

게시물	댓글 수	작성자 수
UNIQ	20,000	20,000
A01	10,062	2,682
A02	19,376	7,784

[표 3]에서 *UNIQ*는 실제 게시물이 아니라 댓글 집합으로 모든 작성자가 하나의 댓글을 썼다고 가정한 임의의 댓글 목록으로 구성된 가상의 게시물이다. 이것은 작성자의 중복이 없는 일반적인 데이터 집합에 대해 제안한 방법의 성능을 평가하기 위한 것이다. 그리고 *A01*과 *A02*는 아고라 게시판에서 수집된 게시물이다. 각각 10,062 개와 19,376 개의 댓글이 달린 것으로 보아 많은 사람들의 관심을 받은 게시물임을 알 수 있다. 댓글에 참여한 작성자들은 2,682 명과 7,784 명으로 해당 댓글 목록에 작성자들의 중복이 많음을 알 수 있다.

제안 방법으로 스킵리스트를 구성하기 위해 각 노드의 레벨 증가 확률 p는 0.5, 초기 노드의 최대 레벨은 10으로 설정하였다. 스킵리스트에서 삽입과 삭제 연산은 해당 key의 검색과 포인터 재설정으로 이루어지므로 검색 성능에 의존적이다[15]. 따라서 본 논문에서는 제안 방법으로 구성한 스킵리스트에서 노드 검색 성능을 측정하였다.

본 실험에서는 제안 방법으로 구성한 스킵리스트에서 노드 검색 시간을 측정하였다. 실험은 [표 3]의 실험 게시물 각각에 대해 댓글을 100개 단위로 스킵리스트에 삽입한 후 임의의 노드 검색 시간을 측정하였다. 측정된 검색 시간은 하나의 노드 비교에 걸리는 시간을 단위시간으로 하여 정규화 하였으며, 각 실험 당 10,000회 이상의 실행으로 얻은 측정값들의 평균을 취하였다. 실험 결과는 [그림 13]과 같다.

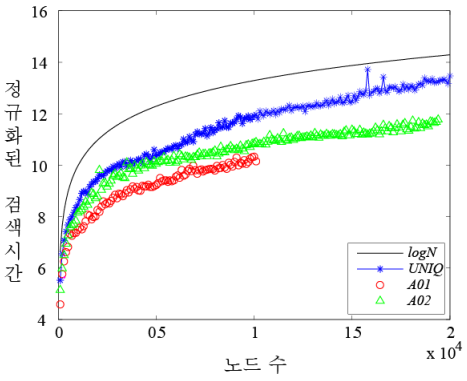


그림 13. 검색 연산 수행 시간

[그림 13]의 그래프에서 검은색 커브는 $\log N$ 함수 곡선이다. 이것은 B-트리나 스킵리스트와 같은 인덱스 검색 구조의 이론적인 계산 복잡도를 나타낸 것이다. 파란색 커브(*)는 *UNIQ*의 검색 시간을 나타낸 그래프로 댓글 작성자의 중복이 전혀 없는 상태이므로 일반적인 스킵리스트에서 검색 시간 복잡도인 $O(\log N)$과 비슷하게 측정되었다. 다음으로 붉은색(○)과 녹색(△) 커브는 각각 *A01*과 *A02*로 구성한 스킵리스트의 검색 속도를 나타낸 그래프이다. 두 가지 목록 모두 댓글 작성자의 중복이 있는 댓글 목록으로 실제 검색 속도는 이론적인 검색 속도 $\log N$보다 빠름을 알 수 있다.

위의 실험 결과를 통해 본 논문에서 제안 방법이 일반적인 데이터 집합에서보다 작성자의 중복이 있는 댓글집합에서 빠른 검색 성능을 나타냄을 알 수 있다.

VI. 결론

본 논문에서는 인터넷 토론 게시판의 댓글이 지닌 통계적 특성을 이해하기 위해서 인터넷 토론 게시판의 댓글을 분석하였다. 이를 위해 이용자 수가 많은 '아고라' 게시판에 2009년 1월 1일부터 2009년 12월 31일까지 게시된 68만 개 이상의 게시물과 3백만 개 이상의 댓글을 수집하여 분석하였으며 댓글 작성 빈도에 따른 작성자들의 분포가 거듭제곱 법칙을 따르는 것을 확인하였다.

댓글 작성자의 분포가 거듭제곱 법칙을 따른다는 것은 댓글에 참여하는 작성자들 간에 참여 빈도에 있어 쏠림 현상이 나타남을 의미한다. 즉, 소수의 작성자가 많은 댓글을 작성한다고 할 수 있다. 댓글 내에 포함된 논쟁 관계 파악이나 스팸 또는 중복 댓글의 파악을 위해서 댓글 집합에서 작성자 정보는 중요한 데이터가 될 수 있다. 따라서 인터넷 토론 게시판의 효율적인 운영을 위해 이러한 작성자들의 효율적인 관리 방법이 요구된다.

이를 위해 본 논문에서는 스킵리스트를 사용한 댓글 검색 구조를 제안하였다. 스킵리스트는 구현하기 쉬울 뿐만 아니라 빠른 삽입과 삭제 연산이 가능하여 삽입과 삭제가 빈번한 동적인 댓글 특성을 반영할 수 있다.

또한 본 논문에서는 일반적인 스킵리스트의 노드에 value 필드를 추가하여 스킵리스트의 각 레벨이 댓글 작성 횟수를 나타낼 수 있도록 하였다. 제안 방법에서 기존의 스킵리스트에 value 필드를 추가한 것은 매우 간단한 변경이라고 할 수 있으나 이로 인해 댓글 데이터의 분포적 특성을 검색 구조에 반영할 수 있다는 것은 중요한 결과라고 할 수 있다.

실제 인터넷 토론 게시판의 댓글 목록을 이용한 실험을 통해 제안 방법이 일반적인 스킵리스트의 계산 복잡도인 $O(\log N)$보다 빠른 검색 성능을 보여 효율적인 댓글 관리가 가능함을 알 수 있었다.

본 논문에서 제시한 바와 같이 인터넷 토론 게시판에서 댓글 작성자의 분포는 거듭제곱 법칙을 따르므로 임계치 이상의 댓글이 달린 경우는 이후에 댓글 수가 지수적으로 증가하게 될 것을 예상할 수 있다. 이러한 게시물의 경우 다수의 공감을 얻는 정상적인 게시물일 수도 있지만, 의도적인 조작이나 불필요한 중복 댓글을 통한 결과일 수도 있다. 따라서 향후 연구로 게시물이나 댓글의 유용성과 정보성 평가 방법에 대한 연구가 필요할 것이다.

참 고 문 헌

[1] David Sifry, "The State of the Live Web," a web article, April 2007. http://www.sifry.com/alerts/archives/000493.html

[2] M. Gumbrecht, "Blogs as protected space," Presented at the Workshop on the Weblogging Ecosystem: Aggregation, Analysis, and Dynamics, May 2004.

[3] E. Trevino, "Blogger motivations: Power, pull, and positive feedback," Presented at Internet Research 6.0, 2005.

[4] J. Sim, H. Cho, H. Yang, I. Ahn, and E. Na, The state of Netizen Internet Utilization on Web 2.0, National Internet Development Agency of Korea, 2006.

[5] G. Mishne and N. Glance, “Leave a reply: An analysis of weblog comments,” Presented at 3rd Annual Workshop on the Weblogging Ecosystem, 2006.

[6] Daum 아고라, http://agora.media.daum.net/

[7] S. Herring, L. Scheidt, S. Bonus, and E. Wright. “Bridging the gap: A genre analysis of weblogs,” In Proc. of the 37th Annual Hawaii International Conference, 2004.

[8] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst, “Patterns of Cascading Behavior in Large Blog Graphs,” In Proc. SDM2007, pp.551-556, 2007.

[9] J. Y. Delort, “Identifying commented passages of documents using implicit hyperlinks,” In Proc. of HYPERTEXT'06, pp.89 - 98, 2006.

[10] Meishan Hu, Aixin Sun, and E. P. Lim, “Comments-oriented blog summarization by sentence extraction,” In Proc. of CIKM'07, pp.901-904, 2007.

[11] 이윤정, 지정훈, 우균, 조환규, “인터넷 게시물의 댓글 분석 및 시각화”, 한국콘텐츠학회논문지, 제9권, 제7호, pp.45-56, 2009.

[12] M. Newman. “Power laws, Pareto distributions and Zipf's law,” Contemporary physics, vol.46, No.5, pp.323-351, 2005.

[13] D. Wang and J. Liu, “Peer-to-Peer asynchronous video streaming using Skip List,” In Proc. of ICME2006, pp.1397-1400, 2006.

[14] T. Ge and S. Zdonik, “A skip-list approach for efficiently processing forecasting queries,” In Proc. VLDB Endow, Vol.1, No.1, pp.984-995, 2008.

[15] W. Pugh, “Skip Lists: A Probabilistic Alternative to Balanced Trees,” Communications of the ACM, Vol.33, No.6, pp.668-676, ACM, 1990.

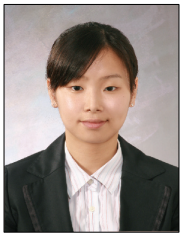
저 자 소 개

이 윤 정(Yun-Jung Lee) 정회원

- 1995년 : 부경대학교 전자계산학과(이학사)
- 1999년 : 부경대학교 전산정보학과(이학석사)
- 2008년 : 부경대학교 전자계산학과(이학박사)
- 2008년 ~ 현재 : 부산대학교 U-Port 정보기술 사업단 박사후연구원

<관심분야> : 얼굴 애니메이션, 웹 콘텐츠 시각화

김 은 경(Eun-Kyung Kim) 준회원



- 2010년 : 부산대학교 정보컴퓨터공학(학사)
- 2010년 ~ 현재 : 부산대학교 컴퓨터공학과 석사과정

<관심분야> : 프로그래밍언어 및 컴파일러, 프로그램 시각화

조 환 규(Hwan-Gue Cho) 정회원

- 1884년 : 서울대학교 계산통계학과(학사)
- 1990년 : 한국과학기술원 전산학(공학박사)
- 1991년 ~ 현재 : 부산대학교 컴퓨터공학과 교수

<관심분야> : 알고리즘 이론, 생물정보학

우 균(Gyun Woo) 정회원

- 1991년 : 한국과학기술원 전산학(학사)
- 1993년 : 한국과학기술원 전산학(석사)
- 2000년 : 한국과학기술원 전산학(박사)
- 2000년 ~ 2002년 : 동아대학교 컴퓨터공학과 전임강사
- 2002년 ~ 2004년 : 동아대학교 컴퓨터공학과 조교수
- 2004년 ~ 현재 : 부산대학교 컴퓨터공학과 조교수

<관심분야> : 프로그래밍언어 및 컴파일러, 함수형 언어, 그리드컴퓨팅, 소프트웨어 메트릭, 프로그램 시각화