복합동반쓰기를 사용하는 디스크 시스템의 성능 평가

장 윤 석† · 김 홍 일† · 김 국 보†

요 약

본 논문에서는 디스크캐쉬 관리 기법의 하나인 동반쓰기 기법을 개선한 복합동반쓰기 기법을 제안하고 그 성능을 평가하였다. 제안된 기법의 성능 평가를 위하여 디스크 시스템을 모델화 하여 실제 디스크시스템에 근접한 시뮬레이션 모델을 구현하고 trace-driven 시뮬레이션을 수행하였다. 시뮬레이션의 결과 본 연구에서 제안된 복합동반쓰기 기법이 기존의 동반쓰기 기법에 비하여 우수한 성능을 나타냄을 증명하였다.

Performance Evaluations of Hybrid Write-Piggybacking Technique for Disk System

Yun Seok Chang† · Hong Il Kim† · Guk Boh Kim†

ABSTRACT

This paper proposes an improved write-piggybacking technique called hybrid write-piggybacking, and evaluates its performance. The performance of the proposed hybrid write-piggybacking technique is done through a trace-driven simulation using a model of a real disk system. The results of simulation show that the proposed hybrid write-piggybacking has better performance compared to the original write-piggybacking technique.

1. 서 론

컴퓨터 시스템의 하드웨어는 CPU를 중심으로 하는 처리장치와 디스크, 테이프장치 등을 포함하는 입출력장치의 두 부분으로 크게 구분될 수 있다. 최근에 들어서 반도체 공학의 발달에 따른 고성능 마이크로프로세서의 개발에 힘입어 컴퓨터의 처리 속도는 과거에 비하여 매우 급속하게 증대되고 있다. 그러나 입출력장치는 기계적인 요소를 포함하므로 그 동작 속도는 전자적인 장치들에 비하여 매우 느린 편이다. 비록 최근에 들어서 입출력장치에 대한 연구가 많이 수행되고 있고, 이에 따라서 입출력장치의 처리 속도도 과거에 비하여서 상당히 향상되고 있지만, 전자적인 요소들의 발달 속도에 비해서는 그 성능 향상도가 비교적 낮은 편이므로 입출력시스템의 처리 속도와 처리량은 프로세서와 같은 장치의 처리 속도에 크게 미치지 못하고 있는 실정이다.

컴퓨터 시스템 상에서 사용자의 프로세스(user process)가 처리되는 도중에 입출력 요구가 발생하면, 프로세스는 입출력 장치가 입출력을 완료할 때까지 기다려야만 한다. 고성능, 대형의 컴퓨터 시스템일수록 처리하여야 할 프로세스의 수가 많아지게 되고, 이에 따라서 입출력 장치는 프로세서와의 처리속도의 차이로 인하여 병목 현상을 일으키게 된다. 이러한 병목 현상은 프로세스가 수행대기상태로 있는 시간을

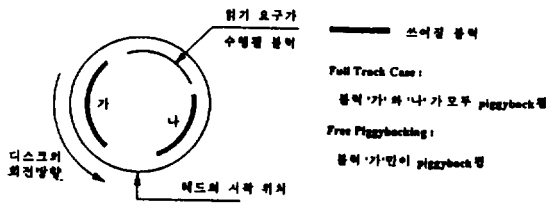
†정 회 원:대진대학교 컴퓨터공학과
논문접수:1995년 7월 18일, 심사완료:1995년 11월 2일

증가시키므로 전체적인 컴퓨터 시스템의 성능을 저하시키는 중대한 원인이 된다. 따라서 대부분의 고성능 상용 시스템에서는 대용량의 버퍼캐쉬나 디스크 캐쉬를 사용하여 입출력 성능을 개선하려 하고 있다. 이러한 시스템에서는 캐쉬의 크기가 비교적 크고 읽기 적중률(read hit ratio)이 높으므로 실제적인 디스크 동작은 대부분 쓰기 요구가 차지하게 된다.[3] 입출력 처리량이 증가됨에 따라서 쓰기 요구, 특히 동기적 쓰기 요구가 증가되면 상대적으로 읽기 요구들이 대기 지연될 확률이 증가되어 프로세스의 수행이 지연되므로 쓰기 요구들에 대한 처리가 최적화 되어져야 할 필요성이 생기게 된다. 그러나 대부분의 캐쉬 기법들이 읽기 요구에 대해서는 많은 이점을 제공하는 반면에 쓰기 요구에 대하여서는 그다지 많은 이점을 제공하지 못하고 있다.[8, 10]

본 연구에서는 디스크캐쉬의 최적화 중에서도 쓰기 요구에 대한 최적화에 중점을 두고 쓰기 요구에 의한 읽기 요구의 지연을 가능한 한 최소화하기 위하여 기존의 디스크캐쉬 관리 기법을 개선하고, 개선된 디스크캐쉬 관리 기법이 기존의 관리 기법에 비하여 우수한 성능을 나타낼 수 있음을 시뮬레이션을 통하여 검증하였다.

2. 복합 동반쓰기(Hybrid Write-piggybacking)

대부분의 캐쉬들처럼 디스크캐쉬도 읽기 요구에 대하여서는 많은 이득을 제공하여 준다. 그러나 쓰기 요구의 경우에는 시스템의 무결성을 유지하기 위하여 많은 상용 시스템에서는 write-through 기법을 사용하고 있으므로 쓰기 요구에 대한 캐싱 이득은 읽기 요구에 비하여 상대적으로 적게 된다. 만약에 쓰기 요구들이 캐싱 되어도 무결성이 유지될 수 있음을 보장한다면 write-behind 방식으로 쓰기 요구들을 관리할 수 있게 된다. 그러므로 디스크캐쉬는 보통의 캐쉬 메모리보다는 NVRAM(Non-Volatile RAM)을 사용할 경우 전체적인 캐싱 효율이 증대되어 디스크 시스템의 성능을 증대시킬 수 있다.[2, 10] 쓰기 전용 캐쉬를 NVRAM으로 구성할 경우 (그림 1)과 같은 2가지 동반쓰기(write-piggybacking) 기법들이 사용될 수 있다. 동반쓰기 기법에서는 읽기 요구가 발생한 경우, 해당 읽기 요구가 발생된 트랙에 쓰기 요구들이 있으면 이들을 같이 처리하므로 쓰기 요구들을 보다 효율적으로 처리할 수 있다. Biswas는 2가지 동반쓰기 기법 중에서 Full-Track 방식이 Free 방식에 비하여 우수한 성능을 나타냄을 보인 바 있다.[10]



(그림 1) 동반쓰기 기법
(Fig. 1) Write-Piggybacking Technique

NVRAM으로 이루어진 디스크캐쉬를 가지는 디스크 시스템에서 Full-Track 방식의 동반쓰기 기법을 사용할 경우, 읽기 요구가 발생되면 해당 트랙 전체가 탐색될 확률이 높으며 이는 디스크 시스템의 부하가 증대될수록 높아진다. 또한 대부분의 고성능 상용 시스템들은 대용량의 버퍼캐쉬를 가지고 있으므로 읽기 요구들은 대부분 연속적읽기이고, 이 경우 미리읽기(readahead)기법을 사용하면 캐쉬 적중률이 매우 높아진다. 그러므로 동반쓰기 기법에서 읽기 요구가 발생될 경우 해당 트랙의 전체 섹터를 모두 트랙 버퍼, 또는 읽기 캐쉬에 캐싱하는 On-arrival Readahead 기법을 병행하여 사용하면 낭비되는 회전 지연 시간을 미리읽기에 이용할 수 있을 뿐 아니라 읽기 요구에 대한 적중률도 상당히 개선될 수 있을 것으로 예상된다. 이와 같은 동반쓰기 기법을 본 연구에서는 복합동반쓰기(Hybrid Write-piggybackng)로 정의하였다.

사실상, 현재 사용되고 있는 대부분의 고성능 디스크 시스템에서는 디스크캐쉬, 또는 트랙 버퍼(track buffer)를 가지고 있으므로[18, 20] 이와 같은 readahead기법을 채택하여 입출력 성능을 향상시키고 있다. 그러나 readahead 기법을 사용하는 디스크캐쉬에 동반쓰기 기법을 적용할 경우 읽기 요구와 쓰기 요구들이 다같이 효율적으로 처리될 수 있으므로 디스크 시스템의 전체적인 성능이 향상될 수 있다. 따라서 본 연구에서는 새로운 디스크캐싱 기법으로 복합동반쓰기 기법을 제안하고 시뮬레이션을 통하여 복합

동반쓰기 기법과 기존의 동반쓰기 기법을 비교하여 그 성능을 평가하였다.

3. 디스크 시스템의 모델링

3.1 모델링 요소

디스크 시스템을 정확하게 모델링하기 위해서는 디스크 시스템을 구성하는 각종 요소들의 구조와 상호 관계, 그리고 동작 알고리즘등이 정확하게 모델화되어야 한다. 일반적으로 디스크 시스템은 물리적인 저장 장치(disk component)와 제어기(controller)의 2부분으로 크게 구분되어질 수 있다.[1]

물리적인 저장 장치는 데이타를 디스크에 물리적으로 수록하는 기록 장치와 디스크상에서의 헤드 이동을 위한 탐색(seeking), 트랙 보정 등을 수행하는 위치 조정 장치로 구성된다. 탐색 시간의 모델링은 탐색 시간이 헤드가 이동하는 실린더의 수에 비례하여 증가되는 선형 탐색 시간에 탐색 거리에 따른 특성을 보정하여 실제적인 탐색 시간에 가깝도록 하여야 한다. 또한 디스크와 헤드가 움직이는 상황을 최대한 실제에 가깝게 모델링하기 위하여 디스크의 실린더 위치 탐색을 위한 트랙 보정(track following), 논리적 블럭과 디스크상의 물리적인 블럭들 간의 매핑, Zonning, 트랙 skewing등의 요소들이 충분히 고려되어야 한다.[1, 9]

제어기는 디스크 저장 장치와 입출력 버스간의 데이타 입출력 및 제어 기능을 수행한다. 여기에서는 주로 입출력을 수행하는 버스 인터페이스와 디스크 캐쉥 및 입출력 큐잉요소가 고려되어질 필요가 있다. 버스 인터페이스의 경우 최근에는 대부분의 대용량의 디스크 시스템들이 SCSI(Small Computer System Interface)라고 하는 5~40MBPS의 동기식 인터페이스를 사용하고 있다. SCSI 제어기를 사용할 경우 하나의 제어기에는 8개까지의 주변 장치들이 연결될 수 있으며 이들 장치간의 데이타 전송은 별도의 데이타 전송 규약(SCSI protocol)에 의하여 수행되어진다. 그러므로 입출력 인터페이스를 모델링하기 위해서는 SCSI인터페이스에 대한 모델링이 필요하게 된다.[14]

디스크 캐쉥은 디스크 시스템의 성능에 많은 영향을 줄 수 있는 요소이다.[2, 8, 10] 대부분의 고성능, 대용량의 디스크 시스템에서 사용되는 디스크 제어기들은 내부적으로 디스크캐쉬(Disk Cache)를 사용하므로 디스크와 호스트 컴퓨터간에 전송되는 데이타는 먼저 디스크캐쉬에 저장된다. 호스트 컴퓨터로부터 전송되는 입출력 요구가 디스크캐쉬에서 적중되면 물리적인 디스크 동작을 수행할 필요 없이 입출력 요구가 바로 서비스될 수 있으므로 입출력 시간을 크게 줄일 수 있다. 그러므로 디스크 제어기는 가능한 한 많은 입출력 요구들이 디스크캐쉬에서 적중되도록 할 필요가 있다. 이를 위해서는 디스크캐쉬의 속도, 크기, 제어기상의 위치 등이 고려되어지며 캐쉥 효율을 높이기 위하여 Readahead, Immediate reporting 등의 여러 가지 캐쉥 알고리즘들도 사용되어진다.

3.2 모의 실험 도구(Simulator)

디스크 시스템을 모델링하고 이를 바탕으로 디스크 시스템의 성능을 개선하기 위하여 본 연구에서는 trace-driven 시뮬레이션을 통한 모의 실험을 수행하였다.[2, 3, 6, 7, 10, 11] 이를 위해서 본 연구에서는 실제 디스크 시스템으로부터 디스크 동작을 기록한 입출력 트레이스(I/O Trace)와 이를 입력으로 하여 디스크 시스템의 동작을 시뮬레이션하고 그 결과를 출력하여 주는 시뮬레이션 프로그램을 사용하였다.

3.2.1 입출력 트레이스

실제의 디스크 시스템에 대한 입출력 데이타로는 HP Laboratory의 디스크 시스템에 대한 연구에서 여러 차례 사용되어진 바 있는 트레이스 데이타를 사용하였다.[2] 이 트레이스 데이타는 HP9000시스템에서 사용되는 운영 체제인 HP-UX ver.8에 내장되어 있는 트레이싱 유틸리티(tracing utility)를 사용하여 화일 서버로부터 추출한 것으로 이 유틸리티의 동작은 운영 체제나 사용자 프로세스에 영향을 주지 않으므로 실제의 디스크 사용 정보들을 정확하게 추출할 수 있다. 입출력 트레이스 데이타는 1일 단위로 10일간에 걸쳐 수집된 화일들로 구성되어 있으며 각 화일의 트레이스 내용은 다음과 같다.

- 입출력 요구들이 제어기에 큐잉된 시각(milisecond)
- 입출력 요구들이 디스크에서 처리되기 시작한 시간과 처리가 종료된 시간(milisecond)
- 디스크 드라이브 번호 및 영역 번호(drive num-

ber/partition number)
- 입출력 데이타에 대한 실린더번호 및 블럭번호 (Cylinder number/Address)
- 입출력되는 데이타의 크기(data size, KB)
- 입출력 데이타 특성(synchromous/asynchronous)
- 현재 제어기에 대기되어 있는 입출력 요구의 수 (queueing length)

트레이스 화일은 개개의 입출력 요구마다 위와 같은 내용의 데이타를 포함하는 데이타 레코드로 구성되어 있으며 이는 직접 시뮬레이션 프로그램의 입력으로 사용되었다.

3.2.2 디스크 특성

입출력 트레이스가 추출되어진 시스템은 HP9000 화일 서버로 이는 다수의 워크스테이션과 함께 client-server로 구성되어 있다. 그러므로 여기에 장착된 디스크들은 다수 사용자, 다수 응용에 대한 특성을 잘 나타낼 수 있다고 볼 수 있다. 본 연구의 시뮬레이션에서 사용되어진 디스크 모델은 이 화일 서버에 장착된 1.3GB 용량의 HP97560 SCSI디스크를 모델화한 것으로 이에 대한 특성은 [표 1]에 나타나 있다.[2, 13] 디스크/호스트 인터페이스로는 10MBps의 SCSI-II 버스를 사용하고 있고 디스트 컨트롤러에는 128KB의 디스크캐쉬를 내장하고 있다.

3.2.3 시뮬레이션 프로그램(Simulator)

본 연구에서는 HP97560 SCSI 디스크와 동일한 특성을 가지는 모델을 시뮬레이션하여 주는 시뮬레이션 프로그램을 작성하여 실험에 사용하였다. 이 시뮬레이션 프로그램은 Ansi-C 문법에 의하여 작성된 약 3,000줄의 코드로 구성되어 있고 HP-UX 운영 체제 및 기타 대부분의 UNIX 운영체제 하에서 동작 가능하도록 되어 있다. 이는 SCSI bus simulator와 디스크 저장 장치 simulator, 그리고 디스크캐쉬 simulator의 세 부분으로 구성되어 있다(그림 2).

SCSI bus simulator는 1채널의 10MBps SCSI-II 버스의 동작을 시뮬레이션하여 준다. 디스크 저장 장치 simulator는 제어기로부터의 입출력 명령을 수행하는 과정을 시뮬레이션하는 부분으로 탐색 시간과 회전 지연 시간, 그리고 데이타 전송 시간에 대한 계산을

〈표 1〉 HP97560 SCSI 디스크 특성
〈Table 1〉 HP97560 SCSI Disk Specifications

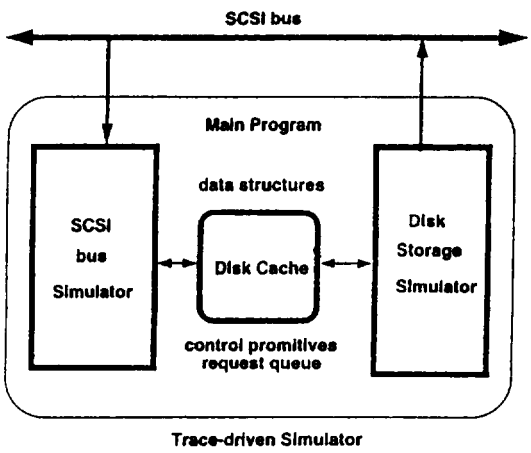
Parameters		Values
Sector Size		512 bytes
Cylinders		1962
Tracks/cylinder		19
Sectors/track		72
Number of zone		1
Revolution speed		4002 rpm
Disk interface		SCSI-II
Seek time	short	$3.24 + 0.4\sqrt{d}$
	long	$8 + 0.008 * d$
	threshold	d = 383 cyl.
Track switch time		1.6 ms
Head switch time		0.7 ms
Sparing type		track sparing
Track skew		8 sectors
Cylinder skew		18 sectors
Controller Overhead		2.2 ms
Disk cache size		128 KB
Read		72 KB
Write		56 KB

수행하여 준다. 탐색 시간은 탐색 거리가 짧은 경우에는 거리의 제곱근에 비례하고 거리가 긴 경우에는 탐색 거리에 비례하도록 하였다(〈표 1〉 참조). 디스크의 회전 지연 시간 및 데이타 전송 시간은 디스크의 회전 속도와 현재 헤드가 위치하고 있는 섹터의 주소, 그리고 입출력될 데이타의 크기로부터 계산되어진다.

디스크캐쉬 simulator는 디스크 제어기 내에 내장된 디스크캐쉬의 동작을 묘사하여준다. 디스크캐쉬는 SCSI bus simulator와 디스크 저장 장치 simulator 사이에서 producer-consumer 방식으로 동작되어 비동기적인 상호 연결 역할을 수행한다. 본 연구에서 사용된 HP97560 디스크 모델의 디스크캐쉬에 적용되는 요소(parameter)들은 다음과 같다.

- Disk Cache:분할된 읽기/쓰기 캐쉬(cache splitting)
- Cache Size:128~2,000KB

- Caching Algorithm : LRU, Delayed back-to-back write, On-Arrival Readahead, Immediate-Reporting



(그림 2) 디스크 Simulator의 구조
(Fig. 2) Structure of Disk Simulator

다음 절에서는 3.1의 모델링 요소들을 반영하여 구현된 디스크 시뮬레이션 프로그램이 실제의 디스크 시스템을 적절한 오차 내에서 잘 반영하고 있는가를 검증한다. 그리고 나서 본 시뮬레이션 프로그램을 이용하여 디스크캐쉬에 여러 piggybacking 알고리즘들을 적용하였을 때의 결과를 분석하여 그 성능을 평가한다.

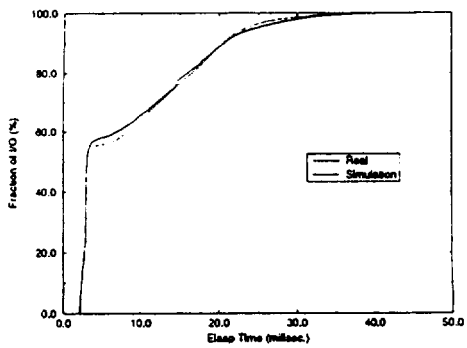
3.3 모델의 검증

본 연구에서 작성되어진 시뮬레이션 프로그램의 적합성은 시뮬레이션된 결과와 실제 디스크의 동작 결과를 비교하여 검증하였다. 즉 트레이스 데이타로부터 얻어지는 실제 디스크 입출력 수행시간과 시뮬레이션 결과로 얻어지는 입출력 수행 시간을 비교하여 그 결과가 작은 오차를 가지는 사실을 보임으로써 시뮬레이션의 정당성을 검증하는 것이다. 이를 위하여 본 연구에서는 Wilkes에서 보는 바와 같은 demerit curve를 이용하였다.[1, 2] 이 그래프의 가로축은 입출력 요구들이 디스크 시스템에서 처리되는 데 걸리는 시간을 0.1 milisecond 단위로 나타낸 것이며 세로축(Y(t))은 각 처리 시간에 해당되는 입출력 요구들에 대한 누적된 백분율 값(%)을 나타내는 것으로 이는 다음과 같이 계산되었다.

$$Y(t) = \frac{\sum_{t=0}^{\infty} f(t)}{N} \times 100(\%)$$

$f(t)$: 입출력에 t 시간이 걸리는 입출력 요구들의 수
N : 전체 입출력 트레이스(요구)의 수

모델화된 디스크 시스템이 실제 디스크 시스템을 얼마나 잘 반영하고 있는가를 계산하기 위해서는 Y 축의 각 점(1% 단위)들에 대한 두 그래프의 수평적인 차이(시간)의 RMS 값인 demerit값을 이용한다. demerit이 작을수록 실제의 디스크 시스템에 가깝게 모델된 것이라고 할 수 있다. (그림 3)에서 모델화된 디스크 시스템의 demerit는 평균 0.54(ms)로 Wilkes의 모델과 비교하여 볼 때 이러한 정밀도는 실제의 디스크 시스템을 잘 반영하고 있다고 할 수 있다.



(그림 3) 모델화된 디스크에 대한 정밀도
(Fig. 3) Precision on Disk Model

4. 모의 실험

본 연구에서는 앞 절에서 구현한 시뮬레이션 프로그램을 이용한 모의 실험을 통하여 기존의 동반쓰기 기법과 복합동반쓰기 기법의 두 가지 디스크캐쉬 알고리즘에 따른 디스크 시스템의 성능 평가를 수행하

였다. 모의 실험에서 사용된 디스크 인자(parameter)들은 〈표 1〉과 같다.

모의 실험의 결과로부터 성능 평가를 수행하기 위해서는 먼저 객관적인 성능 평가 지표가 필요하게 된다. 본 연구에서 사용된 성능 평가 지수는 다음과 같다.

1) Average I/O time: 입출력 요구들이 수행되는 데 걸리는 평균 입출력 시간.
   이는 SCSI bus상에서 측정되는 시간으로 디스크 캐쉬의 크기와 캐싱 알고리즘에 따른 효과를 사용자 프로세스의 관점에서 나타낼 수 있다.
2) cache hit ratio: SCSI bus를 통하여 디스크로 전송되는 모든 입출력 요구들에 대한 평균 디스크 캐쉬 적중률.
3) Purged-write ratio: 전체 write block 중 capacity miss[4]에 의하여 디스크에 쓰여진 write block의 비
4) Piggyback-read ratio: 물리적인 읽기 요구 중 동반쓰기를 수행한 읽기 요구의 비
5) Piggyback-write ratio: 전체 쓰기 블럭들 중 동반쓰기에 의하여 디스크에 쓰여진 쓰기 블럭들의 비

모의 실험은 시뮬레이션 프로그램을 캐싱 알고리즘별로 디스크캐쉬의 용량을 변화시키면서 반복적으로 수행되었다. 여기서 시뮬레이션의 입력은 동일한 트레이스 데이타가 반복적으로 사용되었다. 시뮬레이션 프로그램은 매회 수행될 때마다 해당 조건에 대한 결과를 출력한다. 성능 평가에 필요한 지수들은 이 결과들로부터 산출되어진다.

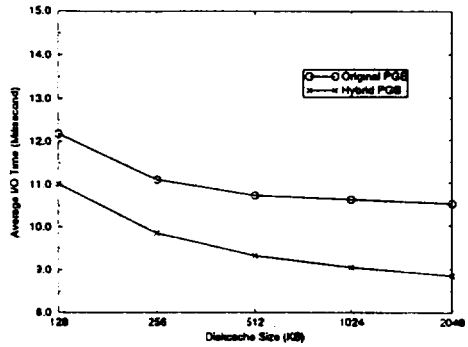
5. 결과 및 분석

시뮬레이션 결과는 디스크캐쉬의 크기 변화에 대한 성능 평가 지수의 변화로 나타내었다. 기존의 동반쓰기 기법을 사용한 결과는 Original PGB로, 복합동반쓰기 기법을 사용한 결과는 Hybrid PGB로 표시되었다.

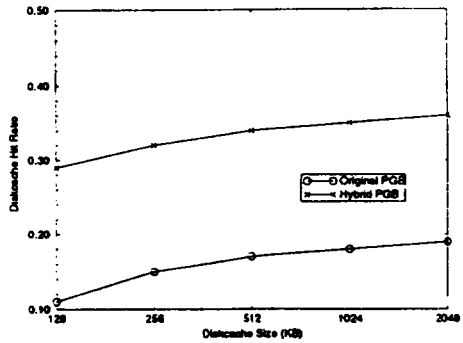
(그림 4)는 디스크캐쉬의 크기에 따른 평균 입출력 처리 시간(Average I/O time)을 나타낸다. 복합동반쓰기를 사용할 경우에는 기존의 동반쓰기 기법에 비하여 처리 시간이 줄어든 것을 볼 수 있다. 이는 On-arrival readahead 기법에 의하여 읽기 요구들에 대한 처리시간이 줄어드는 효과와, 동반쓰기 기법에 의하여, NVRAM으로 구성된 디스크캐쉬로부터 capacity miss 때문에 발생되는 강제적인 쓰기(purged-write) 요구의 양이 줄어듦으로써 읽기 요구들에 대한 처리가 지연될 확률이 감소되는 효과의 2가지 요인이 복합되어 나타난 것으로 분석될 수 있다.

(그림 5)는 디스크캐쉬의 크기에 따른, 입출력 요구들에 대한 디스크캐쉬 적중률(Diskcache hit ratio)을 나타내는 결과로 On-arrival Readahead의 효과가 분명하게 나타나고 있다. 즉 복합동반쓰기 기법을 사용함으로써 readahead가 수행된 경우는 그렇지 않은 경우에 비하여 높은 디스크캐쉬 적중률을 보이므로 읽기 요구에 대한 처리시간이 상당히 줄어들게 된다.

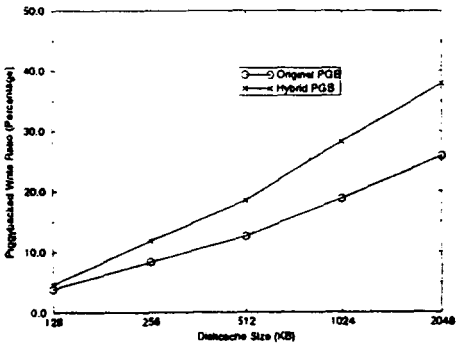
(그림 6)은 디스크캐쉬의 크기에 따른 Purged-write ratio를 나타낸다. 이 결과는 복합동반쓰기 기법을 사



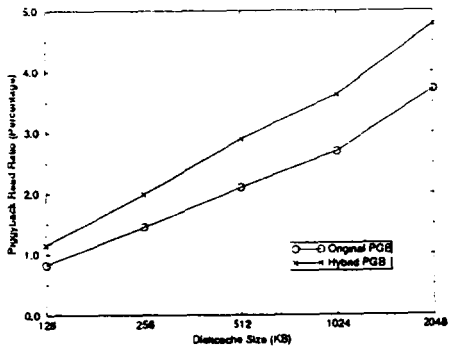
(그림 4) 평균 입출력 처리 시간
(Fig. 4) Average I/O Time



(그림 5) 디스크캐쉬 적중률
(Fig. 5) Diskcache Hit Ratio



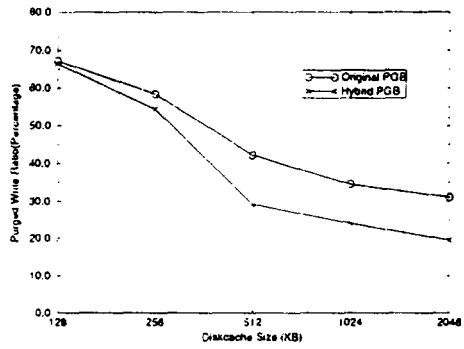
(그림 6) Purged-write Ratio
(Fig. 6) Purged-write Ratio



(그림 7) Piggyback-read Ratio
(Fig. 7) Piggyback-read Ratio

용할 경우 전체 쓰기 요구된 블럭 수에 대한 purge-write 블럭의 비가 기존의 동반쓰기 기법을 사용한 경우에 비하여 최고 15%까지 감소됨을 나타내고 있다. 이는 purge 동작이 수행될 때, 다음의 읽기 요구들이 대기 지연될 확률을 줄여서 읽기 요구들에 대한 처리가 가능한 한 빨리 수행되도록 할 수 있음을 나타내고 있다.

(그림 7)은 물리적으로 수행되는 읽기 요구들 중 동반쓰기를 수행하는 읽기 요구의 비(Piggyback-read ratio)를 나타내고 있다. 복합동반쓰기에서는 readahead 효과 때문에 기존의 동반쓰기에 비하여 물리적인 읽기 요구의 양이 감소된다. 그러나 piggyback-read ratio는 기존의 동반쓰기 기법보다는 복합동반쓰기 기법에 더 높은 결과를 볼 수 있다. (그림 8)의 경우에도 전체 쓰기 요구된 블럭들 중 동반쓰기에 의하여 디스크에 쓰여진 쓰기 블럭의 비(piggyback-write ratio)가 기존의 동반쓰기 기법보다는 복합동반쓰기에서 더 높은 것을 볼 수 있다. 이 두 성능 평가 지수는 결국 동반쓰기의 효율을 나타내는 것으로 복합동반쓰기가 기존의 동반쓰기 기법보다 효율이 좋음을 나타내고 있다.



(그림 8) Piggyback-write Ratio
(Fig. 8) Piggyback-write Ratio

6. 결 론

본 연구에서는 동반쓰기 기법을 개선한 복합동반쓰기 기법을 제시하고 시뮬레이션을 통하여 그 성능을 평가하였다. 성능 평가의 결과, 복합동반쓰기 기법이 기존의 동반쓰기 기법에 비하여 평균 입출력 수행시간, 디스크캐쉬 적중률, 물리적인 쓰기 동작의 비등에서 향상된 성능을 나타냄을 확인하였다. 또한 동반쓰기 효율에 있어서도 복합동반쓰기 기법이 효과적임을 알 수 있었다.

복합동반쓰기는 기존의 디스크 시스템에 쉽게 구현될 수 있다. 특히 디스크캐쉬를 사용하는 상용의 고성능 디스크 시스템은 대부분 on-arrival readahead 기법을 사용하고 있으므로 복합동반쓰기를 지원하기 위해서는 단지 기존의 디스크캐싱 기법에 동반쓰기 기법을 추가하기만 하면 구현된다. 본 연구의 결과는 이러한 디스크 시스템에 복합동반쓰기 기법을 적용할 경우, 전체적인 입출력 성능이 향상될 수 있음을 입증하였다.

참 고 문 헌

[1] C. Ruemmler and J. Wilkes, "An Introduction to Disk Drive Modeling", IEEE Computer Magazine 27(3):17-28, Mar. 1994.

[2] C. Ruemmler and J. Wilkes, "UNIX Disk Access Pattern", '93 Winter USENIX Conference Proceeding, San Diego 405-420, Jan. 1993.

[3] M. G. Baker, J. H. Hartman, M. D. Kupfer, K. W. Shirriff and J. K. Ousterhout, "Measurements of a distributed filesystem", Proceeding of 13th ACM Symposium on Operating Systems Principles, Published as Operating System Review 25(5) :198-212, Oct. 1991.

[4] D. A. Patterson and J. L. Hennessy, "Computer Architecture:A Quantitative approach", Morgan Kauffman, San Meteo, 1990.

[5] C. A. Thekkath, J. Wilkes and E. D. Lasowska, "Techniques for File System Simulation", Technical Report:Department of Computer Science and Engineering, University of Washington, Oct. 1992.

[6] J. K. Ousterhout, H. D. Costa, D. Harrison, J. A. Kunze, M. Kupfer amd J. G. Thompson, "A trace-driven analysis of the UNIX 4.2BSD file system", Proceedings of 10th ACM Symposium on Operating System Principles. Published as Operating System Review 19(5):15-24, Dec. 1985.

[7] K. K. Ramakrishnan, P. Biswas and R. Karedla, "Analysis of file I/O trace in commercial computing environments", Proceedings of 1992 ACM SIGMETRICS. Published as Performance Evaluation Review 20(1):78-90, Jun. 1992.

[8] A. J. Smith, "Disk Cache-miss ratio analysis and design considerations", ACM Transactions on Computer Systems 3(3):161-203, Aug. 1985.

[9] R. P. King, "Disk Arm Movement in Anticipation of Future Requests", ACM Tracnsactions on Computer Systems 8(3):214-229, Aug. 1990.

[10] P. Biswas, K. K. Ramakrishnan, D. Towsley, "Trace Driven Analysis of Write Caching Policies for Disks", Proceedings of 1993 ACM SIGMETRICS 13-23, Jun. 1993.

[11] G. R. Ganger and Y. R. Patt, "The Process-Flow Model:Examining I/O Performance from the System's Point of View", Proceedings of 1993 ACM SIGMETRICS 86-97, Jun. 1993.

[12] 전주식 외, "워크스테이션용 디스크 캐쉬 콘트롤러의 개발", 상공부, 서울대학교 컴퓨터 신기술 공동연구소, 1993, 7

[13] Hewlet-Packard Company, Boise, Idaho. HP 97555, 97558 and 97560 5.25-inch SCSI disk drives:technical reference mannual.

[14] Adaptec, AHA-1540B/1542B SCSI Controller Technical Mannual.

장 윤 석

1988년 2월 서울대학교 자연과학대학 물리학과를 졸업(이학사)

1990년 2월 동 대학교 대학원 컴퓨터공학과 졸업(공학석사)

1992년 2월 동 대학원 박사과정 수료.

1994년 3월~현재 대진대학교 컴퓨터공학과 전임강사

※주관심분야:컴퓨터 구조 설계, 로보틱스, 성능 평가

김 홍 일

1986년 2월 홍익대학교 공과대학 전자계산학과를 졸업(이학사)

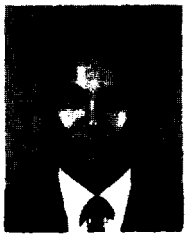
1989년 2월 인하대학교 대학원 전자계산학과 졸업(이학석사)

1993년 홍익대학교 대학원 박사과정 수료.

1993년 8월~1994년 3월 혜전전문대학교 전임강사.

현재 대진대학교 컴퓨터공학과 전임강사

※주관심분야:정보 통신 및 네트워크.



김 국 보

1984년 2월 서울산업대학교 전자계산학과를 졸업 (공학사)

1986년 8월 연세대학교 대학원 전자계산학과 졸업 (공학석사)

현재 대구효성카톨릭대학교 대학원 박사과정

1970년 10월~1990년 3월 해군중앙전산소장

1990년 3월~1993년 2월 부산수산대학교 전자계산학과 조교수 역임.

1993년 3월~현재 대진대학교 컴퓨터공학과 조교수 및 전자계산소장

※주관심분야: 데이타베이스, 시스템 분석 및 설계, 소프트웨어공학