리눅스 멀티미디어 서버에서 정밀한 회전 지연 시간 모델에 의한 디스크 서비스 시간의 예측*

박상수ᵁ 박은정 이수형 신현식
서 울 대 학 교 컴 퓨 터 공 학 과
{sspark, ejpark, onlyjazz, shinhs}@cslab.snu.ac.kr

Estimating disk service time based on a rigorous rotational delay model for Linux multimedia servers

Sang-Soo Parkᵁ Eun-Jeong Park Soo-Hyung Lee Heon-Shik Shin
Dept. of Computer Engineering, Seoul National University

요 약

멀티미디어 서버는 시간 제약성을 가지는 멀티미디어 데이터를 정해진 시간 이내에 디스크로부터 검색해야 하므로 디스크 검색 요청 이전에 디스크 서비스 시간을 예측하여 제한된 시간 이내에 종료된다는 것을 보장해야 한다. 리눅스는 자원의 이용률 및 시스템 성능을 향상시키기 위해서 선행 읽기와 페이지 캐쉬 등의 기법들을 사용하고 있는데 이러한 기법들의 사용으로 디스크 서비스 시간은 매우 가변적이게 된다. 본 논문에서는 리눅스 파일 시스템을 개선하여 디스크 서비스 시간의 가변성이 최소화되도록 하고 이를 기반으로 정밀한 회전 지연 시간 모델에 의한 디스크 서비스 시간의 예측 기법을 제안한다.

1. 서론

멀티미디어 데이터는 기존의 문자 데이터와는 달리 시간 제약성을 가진다. 따라서 주문형 비디오 서비스(Video on Demand)를 제공하기 위한 멀티미디어 서버의 경우 멀티미디어 데이터의 연속적인 재생을 위해서 주어진 데이터를 정해진 시간 이내에 디스크 드라이브로부터 검색한 후 전송해주는 과정이 필요하다. 이를 위해서 멀티미디어 서버는 디스크 검색 요청 이전에 디스크 서비스 시간을 예측하여 디스크 검색이 제한된 시간 이내에 종료된다는 것을 보장해야 한다. 디스크 서비스 시간은 디스크 드라이브의 상태에 따라 가변적이기 때문에 멀티미디어 서버를 위한 디스크 서비스 시간의 예측 기법은 멀티미디어 데이터의 시간 제약성을 만족시키기 위해 최대 디스크 서비스 시간을 예측하게 된다. 멀티미디어 서버는 예측된 디스크 서비스 시간만큼 대역폭을 예약하기 때문에 예측 값의 정확도에 따라 디스크 드라이브의 이용률이 결정되게 되므로 정밀한 디스크 서비스 시간을 예측할 수 있는 예측 기법이 필요하다.

본 논문에서는 리눅스(Linux) 운영체제에서 멀티미디어 서버를 지원하기 위해 온라인에 정밀한 디스크 서비스 시간을 예측할 수 있는 기법을 제안한다.

*본 연구는 한국과학재단 지원(과제번호: 97-0100-09-01-3)에 의해 수행되었습니다.

논문의 구성은 2장에서 리눅스의 선행 읽기와 페이지 캐쉬 등의 기법으로 인한 디스크 서비스 시간의 가변성을 최소화하는 개선 방법과 측정 결과를 살펴보고, 3장에서는 최대 회전 지연 시간에 의한 예측 기법에 비해 정밀한 예측 값을 얻을 수 있는 정밀한 회전 지연 시간 모델에 의한 디스크 서비스 시간의 예측 기법을 제안하고 실험 결과를 살펴보며 마지막으로 4장에서 결론을 살펴본다.

2. 리눅스 파일 시스템의 개선

리눅스의 파일 시스템은 메모리 및 디스크 공간과 같은 자원을 효율적으로 이용하고 디스크 입출력 성능을 높이기 위해서 선행 읽기, 페이지 캐쉬, 클러스터링 등의 기법을 사용하고 있다[1][2]. 그러나, 이러한 기법들은 문자 위주의 응용에 최적화 되어 있고 사용자 수준에서 제어할 수 없으므로 디스크 서비스 시간은 매우 가변적이다. 실험에서는 IBM DCAS-34330W 디스크 드라이브[3]를 사용하였는데 그림 1은 디스크 영역 0에 연속적인 블록 할당된 100MB의 파일을 매 요청 당 256KB 씩 검색하였을 때 디스크 서비스 시간을 나타낸다.

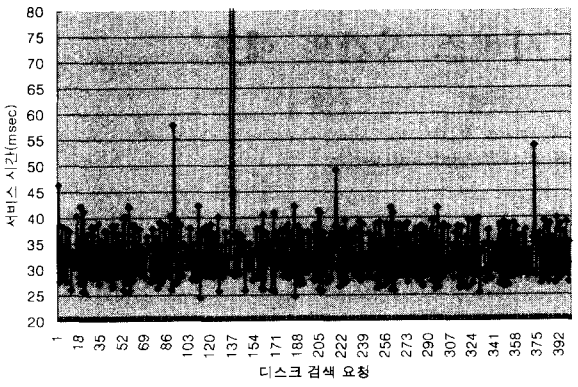

그림 1. 리눅스의 디스크 서비스 시간

디스크 드라이브의 특성상 연속적인 블록에 할당된 데이터를 검색할 경우 디스크 서비스 시간은 데이터 전송 시간과 동일 실린더에서 헤드를 변경하는 트랙 스큐(track-skew) 시간, 인접한 실린더에서 헤드를 이동하는 실린더 스큐(cylinder-skew) 시간의 합이다[4]. 이때 데이터 전송 시간은 일정하므로 서비스 시간의 가변 폭은 두 스큐 시간에 의해 결정되는데, 측정한 스큐 시간의 가변 폭과 측정값의 오차 및 불량 섹터로 인한 지연 시간의 합이 약 3.90 msec이다. 따라서 리눅스 디스크 서비스 시간의 가변 폭은 디스크 드라이브 특성에 의한 것보다 매우 큰 것을 알 수 있다. 리눅스는 최대 120KB를 선행 읽기하고 그 정책을 사용자 수준에서 제어를 하거나 예측할 수 없다. 그림 1에서 리눅스의 선행 읽기 기법에 의해 같은 크기의 요청에 대해 376KB를 검색하는 경우도 있고 136KB를 검색하는 경우도 있으므로 서비스 시간이 가변적이다. 또, 리눅스는 가상 메모리 부분과 통합된 페이지 캐쉬와 슬라브 할당기(Slab Allocator)의 사용으로 현재 프로그램들에 의해 사용되고 있는 메모리 영역을 제외한 모든 영역을 캐쉬에 사용할 수 있으므로 메모리를 매우 효율적으로 사용하지만 디스크 검색 요청 수행 중에 메모리가 부족할 경우 페이지 교체가 발생하기 때문에 그림 1에서 80 msec가 넘는 서비스 시간의 지연이 발생할 수 있다.

본 논문에서는 위와 같은 리눅스의 디스크 서비스 시간 가변성을 최소화하기 위해서 선행 읽기와 페이지 캐쉬를 사용하지 않고 데이터를 디스크 드라이브로부터 사용자 메모리 영역으로 직접 전송하는 직접 입출력 기법(Direct I/O)을 구현하였다. 또, 리눅스는 연속적인 블록을 효율적으로 검색하기 위해서 클러스터링 기법을 사용하는데 최대 클러스터링 크기는 122KB로 제한되어 있다. 이때 디스크 드라이브 제어 보드의 캐쉬가 항상 동작하기 위해 최대 클러스터링 크기를 디스크 드라이브의 DPTL(Disable Pre-fetch Transfer Length)인 64KB로 설정하였다. 그 외에도 측정 프로세스가 리눅스에서 지원하는 POSIX.4의 우선 순위 기반 실시간 스케줄러와 메모리 잠금 기능을 사용하고 다른 사용자 프로세스의 디스크 검색 요청에 의한 간섭을 최소화하기 위해 측정 프로세스에 의한 검색 요청이 다른 사용자 프로세스에 의한 검색 요청보다 먼저 수행되도록 수정하였다.

그림 2는 개선된 리눅스 파일 시스템에서 같은 실험을 수행하였을 때 디스크 서비스 시간을 나타낸다.

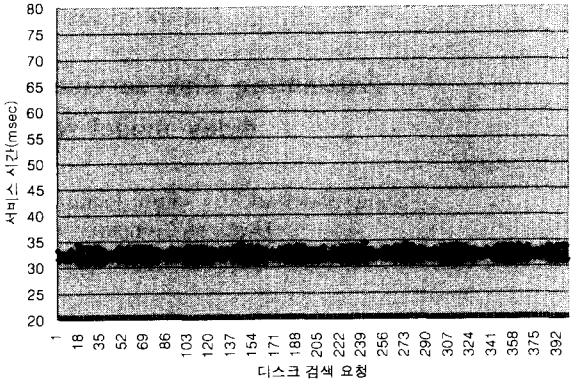

그림 2. 직접 입출력 경우 디스크 서비스 시간

3. 디스크 서비스 시간의 예측

사용자 프로세스의 디스크 검색 요청들은 여러 개의 디스크 검색 요청 명령어로 변환된 후 디스크 스케줄러에 의해서 C-SCAN 방식으로 정렬되고 차례로 디스크 드라이브에 전송되므로 리눅스의 디스크 검색 요청 형태는 디스크 검색 요청 명령어 $\mathcal{R}_i$의 정렬된 집합(ordered set)으로 나타낼 수 있다. 표 1은 논문에서 사용되는 기호를 나타내고 있다.

표 1. 논문에서 사용되는 기호

$\mathcal{R}_i$	i번째 디스크 검색 명령어
B_i	i번째 디스크 검색 명령의 시작 블록 번호
L_i	i번째 디스크 검색 명령의 블록 수
t_i	$\mathcal{R}_i$의 서비스 시간
$t_{move}(a,b)$	블록 a에서 블록 b까지 이동시간
B	파일 시스템 블록 크기
$r_{disk}(i)$	$\mathcal{R}_i$가 위치한 디스크 영역의 전송률
$t_{skew}(i)$	$\mathcal{R}_i$의 트랙 스큐 시간 합

임의의 $\mathcal{R}_i$의 정렬된 집합에 대해서 디스크 서비스 시간 t는 각 디스크 검색 요청 명령어의 서비스 시간의 합과 같으므로 $t=\sum_i t_i$ 이고 t_i는 $\mathcal{R}_{(i-1)}$의 마지막 블록에서 $\mathcal{R}_i$의 첫 블록까지의 이동시간에 $\mathcal{R}_i$의 전송 시간의 합 이므로

$$t_i = t_{move}(B_{(i-1)}+L_{(i-1)}-1, B_i)+\frac{L_i \times B}{r_{disk}(i)}+t_{skew}(i)$$

이다. 기존의 예측 기법에서는 실린더 수 별로 헤드 이동시간을 측정하고 $\mathcal{R}_i$, $\mathcal{R}_{(i-1)}$의 실린더 수 차이로 헤드 이동시간을 예측한 뒤 최악의 경우인 최대 회전 시간을 합하여 요청간 이동시간을 구하는 최대 회전 지연 시간 모델을 사용한다. 일반적인 디스크 검색 요청은 지역성이 있기 때문에 대부분의 경우 요청간 이동시간은 평균 헤드 이동시간보다 작고 실험 환경에서 디스크 드라이브의 평균 이동시간은 8 msec이고 최대 회전 지연 시간은 11 msec이기 때문에 회전 지연 시간이 헤드 이동시간보다 디스크 서비스 시간에 미치는 영향이 크다.

따라서 본 논문에서는 정밀한 회전 지연 시간 모델에 의한 예측 기법을 사용하는데 정밀한 회전 지연 시간을 구하기 위해서는 우선 이동 시작 블록 $B_{(i-1)}+L_{(i-1)}-1$과 끝 블록 B_i의 위치를 정확히 알아야 한다. 실험에서는 측정 프로그램을 사용하여 0번 블록의 물리적인 위치를 기준으로 각 트랙의 첫 번째 블록의 각도를 측정하여 시작 블록 $B_{(i-1)}+L_{(i-1)}-1$과 끝 블록 B_i의 각도를 계산하였으며 이를 바탕으로 디스크 회전 방향으로 회전 지연 시간 rotdelay를 구하였다.

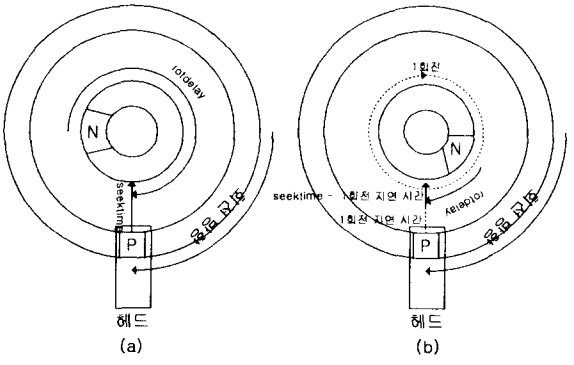

그림 3. 디스크 검색 요청간 이동시간

이때 두 블록 간 헤드 이동시간을 seektime이라고 하면 다음의 3가지 경우에 대해 이동시간을 구할 수 있다. 단, 그림 3에서 P는 시작 블록이고 N은 끝 블록이다.

① seektime < rotdelay : rotdelay [그림 3.(a)]
② seektime > rotdelay : rotdelay + 1회전 지연 시간
③ seektime > 1회전 지연 시간 : 헤드 이동시간에 1회전 지연 시간을 뺀 후 ①,②에 적용해서 얻은 요청 간 이동시간에 1회전 지연 시간을 더한 값 [그림 3.(b)]

헤드 이동시간은 디스크 드라이브의 상태에 따라서 가변적인데 최대 디스크 서비스 시간을 구해야 하기 때문에 측정 프로그램을 사용하여 측정한 실린더 수 별 헤드 이동시간은 최대값을 기준으로 한다.

실험에서는 무작위의 디스크 검색 요청을 생성하고 매 디스크 검색 요청 당 512KB, 1MB, 1.5MB, 2MB를 검색하는 요청을 수행하여 측정한 실제 디스크 서비스 시간과 본 논문의 예측 기법을 사용하여 예측한 디스크 서비스 시간을 비교하였다. 또, 기존의 예측 기법과의 비교를 위해 요청 간 이동시간을 헤드 이동시간과 최대 회전 지연 시간의 합으로 했을 경우 예측 값과 비교하였다. 그림 4는 요청의 크기가 1MB일 경우의 예이고 표 2는 실험 결과 통계를 나타낸다.

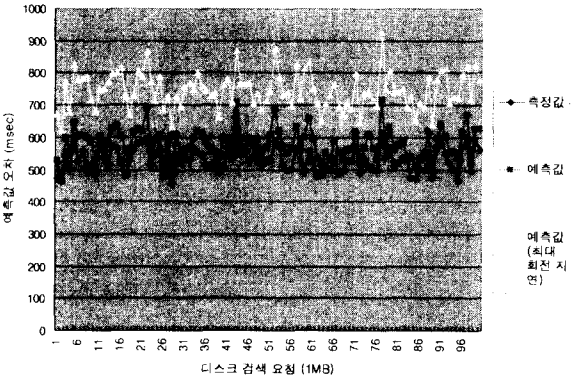

그림 4. 실험 결과

표 2. 실험 결과 통계

	512KB	1MB	1.5MB	2MB
측정값 평균 (msec)	285.26	530.87	774.19	1011.27
오차값 평균 (msec)	28.00	51.03	71.74	92.63
오차값 표준 편차 (msec)	17.11	24.16	27.49	34.25
오차값 비율 (%)	9.81	9.61	9.26	9.15
오차값 (최대 회전 지연)	113.02	214.62	312.43	412.21
오차값 표준 편차 (최대 회전 지연)	25.91	36.04	35.80	54.24
오차값 (최대 회전 지연) 비율	39.63	40.42	40.35	40.76

4. 결론

본 논문에서는 리눅스 파일 시스템을 개선하여 디스크 서비스 시간의 가변성을 최소화하도록 하고 정밀한 회전 지연 시간 모델에 의한 디스크 서비스 시간 예측 기법을 제안하였다. 실험을 통해서 본 논문의 예측 기법에 의한 디스크 서비스 시간의 예측 값은 측정값과 9% 정도의 오차를 나타내며 기존 기법과 비교해서 오차 값은 평균적으로 30% 감소하고 오차 값의 표준 편차는 30%~40% 감소하는 것을 확인하였다.

5. 참고 문헌

[1] Michael Beck and Harald Bohme. *Linux Kernel Internals 2nd Edition*. Addison Wesley, 1997.
[2] Alessandro Rubini and Andy Oram. *Linux Device Drivers*. O'Reilly, 1998.
[3] IBM. Hard disk drive specifications for DCAS-34330W, 1996
[4] C. Ruemmler and J. Wilkes. An Introduction to Disk Drive Modeling. *IEEE Computer*, 1994.