멀티미디어 DBMS에서 3차 저장장치의 효율적 활용 기법

문 찬 호† · 강 현 철††

요 약

멀티미디어 데이터 서비스 응용에서는 멀티미디어 데이터를 구성하는 LOB(unstructured large object)을 다량 저장하고 다룰 수 있어야 한다. 따라서, 대용량 데이터의 효율적 저장 및 처리를 지원하는 DBMS의 저장장치로 기존의 디스크뿐만이 아니라 복수 개의 플래터(광 디스크 쥬크박스의 경우 디스크, 테이프의 경우 카트리지 테이프)로 구성된 광 디스크 쥬크박스 또는 테이프 라이브러리와 같은 3차 저장장치가 중요하게 고려되고 있다. 3차 저장장치는 데이터 접근 시 발생하는 지연 시간이 매우 크기 때문에 3차 저장장치에 저장된 LOB에 대한 효율적인 검색 기법에 관한 연구가 필요하다. 본 논문에서는 LOB의 주 저장소로서 3차 저장장치를 활용할 경우, 3차 저장장치로부터 LOB을 효율적으로 검색하기 위한 입출력 스케쥴링에 대해 연구하였다. 3차 저장장치의 성능 특성과 LOB 데이터의 특성을 고려하여 3차 저장장치로부터 LOB을 검색하는 질의 처리에 있어 발생하는 지연 시간을 줄일 수 있는 여러 가지 입출력 스케쥴링 휴리스틱을 제시하고 이들의 성능을 시뮬레이션을 통하여 평가하였다.

Efficient Incorporation of Tertiary Storage in a Multimedia DBMS

Chan-Ho Moon† · Hyunchul Kang††

ABSTRACT

Multimedia data service applications have to store and manipulate LOBs(unstructured large objects) composing multimedia data. As such, the tertiary storage devices such as an optical disk jukebox and a tape library that consist of a number of platters (the disks in case of an optical disk jukebox and the cartridge tapes in case of a tape library) have been considered essential for the storage system of a DBMS in order to efficiently support storage and management of vary large volume of data. Since the latency with tertiary storage is too long, the schemes for efficient retrieval of LOBs out of tertiary storage need to be investigated. In this paper, we investigated the tertiary I/O scheduling for efficient retrieving of LOBs when tertiary storage is utilized as the main storage for the LOBs. Considering the performance characteristics of tertiary storage and the characteristics of the LOBs, we proposed various I/O scheduling heuristic algorithms that reduce latency in query processing with LOB retrieval from tertiary storage, and evaluated their performance through a detailed simulation.

※ 이 논문은 1998학년도 중앙대학교 학술연구비 지원에 의한 것임.
† 준 회 원 : 중앙대학교 대학원 컴퓨터공학과
†† 정 회 원 : 중앙대학교 컴퓨터공학과 교수
논문접수 : 1999년 3월 23일, 심사완료 : 1999년 5월 19일

1. 서 론

근래에 들어 과학 계산, 통계 처리와 같은 여러 데이터베이스 응용 분야에서 데이터의 양이 기하급수적으로 증가하고 있고, 특히 멀티미디어 데이터 서비스 응용에서는 멀티미디어 데이터를 구성하는 비정형의 LOB(unstructured large object)를 다량 저장하고 다룰 수 있어야 한다. 이와 같이 근래의 데이터베이스 응용들에서는 데이터의 양이 증가할 뿐만 아니라 데이터의 크기도 증가함에 따라 기존 DBMS의 저장장치인 디스크만으로 이들 데이터를 저장하는 데 한계가 있다. 이에 디스크에 비해 낮은 가격으로 방대한 양의 저장 공간을 제공할 수 있는 광 디스크 쥬크박스 또는 테이프 라이브러리와 같은 3차 저장장치가 DBMS에서 중요하게 고려되기 시작했다[1][2][3][4]. 기존 DBMS 환경에서도 3차 저장장치의 사용은 있어왔지만 데이터 백업이나 보관 목적의 오프라인 장치로서 주로 사용되었다. 그러나 이제는 온라인 저장장치로서 3차 저장장치에 저장된 데이터에 대한 효과적인 접근을 지원하는 DBMS가 필요하게 되었다.

3차 저장장치는 그 성능 특성상 데이터 접근 시 발생하는 지연 시간이 기존 DBMS의 저장 매체인 디스크보다 매우 크다는 단점을 지닌다. 광 디스크 쥬크박스 또는 테이프 라이브러리와 같은 3차 저장장치는 복수 개의 플래터(platter, 광 디스크 쥬크박스의 경우 디스크, 테이프의 경우 카트리지 테이프)로 구성된다. 이들에 저장된 데이터의 입출력 시 발생하는 지연 시간으로는, 로봇장치가 입출력 드라이브에 적재된 플래터를 들어내고 새 플래터를 적재하는 데 드는 플래터 교체 시간, 플래터 내에서의 탐색 시간 등을 들 수 있다. 이러한 긴 지연 시간은 3차 저장장치에 저장된 데이터를 대상으로 하는 질의 처리의 성능을 크게 떨어뜨리는 요인이 된다. 따라서 기존 디스크 기반의 DBMS에 3차 저장장치를 부착하여 확장할 경우 가장 중요하게 고려해야 할 사항은 3차 저장장치에 대한 접근 시 지연 부담을 최소화하는 것이다. 현재까지 이를 위해 다음과 같은 많은 연구들이 진행되어 왔다.

DBMS의 저장 계층을 '메모리 버퍼-디스크-3차 저장장치'로 구성하여, 3차 저장장치에 저장되어 있는 데이터 중 비교적 자주 접근되는 데이터를 디스크에 캐쉬해 두어 3차 저장장치에 대한 입출력 요청 횟수를 줄이는 기법이 제시되었고[2][5][6][7][8][9][10], 3차 저장장치에 대한 입출력 요청들이 주어졌을 때 그 입출력들을 요구된 순서대로 수행하지 않고 3차 저장장치의 현재 상태와 하드웨어 구성을 고려한 스케줄링을 수행함으로써 3차 저장장치에 저장된 데이터에 대한 접근 시간을 줄이는 기법이 제시되었다[5][7][8][9][11][12]. 이외에도 테이프에 저장된 두 릴레이션 간의 조인 연산과 디스크와 테이프에 각각 따로 저장된 두 릴레이션 간의 조인 연산을 효율적으로 수행하는 알고리즘에 관한 연구[13][14][15][16], 3차 저장장치에 저장되어 있는 데이터에 대한 접근 시간을 줄이기 위해서 3차 저장장치에 데이터를 최적으로 배치하는 기법을 제시한 연구[17], VOD와 같은 특정 응용 서비스를 위해 3차 저장장치를 저장 시스템으로 사용하는 연구[18][19], 데이터를 저장하는 화일 시스템을 로그 구조를 갖는 LFS(Log-structured File System)로 구성하는 기법에 대한 연구[20][21] 등이 있다.

본 논문에서는 멀티미디어 DBMS에서 대용량 LOB 데이터의 주 저장소로서 복수 개의 플래터로 구성된 광 디스크 쥬크박스 또는 테이프 라이브러리와 같은 3차 저장장치를 활용할 경우, 3차 저장장치로부터 LOB을 효율적으로 검색하기 위한 입출력 스케줄링에 대해 연구하였다. 3차 저장장치의 성능 특성과 LOB 데이터의 특성을 고려하여 3차 저장장치로부터 LOB을 검색하는 질의 처리에 있어 발생하는 지연 시간을 줄일 수 있는 여러 가지 입출력 스케줄링 휴리스틱을 제시하고 이들의 성능을 비교·평가하였다.

본 논문의 구성은 다음과 같다. 2절에서는 LOB의 주 저장소로 사용되는 3차 저장장치를 지원하는 DBMS 모델을 기술한다. 3절에서는 3차 저장장치에 저장된 LOB 검색을 효율적으로 수행하기 위한 여러 가지 입출력 스케줄링 휴리스틱 알고리즘에 대해 기술한다. 4절에서는 이들 알고리즘들을 이용한 LOB 검색의 성능을 평가하기 위한 시뮬레이션 모델을 기술한다. 5절에서는 시뮬레이션 결과를 기술하고 입출력 스케줄링 휴리스틱들을 비교, 평가한다. 마지막으로 6절에서는 결론을 기술한다.

2. LOB의 주 저장소로 사용되는 3차 저장장치를 지원하는 DBMS 모델

본 절에서는 LOB의 주 저장소로 사용되는 3차 저장장치를 지원하는 DBMS 모델을 기술한다. 이 모델은 본 본문에서 입출력 스케줄링을 통한 LOB 검색 질의

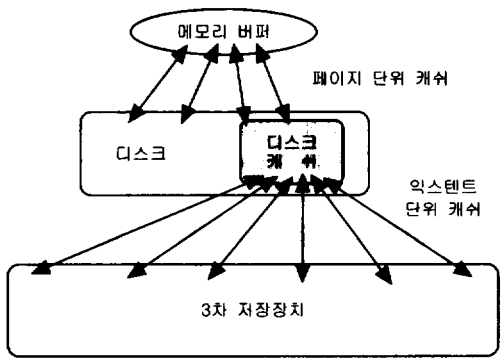
처리의 성능을 평가하기 위한 것으로서, 저장 계층 구조, 이중 버퍼링을 통한 3차 저장장치에서의 LOB 검색 모델, 디스크 캐싱, LOB을 저장하는 플래터의 구조 및 LOB 저장 모델, 그리고 LOB 검색 질의 모델 등에 대해 기술한다.

2.1 저장 계층 구조

(그림 1)은 3차 저장장치를 온라인 저장장치로 지원하는 DBMS의 저장 계층 구조를 나타낸 것이다. 저장 계층 구조 상에서 3차 저장장치는 최하위에 위치하고, 3차 저장장치에 저장된 데이터의 캐쉬 영역으로 디스크의 일부를 사용한다. 그리고 디스크에 저장된 데이터의 캐쉬 영역으로 메모리 버퍼가 사용된다.

이러한 저장 계층 구조에서 전통적인 정형 데이터를 검색하는 경우, 먼저 메모리 버퍼를 검색하고, 해당 데이터가 없으면 디스크 캐쉬를 검색하고, 거기에도 없으면 마지막으로 3차 저장장치에 대한 입출력을 통해 해당 데이터를 검색한다. 대용량의 멀티미디어 데이터를 검색하는 경우에는 일반적으로 메모리 버퍼에 해당 데이터 전체가 저장되기 어려우므로 디스크 또는 디스크 캐쉬를 먼저 검색하고 없을 경우, 3차 저장장치 입출력을 수행한다.

메모리 버퍼와 디스크 간의 캐쉬 단위는 기존 DBMS에서 메모리 버퍼와 디스크 간의 캐쉬 단위와 같은 페이지이다. 그러나 3차 저장장치와 디스크 간의 캐쉬 단위는 디스크 페이지보다 훨씬 큰 단위인 익스텐트(extent : 물리적으로 연속된 페이지들의 집합)이다. 이와 같이 페이지보다 큰 캐쉬 단위를 사용하는 이유는 3차 저장장치의 경우 입출력 지연 부담이 크기 때문이다.

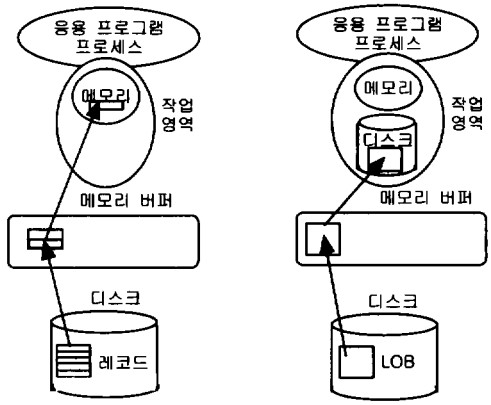


(그림 1) 3차 저장장치를 지원하는 DBMS의 저장 계층 구조

2.2 3차 저장장치로부터의 LOB 검색 모델

전통적인 디스크 기반 DBMS에서 정형 데이터 및 LOB의 검색 과정은 (그림 2)와 같다. (그림 2)(a)는 정형 레코드 검색 과정을 나타낸 것이다. 응용 프로그램이 요청하는 레코드 검색 질의를 처리하기 위해, 응용 프로그램 프로세스는 메모리 버퍼에서 해당 레코드를 저장하고 있는 페이지를 검색한다. 만약, 해당 페이지가 메모리 버퍼에 없으면 디스크로부터 읽어온다. 이후 해당 레코드는 응용 프로그램의 작업 영역 내 메모리로 옮겨지고 처리된다.

(그림 2)(b)는 LOB 검색 과정을 나타낸 것이다. 정형 레코드와는 달리 LOB은 데이터의 양이 크기 때문에 일반적으로 검색된 LOB을 응용 프로그램의 작업 영역 내 메모리에 전부 저장할 수 없는 것이 보통이다. 따라서 응용 프로그램 프로세스는 메모리 버퍼를 경유하여 작업 영역 내 디스크로 검색된 LOB을 저장한다. 이때, 검색할 LOB의 용량에 비해 사용 가능한 메모리 버퍼의 공간이 충분하지 못할 경우에는 LOB을 디스크로부터 부분부분 읽어와서 작업 영역 내의 디스크로 옮기게 된다.[1] 이후 검색된 LOB의 처리는 해당 LOB의 유형에 따라 적절한 도구(예를 들어, 텍스트 LOB의 경우 텍스트 편집기, 이미지 LOB의 경우 이미지 뷰어)를 이용하여 이루어진다.



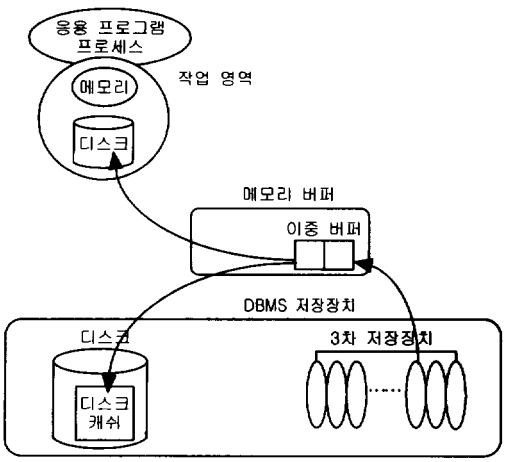
(a) 정형 레코드 검색 과정 (b) LOB 검색과정

(그림 2) 디스크 기반 DBMS에서 데이터 검색 과정

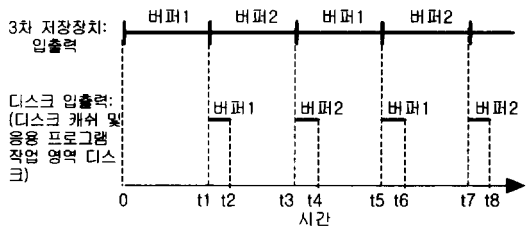
(그림 3)은 복수 개의 플래터로 구성된 광 디스크

1) 예를 들어, MS ODBC의 경우, SQLGetData(), SQLPutData()와 같은 LOB 처리 함수는 LOB을 연속적으로 부분 검색/저장(piecewise retrieval/storage)하는 기능을 제공한다.

쥬크박스, 또는 테이프 라이브러리와 같은 3차 저장장치를 지원하는 DBMS에서 LOB 검색 과정을 나타낸 것이다. 3차 저장장치에 대한 입출력 시간이 3차 저장장치에 저장된 LOB 검색 시간의 대부분을 차지하기 때문에 이중 버퍼(double buffer)를 사용하는 것이 효율적이다. 즉, LOB 검색 요청이 연속적으로 제기될 경우, 3차 저장장치에 대한 입출력을 통해 검색된 LOB을 한 버퍼에 채운 후, 다음 요청에 대해 다른 버퍼를 채울 동안 앞서 채운 버퍼의 LOB을 디스크 캐쉬 및 응용 프로그램의 작업 영역 내 디스크로 동시에 이동시킨다. (그림 4)는 이와 같은 이중 버퍼링의 효율성을 도시한 것이다. 이중 버퍼를 구성하는 두 버퍼를 각각 버퍼1, 버퍼2라 부르자. 첫 LOB 요청에 대해 3차 저장장치에 대한 입출력을 통해 검색된 LOB을 버퍼1에 채운다.(이를 위해 소요된 시간을 t1이라 하자.) 시각 t1에 두 번째 LOB 검색 요청을 위하여 3차 저장장치에 대한 입출력을 시작하고 검색된 LOB은 버퍼2에 채운다. 이때 3차 저장장치 입출력을 수행하는 프로세스와 다른 프로세스가 버퍼1의 LOB을 디스크 캐쉬 및 응용 프로그램의 작업 영역 내 디스크로 동시에 이동시키는 작업을 시작한다. 버퍼2를 채우는 작업이 시각 t3에 끝나고 버퍼1에 대한 디스크 입출력 처리가 t2에 끝난다고 하자. 일반적으로, 3차 저장장치의 입출력 시간이 디스크 입출력 시간보다 훨씬 길다. 즉, 시간(t3-t1)이 시간(t2-t1)보다 훨씬 크다. 따라서, 여러 응용 프로그램이 3차 저장장치에 저장된 LOB의 검색 요청을 연속적으로 해올 경우, 모든 요청에 대해 3차 저장장치로부터 각 응용 프로그램의 작업 영역 내 디스크로 LOB들이 이동되는 데 드는 총 시간은 이중 버퍼링을 사용함으로써 단일 버퍼링을 사용하는 경우에 비해 대략 디스크 입출력에 드는 총 시간만큼 줄일 수 있다. 이는 이중 버퍼링을 사용할 경우 3차 저장장치로부터 일련의 LOB 검색에 드는 총 시간은 대략 3차 저장장치에 대한 입출력 시간과 동일함을 의미한다.



(그림 3) 3차 저장장치로부터 이중 버퍼링을 이용한 LOB 검색 과정



(그림 4) 3차 저장장치 및 디스크에 대한 효율적 입출력 수행을 위한 이중 버퍼링

2.3 디스크 캐싱

디스크 캐쉬는 3차 저장장치에 저장되어 있는 데이터 중에서 자주 접근되는 데이터를 디스크에 캐쉬해 두어 3차 저장장치에 대한 입출력을 줄이려는 목적으로 사용된다. 그러나 대용량 LOB의 검색에 있어서 디스크 캐쉬의 효과는 다음과 같은 이유로 높지 않을 수 있다. <표 1>은 4가지 3차 저장장치의 용량과 디스크 캐쉬 대 3차 저장장치의 용량 비율을 나타낸 것이다. Sony의 광 쥬크박스(WORM optical jukebox)나 Exabyte의 테이프 라이브러리인 소형의 3차 저장장치 [8]의 경우, 3차 저장장치의 전체 용량에 대한 디스크 캐쉬 용량의 비율은 디스크가 10 GB일 때 1% 정도이고, 디스크가 50 GB일 때 1%~7% 정도이다. Metrum이나 DMS의 대형 테이프 라이브러리의 경우, 3차 저장장치의 전체 용량에 대한 디스크 캐쉬 용량의 비율은 1% 미만이다.[2] 이는 대용량 LOB의 검색에 있어, 디스크 캐쉬에서 해당 LOB을 찾는(cache hit) 경우가 빈번하지 못할 수 있음을 의미한다. 특히, 빈번히 참조되는 핫 데이터인 LOB에 대해서는 DBA가 데이터 배치 스키마 결정 시 3차 저장장치 대신 디스크에 상주시킬 것이다. 따라서, 본 논문에서는, 3차 저장장치에 저장

2) 이와 같은 수치는 현재의 일반적인 시스템 사양과 비교했을 때, 디스크의 용량 그리고 그 중 캐쉬 영역으로 할당된 용량을 충분히 크게 설정했을 때 얻어진 값이다.

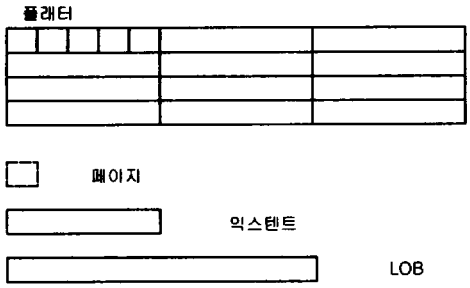
〈표 1〉 3차 저장장치 [8]의 용량과 3차 저장장치에 대한 디스크 캐쉬의 용량 비율

사양 \ 3차 저장장치			Sony optical jukebox	Exabyte small tape library	Metrum large tape library	DMS large tape library
플래터 크기(GB)			3.27	5	14.5	41
플래터 개수			100	116	600	320
총 용량(GB)			327	580	8700	13120
10GB 디스크 경우 3차 저장장치에 대한 디스크 캐쉬의 용량 비율(%)	캐쉬로 사용되는 디스크의 비율(%)	10(1GB)	0.31	0.17	0.01	0.01
		20(2GB)	0.61	0.34	0.02	0.02
		50(5GB)	1.53	0.86	0.06	0.04
50GB 디스크 경우 3차 저장장치에 대한 디스크 캐쉬의 용량 비율(%)	캐쉬로 사용되는 디스크의 비율(%)	10(5GB)	1.53	0.86	0.06	0.04
		20(10GB)	3.06	1.72	0.11	0.08
		50(25GB)	7.65	4.31	0.29	0.19

된 LOB의 검색 요청을 3차 저장장치 저장 공간 전체에 대해 임의의 위치에 저장된 LOB에 대해 발생한다고 가정하였다. 따라서, 디스크 캐싱은 고려하지 않고 3차 저장장치로부터의 LOB 검색을 대상으로 하였다.

2.4 플래터의 구조 및 LOB 저장 모델

광 디스크 쥬크박스 또는 테이프 라이브러리와 같은 3차 저장장치는 복수 개의 플래터로 구성된다. LOB이 저장되는 플래터의 모델은 (그림 5)와 같은 구조를 갖는다. 페이지는 3차 저장장치를 지원하는 DBMS가 관리하는 객체들 중 가장 작은 자료 저장 단위이다. 익스텐트는 인접한 페이지들의 집합으로 각각 동일한 크기를 갖는다. 플래터는 여러 개의 익스텐트들로 구성된다. LOB의 저장을 위하여 물리적으로 연속된 익스텐트를 할당하고, 같은 익스텐트 내에 서로 다른 LOB을 구성하는 데이터 페이지가 공존할 수 없다.



(그림 5) 플래터의 구조 및 LOB 저장 모델

2.5 LOB 검색 질의 모델

3차 저장장치에 저장된 LOB 검색의 질의 유형으로는 LOB 전체 검색과 LOB 부분 검색이 있다. LOB 전체 검색은 3차 저장장치에 저장된 LOB의 바이트 스트림(byte stream)을 처음에서 끝까지 모두 검색하는 것이고, LOB 부분 검색은 LOB의 일부 바이트 범위(byte range)만을 검색하는 것이다.

응용 프로그램은 자신이 요청하는 LOB이 디스크에 저장되어 있는지, 3차 저장장치에 저장되어 있는지, 3차 저장장치의 경우 어느 플래터에 저장되어 있는지 알 수 없다. DBMS의 질의 처리기는 요청된 질의를 처리하는 과정에서 시스템 카탈로그를 통해 LOB이 저장된 매체(디스크 또는 3차 저장장치) 및 볼륨과 3차 저장장치의 경우 어느 플래터에 저장되어 있는지를 파악하여 하부 저장 시스템 모듈에 해당 LOB의 검색을 요청하게 된다. 저장 시스템에 요청되는 3차 저장장치의 LOB 검색 질의는 (i, j, k)의 형태로 나타낼 수 있다. 여기서, i는 검색하려는 LOB이 저장된 플래터의 식별자이고, j와 k는 LOB을 저장하고 있는 일련의 익스텐트들 중 시작 익스텐트와 마지막 익스텐트의 식별자를 각각 나타낸다.

3. LOB 입출력 스케줄링 알고리즘

본 절에서는 3차 저장장치에 저장된 LOB의 검색을 효율적으로 수행하기 위한 3차 저장장치 입출력 스케줄링 휴리스틱 알고리즘을 기술한다.

3.1 개 요

3차 저장장치 입출력 스케줄링은 3차 저장장치 입출력 시간 중 플래터 교체 시간과 탐색 시간 등과 같은 지연 시간을 일련의 3차 저장장치 입출력 요청들에 대

하여 줄이는 것을 목표로 한다. 복수 개의 플래터로 구성된 3차 저장장치에 저장된 LOB 검색 질의가 연속적으로 제기되는 경우, 본 논문에서는 아래와 같은 3차 저장장치 입출력 스케줄링 알고리즘을 사용하여 LOB 검색을 수행한다. 먼저 스케줄링 알고리즘을 설명하기 위해 사용되는 변수와 함수들은 다음과 같다.

q : 3차 저장장치에 저장된 LOB 검색 질의
P : 플래터의 총 수
Group_i : 플래터 i에 저장된 LOB을 검색하고자 하는 질의들의 집합, $1 \le i \le P$
H : 입출력 요청에 대한 지연 시간을 줄이고자 사용되는 입출력 스케줄링 휴리스틱
select_platter(H) : 입출력 스케줄링 휴리스틱 H에 의해 한 플래터를 선택하는 함수
p : LOB을 검색하기 위해 선택된 플래터
execute_IO(p) : 플래터 p를 대상으로 LOB을 검색하기 위한 입출력을 수행하는 함수

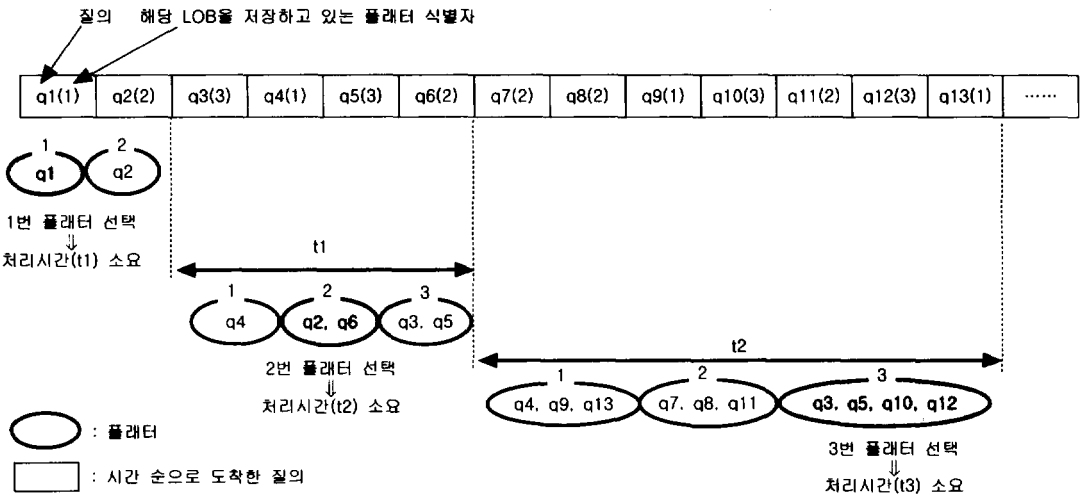
3차 저장장치 입출력 스케줄링 알고리즘을 C-like 언어를 사용하여 의사 코드 형태로 기술하면 다음과 같다.

```
do {      /* 모든 LOB 검색 질의들의 처리가 종료될
             때까지 아래의 과정을 반복 수행 */

   for each query q arrived thus far do
          /* 현재까지 도착된 질의들을 대상으로,
      if q is to access a LOB on platter i then
             검색하고자 하는 LOB이 저장된 플래터
          Group_i ← Group_i ∪ {q} , 1≤i≤P
             별로 질의들을 나눔
   endfor
          (플래터 별 질의 그룹핑) */

   p ← select_platter(H);
          /* 입출력 스케줄링 휴리스틱에 의해 한
             플래터를 선택 */

   execute_IO(p);
          /* 선택된 플래터를 대상으로 저장된 LOB
             을 검색하기 위한 입출력을 수행 */
} while(LOB retrieval query q exists);
```

플래터 별로 질의들을 그룹핑함으로써 3차 저장장치 접근 시 큰 지연 시간을 소요하는 플래터 교체 시간을 줄일 수 있다. 또한 각 플래터 별로 그룹핑된 질의들을 함께 처리함으로써 탐색시간 소요를 최적화할 수 있다. 플래터 별 질의 그룹핑은 3.2절에서 기술하고, 여러 가지 플래터 선택 휴리스틱에 대해서는 3.3절에서 기술한다.

3.2 플래터 별 질의 그룹핑

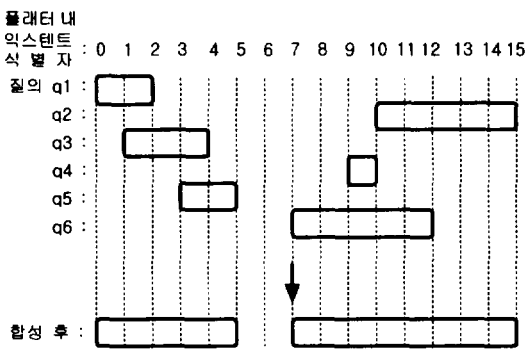
효율적인 LOB 검색을 위한 입출력 스케줄링 알고리즘의 첫 번째 단계는, 현재까지 도착한 질의들을 대상으로 해당 LOB을 저장하고 있는 플래터 별로 질의들을 나누는 것이다. 같은 플래터를 대상으로 하는 LOB 검색 질의들을 그 플래터가 입출력 드라이브에 적재되었을 때 모두 처리하는 것이 플래터 교체 횟수를 줄일 뿐 아니라, 이들 질의들이 요구하는 플래터 내 LOB들의 검색을 위한 탐색 시간을 최소화할 수 있다. (그림 6)은 시간 흐름에 따라 도착된 질의들을 대상으로 해당 LOB을 저장하고 있는 플래터 별로 질의들을 그룹핑하고 플래터를 선택하여 LOB 검색을 수행하는 과정을 예를 들어 나타낸 것이다. 3차 저장장치에 모두 3개의 플래터(1, 2, 3번 플래터)가 있다고 하자. q1~q13은 LOB에 대한 검색 질의들이고 특정 시간 간격으로 도착한다고 하자. 먼저 현재까지 질의 q1(1번 플래터에 저장된 LOB의 검색 요청)과 q2(2번 플래터에 저장된 LOB의 검색 요청)가 도착했다고 하자. q1과 q2를 대상으로 플래터 별 그룹핑을 수행하면 1번 플래터에 해당되는 q1과 2번 플래터에 해당되는 q2가 각각 한 질의 그룹을 형성한다. 라운드 로빈(플래터들을 한 번씩 순회하면서 질의 처리 요청이 있는 플래터를 선택) 정책에 의해 플래터를 선택하여 3차 저장장치 입출력을 수행할 경우, 먼저 1번 플래터가 선택되어 1번 플래터를 입출력 드라이브에 적재하고 q1을 처리한다. 이에 소요된 시간을 t1이라 하자. 이 t1의 시간동안 새로 도착된 질의들(q3, q4, q5, q6)을 추가하여 플래터 별 질의 그룹핑을 다시 수행한다. 그룹핑 결과, 1번 플래터에 해당되는 질의는 q4이고, 2번 플래터에 해당되는 질의들은 q2, q6이고, 3번 플래터에 해당되는 질의들은 q3, q5가 된다. 이 세 그룹 중 라운드 로빈에 의해 2번 플래터가 선택이 되어 1번 플래터를 들어내고 2번 플



(그림 6) 플래터 별 질의 그룹핑 및 플래터 선택

래터를 적재한 후 q2와 q6을 처리한다. 이때 t2의 시간이 소요되었다고 하자. 그 동안 새로 도착된 질의들(q7, q8, q9, q10, q11, q12, q13)을 추가하여 플래터 별 질의 그룹핑을 다시 수행하고 라운드 로빈에 의해 3번 플래터의 질의들을 처리하는 과정이 반복된다.

플래터 별로 그룹핑된 LOB 검색 질의들은 서로 같은 LOB을 대상으로 할 수 있기 때문에 요청된 LOB 데이터들의 바이트 범위는 서로 중복될 수 있다. 이와 같은 경우 중복되는 LOB 데이터의 바이트 스트림들을 합쳐서(merge) 한 번에 모두 읽어들임으로써 플래터 내에서 탐색 시간을 최소화할 수 있다. (그림 7)은 한 플래터 내에 저장된 LOB 데이터들에 대한 검색 질의들(q1, q2, q3, q4, q5, q6)에 대해 질의들 간에 중복된 바이트 스트림을 합치는 과정을 나타낸 것이다. 그림 상단의 0, 1, ···, 15는 플래터 내 익스텐트의 식별자를 표시한 것이다. 예를 들어, 질의 q1은 0번 익스텐트에서부터 2번 익스텐트에 걸쳐 저장된 LOB을 요청하는 질의다. 질의 q1, q3, q5 들이 요청한 LOB의 바이트 스트림은 서로 중복되므로 합쳐서 한번에 읽어들이는 것이 효율적이다. 즉, 0번 익스텐트에서부터 5번 익스텐트까지를 검색하면 질의 q1, q3, q5가 모두 처리되며 이 경우 세 질의에 대해 1회의 탐색만 요구된다. 마찬가지로 질의 q2, q4, q6 들이 요청한 LOB의 바이트 스트림이 서로 합쳐져 7번 익스텐트에서부터 15번 익스텐트까지가 검색 대상이 된다.



(그림 7) 플래터 내 중복 질의 합성

3.3 플래터 선택 휴리스틱

플래터 별 질의 그룹핑 이후, 특정 플래터를 선택하는 휴리스틱에는 여러 가지가 있다. 가장 간단한 휴리스틱으로는 (그림 6)의 예에서 사용한 플래터들 간에 차례를 정해놓고 이들을 한번씩 순회하면서 질의가 존재하는 플래터를 선택하는 라운드 로빈이 있다. 본 논문에서는 라운드 로빈 외에 다음과 같은 두 가지 플래터 선택 휴리스틱들을 제시하고 성능 평가에 사용하였다. 첫째 휴리스틱은 플래터 내의 모든 질의들의 수행에 드는 총 시간이 가장 긴 플래터를 선택하는 것이다. 둘째 휴리스틱은 플래터 내의 질의 수가 가장 많은 플래터를 선택하는 것이다.[3] 이들 두 휴리스틱은

3) 본 논문에서는 이 두 가지 휴리스틱 외에 다양한 휴리스틱들을 다수 고려하였다. 그들 중 상기 두 가지 휴리스틱이 4. 5절에서 기술할 성능 평가 실험에서 가장 좋은 성능을 나타내었다.

모두 작업량이 적은 플래터 처리를 지연하고 작업량이 많이 누적된 플래터를 우선 처리함으로써, 3차 저장장치의 높은 지연 부담(플래터 교체 및 탐색)에 대해 질의 처리량을 늘리고 작업량 누적이 적은 플래터에는 후속 질의 요청이 더 누적되기를 기다리고자 하는 목적을 갖는다. 즉, 시간이 오래 걸리는 플래터 교체가 한 번 일어날 때마다 그 부담을 충분히 상쇄할 수 있을 만큼의 많은 양의 질의를 처리함으로써 시스템 전체적인 성능 향상을 의도하는 것이다.

본 논문에서는 라운드 로빈 정책을 RR이라 나타내고, 상기 두 휴리스틱을 각각 MPT(Max. Processing Time), MQN(Max. Query Number)이라 나타내기로 한다. 이들 간의 성능 비교·평가에 있어, 일체의 입출력 스케쥴링을 수행하지 않고 질의들이 도착되는 대로 하나씩 차례로 수행하는 정책 즉, FCFS(First Come First Serve)와 최적의 플래터 선택을 수행하는 Opt (Optimal)를 각각 최악의 성능과 최상의 성능을 나타내는 경우로서 비교 대상으로 하였다.[4)]

4. 시뮬레이션 모델

본 절에서는 전 절에서 기술한 3차 저장장치로부터의 LOB 검색을 위한 입출력 스케쥴링 알고리즘들(FCFS, RR, MPT, MQN, Opt)의 성능을 비교·평가하기 위한 시뮬레이션 모델을 기술한다.

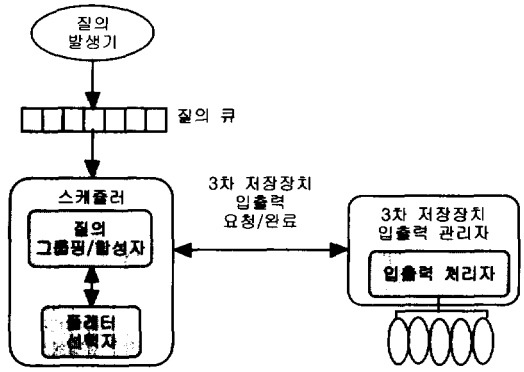
4.1 시스템 모델링

시뮬레이터 구현은 C 언어를 사용하였고, Pentium PC (NT 4.0)환경에서 실험하였다. 3차 저장장치의 종류로서 광 디스크 쥬크박스와 테이프 라이브러리의 두 가지 경우에 대하여 상기 알고리즘들의 성능을 평가하였다. 시스템 모델은 (그림 8)과 같으며, 아래 모듈들로 구성된다.

1. 질의 발생기(Query Generator) : 시스템에서 수행되는 다양한 LOB 검색 질의를 발생시켜 질의 큐에 저장한다.
2. 스케쥴러(Scheduler) : 3차 저장장치에 대한 입출력 요청들을 스케쥴링한다.
   1) 질의 그룹핑/합성자(Query Merger) : 현재까지 도착된 질의들을 대상으로, 플래터 별 질의 그룹핑을 수행하고 플래터 내 중복 질의 합성을 수행한다.
   2) 플래터 선택자(Platter Selector) : 3차 저장장치 입출력을 수행할 플래터를 선택한다.
3. 3차 저장장치 입출력 관리자(Tertiary I/O Manager) : 입출력 처리자(I/O Processor)에 의해 3차 저장장치 입출력을 수행한다.

4.2 질의 처리 모델링

질의가 처리되는 과정은 다음과 같다. 질의 발생기는 검색할 LOB의 크기, 총 질의 개수에 따라 질의를 특정 시간 간격으로 생성하여 질의 큐에 차례로 저장한다. 스케쥴러는 질의 큐에 현재까지 도착한 질의들을 대상으로 스케쥴링을 수행한다. 먼저 스케쥴러 내의 질의 그룹핑/합성자가 검색하고자 하는 LOB이 저장된 플래터 별로 질의들을 나누고 필요시 중복 질의를 합성한다. 다음은 스케쥴러 내의 플래터 선택자가 3차 저장장치 입출력을 수행할 대상 플래터를 스케쥴링 알고리즘에 의해 선택한다. 이후 스케쥴러는 3차 저장장치 입출력 관리자에게 3차 저장장치 입출력 요청 메시지를 보낸다. 3차 저장장치 입출력 관리자 내 입출력 처리자는 3차 저장장치에서 버퍼로 LOB을 전송하고, 전송이 끝나면 스케쥴러에게 3차 저장장치 입출력 완료 메시지를 보낸다.



(그림 8) 시스템 모델링

4.3 시뮬레이션 파라미터

본 논문에서 제시된 기법의 성능 실험을 위해서는 DBMS를 구성하는 페이지 및 익스텐트의 크기, 3차 저장 장치로부터 검색하고자 하는 LOB의 크기, 그리고 3차 저장장치의 플래터 크기, 개수, 드라이브 수와 플래

4) 최적의 플래터 선택은 물론 질의 도착 도중에 동적으로 구할 수 없다. 이는 4, 5절의 성능 평가 실험에 사용된 질의 스트림 전체에 대해 최적해를 해 공간 전체를 대상으로 구한 것이다.

터 교체 시간, 데이터 전송율 및 탐색율 등의 정의가 필요하다. 실험에서 사용되는 파라미터와 그 설정값은 <표 2>~<표 5>와 같다. <표 2>는 3차 저장장치를 지원하는 DBMS 관련 파라미터를 나타낸 것으로 <표 2>의 설정값에 의해 한 익스텐트의 크기는 512KB이다. <표 3>은 3차 저장장치 파라미터를 나타낸 것으로 광 디스크 쥬크박스의 경우에는 Sony사의 소형 광 디스크 쥬크박스를, 테이프 라이브러리의 경우에는 Exabyte사의 소형 테이프 라이브러리의 명세서를 참조하였다[8]. 광 디스크 쥬크박스와 테이프 라이브러리 모두 한 플래터의 크기를 3GB로 하였고, 3차 저장장치의 총 용량은 30GB로 하였다. <표 4>는 데이터베이스 파라미터를 나타낸 것이다. 3차 저장장치에 저장된 LOB의 크기로는 1M, 10M, 50M, 100M의 다섯 종류를 고려하였고, 하나의 LOB은 2개의 플래터에 분할되어 저장되지 않고 한 플래터 내에 모두 저장된다고 가정하였다. <표 5>는 질의 관련 파라미터를 나타낸 것이다. 질의 도착률이란 한 질의가 도착한 후 다음 질의가 도착할 때까지의 시간 간격을 정하기 위한 값으로 시스템의 질의 부하를 결정하는 파라미터다. 질의 도착률 값의 설정은 다음과 같이 하였다: 하나의 LOB 검색 질의가 도착하였을 때 아무 기다림 없이 곧바로 3차 저장장치 입출력을 수행하여 그 질의를 처리하는 데 소요되는 총 시간의 일정 비율(%)에 해당되는 시간이 경과한 후 다음 LOB 검색 질의가 도착한다.

〈표 2〉 3차 저장장치를 지원하는 DBMS 관련 파라미터

파라미터	내 용	설정값
PageSize	한 페이지의 크기	16KB
ExtentSize	한 익스텐트를 구성하는 페이지 수	32페이지

〈표 3〉 3차 저장장치 파라미터

파라미터	내 용	광 디스크 쥬크박스 실험에서의 설정값	테이프 라이브러리 실험에서의 설정값
SwitchTime	플래터 교체 시간	8sec	171sec
TransferRate	데이터 전송율	0.8MB/sec	0.47MB/sec
SeekRate	탐색율	-	36.2MB/sec
SeekStartTime	탐색 시작 시간	0.5sec	16sec
DriveNum	드라이브 개수	1개	1개
PlatterSize	한 플래터의 크기	3GB	3GB
PlatterNum	플래터의 개수	10개	10개
TotalCapacity	3차 저장장치의 총 용량	30GB	30GB

〈표 4〉 데이터베이스 파라미터

파라미터	내 용	설정값
ExtentNum	한 플래터 내의 익스텐트 개수	6,144개(30G)
LOBSize	플래터에 저장된 LOB의 크기	1, 10, 50, 100MB

〈표 5〉 질의 관련 파라미터

파라미터	내 용	설정값
ArrivalRate	질의 도착률 (본문 참조)	10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, 100%

5. 성능 평가 결과

본 절에서는 4절에서 제시한 시뮬레이션 모델을 이용하여 3차 저장장치에 대한 입출력 스케줄링 알고리즘에 의한 LOB 검색 질의 처리의 성능 평가 결과를 기술한다.

5.1 성능 평가 척도

성능 평가 척도로 질의 별 평균 응답 시간과 총 질의 처리 시간을 사용하였다. 질의 별 평균 응답 시간은 각 질의가 시스템에 도착한 후 처리되기까지의 평균 응답 시간을 의미하고, 총 질의 처리 시간은 실험에 사용된 질의 스트림의 첫 질의가 도착한 후부터 마지막 질의가 모두 처리되기까지의 총 시간을 의미한다.

5.2 실험 결과

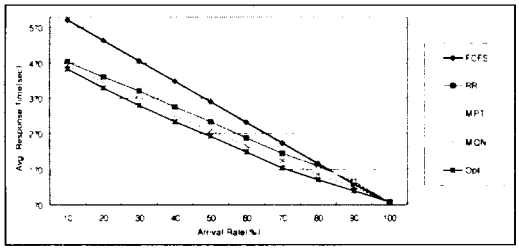
복수 개의 플래터로 구성된 광 디스크 쥬크박스나 테이프 라이브러리와 같은 3차 저장장치는 서로 다른 성능 특성을 갖는다. 광 디스크 쥬크박스의 경우, 플래터 교체 시간에 비해, 데이터 전송 시간이 길고 플래터 내 탐색 시간이 각 질의에 대해 거의 동일하다는 특성을 갖는다. 반면에 테이프 라이브러리 경우, 플래터를 들어내기 위해서 테이프를 되감는 시간을 소요하기 때문에 플래터 교체 시간이 상당히 길고, 플래터 내 탐색 시간이 데이터의 위치에 따라 크게 다를 수 있다는 특성을 갖는다. 이와 같은 3차 저장장치들의 특성을 고려해서 성능 평가 실험 결과는 3차 저장장치가 광 디스크 쥬크박스인 경우와 테이프 라이브러리인 경우로 나누어 기술한다.

5.2.1 광 디스크 쥬크박스

검색하고자 하는 LOB 크기가 매 질의마다 1M, 10M,

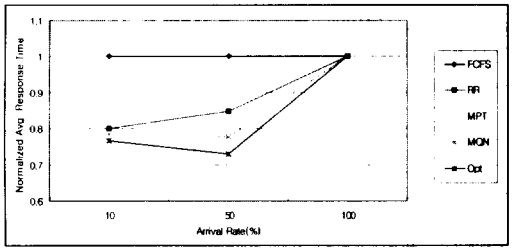
50M, 100M 중의 하나를 임의로 선택하여 실험하였다. 1회의 실험에서 요청되는 질의의 수는 20개이고, 실험의 총 횟수는 20번으로 그 평균을 구했다.[5] 먼저 질의별 평균 응답 시간은 다음과 같다.

(그림 9)는 질의 도착률 변화에 따른 질의 별 평균 응답 시간을 나타낸 것이다. 모든 알고리즘의 경우에 질의 도착률이 증가할수록(질의 도착 시간 간격이 커질수록) 질의 별 평균 응답 시간이 줄어들었다. 이는 질의 도착률이 증가할수록 질의마다 처리되기까지 질의 큐에서 기다리는 시간이 줄기 때문이다. Opt 다음으로 MQN의 성능이 우수하고, MPT, RR, FCFS 순으로 성능 차이를 나타냈다. 질의 도착률이 10%~30%인 경우에는 MQN과 MPT가 성능 차이를 보이지 않았지만, 그보다 큰 값의 질의 도착률에 대해서는 MQN의 성능이 더 우수하였다.



(그림 9) 광 디스크 쥬크박스로부터 LOB 검색 : 질의 도착률 변화에 따른 질의 별 평균 응답 시간

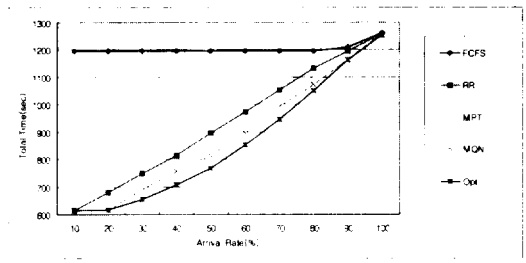
(그림 10)은 FCFS를 기준으로 다른 입출력 스케쥴링 휴리스틱의 질의 별 평균 응답 시간을 정규화하여 나타낸 것이다. Opt는 FCFS와 비교하여 질의 도착률이 10%일 때 질의 별 평균 응답 시간이 0.76배이고, 50%일 때 0.74배로 나타났다. MQN의 경우에는 질의 도착률이 10%와 50%일 때 모두 0.78배로 나타났다.



(그림 10) 광 디스크 쥬크박스로부터 LOB 검색 : 질의 도착률 변화에 따른 정규화된 질의 별 평균 응답 시간(FCFS 기준)

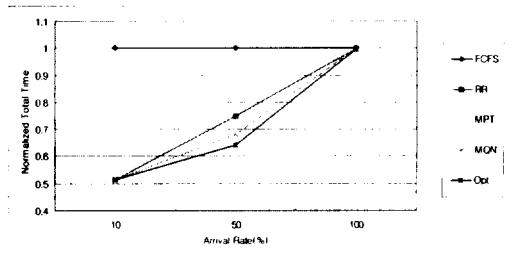
5) 최적 알고리즘 Opt 이외의 알고리즘에 대한 더 많은 질의 수와 실험 횟수에 의한 결과는 5.2.3절을 참조. 본 절에서는 Opt의 경우 때문에 20개의 질의에 대해 20회의 실험으로 제한하였다.

(그림 11)은 질의 도착률 변화에 따른 총 질의 처리 시간을 나타낸 것이다. FCFS의 경우에는 질의 도착률이 80%가 될 때까지 총 질의 처리 시간에 변화가 없다가 그 이후 증가하기 시작하였다. FCFS를 제외한 나머지 알고리즘의 경우에는 질의 도착률이 증가할수록 총 질의 처리 시간이 길어졌다. 이는 이들 알고리즘의 경우 FCFS와는 달리 입출력 스케쥴링의 효과가 있음을 반증하는 것이다. 질의 별 평균 응답 시간과 마찬가지로 Opt 다음으로 MQN의 성능이 우수하고, MPT, RR, FCFS 순으로 성능 차이를 나타냈다.



(그림 11) 광 디스크 쥬크박스로부터 LOB 검색 : 질의 도착률 변화에 따른 총 질의 처리 시간

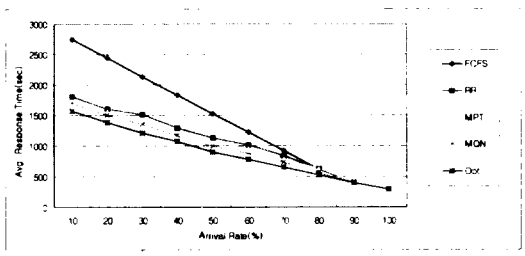
(그림 12)는 FCFS를 기준으로 다른 입출력 스케쥴링 휴리스틱의 총 질의 처리 시간을 정규화하여 나타낸 것이다. Opt는 FCFS와 비교하여 질의 도착률이 10%일 때 총 질의 처리시간이 0.51배 정도이고, 50%일 때 0.64배로 나타났다. MQN의 경우에는 질의 도착률이 10%일 때 Opt와 마찬가지로 0.51배이고, 50%일 때 0.68배로 나타났다.



(그림 12) 광 디스크 쥬크박스로부터 LOB 검색 : 질의 도착률 변화에 따른 정규화된 총 질의 처리 시간(FCFS 기준)

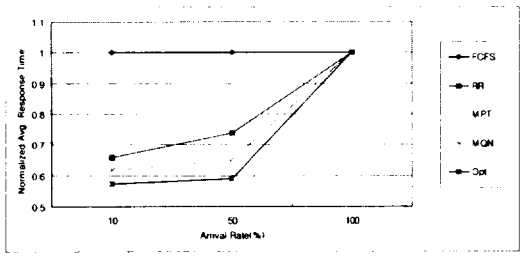
5.2.2 테이프 라이브러리

검색하고자 하는 LOB 크기가 매 질의마다 1M, 10M, 50M, 100M 중의 하나를 임의로 선택하여 실험하였다. 1회의 실험에서 요청되는 질의의 수는 20개이고, 실험의 총 횟수는 20번으로 그 평균을 구했다. 먼저 질의별 평균 응답 시간은 다음과 같다. (그림 13)은 질의 도착률 변화에 따른 질의 별 평균 응답 시간을 나타낸 것이다. 광 디스크 쥬크박스에서와 같은 이유로 모든 알고리즘의 경우에 질의 도착률이 증가할수록 질의 별 평균 응답 시간이 줄어들었다. 그러나 테이프 라이브러리는 광 디스크 쥬크박스에 비해 매체 교체 시간이 느리고 데이터 전송률이 낮기 때문에 질의 별 평균 응답 시간이 광 디스크 쥬크박스에 비해 더 컸다. 광 디스크 쥬크박스의 경우와 같이 Opt 다음으로 MQN의 성능이 우수하고, MPT, RR, FCFS 순으로 성능 차이를 나타냈다. 그러나 FCFS와 다른 알고리즘 간의 성능 차이는 광 디스크 쥬크박스인 경우보다 더 큰 격차를 나타냈다.



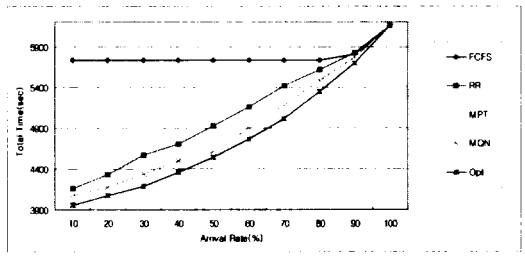
(그림 13) 테이프 라이브러리로부터 LOB 검색 : 질의 도착률 변화에 따른 질의 별 평균 응답 시간

(그림 14)는 FCFS를 기준으로 다른 입출력 스케쥴링 휴리스틱의 질의 별 평균 응답 시간을 정규화하여 나타낸 것이다. Opt는 FCFS와 비교하여 질의 도착률이, 10%일 때 질의 별 평균 응답 시간이 0.57배 정도이고, 50%일 때 0.59배로 나타났다. MQN의 경우에는 질의 도착률이 10%일 때 0.61배이고, 50%일 때 0.65배로 나타났다.



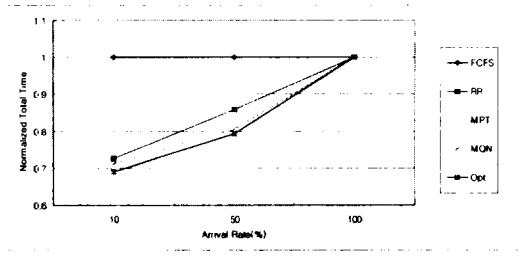
(그림 14) 테이프 라이브러리로부터 LOB 검색 : 질의 도착률 변화에 따른 정규화된 질의 별 평균 응답 시간(FCFS 기준)

(그림 15)는 질의 도착률 변화에 따른 총 질의 처리 시간을 나타낸 것이다. 광 디스크 쥬크박스의 경우와 같이 FCFS의 경우에는 질의 도착률이 80%가 될 때까지 총 질의 처리 시간에 변화가 없다가 그 이후 증가하였고 나머지 알고리즘의 경우에는 질의 도착률이 증가할수록 총 질의 처리 시간이 길어졌다. 또한 질의 별 평균 응답 시간의 경우와 같이 테이프 라이브러리는 광 디스크 쥬크박스에 비해 매체 교체 시간이 느리고 데이터 전송률이 낮기 때문에 총 질의 처리 시간이 광 디스크 쥬크박스에 비해 더 컸다. 광 디스크 쥬크박스의 경우와 같이 Opt 다음으로 MQN의 성능이 우수하고, MPT, RR, FCFS 순으로 성능 차이를 나타냈다.



(그림 15) 테이프 라이브러리로부터 LOB 검색 : 질의 도착률 변화에 따른 총 질의 처리 시간

(그림 16)은 FCFS를 기준으로 다른 입출력 스케쥴링 휴리스틱의 총 질의 처리 시간을 정규화하여 나타낸 것이다. Opt는 FCFS와 비교하여 질의 도착률이 10%일 때 총 질의 처리시간이 0.68배 정도이고, 50%일 때 0.79배로 나타났다. MQN의 경우에는 질의 도착률이 10%일 때 0.71배이고, 50%일 때 0.8배로 나타났다.



(그림 16) 테이프 라이브러리로부터 LOB 검색 : 질의 도착률 변화에 따른 정규화된 총 질의 처리 시간(FCFS 기준)

5.2.3 대규모 질의 부하 실험 결과

검색하고자 하는 LOB의 크기가 1M, 10M, 50M, 100M 중의 하나를 임의로 선택하였고, 1회의 실험에서 요청되는 질의의 수는 100개이고, 실험의 총 횟수는 100회로 그 평균을 구하였다. 5.2.1절과 5.2.2절에서 기술한 20개의 질의/20번의 실험 결과와 비교해서 모든 알고리즘에 대해 질의 별 평균 응답 시간과 총 질의 처리 시간이 증가했다. 질의 별 평균 응답 시간이 증가한 이유는 질의의 개수가 증가할수록 질의마다 처리되기까지 질의 큐에서 기다리는 시간이 늘어나기 때문이고, 총 질의 처리 시간이 증가한 이유는 질의의 개수가 증가할수록 마지막 질의가 모두 처리되기까지의 시간이 늘어나기 때문이다. <표 6>은 5.2.1절과 5.2.2절에서 기술한 20개의 질의/20번의 실험 결과와 100개의 질의/100번의 실험 결과에 따라 광 디스크 쥬크박스와 테이프 라이브러리의 경우로 나누어 FCFS를 기준으로 다른 입출력 스케쥴링 휴리스틱의 질의 별 평균 응답 시간과 총 질의 처리 시간을 정규화하여 비교한 것이다. 20개의 질의/20번의 실험 결과와 비교하여 모든 휴리스틱 알고리즘에서 질의 별 평균 응답 시간과 총 질의 처리 시간이 낮게 나타났다. 이는 질의의 개수가 증가할수록 플래터 별 질의 그룹핑을 이용한 입출력 스케쥴링 효과가 더 커지기 때문이다.

6. 결 론

본 논문에서는 전통적인 디스크 기반의 DBMS에 광 디스크 쥬크박스 또는 테이프 라이브러리와 같은 3차 저장장치를 부착하여 온라인 저장장치로 사용할 경우, 3차 저장장치에 저장된 LOB을 효율적으로 검색하기 위한 입출력 스케쥴링 알고리즘에 대해 연구하였다.

데이터 접근 시 기존 DBMS의 저장 매체인 디스크보다 지연 시간이 매우 큰 3차 저장장치의 성능 특성과 데이터의 용량이 큰 LOB의 특성을 고려해서 본 논문에서는 3차 저장장치에 대한 입출력을 효율적으로 수행하기 위해 다음과 같은 입출력 스케쥴링 알고리즘을 연구하였다.

연속해서 발생하는 LOB 검색 질의들에 대하여 현재까지 도착된 질의들을 대상으로, 검색하고자 하는 LOB이 저장된 플래터 별로 질의들을 나눈다(플래터 별 질의 그룹핑). 플래터 선택 휴리스틱에 의해 한 플래터를 선택하고, 선택된 플래터를 대상으로 3차 저장장치 입출력을 수행한다. 모든 LOB 검색 질의들의 처리가 종료될 때까지 위의 과정을 반복한다.

상기 알고리즘에서 플래터 선택 휴리스틱으로는 RR(라운드 로빈 : 플래터들간에 차례를 정해놓고 이들을 한번씩 순회하면서 질의가 존재하는 플래터를 선택), MPT(Max. Processing Time : 플래터 내의 모든 질의들의 수행에 드는 총 시간이 가장 긴 플래터를 선택), MQN(Max. Query Number : 플래터 내의 질의 수가 가장 많은 플래터를 선택)을 고려하였고, 이들 간의 성능 비교·평가에 있어, 일체의 입출력 스케쥴링을 수행하지 않고 질의들이 도착되는 대로 하나씩 차례로 수행하는 FCFS(First Come First Serve)와 최적의 플

〈표 6〉 질의 부하에 따른 입출력 스케쥴링 휴리스틱 간의 비교

3차 저장장치의 종류	실험 / 질의 도착률 / 입출력 스케쥴링 알고리즘	100개의 질의/100번의 실험				20개의 질의/20번의 실험			
		10%		50%		10%		50%	
		질의 별 평균 응답 시간	총 질의 처리시간	질의 별 평균 응답 시간	총 질의 처리시간	질의 별 평균 응답 시간	총 질의 처리시간	질의 별 평균 응답 시간	총 질의 처리시간
광 디스크 쥬크박스	FCFS	1	1	1	1	1	1	1	1
	RR	0.37	0.43	0.32	0.61	0.80	0.66	0.85	0.74
	MPT	0.38	0.42	0.32	0.61	0.78	0.63	0.79	0.69
	MQN	0.35	0.42	0.27	0.60	0.78	0.62	0.78	0.65
	Opt	-	-	-	-	0.77	0.57	0.73	0.59
테이프 라이브러리	FCFS	1	1	1	1	1	1	1	1
	RR	0.50	0.53	0.45	0.68	0.51	0.73	0.75	0.86
	MPT	0.48	0.53	0.42	0.67	0.51	0.70	0.72	0.83
	MQN	0.48	0.53	0.41	0.67	0.51	0.71	0.68	0.81
	Opt	-	-	-	-	0.51	0.69	0.64	0.79

래터 선택을 수행하는 Opt(Optimal)를 각각 최악의 성능과 최상의 성능을 나타내는 경우로서 비교 대상(baseline algorithm)으로 하였다.

3차 저장장치 입출력 스케줄링 알고리즘에 따른 질의 처리 성능 평가를 위한 실험은 크게 광 디스크 쥬크박스의 경우와 테이프 라이브러리의 경우로 나누어 수행하였다. 성능 평가 척도로는 질의 별 평균 응답 시간과 총 질의 처리 시간을 사용하였다. 질의 별 평균 응답 시간은 각 질의가 시스템에 도착한 후 처리되기까지의 평균 응답 시간을 의미하고, 총 질의 처리 시간은 실험에 사용된 질의 스트림의 첫 질의가 도착한 후부터 마지막 질의가 모두 처리되기까지의 총 시간을 의미한다.

실험 결과를 정리하면 다음과 같다.

1. 광 디스크 쥬크박스와 테이프 라이브러리 경우 모두 Opt 다음으로 MQN의 성능이 우수하고, MPT, RR, FCFS 순으로 성능 차이를 나타냈다. 즉, 플래터 처리를 균등히 하기 위하여 단순 스케줄링을 수행하는 RR보다 작업량이 많이 누적된 플래터부터 처리하는 휴리스틱(MQN, MPT)이 더 우수하였다.
2. 광 디스크 쥬크박스와 테이프 라이브러리 경우 모두 MQN과 MPT의 성능이 Opt의 성능에 근접하게 나타나 준 최적 스케줄링의 효과를 보였다.
3. 질의 별 평균 응답 시간에 대한 성능 평가 : 모든 알고리즘에 대하여 질의 도착률이 증가할수록(질의 도착 시간 간격이 커질수록) 질의 별 평균 응답 시간이 줄어들었다. 이는 3차 저장장치 접근 시 입출력 자체가 소요하는 시간뿐 아니라 다른 질의를 위한 입출력 시간 동안 기다리는 시간이 큼을 의미한다.
4. 총 질의 처리 시간에 대한 성능 평가 : FCFS의 경우에는 질의 도착률이 80%가 될 때까지 총 질의 처리 시간에 변화가 없다가 그 이후 증가하였고 나머지 알고리즘의 경우에는 질의 도착률이 증가할수록 총 질의 처리 시간이 길어졌다. 이는 FCFS 외의 알고리즘에서 입출력 스케줄링의 효과가 크다는 것을 반증하는 것이다.
5. 광 디스크 쥬크박스의 경우, 질의 별 평균 응답 시간에 대한 평가 : 질의 도착률이 10%~30%인 경우에는 MQN과 MPT가 성능 차이를 보이지 않았지만, 그보다 큰 값의 질의 도착률에 대해서는 MQN의 성능이 더 우수하였다.
6. 테이프 라이브러리의 경우, 질의 별 평균 응답 시간과 총 질의 처리 시간에 대한 평가 : FCFS와 다른 알고리즘 간의 성능 차이가 광 디스크 쥬크박스인 경우보다 더 크게 나타났다. 이는 테이프 라이브러리의 경우, 플래터 교체 시 되감는 시간의 소요로 플래터 교체 시간이 크고, 탐색 시간도 디스크에 비해 크기 때문이다.
7. 100개의 질의/100번의 실험을 통한 대규모 실험에서는 20개의 질의/20번의 실험 결과와 비교해서 모든 알고리즘에 대해 질의 별 평균 응답 시간과 총 질의 처리 시간이 커졌다. 이는 총 질의 처리시간의 경우는 질의 수가 많아졌으므로 당연한 결과이고, 질의 별 평균 응답 시간의 경우에는 각 질의가 처리되기까지 기다리는 시간이 커진 탓이다. 그러나, FCFS를 기준으로 다른 입출력 스케줄링 휴리스틱의 질의 별 평균 응답 시간과 총 질의 처리 시간을 정규화하여 비교한 결과에서는 20개의 질의/20번의 실험 결과와 비교하여 모든 휴리스틱 알고리즘에서 질의 별 평균 응답 시간과 총 질의 처리 시간이 개선되었다. 즉, 시스템에 질의 처리량이 많을수록 입출력 스케줄링의 효과가 더 크게 나타났다.

참 고 문 헌

[1] M. Carey et al., "Tapes Hold Data, Too : Challenges of Tuples on Tertiary Store," Proc. ACM SIGMOD Int'l Conf., pp.413-417, 1993.

[2] S. Prabhaker et al., "Tertiary Storage : Current Status and Future Trends," Tech. Rep. TRCS 96-21, U. of California, Santa Barbara, 1996.

[3] M. Stonebraker, "Managing Persistent Objects in a Multi-level Store," Proc. ACM SIGMOD Int'l Conf., pp.2-11, 1991.

[4] T. Johnson and E. Miller, "Performance Measurements of Tertiary Storage Devices," Proc. Int'l Conf. on VLDB, pp.50-61, Aug. 1998.

[5] J. Yu and D. Dewitt, "Query Pre-execution and Batching in Paradise : A Two-Pronged Approach to the Efficient Processing of Queries on Tape-Resident Data Sets," Int'l Conf. on Scientific and Statistical Database Management, 1997.

[6] S. Sarawagi and M. Stonebraker, "Single Query Optimization for Tertiary Memory," Tech. Rep.

S2K-94-45, Computer Science Div., U. C. Berkeley, 1993.

[7] S. Sarawagi, "Database Systems for Efficient Access to Tertiary Memory," Proc. of Int'l IEEE Symp. on Mass Storage Systems, pp.120-126, 1995.

[8] S. Sarawagi, "Query Processing in Tertiary Memory Databases," Proc. Int'l Conf. on VLDB, pp.585-595, 1995.

[9] S. Sarawagi, "Query Processing in Tertiary Memory Databases", Ph. D thesis, Univ. of California, Berkeley, Dec. 1996.

[10] S. Sarawagi and M. Stonebraker, "Reordering Query Execution in Tertiary Memory Databases," Proc. Int'l Conf. on VLDB, pp.156-167, 1996.

[11] B. Hillyer and A. Silberschatz, "Random I/O Scheduling in Online Tertiary Storage," Proc. ACM SIGMOD Int'l Conf. on Management of Data, pp.195-204, 1996.

[12] B. Hillyer et al., "Scheduling and Data Replication to Improve Tape Jukebox Performance," Proc. Int'l Conf. on Data Eng., pp.532- 541, Mar. 1999.

[13] J. Myllymaki and M. Livny, "Disk-Tape Joins: Synchronizing Disk and Tape Access," Proc. ACM SIGMETRICS, pp.279-290, 1995.

[14] J. Myllymaki and M. Livny, "Efficient Buffering for Concurrent Disk and Tape I/O," Proc. Performance, pp.453-471, 1996.

[15] J. Myllymaki and M. Livny, "Relational Joins for Data on Tertiary Storage," Proc. Int'l Conf. on Data Eng., pp.159-168, 1997.

[16] A. Kraiss et al., "Tape-Disk Join Strategies under Disk Contention," Proc. Int'l Conf. on Data Eng., pp.552-559, Mar. 1999.

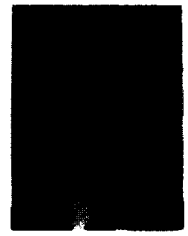
[17] S. Christodoulakis et al., "Principles of Optimally Placing Data in Tertiary Storage Libraries," Proc. Int'l Conf. on VLDB, pp.236-245, 1997.

[18] A. Chervenak et al., "Storage Systems for Movies-on-Demand Video Servers," Proc. IEEE Symposium on Mass Storage Systems, 1995.

[19] M. Kienzle et al., "Using Tertiary Storage in Video-On-Demand Servers," Proc. CompCon, pp. 225-233, 1995.

[20] J. Kohl et al., "HighLight : Using a Log-structured File System for Tertiary Storage Management," Proc. Winter USENIX, pp.435-447, 1993.

[21] J. Kohl et al., "HighLight : A File System for Tertiary Storage," Proc. IEEE Symp. on Mass Storage Systems, Apr. 1993.



문 찬 호

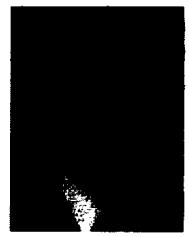
e-mail : moonch@rose.cse.cau.ac.kr

1997년 중앙대학교 컴퓨터공학과 졸업(공학사)

1999년 중앙대학교 대학원 컴퓨터공학과(공학석사)

1999년~현재 중앙대학교 대학원 컴퓨터공학과 박사과정 재학중

관심분야 : Web 데이터베이스, 대용량 데이터베이스, 멀티미디어 데이터베이스



강 현 철

e-mail : hckang@rose.cse.cau.ac.kr

1983년 서울대학교 컴퓨터공학과 졸업(공학사)

1985년 U. of Maryland at College Park, Computer Science (M.S.)

1987년 U. of Maryland at College Park, Computer Science(Ph.D.)

1988년~현재 중앙대학교 컴퓨터공학과 교수

관심분야 : 클라이언트-서버 DBMS, 분산 데이터베이스, 멀티미디어 데이터베이스, 실시간 데이터베이스, 이동 데이터베이스 등