

데이터 방송에서 요청확률을 고려한 데이터 스케줄링 알고리즘

민재호[†], 황주연^{**}, 백두원^{***}

요 약

비대칭 통신환경에서는 서버가 사용자에게 데이터를 전송하는 방법으로 데이터 방송을 사용한다. 이 방법은 서버에서 사용자에게 필요한 모든 데이터를 주기적으로 반복해서 전송하고 사용자는 필요한 데이터가 방송되면 그 데이터를 사용한다. 그러므로 사용자는 필요한 데이터가 방송될 때까지 기다려야한다. 이러한 사용자의 대기시간은 줄여야만 한다. 대기시간을 줄이기 위하여 서버에서는 각각의 데이터의 순서를 스케줄링하여 방송해야 한다. 본 논문에서는 사용자가 각각의 데이터를 요청할 확률, 데이터의 크기와 데이터의 최적주기를 이용하여 데이터를 스케줄링하는 방법을 제안하고 이 방법들의 효용성을 검증했다. 이를 위해 제안한 방법으로 데이터를 스케줄링하여 사용자의 평균대기시간을 구하는 실험을 하였다. 실험 결과 빈도만 고려한 방법에 비해 약 13%의 성능이 증가되었다.

A Data Scheduling Algorithm Considering the Request Probability for Data Broadcasting

Jae-Ho Min[†], Jooyeon Hwang^{**}, Doowon Paik^{***}

ABSTRACT

Data broadcasting is a mechanism to transmit information to clients in simplex transmission environments. In this mechanism, server transmits to repeat all information to all users and users use a necessary data when the data is transmitted. Therefore users are waiting the necessary data until the data is transmitted. It is important to reduce user's waiting time. Server schedules the data to reduce waiting time. In this paper, we study efficient data scheduling algorithm. The result shows that the performance of the proposed method is increased 13 percent compared to the method considering only request frequency.

Key words: Data Broadcasting(데이터 방송), Simplex Transmission(비대칭 통신), Scheduling(스케줄링)

1. 서 론

최근 방송은 아날로그방송에서 디지털 방송으로 급속히 전환되어 가고 있다. 디지털 방송의 특징 중 하나는 음성이나 영상 이외에 데이터를 보낼 수 있다

는 것이다[1]. 하지만 대부분의 데이터방송 환경은 서버가 다수의 사용자에게 데이터를 일방적으로 방송하는 단방향 환경인 경우가 많다. 단방향 방송 환경인 경우 사용자는 필요한 데이터가 있을 때 서버에 데이터 요청을 할 수 없고, 서버는 사용자가 어떤 데

※ 교신저자(Corresponding Author): 백두원, 주소: 서울시 동작구 상도5동 숭실대학교 정보과학관 622호(156-743), 전화: 02)820-0916, FAX: 02)822-3622, E-mail: ssudlab@gmail.com

접수일: 2007년 11월 12일, 완료일: 2008년 1월 8일

[†] 숭실대학교 미디어학과 석사

(E-mail: morehuman@hanmail.net)

^{**} 정회원, 숭실대학교 대학원 미디어학부 박사과정

(E-mail: zooyouny@ssu.ac.kr)

^{***} 정회원, 숭실대학교 미디어학과 부교수

※ 본 연구는 숭실대학교 교내 연구비 지원으로 수행되었음.

이터를 필요로 하는지 알 수 없다. 이 문제를 해결하기 위해 단방향 데이터방송 환경에서는 데이터를 주기적으로 반복하여 보내는 object carousel이라는 방식을 이용한다. object carousel이란 MPEG-2의 DSM-CC에서 제공하는 파일 전송 프로토콜로써 서버는 object carousel을 이용하여 모든 데이터를 주기적으로 반복하여 불특정 다수에게 전송한다[2,3]. 그러면 사용자는 필요한 데이터가 전송될 때까지 대기하고 있다가 원하는 데이터가 도착하였을 때 다운로드 하여 사용할 수 있다.

object carousel에서 전송될 데이터는 차례로 전송된다. 서버는 모든 데이터를 전송한 후에 다시 처음부터 데이터를 차례로 재전송한다. 이 데이터는 Transport Stream에 삽입되어 전송되는데, 수신기가 전송되어진 Transport Stream의 중간 부분부터 수신하거나 전송과정 중에 에러가 발생하여 필요한 데이터의 일부를 읽지 못하면 수신기는 해당하는 데이터가 다시 전송되어 올 때까지 대기하여야 한다. 그러나 다시 전송하기 까지 주기가 길면 수신기의 대기시간이 길어져 사용자가 그 데이터를 다운로드 하는데 필요한 시간이 증가한다. 이 문제를 해결하기 위해서 서버는 사용자가 자주 요청하는 데이터를 다른 데이터에 비해 더 빈번히 전송해야 한다.

본 논문에서는 사용자가 서버에게 필요한 데이터를 요청할 수 없는 단방향 네트워크 환경에서 사용자의 대기시간을 최소화하는 데이터 스케줄링 방법을 제안하고 이 방법을 통해 만든 스케줄을 실험을 통해 검증하였다. 데이터를 스케줄 할 때 각 데이터의 전송 주기가 중요하다. 따라서 각 데이터의 적당한 전송 주기에 맞추어 데이터를 스케줄링해야 한다[4]. 그런데 크기가 큰 데이터를 전송하면 크기가 작은 데이터들은 크기가 큰 데이터가 전송 될 때까지 전송 되지 못한다. 만약 크기가 작은 데이터들의 주기가 밀리게 된 경우에 사용자가 크기가 작은 데이터를 필요로 하면 오랫동안 기다려야 하므로 사용자 대기시간이 커진다. 본 논문에서는 이러한 문제를 해결하기 위해서 다른 데이터의 주기를 해치는 데이터의 전송 주기를 길게 하여 전체적인 사용자 대기시간을 최소화 하였다.

2. 관련연구

2.1 비실시간 방법

[5,6]에서는 크기가 같은 데이터들의 이론적 최적

주기를 계산하였다. 그리고 이론적 최적주기를 이용하여 평균대기시간의 이론적 최적 값을 구하는 식을 증명하였다. 이 연구에서는 사용자 대기시간을 계산하여 이 값이 이론적 최적 대기시간임을 증명하였다. 그리고 일정한 길이의 주기를 스케줄링하여 반복적으로 전송하는 알고리즘을 제안하였다. 이 알고리즘은 다양한 길이의 주기를 갖는 스케줄을 만들고 각 주기의 평균대기시간을 계산하여 평균대기시간이 최소가 되는 주기를 선택하여 전송하는 방법이다. 따라서 다양한 길이의 주기를 생성하기 위해 많은 시간이 필요하고 실시간으로 스케줄을 생성하지 못하는 단점이 있다.

2.2 실시간 방법

본 논문에서는 크기가 다른 데이터들을 실시간으로 스케줄링하는 방법을 제안한다. 기존에도 크기가 다른 데이터를 스케줄링하는 방법에 대한 연구가 있었다. [7]에서는 크기가 다른 데이터들의 이론적 최적주기를 계산하였고 이 주기를 이용하여 평균대기시간의 이론적 최적 값을 구하는 식을 증명하였다.

[8]에서 제안한 방법은 데이터의 최적전송주기 s_i 와 지금 이전에 데이터가 전송된 횟수 n_i 를 이용하여 각 데이터의 빈도를 유지하는 방법이다. 여기에서 최적전송주기는 [7]에서 증명된 최적전송주기를 구하는 식을 사용하여 구한다. 데이터가 요청될 확률 p_i 와 데이터의 크기 l_i 를 고려해서 각 데이터의 최적의 전송주기를 구할 수 있다. 데이터의 전송주기는 데이터가 요청될 확률에 반비례하고 데이터의 크기에 비례해야 한다. 데이터의 총 개수를 M 이라고 한다면 최적주기를 구하는 식은 식 (2.1)과 같이 기술할 수 있다.

$$s_i = \left(\sum_{j=1}^M \sqrt{p_j l_j} \right) \sqrt{\frac{l_i}{p_i}} \quad (2.1)$$

알고리즘

Step 1 : (2.1)을 이용하여 모든 데이터 i 의 최적주기 s_i 를 구한다.

현재 시간 T 는 0으로 초기화한다.

모든 n_i 는 0으로 초기화한다.

Step 2 : $n_i \cdot s_i \leq T$ 인 데이터들의 집합 S 를 구한다.

$$S = \{i | n_i \cdot s_i \leq T, 1 \leq i \leq M\}$$

Step 3 : S에 속하는 데이터들 중에서 $(n_i + 1) \cdot s_i$ 값이 최소인 데이터 j 를 찾는다.

Step 4 : T에서 데이터 j 를 전송한다.

$$n_j = n_j + 1$$

$$T = T + l_j$$

본 연구에서는 [8]에서 제안된 방법의 단점인 크기가 큰 데이터에 의해서 다른 데이터의 주기가 밀려서 전체 평균 대기시간이 증가하는 것을 보완하는 방법을 제안한다.

3. 데이터 스케줄링 기법

본 논문에서 제안한 데이터 스케줄링 방법은 데이터의 최적전송주기와 데이터의 길이, 데이터가 요청될 확률을 고려한 방법이다. 이 방법은 [8]에서 제안한 방법처럼 각 데이터의 빈도를 유지하지만 만약 임의의 데이터가 다른 데이터의 주기를 어긋나게 한다면 그 데이터의 빈도를 감소시키는 방법이다. 데이터의 크기와 데이터가 요청될 확률이 정해지면 (2.1)을 사용하여 각 데이터의 최적주기(s_i)를 구한다. 현재시간까지 데이터의 최적 전송빈도보다 실제로 데이터가 전송된 빈도가 낮은 데이터들의 집합 S를 구한다. 그리고 집합 S에 포함된 데이터들 중에서 현재시간에서 전송한다면 최적 전송빈도에 비해 전송 빈도가 가장 낮은 데이터를 선택한다. 만약 앞에서 선택된 데이터의 크기가 다른 데이터의 최적주기보다 크다면 선택된 데이터의 크기보다 최적주기가 작고 선택된 데이터의 확률보다 확률이 큰 데이터들의 집합 D를 구한다. 여기에서 전송하려는 데이터의 크기가 다른 데이터의 최적주기보다 크다면 데이터의 크기가 큰 데이터를 전송하면 데이터의 최적주기가 작은 데이터들은 최적주기보다 더 밀리게 된다. 이런 경우에 최적주기가 작은 데이터들의 평균대기시간이 증가해서 전체평균대기시간이 증가하게 된다. 그래서 전송하려고 한 데이터의 크기가 다른 데이터들의 최적주기보다 크다면 다른 데이터를 선택하여 전송한다. 여기에서 데이터 i 가 현재시간에 전송될 경우의 전체 평균 대기시간 TAT_i 가 최소가 되는 데이터를 전송한다. 그리고 전송된 데이터의 크기만큼 크기가 큰 데이터를 연기한다. 하지만 데이터의 크기가 큰 데이터도 전송 시간이 밀려나게 되면 평균대기시

간이 증가하므로 그 데이터의 최적주기보다 더 연기됐을 경우에는 연기하지 않고 전송한다.

이 방법으로 스케줄링하려면 모든 데이터에 대해서 데이터 i 가 전송 될 경우의 전체 평균 대기시간 TAT_i 를 계산해야 한다. 본 연구에서는 현재시간에 데이터 i 가 놓였을 때, 다른 데이터가 다음으로 전송될 시간을 예측하여 전체평균대기시간을 구하는 방법을 제안한다. 전체평균대기시간은 모든 데이터에 대해서 각 데이터의 평균대기시간에 해당 데이터의 요청될 확률을 곱한 값의 누적한 값이다. 전체평균대기시간을 구하는 식은 (3.1)로 나타낼 수 있다.

$$TAT_i = \sum_{j=1}^M AT_j \cdot P_j \tag{3.1}$$

(3.1)에서 AT_j 는 각 데이터 j 의 평균대기시간이다. AT_j 는 현재시간에서 어떤 아이템이 선택되느냐에 따라 달라진다. AT_j 를 구하는 식은 (3.2)로 나타낼 수 있다.

$$AT_j = \sum_{k=0}^{n_j-1} \frac{s_j^k}{T} \cdot \frac{s_j^k}{2} + \frac{s_j^{n_j}}{T} \cdot \frac{s_j^{n_j}}{2} \tag{3.2}$$

(3.2)에서 s_j^k 는 데이터 j 가 k 번째 전송된 시점과 $k+1$ 번째 전송된 시점 사이의 거리이다. 그리고 $s_j^{n_j}$ 은 데이터 j 가 마지막으로 전송된 시점에서 다음에 전송될 시점 사이의 거리이다. 그런데 데이터 j 가 다음에 언제 전송 되는지 알 수 없기 때문에 본 연구에서는 이 값을 예측하는 방법을 사용하였다. 만약 현재시간 T에서 데이터 j 가 전송된다면 $s_j^{n_j}$ 은 현재시간과 데이터 j 가 마지막으로 전송된 시간의 차이가 된다. 현재시간에서 전송되지 않는 데이터들의 $s_j^{n_j}$ 은 현재시간에 데이터 j 의 크기를 더한 값과 데이터 i 가 마지막으로 전송된 시간과의 차이가 된다.

알고리즘

Step 1 : (2.1)을 이용하여 모든 데이터 i 의 최적거리 s_i 를 구한다.

현재 시간 T를 0으로 초기화한다.

모든 데이터 i 의 n_i 를 0으로 초기화한다.

모든 데이터 i 의 d_i 를 0으로 초기화한다.

Step 2 : $n_i \cdot s_i \leq T$ 를 만족하는 데이터 i 의 집합 S를 구한다.

$$S = \{i | n_i \cdot s_i \leq T, 1 \leq i \leq M\}$$

Step 3 : S에 속하는 데이터들 중에서 $(n_i+1) \cdot s_i$ 값이 최소인 데이터 j 를 찾는다.

Step 4 : 만약 $d_j < s_j$ 이면 $s_i < l_j$ 와 $p_i > p_j$ 를 동시에 만족하는 데이터 i 의 집합 D 를 구한다.
 $D = \{i | s_i < l_j, 1 \leq i \leq M\}$

Step 5 : 만약 $D = \emptyset$ 이면 $k = j, d_k = 0$ 이고, $D \neq \emptyset$ 이면 D 에 포함된 데이터들 중에서 TAT_i 가 최소가 되는 데이터 k 를 선택한다. 그리고 $d_j = d_j + l_k$ 이다.

Step 6 : T에서 데이터 k 를 전송한다.
 $n_k = n_k + 1$
 $T = T + l_k$

4. 실험환경 및 결과

4.1 실험 환경

본 논문에서는 전체평균대기시간이 최소가 되는 데이터를 선택하여 스케줄링하는 방법을 구현했다. 제안된 방법을 검증하기 위해 데이터의 크기와 데이터가 요청될 확률을 다양하게 정해주고 데이터의 전송빈도를 고려한 방법[8]와 비교하였다.

본 실험에서는 데이터의 개수 M 은 10으로 가정한다. 그리고 각 실험은 10만 단위시간의 스케줄을 만든다. 데이터가 요청될 확률 p_i 는 Zipf분포를 사용하여 결정했다. Zipf분포는 기존의 데이터 스케줄링 연구에서 데이터가 요청될 확률을 결정하는데 많이 사용된 분포이다. 이 분포는 (4.1)처럼 표현된다.

$$p_i = \frac{(1/i)^\theta}{\sum_{i=1}^M (1/i)^\theta} \tag{4.1}$$

이 분포는 θ 가 0일 경우에는 모든 데이터의 요청될 확률이 $1/p_i$ 로 동일하고 θ 가 증가할수록 데이터들 사이의 요청될 확률이 비대칭으로 정해진다.

데이터의 크기는 [8]에서 사용한 방법과 임의로 정해주는 방법을 사용했다. [8]에서 데이터의 크기를 정해주는 방법은 (4.2)와 같이 나타내어진다.

$$l_i = \text{round}(\frac{L_1 - L_0}{M-1}(i-1) + L_0) \tag{4.2}$$

(4.2)에서 L_0 와 L_1 은 분포의 특징을 결정하는 인자이다. L_0 와 L_1 은 0이 아닌 정수이다. $\text{round}()$ 함수는

반올림을 하는 함수이다.

4.2 실험 결과

본 절에서는 본 논문에서 제안한 데이터 스케줄링 방법과 최적전송빈도에 맞게 데이터를 스케줄링하는 방법을 이용하여 스케줄링했을 경우에 전체평균 대기시간을 계산하여 두 방법의 성능을 비교한다.

그림 1은 본 논문에서 제안한 데이터 스케줄링 방법과 최적전송빈도에 맞게 데이터를 스케줄링하는 방법을 비교하기 위해 두 방법을 사용하여 생성된 데이터 스케줄의 θ 에 따른 전체평균대기시간과 이론적 최적의 전체평균대기시간을 나타낸 그래프이다. 이 실험은 θ 값을 0부터 2까지 증가시켜서 데이터가 요청될 확률을 변화 시키고 데이터의 크기는 확률이 가장 큰 데이터의 크기를 1로하고 확률이 작아질수록 1에서 10사이로 데이터의 크기가 증가하는 데이터를 스케줄링하였다. 이 경우에 데이터의 크기가 최적전송주기보다 큰 것이 없어서 데이터를 최적전송주기와 비슷하게 스케줄링하여 전송 할 수 있다. 따라서 두 방법을 사용하여 스케줄링했을 경우의 전체평균대기시간과 이론적 최적의 전체평균대기시간과의 차이가 크지 않다.

그림 2는 θ 를 0부터 2까지 증가시켜서 데이터가 요청될 확률을 변화시키고 확률이 가장 작은 데이터의 크기가 100이고 나머지 데이터들의 크기는 1인 데이터들을 스케줄링한 경우의 θ 에 따른 전체평균대기시간을 그린 그래프이다. 이 경우에 크기가 1인 데이터의 최적전송주기보다 확률이 가장 작은 데이터의 크기가 크기 때문에 크기가 100인 데이터가 전송될 경우 다른 데이터들은 자신의 주기보다 밀리게

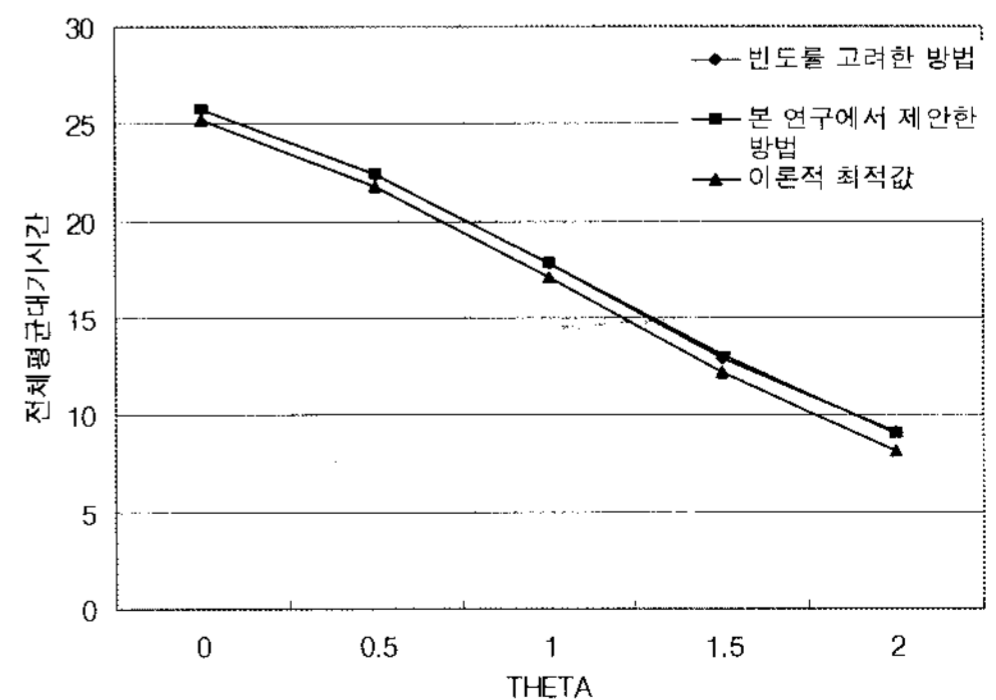


그림 1. 확률이 작아질수록 데이터의 크기가 1에서 10으로 증가하는 데이터를 스케줄링했을 경우

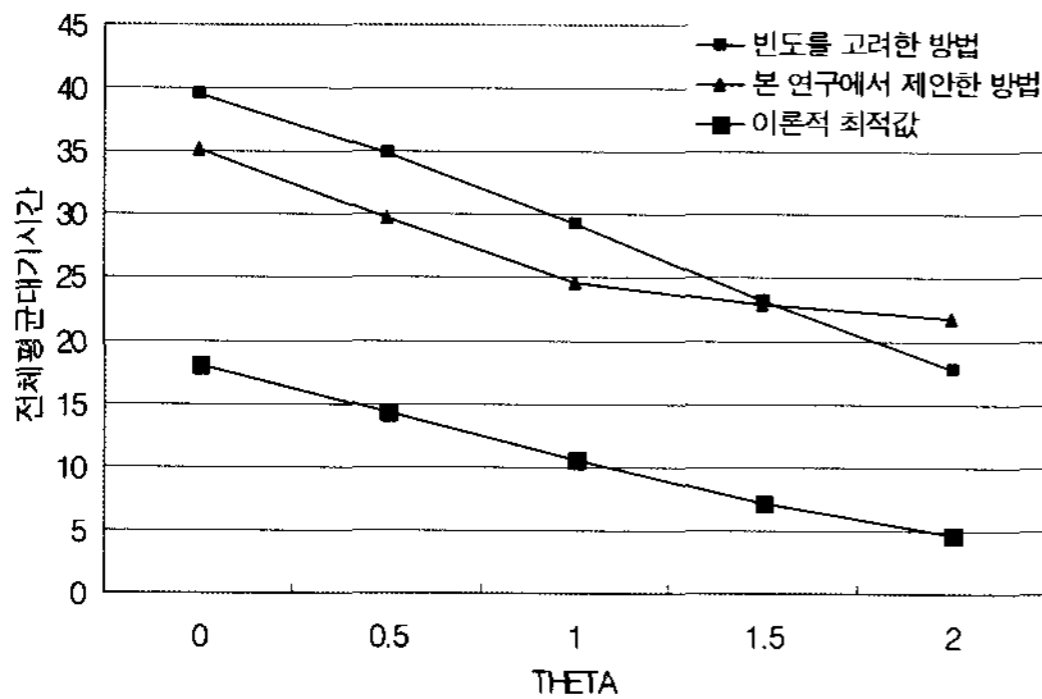


그림 2. 확률이 가장 작은 데이터의 길이가 100이고 나머지 모든 데이터의 길이가 1인 데이터들을 스케줄링했을 경우

된다. 그래서 결과가 이론적 최적의 전체평균대기시간보다 크게 나온다. 하지만 본 연구에서 제안한 방법은 길이가 100인 데이터의 전송 빈도를 낮추고 나머지 데이터들의 전송 빈도를 높여서 길이가 100인 데이터의 평균대기시간은 증가하지만 나머지 데이터들의 평균대기시간이 감소하여서 기존에 연구되었던 데이터의 최적빈도만을 고려한 방법보다 좋은 결과를 보인다.

그림 3은 θ 를 0부터 2까지 증가시켜서 데이터가 요청될 확률을 변화시키고 각 확률분포마다 하나의 데이터의 크기는 100이고 나머지는 1로 하여 모든 경우의 전체평균대기시간의 평균을 그래프로 나타낸 그림이다. 확률이 가장 큰 데이터부터 확률이 가장 작은 데이터까지 데이터의 크기가 100인 경우의 전체평균대기시간의 변화를 비교할 수 있다. 그림 3에서

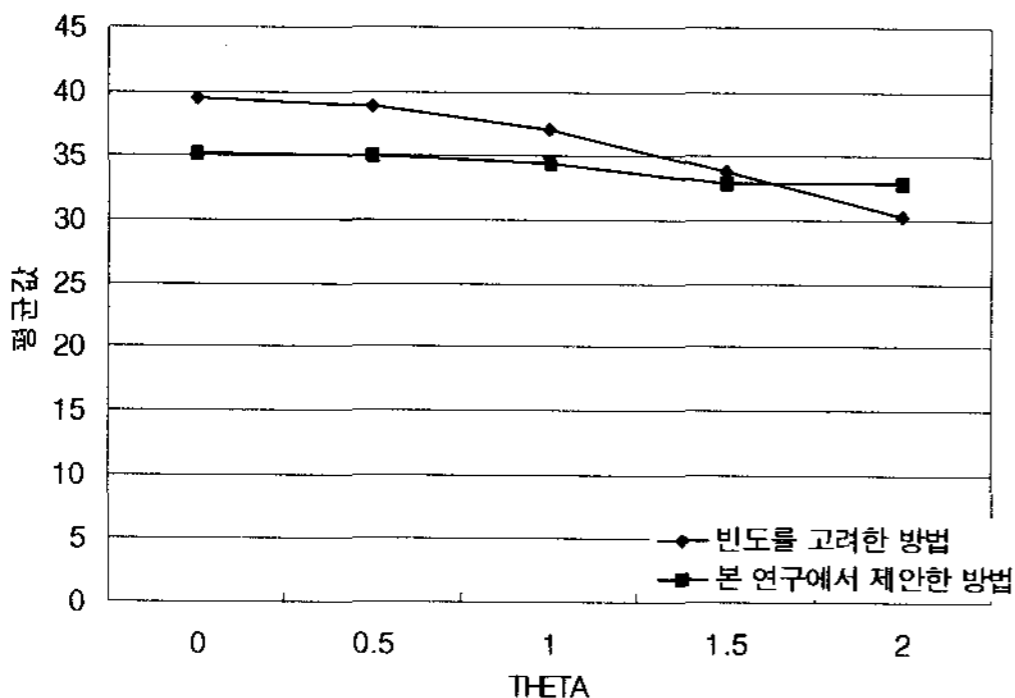


그림 3. 각 확률 분포마다 확률이 작은 데이터부터 확률이 가장 큰 데이터까지 하나의 데이터의 크기를 100으로 바꾸고 나머지 데이터의 크기를 1로 하였을 경우의 전체평균대기시간의 평균값

볼 수 있듯이 본 연구에서 제안한 방법을 사용한 스케줄이 빈도만을 고려한 방법보다 평균값이 작다.

5. 결 론

기존의 데이터 스케줄링 방법은 각 데이터의 최적 빈도만을 고려하여서 임의의 데이터가 다른 데이터에 의해서 주기에서 밀릴 경우에 평균대기시간이 커지는 문제가 있다. 본 연구에서는 이러한 문제를 해결하기 위해 임의의 데이터의 크기가 다른 데이터의 최적주기보다 커서 다른 데이터들이 밀릴 것으로 예상될 경우 밀릴 것으로 예상되는 데이터들을 현재 시간에 스케줄링했을 경우에 전체평균대기시간이 최소가 되는 데이터를 선택하여 전송하고 원래 전송하려고 하였던 크기가 큰 데이터의 전송은 연기한다. 이와 같은 방법을 통해서 기존 방법에서 크기가 큰 데이터가 있을 경우에 전체평균대기시간이 커지는 문제점을 개선하는 것을 제안하였다. 실험을 통해서 기존의 빈도만을 고려한 방법과 본 연구에서 제안한 방법을 비교하여 본 연구에서 제안한 방법이 기존의 방법보다 좋은 것을 증명하였다.

참 고 문 헌

- [1] R. Chernock., Data Broadcasting, McGraw-Hill, 2001.
- [2] ISO/IEC 13818-6, MPEG-2 Digital Storage Media Command & Control, 1998.
- [3] ATSC Standard A/90, ATSC Data Broadcast Standard, 2000.
- [4] Swarup Acharya, Rafael Alonso, Michael Franklin and Stanley Zdonik, "Broadcast Disks: Data Management for Asymmetric Communication Environments," Proc. of the 1995 ACM SIGMOD international conference on Management of Data, ACM Press new York, 24, 2, 199-210, 1995.
- [5] Ammar, M. H., and Wong, J. W., "The design of Teletext broadcast cycles," In Performance Evaluation, Vol.5 No.4, pp. 235-242, 1985.
- [6] J. W. Wong, "Broadcast delivery," Proc. of the IEEE, Vol.76, No.12, pp. 1566-1577, 1988.

- [7] Nitin H. Vaidya, and Sohail Hameed, "Scheduling Data Broadcast in Asymmetric Communication Environments," *Wireless Network*, Vol.5, No. 3, pp. 171-182, 1996.
- [8] S. Hameed, and N. H. Vaidya, "Efficient Algorithms for Scheduling Data Broadcast," *Wireless Networks*, Vol.5 No.3, pp. 183-193, 1999.



백 두 원

1983년 서울대학교 수학과
 1990년 Univ. of Minnesota
 Computer Science M.S.
 1991년 Univ. of Minnesota
 Computer Science Ph.D.
 1992년~1994년 AT & T Bell
 Labs Member of
 Technical Staff

2001년~2002년 Cadence Design System Member of
 Consulting Staff

1995년~현재 숭실대학교 미디어학과 부교수
 관심분야: 디지털 방송, 알고리즘, 컴퓨터 그래픽스



민 재 호

2004년 숭실대학교 미디어학부
 (공학학사)

2006년 숭실대학교 미디어학과
 석사

관심분야: 디지털 방송, 알고리즘



황 주 연

2005년 숭실대학교 미디어학부
 (공학학사)

2005년~현재 숭실대학교 대학
 원 미디어학부 박사과정

관심분야: 디지털 방송, 컴퓨터
 그래픽스, 영상처리