주문형 전자신문 시스템에서 사용자 접근 패턴을 이용한 기사 프리패칭 기법

김 영 주[†] · 최 태 욱[††] · 정 기 동[†††]

요 약

NOD 기사 데이터는 VOD의 비디오 데이터와 비교하여 생성주기가 짧고, 수시로 생성되며, 사용자가 평균적으로 선택하는 데이터의 개수가 다를 뿐만 아니라 접근 시간 등에서 높은 편기성을 나타낸다. 이러한 고유의 특성으로 인하여 NOD 기사에 대한 사용자 접근 패턴이 VOD의 경우와는 다르게 나타나는데, 이는 기존의 VOD 서버와 관련된 기술을 이용하여 NOD 서버 시스템을 구축할 경우 최적의 성능을 얻을 수 없음을 의미한다.

본 논문에서는 먼저 현재 운영중인 전자신문의 로그 파일을 분석하여 NOD 기사 데이터의 인기도 모형이 멀티미디어 응용에서 널리 사용하는 Zipf 분포와는 다름을 보인 후에 새로운 인기도 모형으로서 MS-Zipf(Multi-Selection Zipf) 분포와 그에 대한 근사적인 알고리즘을 제시한다. 그리고 시간에 따른 인기도 변화를 나타내는 기사 데이터의 생명주기(Life Cycle) 모형을 제시하고, 이를 이용하여 접근 가능성이 높은 기사를 예측하여 프리패칭하는 LLBF(Largest Life-cycle Based Frequency) 프리패칭 알고리즘을 제안하고 성능을 분석한다. 제안한 LLBF 알고리즘은 LRU 등의 다른 재배치 알고리즘에 비해 비슷한 적중률을 보이면서 불필요한 재배치 횟수를 줄임으로써 재배치 비용의 부담을 줄일 수 있었다. NOD 기사 데이터의 사용자 접근 패턴에 대해 정확한 모형을 제안함으로써 NOD 서버 시스템의 성능을 정확히 분석할 수 있을 뿐만 아니라 NOD 서버 시스템을 구현하는데 있어 보다 효과적인 기법들을 개발할 수 있게 되었다.

Article Data Prefetching Policy using User Access Patterns in News-On-Demand System

Young-Ju Kim[†] · Tae-Uk Choi[††] · Ki-Dong Chung[†††]

ABSTRACT

As compared with VOD data, NOD article data has the following characteristics: it is created at any time, has a short life cycle, is selected as not one article but several articles by a user, and has high access locality in time. Because of these intrinsic features, user access patterns of NOD article data are different from those of VOD. Thus, building a NOD system using the existing techniques of VOD system leads to poor performance.

In this paper, we analysis the log file of a currently running electronic newspaper, show that the popularity distribution of NOD articles is different from Zipf distribution of VOD data, and suggest a new popularity model of NOD article data MS-Zipf(Multi-Selection Zipf) distribution and its approximate solution. Also we present a life cycle model of NOD article data, which shows changes of popularity over time. Using this life cycle model, we develop LLBF (Largest Life-cycle Based Frequency) prefetching algorithm and analysis the performance by simulation. The developed

† 준 회 원 : 부산대학교 대학원 전자계산학과
†† 준 회 원 : 부산대학교 대학원 멀티미디어 협동과정학과
††† 종신회원 : 부산대학교 전자계산학과 교수
논문접수 : 1999년 2월 11일, 심사완료 : 1999년 3월 6일

LLBF algorithm supports the similar level in hit-ratio to the other prefetching algorithms such as LRU(Least RecentlyUsed) etc, while decreasing the number of data replacement in article prefetching and reducing the overhead of the prefetching in system performance.

Using the accurate user access patterns of NOD article data, we could analysis correctly the performance of NOD server system and develop the efficient policies in the implementation of NOD server system.

1. 서 론

컴퓨터 및 네트워크, 그리고 통신기술의 발달과 멀티미디어 관련기술의 발전으로 인하여 활자인쇄의 신문을 통한 뉴스정보 보다는 멀티미디어를 통한 뉴스정보에 대한 요구가 증가하고 있으며, 이에 NOD(News On Demand) 멀티미디어 응용에 대한 관심이 집중되고 있다. 멀티미디어 NOD 서버는 텍스트, 이미지, 비디오 그리고 오디오 등을 조합하여 구성되는 NOD 기사(article) 데이터를 사용자의 요구에 따라 실시간으로 서비스한다. NOD 기사 데이터는 VOD(Video On Demand) 데이터와 달리 미디어의 다양성, 데이터의 크기, 그리고 요구 대역폭 등에서 차이점을 가진다[1]. 특히, NOD 기사 데이터는 다음과 같은 주요한 특성을 가진다.

첫째, 시간적, 공간적 접근 편기성을 가진다. 사용자 요구는 밤이나 새벽시간보다는 낮 시간에 집중적으로 발생하며, 특정 시간에 사용자는 모든 기사에 접근하는 것이 아니라 관심 있는 주요 기사에만 접근하기 때문에 특정 기사에 접근 편기성이 높아진다.

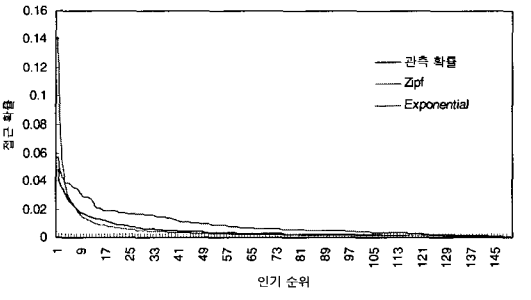
둘째, 기사 데이터는 생성주기가 짧고 수시로 생성된다. 비디오 데이터의 경우에 그 생성주기가 보통 일주일에서 보름 단위이다. 따라서 사용자의 선호도가 그 생성주기에 따라 천천히 변한다. 그러나 뉴스기사는 하루 중에 수시로 생성되기 때문에 생명주기가 비디오 데이터에 비해 매우 짧다.

셋째, 사용자는 평균적으로 한번 접속하여 여러 개의 기사 데이터에 접근한다. VOD의 비디오 데이터의 경우에는 비교적 긴 서비스 시간과 높은 서비스 비용으로 한번 접속하여 한두 편 정도의 비디오 데이터를 접근하지만, NOD 기사 데이터는 그 크기가 작고 짧은 주기로 생성되어 사용자는 대부분 한번 접속하여 여러 개의 기사 데이터에 접근한다.

위와 같은 고유의 특성으로 인하여 NOD 기사 데이터에 대한 사용자 접근 패턴은 VOD의 경우와는 다르게 나타난다.

VOD의 비디오 데이터에 대한 사용자 접근 패턴을 표현하는 대표적인 모형으로서 Zipf 분포가 있는데, 이는 가장 인기 있는 비디오부터 순위를 줄 때, 임의의 순위를 가진 비디오가 접근될 확률을 나타내는 분포로서 멀티미디어 응용 연구에서 널리 사용되는 인기도 모형이다[2]. 그런데 현재 운영중인 전자신문의 로그파일을 분석한 결과, 실제 NOD 기사 데이터의 사용자 접근 패턴이 Zipf 분포와는 상당한 차이가 있음을 알 수 있었다.

(그림 1)은 1998년 6월부터 12월까지의 전자신문 로그 파일을 분석하여 하루 중에 접근되는 기사들에 대해 150개의 상위 기사에 대한 평균 접근 확률 분포를 Zipf 분포 및 지수(Exponential) 분포와 비교하여 보여주고 있는데, 전체적으로 지수 분포와는 큰 차이가 있으며, Zipf 분포와는 사용자의 접근이 집중되는 상위 순위 부분에서 많은 차이를 보이고 있다.



(그림 1) 관측 확률 분포와 다른 확률 분포와 비교

멀티미디어 응용에서 사용자 접근 패턴은 멀티미디어 서버를 구성하는 여러 가지 기능, 즉 미디어 데이터의 배치 및 검색, 그리고 데이터 캐싱과 프리패칭 등에서 중요한 역할을 한다[3].

따라서 본 논문에서는 현재 운영중인 전자신문의 로그 파일에 대한 분석을 통하여 NOD 기사 데이터의

사용자 접근 패턴이 기존의 Zipf 분포나 지수 분포와는 차이가 남을 보이고 새로운 인기도 모형으로서 MS-Zipf(Multi-Selection Zipf) 분포를 제시한다. 그리고 MS-Zipf 분포에 대해 알고리즘적 접근법으로 확률 생성 알고리즘을 제시하고 실제 데이터에 대한 적합성을 검정한다. 또한, 로그 파일의 통계분석을 통하여 기사 데이터의 생명주기 모형을 추출하고, 이를 이용하여 NOD 서버 시스템의 성능을 향상시키는 LLBF(Largest Life-cycle Based Frequency) 프리패칭 알고리즘을 제안하고 성능을 분석한다.

본 논문의 구성은 다음과 같다. 2장에서는 현재까지 진행되어 온 관련 연구에 대하여 고찰하고, 3장에서는 NOD 기사 데이터의 새로운 인기도 모형인 MS-Zipf 분포를 제안하고 이를 다른 인기도 모형과 비교한다. 4장에서 본 논문에서 가정하는 서버 시스템의 구조에 대하여 간단히 살펴본 다음, NOD 기사 데이터의 생명주기 모형을 제시하고 이를 이용한 LLBF 프리패칭 알고리즘을 제안한다. 5장에서는 제안된 LLBF 프리패칭 알고리즘에 대해 모의실험을 통하여 성능을 분석한다. 마지막으로 6장에서 결론과 향후 연구 방향에 대하여 살펴본다.

2. 관련 연구

2.1 멀티미디어 데이터의 인기도 모형 연구

지금까지 멀티미디어 응용에 관한 연구에서는 멀티미디어 데이터의 인기도 모형으로서 Zipf 분포를 널리 사용하였는데, 최근의 몇몇 논문은 실제의 인기도 모형이 Zipf 분포와는 다르다는 의견을 제시하고 있다.

Scott A. Barnett 등은 실제 비디오의 인기도 모형이 Zipf 분포보다는 double exponential 분포에 더 근사함을 보이고, 비디오의 생명주기 모형으로서 Chart Movement를 제안하였다[4]. Carsten Griwodz 등은 계층적인 분산 VOD 시스템에서 비디오 데이터의 인기도 변화를 나타내는 Long-term Movie Popularity 모형을 제시하였는데, 이는 3개의 매개변수를 가진 지수 분포의 변형 모형이었다[5].

위의 논문들은 Zipf 분포가 비디오 대여점 등에서의 인기도에 기반하고 있어 실제로 사용자의 요구를 충분히 반영하지 못하였음을 지적한다. 따라서 위의 논문에서 새롭게 제시하는 인기도 모형은 실제 Zipf 분포보다는 편기성이 더 크게 나타난다.

NOD 기사 데이터의 경우에는 아직 인기도 모형에 대하여 연구된 바가 없으며, 현재는 대부분 Zipf 분포로 가정하고 있는 실정이다.

2.2 멀티미디어 데이터 캐싱에 관한 연구

전통적인 파일시스템이나 데이터 베이스 시스템에서의 캐싱(caching)은 대부분 데이터 접근의 최신도(recency)와 빈도수(frequency)를 기반으로 한다[6,7,8]. 이러한 방법은 크기가 작은 텍스트 위주의 데이터를 다루기 때문에 대용량, 고 대역폭을 요구하는 멀티미디어 데이터를 포함하는 NOD 기사의 캐싱 정책으로는 적합하지 못하다.

비디오 데이터와 같은 연속 매체의 캐싱은 그 크기가 방대하여 데이터 전체를 캐싱하는 것이 불가능하다. 따라서, 주기적이고 순차적으로 요구되는 블록을 위해 할당된 캐쉬 버퍼는 사용 후에 즉시 재사용되어야 한다[9]. Asit Dan 등은 연속적으로 도착하는 동일한 데이터에 대한 요구들을 위하여 일정한 시간간격 동안의 데이터를 캐싱하는 Interval caching 기법을 제안하였다[10]. 그러나 NOD 기사의 경우 연속 매체를 포함하지만, 5~10분 정도의 비디오 클립 등으로 크기가 비교적 작다. 따라서 대용량을 전제로 한 연속 매체의 캐싱 정책을 그대로 적용하기는 무리가 있다.

최근 웹 캐싱에 대하여 관심이 집중되고 있는데 웹 문서는 크기가 다양하고 여러 가지 미디어를 포함한다는 점에서 NOD기사와 비슷하나 웹 문서의 생명주기가 비교적 길다는 차이가 있다. I. Tatarinov 등은 웹 문서의 사용자 접근 패턴이 느리게 변함으로 재배치를 자주하는 것이 역효과가 날 수 있음을 지적하고 일정 시간마다 주기적으로 재배치하는 static caching을 제안하였다[11]. James E. Pitkow 등은 웹 문서의 사용자 접근 패턴을 인간의 기억력에 기반하여 인지학적으로 분석하였으며, 사용자 접근 패턴과 히트율에 적응할 수 있는 캐싱 방법을 주장하였다[12]. 그러나 NOD 기사는 생명주기가 매우 짧고 사용자의 접근 편기성이 다양하게 나타나기 때문에 웹 캐싱 기법을 그대로 적용하는 것은 효율적이지 못하다.

3. MS(Multi-Selection) Zipf 분포 모형

NOD 기사 데이터는 앞 절에서 설명한 것과 같이 VOD 데이터와는 다른 고유의 특성을 가지며, 이러한

특성이 기사 데이터에 대한 사용자의 접근 패턴에 커다란 영향을 주고 있음을 운영중인 전자신문의 로그 파일에 대한 통계분석을 통하여 알 수 있었다. 이 절에서는 NOD 기사 데이터에 대한 사용자의 접근 패턴에 대해 새로운 모형과 알고리즘에 의한 접근법을 제시한다.

3.1 MS-Zipf 분포 모형

일반적으로 VOD와 같은 멀티미디어 응용에 관한 연구에서는 미디어 데이터에 대한 사용자의 접근 패턴 모형으로서 Zipf 분포를 널리 사용하고 있다[2]. 다시 말해, VOD 응용에서는 사용자가 비디오 데이터에 접근할 때에 특정 비디오를 선택할 확률, 즉 특정 비디오의 인기도를 나타내기 위하여 (식 1)의 Zipf-like 분포함수를 사용한다. 그 이유로는 사용자들이 모든 비디오에 골고루 접근하는 것이 아니라 인기 있는 비디오에 편중하여 접근하는 경향을 나타내며, Zipf 분포가 이러한 접근 편기성을 적절히 반영하고 있기 때문이다.

$$Zipf(i) = \frac{F_i}{\sum_{j=1}^{N} F_j}, \quad F_i = \frac{1}{i^{1-\theta}} \qquad \text{(식 1)}$$

단, $i = 1 \cdots N$, N= 비디오데이터 수

NOD 기사의 경우에도 마찬가지로 인기 있는 특정 기사에 사용자의 접근이 편중되는 접근 편기성이 나타나고 있으나 VOD의 경우와는 많은 차이점이 있음을 (그림 1)에서 알 수 있었다. 이러한 차이점은 VOD 사용자의 경우에 비교적 긴 서비스 시간과 높은 서비스 비용으로 한번 접속하여 한 편, 또는 두 편 정도의 비디오 데이터에 접근하나, NOD 사용자의 경우에는 한번 접속하여 여러 개의 기사 데이터에 접근함으로써 접근 편기성이 높은 기사 데이터 수가 많아진다는 사실에 의해 설명되어지며, 또한 NOD 기사 데이터의 인기도 모형이 Zipf 분포와는 다르다는 것을 의미한다.

NOD 기사 데이터에 대한 사용자 접근 패턴을 모형화하는데 있어 사용자가 한번 접속하여 평균적으로 k개의 기사를 접근하는 것은 사용자가 k개의 기사 조합으로 이루어진 기사 그룹들 중에 하나의 기사 그룹을 접근하는 것과 같다고 할 수 있다. 그리고, 하나의 기사 그룹이 접근될 확률은 접근 확률이 높은 기사들로 이루어진 기사 그룹에 크게 편기되어 나타나므로 VOD 데이터와 같이 Zipf-like 분포를 따른다고 할 수 있다. 따라서 다음과 같은 가정들을 세울 수 있다.

- 가정 1

  사용자는 한번 접속하여 N개의 기사 집합 $A = \{a_1, a_2, \cdots, a_N\}$에서 평균적으로 k개의 기사를 접근한다.

- 가정 2

  사용자가 $M = \binom{N}{k}$개의 기사 그룹의 집합 $C = \{C_1, C_2, \cdots, C_M\}$에서 임의의 기사 그룹 C_i에 접근할 확률은 (식 1)의 모수 θ를 가지는 Zipf-like 분포를 따른다.

NOD 응용에서 사용자의 접근 패턴이 가정 1, 2를 만족할 때에 사용자는 기사 단위로 접근하는 것이 아니라 기사 그룹 단위로 접근하게 됨으로 각각의 기사가 접근될 확률은 그 기사가 포함되어있는 기사 그룹들이 접근될 확률의 합이 된다. 그러나 각각의 기사 그룹의 접근 확률은 원소 개수인 k만큼 반복 계산되기 때문에 모든 기사의 접근 확률의 합이 '1'이 되도록 정규화 하여야 한다. 따라서 하나의 기사가 접근될 확률의 분포를 정의하면 다음과 같다.

- 정의 1 : Multi-Selection Zipf 분포

  가정 1, 2하에서 임의의 기사 a_i가 접근될 확률이 기사 a_i을 포함하고 있는 기사 그룹들이 접근될 확률의 합을 k로 나눈 값이라 주어질 때에 기사 a_i의 접근 확률의 분포를 'MS(Multi-Selection) Zipf 분포'라고 정의하고 기사 a_i의 접근 확률은 다음과 같이 나타낸다.

$$\Pr(selection = a_i) = \frac{\sum_{a_i \in C_j} \Pr(C_j)}{k} \qquad \text{(식 2)}$$

[정의 1]에 의하여 기사 a_i의 접근 확률을 계산하기 위해서는 먼저 기사 그룹들의 접근 확률이 계산되어야 한다. Zipf-like 분포함수를 이용하여 기사 그룹의 접근 확률을 계산하기 위해서는 기사 그룹에 순위가 부여되어야 하는데 순위 부여 방법에 따라 기사 a_i의 접근 확률 분포가 달라질 수 있다. 따라서, 본 논문에서는 NOD 기사 데이터의 접근 편기성과 알고리즘의

간편성을 고려하여 (그림 2)의 기사 그룹 순위 부여 규칙에 따라 기사 그룹에 순위를 정하였다.

1) 전체 기사의 순위는 {1, 2, ···, N}으로 주어진다
2) 각각의 기사 그룹에 대하여 그 기사 그룹이 포함하고 있는 기사들의 순위 값의 합인 조합값(Combination Value : CV)을 구하고 조합값을 기준으로 기사 그룹의 순위를 부여한다
3) 조합값이 작은 기사 그룹에 높은 순위를 부여한다
4) 조합값이 같은 기사 그룹들에 대해서는 더 높은 순위의 기사를 포함하는 기사 그룹에 높은 순위를 부여한다

(그림 2) 기사 그룹 순위 부여 규칙

(그림 2)에서 정의된 방법은 NOD 기사 데이터의 특성을 충분히 고려하였다는 장점이 있으나 기사 그룹의 순위를 수식으로 표현하기가 어렵다. 이는 기사 a_i의 접근 확률, 즉 MS-Zipf 분포의 확률 분포 함수를 수학적으로 정의하기 어렵다는 것을 의미한다. 따라서, 본 논문에서는 MS-Zipf 분포의 확률 밀도 함수에 대하여 알고리즘에 의한 방법으로 접근하였으며, (그림 3)에서 MS-Zipf 분포의 확률 생성 알고리즘을 제시한다.

```
procedure mszipf(int N, int k, float θ)
begin
    make (N k) article groups by selecting k
        articles from the N article set;
    calculate CV of each article group;
    sort article groups in CV order;
    calculate the Zipf-like probability
        of each article group;
    for (i=0; i<N; i++)
    begin
        add all probabilites of article groups
            containing article i;
    end
    normalize probability of each article so
        that the total sum of all probabilities of
        articles becomes to 1;
end
```

(그림 3) MS-Zipf 확률 생성 알고리즘

3.2 복잡도를 낮춘 MS-Zipf 알고리즘

(그림 3)의 MS-Zipf 알고리즘은 $\binom{N}{k}$개의 기사 그룹을 생성하고 정렬하는 연산을 수행하기 때문에 $O\left(\left(\frac{N!}{k!(N-k)!}\right)^2\right)$의 시간 복잡도(time complexity)와 $O\left(\frac{N!}{k!(N-k)!}\right)$의 공간 복잡도(space complexity)를 가진다. NOD 응용에서 하루에 평균 500개 이상의 기사 데이터가 접근되고 사용자가 평균적으로 한번에 3개 이상의 기사 데이터에 접근하는 상황을 고려하면 상기의 알고리즘의 복잡도는 매우 크다고 할 수 있다. 따라서 제시된 알고리즘의 복잡도를 낮출 필요가 있다.

원래의 MS-Zipf 알고리즘에서는 모든 기사 그룹을 생성한 후에 정렬하는 연산을 진행하는데 이를 순위가 높은 기사를 기준으로 기사 그룹을 생성하면서 순위를 정하고 정렬함으로써 알고리즘의 복잡도를 낮출 수 있다. 여기서, 기사 그룹이 가질 수 있는 조합값(CV)의 종류와 각각의 조합값을 가지는 기사 그룹의 수를 미리 알 수 있어 기사 그룹의 순위를 쉽게 정할 수 있다. 예를 들면, N이 5이고 k가 3인 경우에, 생성될 수 있는 기사 그룹과 조합값은 <표 1>과 같다.

<표 1> N=5, k=3일 때에 기사 그룹과 조합값

CV	기사 그룹	개수	이전 CV의 누적 개수
6	(1,2,3)	1	0
7	(1,2,4)	1	1
8	(1,2,5) (1,3,4)	2	2
9	(1,3,5) (2,3,4)	2	4
10	(1,4,5) (2.3.5)	2	6
11	(2,4,5)	1	8
12	(3,4,5)	1	9

(그림 2)의 규칙(3)에 의하여 CV가 적은 (1,2,3)이 (1,2,4)보다 높은 순위를 갖도록 정하고, CV = 9인 경우에서 처럼 CV가 같은 경우에는 규칙(4)에 의하여 (1,3,5)가 (2,3,4)보다 높은 순위를 갖도록 정하는데, 이때의 순위값은 이전 CV까지의 누적 개수에 현재 CV의 기사 그룹의 생성 개수를 더하면 된다. 즉, 기사 그룹 (2,3,4)의 순위값은 이전 CV까지의 누적 개수 4와 현재 CV의 기사 그룹의 생성 개수 2를 더하여 6이 된다. 이와 같이 기사 그룹의 순위값을 산출하는 식은 (식 3)과 같으며, (그림 4)에서는 복잡도를 낮춘 MS-Zipf 알고리즘을 구체적으로 제시하고 있는데 $O\left(\frac{N!}{k!(N-k)!}\right)$의 시간 복잡도와 $O(kN)$의 공간 복잡도를 갖는다.

C_i의 순위 = 이전 CV의 누적개수 +
현재까지 생성된 CV의 기사그룹 수 (식 3)

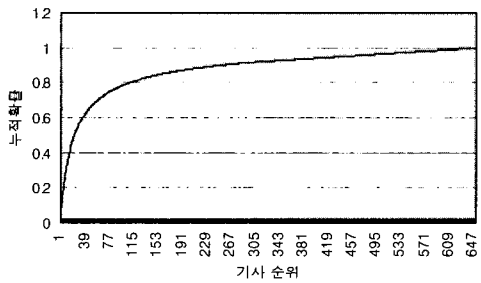
```
procedure modified_mszipf(int N, int k, float θ)
begin
    create CV table;
    M= ( N k );
    for (i=0;i<M;i++)
    begin
        create a article group C_i
        calculate CV of C_i;
        determine rank of C_i based on (Eq. 3);
        update CV table;
    end
end
```

(그림 4) 복잡도를 낮춘 MS-Zipf 알고리즘

3.3 MS-Zipf 근사 알고리즘

(그림 4)의 알고리즘은 (그림 3)에서 제시된 알고리즘에 비해 복잡도가 줄었으나, 여전히 NP-complete 문제로서 전체 기사의 수 N과 사용자의 기사 접근 수 k가 커질수록 계산량이 기하급수적으로 많아진다. 따라서 본 논문에서는 N의 값을 줄임으로써 전체 계산량을 줄이는 근사적인 알고리즘을 제시한다.

(그림 5)는 실측된 기사 데이터의 누적 접근 확률 분포를 나타낸 곡선으로서 기사의 순위값이 250 정도에서 누적확률이 0.9 정도가 되고, 이후의 누적 확률 분포 곡선은 거의 편평한 상태를 유지한다. 이것은 기사의 순위값이 250보다 큰 기사에 대해서는 접근 확률이 거의 변화가 없음을 의미한다. 따라서 전체 기사의 수 N에 대하여 MS-Zipf 알고리즘을 실행하는 대신에 0.9의 누적확률을 가지는 $N_{0.9}$에 대하여 MS-Zipf 확률을 구하고, 총 확률의 합이 0.9가 되도록 정규화 시키면 실제 접근 확률 분포에 근사한 MS-Zipf 확률 분포를 구할 수 있다. (그림 6)은 누적확률 0.9에 근사시키는 MS-Zipf 근사 알고리즘을 나타낸다.



(그림 5) 관측 데이터의 누적 접근 확률 분포

```
procedure approximated_mszipf(
    int N_0.9, int k, float θ)
begin
    initialize variables;
    /* create MS-Zipf probabilites for N_0.9
       articles */
    modified_mszipf( N_0.9, k, θ) ;
    normalize probabilites so that total sum of
       articles probabilities becomes to 0.9;
end
```

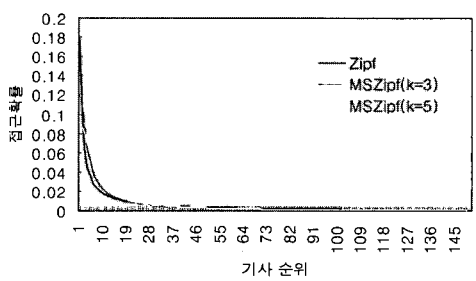
(그림 6) MS-Zipf 근사 알고리즘

3.4 매개변수 k와 θ의 추정

MS-Zipf 분포의 매개변수는 N(전체 기사의 수), k(사용자의 평균 접근 기사 수), θ(기사 그룹의 접근 편기성 정도)이며, 이들의 값에 의하여 MS-Zipf 분포의 모양이 좌우된다. 따라서, N에 대해서는 앞에서 제시한 근사적인 값을 사용한다면 실제 기사 접근 확률 분포에 가까운 MS-Zipf 분포를 구하기 위하여 k와 θ값을 추정하여야 한다. 본 논문에서는 실험을 통하여 k와 θ값을 추정하고 실측 데이터가 추정된 MS-Zipf 분포를 따르는지 여부를 χ^2 적합성 검정을 통하여 검정한다.

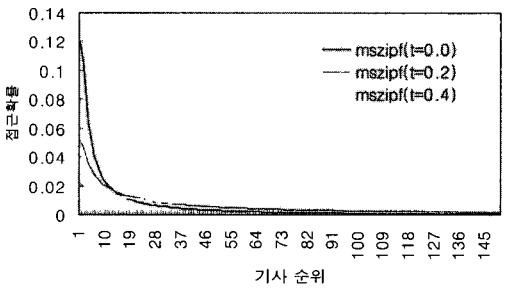
추정실험을 간단히 하기 위하여 접근 데이터가 비교적 적은 '일요일 로그 파일'을 실험 데이터로 사용하였으며, '일요일 로그 파일'에서의 $N_{0.9}$ 값은 150이었다. (그림 6)의 근사 알고리즘을 이용하여 N=150일 때 k와 θ값을 변화시키면서 MS-Zipf 분포의 변화를 측정하였다.

(그림 7)은 N=150, θ=0.0으로 하고, k값이 변할 때에 MS-Zipf 분포의 변화를 측정한 것으로 k값이 커질수록 MS-Zipf 분포의 상단 부분이 넓게 분산됨을 알 수 있다. 이는 사용자들이 한번에 k개의 기사를 접근하기 때문에 상대적으로 접근 확률이 상위 기사들에게 분산되는 것을 의미한다.



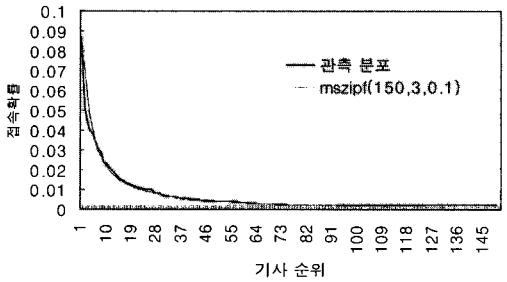
(그림 7) k값에 따른 MS-Zipf 분포 변화

(그림 8)은 N=150, k=3으로 하고, θ값이 변할 때에 MS-Zipf 분포의 변화를 측정한 것으로 θ값이 증가할수록 상위 부분의 확률이 하위 부분으로 분산됨을 알 수 있다, 이는 (그림 7)에서 k값이 증가함에 따라 하위 부분의 확률이 상위 부분으로 집중되는 것과는 상이한 현상을 보이는 것으로 k와 θ가 서로 상충적인 영향을 미치는 요인임을 의미한다. 따라서, 적정한 k와 θ값을 추정함으로써 실제 분포에 근사한 MS-Zipf 분포를 구할 수 있다.



(그림 8) θ값에 따른 MS-Zipf 분포 변화림

본 논문에서는 k와 θ값을 조절하면서 실제 기사 접근 확률 분포에 적합한 MS-Zipf 분포를 위한 최적의 값을 추정하였으며, 추정 결과로 k=3, θ=0.1일 때에 MS-Zipf 분포가 실제의 분포에 적합함을 (그림 9)를 통하여 알 수 있었다.



(그림 9) 관측 분포와 추정한 MS-Zipf 분포의 비교

또한, 실측 데이터가 추정된 MS-Zipf 분포를 따르는지의 여부를 χ^2 적합성 검정을 통하여 검정하였다[20]. <표 2>는 χ^2 검정 통계량을 구하기 위한 관측도수와 기대 도수를 나타낸다.

$$T = \sum_{i=1}^{c} \frac{(O_i - E_i)^2}{E_i} \qquad \text{(식 4)}$$

(식 4)를 이용한 계산한 검정 통계량은 $T = 9.5$로서, $\chi^2(0.05,11)$=19.68 보다 작아 귀무가설이 기각할 수 없다. 따라서, 실측 데이터는 추정된 MS-Zipf 분포를 따른다고 할 수 있다.

<표 2> χ^2 검정통계량을 위한 관측도수와 기대도수

계 급	관측도수(O_i)	기대도수(E_i)
1~2	66	72
3~5	59	70
6~10	67	65
11~20	75	71
21~30	45	42
31~50	28	30
51~65	25	25
66~80	19	19
81~95	15	15
96~110	15	13
111~130	20	16
131~150	20	12

4. LLBF 프리패칭 알고리즘

지금까지 NOD 응용의 사용자 접근 패턴의 하나인 기사 데이터의 인기도 모형을 살펴보았다. 이러한 인기도 모형은 멀티미디어 응용에서 효과적으로 미디어 데이터를 배치하는데 중요하게 사용되기도 한다. 이 절에서는 NOD 응용에서 또 다른 형태의 사용자의 접근 패턴으로서 기사 데이터의 생명주기 모형을 제시하고, 이러한 사용자 접근 패턴 모형을 바탕으로 CMNOD (Clustered Multimedia NOD) 서버의 성능을 향상시키는 LLBF 기사 프리패칭 알고리즘에 대하여 살펴본다.

4.1 CMNOD(Clustered Multimedia NOD) 서버 구조

기존의 멀티미디어 서버에 관한 연구에서는 고성능 병렬 컴퓨터를 기반으로 한 서버 구조나 여러 대의 독립적인 서버들을 연결한 분산 서버 구조 등을 제안하고 구현해왔다[17,18]. 이러한 서버들은 멀티미디어 응용이 요구하는 기본적인 조건인 대용량 저장 공간과 실시간 미디어 데이터 전송 등을 지원하나, 주문형 전자신문, 디지털 라이버러리, 그리고 가상현실 등과 같은 새로이 개발되는 멀티미디어 응용의 요구사항을 지원하기에는 한계가 있다[16].

NOD 응용 등과 같은 새로운 멀티미디어 응용의 요구사항을 간단히 살펴보면,

첫째, 수천 명 이상의 동시 사용자를 지원할 수 있는 확장성,

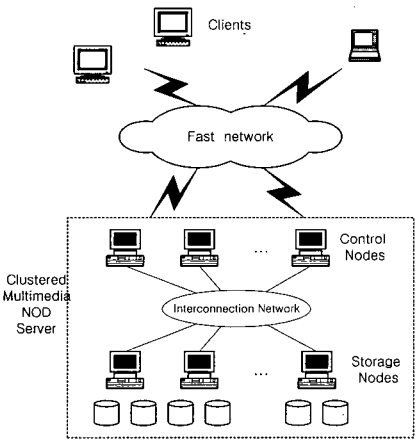
둘째, 텍스트, 이미지 그리고 비디오 등의 다양한 종류의 미디어에 대한 접근 및 전송,

셋째, 클라이언트의 다양한 대화식 데이터 접근,

넷째, 안정적이고 지속적인 서비스를 위한 고장 무결성,

다섯째, 대용량 서버 구성을 위한 가격대비 높은 성능

등이 있는데, 이러한 조건을 만족시키기 위해서는 저가형 서버들을 빠른 통신 네트워크로 연결하여 서로 상호작용 하도록 구성하는 클러스터드 서버 구조(clustered server architecture)가 적합하다[15,16]. 따라서, 본 논문에서 연구하는 대용량의 멀티미디어 NOD 서버는 (그림 10)과 같은 CMNOD(Clustered Multimedia NOD) 서버 구조를 가짐을 가정한다.



(그림 10) CMNOD(Clustered Multimedia NOD) 서버 구조

CMNOD 서버는 기본적으로 제어 서버(Control Server)와 저장 서버(Storage Server)로 구성되는데 제어 서버는 사용자 요구를 받아 처리하고 사용자에게 검색 데이터를 전송하는 기능을 수행하고 저장 서버는 미디어 데이터를 저장하고 검색하는 기능을 수행한다. 제어 서버와 저장 서버를 이루는 각 노드들은 빠른 통신 스위치 등으로 연결되어 노드간에 충분한 통신 대역폭을 지원하며, 제어 서버와 저장 서버의 적절한 상호작용을 통하여 사용자의 요구를 실시간으로 지원한다.

CMNOD 서버의 동작을 간단히 살펴보면, 클라이언트는 외부 네트워크을 통하여 제어 서버에게 데이터 접근을 요구하고, 제어 서버는 사용자 요구에 대하여 승인 제어(admission control)를 거친 후에 적절한 저장 서버들에게 사용자 요구를 분배한다. 그러면, 저장 서버는 제어 서버에서 전송된 요구에 따라 저장된 데이터를 검색하여 제어 서버에게 넘겨 주고, 제어 서버는 데이터를 클라이언트에게 실시간으로 전송한다. 이때, 연속 미디어 데이터에 대해서는 저장 서버의 데이터 검색과 제어 서버의 데이터 전송이 서비스가 종료될 때까지 주기적으로 발생하여 streaming I/O를 지원한다.

현재 클러스터드 서버의 제어 서버와 저장 서버 구조에 대하여 많은 연구가 진행 중이며, 몇몇의 연구에서는 제어 서버가 접근 빈도가 높은 데이터를 미리 프리패칭함으로써 전체 서버의 성능을 크게 향상시킬 수 있음을 지적하고 있다[15]. 따라서, 본 논문에서는 NOD 기사 데이터의 사용자 접근 패턴을 바탕으로 한 프리패칭 알고리즘을 제안함으로써 CMNOD 서버의 성능을 향상시키고자 한다.

4.2 NOD 기사 데이터의 생명주기 모형

NOD 기사 데이터는 하루 중에 수시로 생성되고 사용자는 새로운 기사를 선호하기 때문에 시간에 따라서 인기 있는 기사들이 변하게 된다.

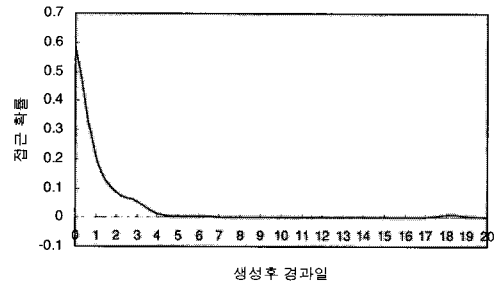
실제 전자신문의 로그파일을 분석하여 생성 후의 경과 시간에 따른 인기도 변화와 생성 후의 경과 날짜에 따른 인기도 변화를 조사하였다. (그림 11)은 날짜에 따른 인기도 변화를 나타내는데, 평균적으로 3~4일이 지나면 거의 접근 가능성이 희박해져 기사 데이터의 생명주기가 VOD의 비디오 데이터에 비하여 매우 짧음을 알 수 있다. 그리고 (그림 12)는 시간에 따른 인기도 변화를 나타내는데, 평균적으로 생성 직후에 접근 확률이 급격히 증가했다가 서서히 감소하는 패턴을 보이고 있으며, 이러한 사용자 접근 패턴은 접근 빈도가 어느 정도 높은 기사에 해당한다. 그러나 접근 빈도가 낮은 기사의 경우에는 생성 직후의 접근 확률에서 계속적으로 감소하는 패턴을 보인다.

그림에서 알 수 있듯이 NOD 기사 데이터의 생명주기에 대해 하나의 수학적 모형을 정의하는 것은 어렵지만 몇 가지 주요한 사실은 알 수 있다.

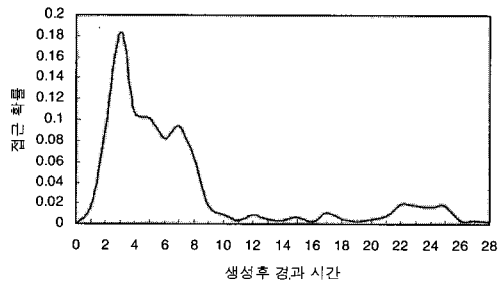
첫째, NOD 기사 데이터는 생명주기가 비교적 긴 VOD의 비디오 데이터나 웹 문서 등에 비하여 매우 짧다.

둘째, NOD 기사 데이터에 대한 사용자 접근은 생성 직후의 몇 시간동안 집중된 후에 서서히 감소한다.

즉, 기사 데이터의 인기도 변화주기가 짧다.



(그림 11) 생성 후의 경과 날짜에 따른 인기도 변화



(그림 12) 생성 후의 경과 시간에 따른 인기도 변화

이와 같은 사실을 바탕으로 기사 프리패칭 정책을 세우고자 할 때는 기본적으로 당일 기사가 접근 확률이 높으므로 당일 기사를 프리패칭하고 당일 기사 중에서도 생성된 직후의 기사를 프리패칭하는 것이 효과적임을 알 수 있다.

4.3 LBF(Life-cycle Based Frequency) : 프리패칭의 재배치 척도

NOD 기사 데이터는 다른 데이터와는 달리 그 생명주기가 뚜렷하고 짧다. 즉, 기사 데이터는 생성된 직후 짧은 시간동안 급격히 인기도가 증가했다가 시간이 흐를수록 서서히 감소하여 인기도 변화가 심하다. 그리고 많은 수의 기사 데이터가 동시에 높은 인기도를 유지한다. 이에 단순히 접근 시간이나 접근 빈도수를 기반으로 한 프리패칭은 제한된 자원환경에서 빈번한 데이터 재배치를 발생시켜 재배치 비용으로 인한 시스템 성능이 저하될 수 있음을 실험을 통하여 알 수 있었다 (그림 18).

본 논문에서는 주기적으로 기사 데이터의 생명주기 정보를 이용하여 접근 가능성이 높은 기사를 예측하고 프리패칭함으로써 불필요한 재배치를 줄일 수 있는 프리패칭 알고리즘을 제안한다.

기사 프리패칭 알고리즘에서 중요한 요소는 프리패칭할 기사를 결정하는 방법인데, 제안된 알고리즘에서는 기사 데이터의 생명주기를 이용하여 프리패칭할 기사를 결정하며, 이러한 기사 데이터의 생명주기에 대한 척도로서는 시간 윈도우(time window)를 바탕으로 한 접근 빈도수의 변화량을 이용한다. 접근 빈도수의 증가량이 큰 기사는 생성된지 얼마 되지 않아 접근 가능성이 높음을 나타내고, 접근 빈도수가 감소하는 기사들은 생성 후에 어느 정도 시간이 경과하여 인기도가 떨어지고 있음을 나타내며, 변화가 거의 없는 기사들은 생성된 지 오래된 기사임을 나타낸다.

따라서, 시간 윈도우별로 접근 빈도수와 증감량 정보를 이용하여 앞으로 인기도가 상승할 기사를 예측하고 프리패칭함으로써 기사에 대한 적중률(hit ratio)을 높일 수 있을 뿐만 아니라 시간 윈도우 단위로 프리패칭을 제한함으로써 프리패칭 횟수를 줄일 수 있다.

기사의 생명주기를 기반으로 한 프리패칭 알고리즘은 시간 윈도우별로 현재 윈도우의 접근 빈도수 f_i와 다음 윈도우에서의 예상 증감량 r_i의 합인 'LBF(Life-cycle Based Frequency)'을 구하고(식 5), 이를 다음 윈도우에서의 접근 빈도수의 예측치로 사용하여 재배치할 기사를 결정한다.

$$LBF_i = f_i + r_i \qquad \text{(식 5)}$$

$$r_i = sf \times (f_i - f_{i-1}) + (1 - sf) \times r_{i-1} \qquad \text{(식 6)}$$

(식 6)은 빈도수 증감량에 대한 가중평균을 구한 것인데, sf는 평활 계수(smoothing factor)로서 현재 윈도우의 빈도수 증감량에 대한 가중치이다. 평활 계수가 클수록 r_i는 현재 윈도우의 빈도수 변화량에 민감하게 반응하고, 적을수록 둔감하게 반응한다.

4.4 LLBF(Largest LBF) 프리패칭 알고리즘

NOD 기사 데이터의 프리패칭 알고리즘은 짧은 주기의 인기도 변화로 인한 빈번한 기사 재배치 문제를 고려하여야 한다.

본 논문에서 제안하는 프리패칭 알고리즘은 기사 데이터에 대하여 인기도가 상승할 때에 프리패칭하였다가 인기도가 하강할 때까지 재배치 하지 않음을 보장

함으로써 단순히 접근 최근도(access recency)나 접근 빈도수(access frequency) 만을 고려한 프리패칭 알고리즘보다 재배치 횟수를 줄이려 하였으며, 기사 데이터의 최적의 재배치 시점을 결정하기 위하여 다음 시간 윈도우의 예측 접근 빈도수인 LBF를 사용한다. 즉, 제안된 프리패칭 알고리즘은 각각의 시간 윈도우마다 가장 LBF가 높은(Largest LBF) 상위 H개의 기사를 선택하여 프리패칭한다.

(그림 13)은 제안한 LLBF(Largest LBF) 프리패칭 알고리즘을 나타낸 것이다.

```
procedure LLBF_prefetching()
begin
    if (new window)
    begin
        for (all articles in new window)
        begin
        r_i = sf×(f_i − f_{i−1}) + (1 − sf)×r_{i−1};
            LBF_i = f_i + r_i;
        end
    end
    select H articles with largest LBF;
    update the cache with the new H articles;
end
```

(그림 13) LLBF 프리패칭 알고리즘

5. 모의 실험

5.1 모의 실험 환경

LLBF프리패칭 알고리즘에 대한 모의 실험은 Solaris 2.5.1이 탑재된 Sun Ultra Sparc 30 시스템에서 C 언어로 작성된 모의 실험 프로그램을 통하여 실시하였다.

<표 3>은 모의실험에서 사용한 매개변수이며 모의실험의 결과를 이용하여 LLBF 알고리즘에 대해 성능을 분석하고 다른 프리패칭 알고리즘과 성능을 비교, 분석하였다.

1998년 6월에서 12월까지의 전자신문의 로그파일을 통계 분석하여 기본적인 통계치를 얻었다. 하루에 생성되는 기사 수는 200~300개, 하루에 접근되는 기사 수는 500~700개, 하루의 총 접근 수는 4000~5000회 정도였다. 이러한 통계량을 토대로 몇몇 모의 실험의 매개변수 값을 결정하였는데, 평균 사용자 도착시간을 20초로, 그리고 평균 기사 발생 시간을 5분으로 설정하였다. 평활 계수(smoothing factor) 값은 모의 실험에서 그 값을 조정하면서 가장 높은 성능을 내는 값으로 정하였고 기사 데이터에 대한 사용자의 접근 패턴은 MS-Zipf 분포를 따르도록 하였다.

〈표 3〉 시스템 성능을 계산하기 위한 매개변수

매개 변수	표 기
저장 서버의 디스크 대역폭	D_{BW}
제어 서버의 캐쉬 메모리 대역폭	C_{BW}
사용자 요구 대역폭	U_{BW}
CMNOD 서버의 처리율	C_{user}
평균 사용자 도착 시간 간격	I_{user}
평균 기사 발생 시간 간격	$I_{article}$
시간 윈도우 크기	I_{window}
기사 프리패칭 양(%)	C_{amount}
평활 계수	sf

5.2 LLBF 프리패칭 알고리즘의 성능 평가

CMNOD 서버 구조에서 제어 서버의 프리패칭은 저장 서버의 접근 횟수를 줄이고 빠른 캐쉬메모리에서 서비스함으로써 사용자 요구 처리량을 늘릴 수 있다. 따라서 모의 실험에서는 LLBF 프리패칭 알고리즘이 서버의 성능에 미치는 영향을 나타내기 위하여 단위시간당 처리율을 측정하였으며 <표 4>와 같은 매개변수 값을 사용하였다.

〈표 4〉 모의 실험에서 사용된 매개변수 값

실험 매개변수	값
데이터 블록의 크기	256 Kbytes
사용자 요구 대역폭	1.5 MB/sec
저장 서버의 대역폭	10 MB/sec
제어 서버의 캐쉬메모리 대역폭	100 MB/sec
평균 사용자 요구 도착시간 간격	20초
평균 기사 발생 시간 간격	5분
평활 계수	0.5
시간 윈도우 크기	입력변수
기사 프리패칭 양	입력변수

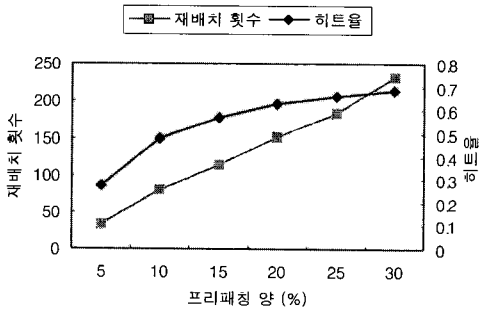
모의 실험에 사용된 매개변수 값들 중에서 데이터 블록의 크기는 멀티미디어 데이터에 대한 검색임을 가정하여 일반적으로 적용하는 256 Kbytes로 설정하였으며, 사용자 요구 대역폭은 MPEG-1 비디오 스트림의 요구 대역폭으로 설정하였다. 그리고 캐쉬메모리의 대역폭과 디스크 대역폭은 기존의 연구에서 적용한 값

으로 가정하였는데[14], 캐쉬메모리의 대역폭이 디스크 대역폭에 대해 10배 정도 큼을 가정하고 서버 노드들을 연결하는 네트워크로 인한 지연은 무시하였다.

본 논문에서는 서비스 대역폭을 요구 대역폭으로 나누어 처리율을 추정하는 방법을 적용하였다. 제어 서버에서 인기도가 높은 기사를 프리패칭함으로써 사용자의 요구가 프리패칭 데이터에 적중하는 경우에 제어 서버의 프리패칭 캐쉬에서 직접 서비스되고, 적중하지 않을 경우에 저장 서버에서 서비스되어진다. 따라서 CMNOD 서버의 처리율은 프리패칭 캐쉬의 적중률(hit ratio)에 의하여 결정되고 (식 7)과 같이 추정할 수 있다.

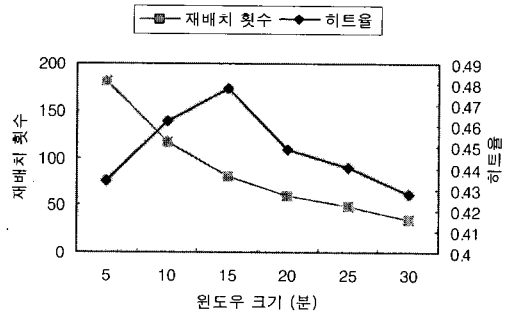
$$C_{user} = \frac{C_{BW}}{U_{BW}} \times hit\_ratio + \frac{D_{BW}}{U_{BW}} \times (1 - hit\_ratio) \qquad \text{(식 7)}$$

(그림 14)는 LLBF 알고리즘에서 시간 윈도우 크기를 15분으로 하고 기사 프리패칭 양을 변화시켰을 때에 적중률 및 재배치 횟수의 추이를 나타낸 것이다. 프리패칭 양이 많아지면 프리패칭하는 기사 수가 절대적으로 많아져 적중률 및 재배치 횟수가 거의 비례적으로 증가함을 알 수 있다.



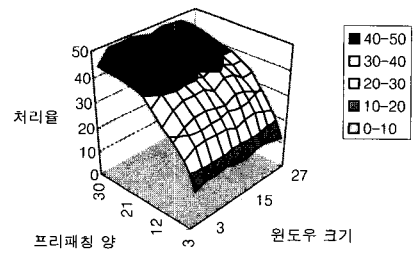
(그림 14) 프리패칭 양에 따른 LLBF 알고리즘의 성능

(그림 15)는 LLBF 알고리즘에서 프리패칭 양을 10%로 하고 시간 윈도우의 크기를 변화시켰을 때에 적중률과 재배치 횟수의 추이를 나타낸 것이다. 시간 윈도우가 일정한 크기 이상이 되면 적중률이 증가하다가 감소함을 볼 수 있는데, 이는 시간 윈도우의 크기가 기사 데이터의 인기도 변화주기보다 커서 인기도 변화를 적절히 반영하지 못하기 때문이다. 따라서, 시간 윈도우 크기는 기사의 발생주기와 인기도 변화주기를 고려하여 적절하게 설정되어야 한다.



(그림 15) 시간 윈도우 크기에 따른 LLBF 알고리즘의 성능

(그림 16)은 LLBF 알고리즘에서 프리패칭 양과 시간 윈도우의 크기를 동시에 변화시켰을 때에 CMNOD 서버의 처리율에 대한 추이를 나타낸 것이다. 서버의 처리율은 프리패칭 양이 커질수록 증가하나 시간 윈도우의 크기가 커질수록 일정한 수준을 유지하다가 다소 떨어진다. (그림 15)의 결과에 의하면 시간 윈도우가 커질수록 처리율이 떨어져야 하는데, 프리패칭 양이 일정한 크기 이상이 되면 재배치 횟수는 늘어나나 적중률을 보상하여 서버의 처리율도 증가한 후에 거의 일정한 수준을 유지하게 된다. 그리고 그래프의 굴곡 형태는 기사 데이터의 발생주기와 모의실험에서 적용한 시간 윈도우의 크기 값에 의하여 발생하는 것으로 주기적이면서 반복적으로 처리율이 다소 떨어지는 현상을 보인다.



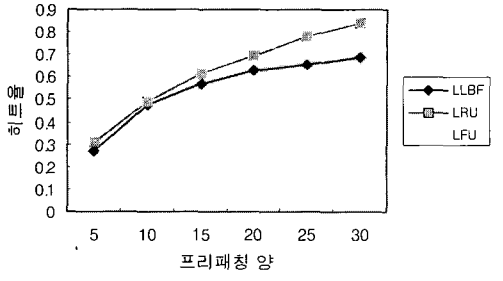
(그림 16) LLBF 알고리즘을 이용한 CMNOD 서버의 처리율

5.3 LLBF 프리패칭 알고리즘의 재배치 횟수 비교

NOD 기사 데이터는 연속 매체와 비연속 매체를 포함한다. 따라서 VOD 데이터 보다는 크기가 작지만, 5~10분 정도의 뉴스 비디오 클립을 가지는 기사의 경우 수십 Mbytes에 달하므로 이를 재배치 할 때의 비용은

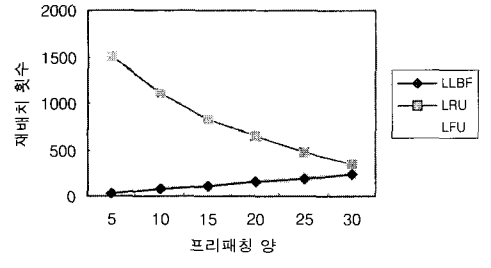
매우 크다. 따라서 프리패칭 정책은 재배치 횟수를 가능한 줄여야 한다. 상기의 LLBF 알고리즘은 주기적으로 기사의 생명주기를 이용함으로써 불필요한 재배치를 줄인다. 이를 보이기 위해 대표적인 재배치 알고리즘 LRU(Least Recently Used)와 LFU(Least Frequently Used) 알고리즘과 비교한다[19].

(그림 17)은 LLBF 알고리즘과 다른 알고리즘들의 적중률을 비교하여 나타냈는데, LLBF는 LFU 보다는 적중률이 좋았으나, LRU 보다는 다소 떨어짐을 알 수 있다.



(그림 17) 다른 알고리즘과의 적중률 비교

(그림 18)은 재배치 횟수를 비교하여 나타낸 것으로, 캐쉬 크기가 작을 때에 LLFU는 LRU나 LFU 보다 재배치 횟수가 훨씬 작으나 캐쉬 크기가 커질수록 차이가 줄어들고 있다. 현실적으로 많은 기사를 프리패칭하기 힘들기 때문에 하루에 접근되는 기사의 약 10%을 프리패칭한다고 했을 경우, LLBF는 LRU와 거의 같은 적중률을 가지면서 LRU 보다는 6배 이상, LFU 보다는 8배 이상의 재배치 비용을 줄일 수 있다.



(그림 18) 다른 알고리즘과의 재배치 횟수 비교

6. 결 론

NOD 멀티미디어 응용에서 다루는 기사 데이터는 기존의 VOD 응용의 비디오 데이터와는 달리 미디어 종류의 다양성, 데이터의 크기, 그리고 사용자 요구 대역폭 등 기본적인 특성에서 차이점을 보일 뿐만 아니라 이러한 특성으로 인하여 사용자의 접근 패턴도 달라져 새로운 기사 데이터의 인기도 모형이나 생명주기 모형을 갖는다.

본 논문에서는 NOD 기사 데이터에 대한 사용자 접근 패턴을 분석하여 NOD 서버 시스템의 성능을 향상시키는데 응용하고자 하였다.

먼저, 현재 운영중인 전자신문의 로그 파일에 대한 통계분석을 통하여 NOD 기사 데이터에 대한 사용자 접근 패턴이 기존의 멀티미디어 응용에서 널리 사용하던 인기도 모형인 Zipf 분포와는 다른 형태를 가짐을 보였다. 이에 기사 데이터의 새로운 인기도 모형으로서 'MS-Zipf 분포'를 제안하고 알고리즘적인 접근법으로 MS-Zipf 확률 생성 알고리즘을 제시하였으며, χ^2 적합성 검정을 통하여 MS-Zipf 분포의 적합성을 검정하였다. 또한, NOD 기사 데이터에 대한 생명주기 모형의 유형을 분석하였는데, 평균적으로 짧은 주기로 급격한 인기도 변화를 보였다.

그리고, NOD 기사 데이터의 사용자 접근 패턴에 대한 정보를 이용하여 접근 가능성이 높은 기사 데이터를 예측하여 프리패칭함으로써 CMNOD(Clustered Multimedia NOD) 서버 시스템의 성능을 향상시키는 LLBF 프리패칭 알고리즘을 제안하고 성능을 분석하였다. LLBF 알고리즘은 제한된 자원환경에서 LRU 알고리즘과 비슷한 적중률을 가지면서 훨씬 적은 재배치 횟수를 보여 재배치 비용의 부담을 줄일 수 있었다.

향후 연구 계획으로는 MS-Zipf 분포에 대한 근사 모형으로서 수학적 모형에 대하여 연구를 진행할 것이며, 현재 구현중인 CMNOD 서버 시스템에서 NOD 기사 데이터에 대한 사용자 접근 패턴을 고려한 승인 제어(admission control) 알고리즘을 제시하고 구현할 예정이다.

참 고 문 헌

[1] 이주경, 박용운, 김영주, 정기동, "NOD 데이타를 위한 배치 방법", 한국정보과학회 가을학술발표논문집, 24권 2호, pp.55-58, 1997.

[2] Robert Flynn and William Tetzlaff, "Disk Striping and Block Replication Algorithms for Video

File Servers," Proc. of IEEE Conference on Multimedia Computing and Systems, pp.590-597, 1996.

[3] T. D. C. Little and V. Venkatesh, "Popularity-based assignment of movies to storage devices in a video-on-demand system," ACM Multimedia Systems, 1994.

[4] Scott A. Barnett, Gary J. Anido, H. W. Beadle, "Caching Polices in a Distributed Video-on-Demand System," Proc. of Australian Telecommunication Networks and Applications Conference, 1995.

[5] Carsten Griwodz, Michael Bar, Lars C. Wolf, "Long-term Movie Popularity Models in Video-on-Demand Systems or The Life of an on-Demand Movie," Proc. of the fifth ACM International Multimedia Conference, pp.349-357, 1997.

[6] J. T. Robinson and N. V. Devarakonda, "Data Cache Management Using Frequency-based Replacement," Proc. of ACM SIGMETRICS Conference, pp.134-142, 1990.

[7] E. J. O'Neil, P. E. O'Neil, and G. Weikum, "The LRU-k page replacement algorithm for database disk buffering," Proc. of International Conference on Management of Data, 1993.

[8] D. Lee, J. Choi, J. Kim, S. H. Min, Y. Cho, and C. S. Kim, "LRFU Replacement Policy : A spectrum of block replacement policies," Seoul National University Technical Report, Korea, 1996.

[9] B. Ozdel, R. Rastogi, and A. Silberschatz, "Buffer Replacement Algorithms for Multimedia Storage Systems," Proc. of International Conference on Multimedia Computing and System, pp.172-180, 1996.

[10] A. Dan, D. Dias, R. Mukherjee, D. Sitaram, and R. Tewari, "Buffering and Caching in Large-Scale Video Servers," Proc. of IEEE COMPCON, 1995

[11] I. Tatarinov, V. Soloview, A. Rousskov, "Static Caching in Web Servers," Proc. of the 6th International Conference on Computer Communications and Networks, 1997.

[12] James E. Pitkow, Margaret M. Recker, "A Simple Yet Robust Caching Algorithm based on Dynamic Access Patterns," Proc. of the 2nd International Multimedia Conference, 1997.

[13] A. Bestavros and et.al., "Application-level Document Caching in the Internet," Proc. of Workshop on Services and Distributed Environments, Jun, 1995.

[14] Renu Tewari, Harrick Vin, Asit Dan and Dinkar Sitaram, "Resource Based Caching for Web Servers," Proc. of SPIE/ACM Conference on Multimedia Computing and Networking(MMCN), San Jose, January 1998.

[15] R. Tewari, R. Mukherjee and D. Dias, "Design and Performance Tradeoffs in Clustered Video Servers," Proc. of International Conference on Multimedia Computing and Systems, pp.144-150, 1996.

[16] Frank Fabbrocino and Renato Santos, "An Implicitly Scalable, Fully Interactive Multimedia Storage Server," 2th International Workshop on Distributed Interactive Simulation and Real Time Applications, 1998.

[17] A. Laursen, J. Olkin, and M. Porter, "Oracle media server : Providing consumer based interactive access to multimedia data," Proc. of ACM SIGMOD 94, pp.194-201, 1994.

[18] C. Bernhardt and E. Biersack, "The server array : A scalable video server architecture," in *High Speed Networking for Multimedia Applications*, Kluwer Publishers, Mar, 1996.

[19] J. Yang and Raymond T. Ng, "An analysis of buffer sharing and prefetching techniques for multimedia systems," Multimedia Systems, 1996.

[20] 김우철 외, "현대통계학", 영지문화사, 1990.

김 영 주

e-mail : yjkim@melon.cs.pusan.ac.kr

1988년 부산대학교 계산통계학과 졸업(이학사)

1990년 부산대학교 대학원 계산통계학과 졸업(이학석사)

1990년~1995년 큐닉스컴퓨터 응용시스템 연구소 근무

1995년~1997년 부산대학교 대학원 전자계산학과 박사 과정 수료
관심분야 : 멀티미디어, 병렬파일시스템, 분산처리

최 태 욱

e-mail : tuchoi@melon.cs.pusan.ac.kr
1997년 동의대학교 전산통계학과 졸업(이학사)
1999년 부산대학교 대학원 전자계산학과 졸업(이학석사)
1999년~현재 부산대학교 대학원 멀티미디어 협동과정학과 박사과정
관심분야 : 멀티미디어, 병렬처리, 분산처리

정 기 동

e-mail : kdchung@melon.cs.pusan.ac.kr
1973년 서울대학교 졸업(학사)
1975년 서울대학교 대학원 졸업(석사)
1986년 서울대학교 대학원 계산통계학과 졸업(이학박사)
1990년~1991년 MIT, South Carolina 대학 교환 교수
1995년~1997년 부산대학교 전자계산소 소장
1978년~현재 부산대학교 전자계산학과 교수
1997년~현재 부산대학교 대학원 멀티미디어 협동과정학과 교수
관심분야 : 병렬처리, 멀티미디어