하이브리드 공간 DBMS에서 질의 분류를 이용한 최적화 기법

Query Optimization Scheme using Query Classification in Hybrid Spatial DBMS

정원일*, 장석규**
호서대학교 정보보호학과, 삼성전자

Weonil Chung(wnchung@hoseo.edu)*, Seok-Kyu Jang(seokkyu.jang@samsung.com)**

요약

본 논문에서는 하이브리드 공간 DBMS에서 질의 분류를 이용한 최적화 기법을 제안한다. 제안 기법은 질의에 이용되는 데이터의 위치에 따라 메모리 질의, 디스크 질의, 하이브리드 질의로 분류하여 처리한다. 특히, 하이브리드 질의의 경우에는 실체화 뷰의 사용률을 높이기 위해 실체화 뷰 생성 조건과 사용자 질의 조건을 비교하여 술어를 분할하는 메커니즘을 적용한다. 또한 질의를 최적화하기 위해 분류된 질의의 비용 계산 결과를 이용하여 최소 비용의 데이터 접근 경로를 선택할 수 있는 데이터 접근 경로 선택 알고리즘을 제안한다. 제안 기법은 대용량 데이터 관리와 빠른 응답 속도를 동시에 만족하는 하이브리드 공간 DBMS의 성능을 기존의 디스크 기반 공간 DBMS보다 최소 20%에서 최대 50%의 성능 향상을 보인다.

■ 중심어 : | 하이브리드 공간 DBMS | 질의 분류 | 질의 최적화 |

Abstract

We propose the query optimization technique using query classification in hybrid spatial DBMS. In our approach, user queries should to be classified into three types: memory query, disk query, and hybrid query. Specially, in the hybrid query processing, the query predicate is divided by comparison between materialized view creating conditions and user query conditions. Then, the deductions of the classified queries' cost formula are used for the query optimization. The optimization is mainly done by the selection algorithm of the smallest cost data access path. Our approach improves the performance of hybrid spatial DBMS than traditional disk-based DBMS by 20%~50%.

■ keyword : | Hybrid Spatial DBMS | Query Classification | Query Optimization |

I. 서 론

최근 텔레매틱스, RFID/USN, U-City 등의 분야에서는 공간 정보 관리에 대한 관심이 급증하고 있다[8][10][18]. 특히, 위치기반서비스, 지리정보시스템, 공공시설물 관리 등[1][7][14]에서 다양하게 사용되고 있는 공간 정보는 방대한 대용량의 정보이며, 이를 이용하는 다양한 응용들은 공간 정보의 처리에 대하여 실시간의 응답 속도를 요구하고 있다. 이에 대한 해결책으로 대용량의 공간 데이터를 관리 및 빠른 응답속도를 제공할 수 있는 하이브리드 공간 DBMS가 사용되고 있다[2][6][21].

* 본 논문은 2007년도 호서대학교의 재원으로 학술연구비 지원을 받아 수행된 연구입니다.(20070088)

접수번호 : #071119-001
접수일자 : 2007년 11월 19일
심사완료일 : 2008년 01월 10일
교신저자 : 정원일 e-mail : wnchung@hoseo.edu

제안 기법이 구현된 하이브리드 공간 DBMS에서는 디스크 데이터베이스에서 대용량의 모든 데이터를 관리하며, 빠른 처리를 요구하는 데이터는 실체화 뷰의 형태로 메인메모리 데이터베이스에 저장하여 처리한다[2]. 대용량의 공간 데이터는 갱신 연산이 빈번하지 않은 특성을 나타내므로 실체화 뷰에 존재하는 데이터를 효율적으로 사용하는 것이 시스템의 성능을 좌우하는 매우 중요한 요소이다[6]. 효율적인 실체화 뷰의 활용을 위해서는 질의가 요구하는 데이터의 실체화 뷰 존재 여부를 판단하여야 한다. 또한, 입력 질의에 대한 데이터 접근 경로가 가장 최소의 비용을 갖는 지를 판단하여 시스템의 성능을 높일 수 있도록 최적화해야 한다.

따라서 본 논문에서는 하이브리드 공간 DBMS에서 질의 분류를 이용한 최적화 기법을 제안한다. 이는 실체화 뷰를 효율적으로 이용하기 위하여, 요구되는 데이터의 저장 위치에 따라 분류된 질의의 비용을 분석하여 가장 최소의 비용을 갖는 실행 계획을 선택하여 처리하는 최적화 기법이다. 제안 기법에서는 입력되는 사용자 질의를 메모리 데이터만을 사용하는 메모리 질의, 디스크 데이터만을 사용하는 디스크 질의, 두 종류의 데이터를 함께 사용하는 하이브리드 질의로 분류한다. 질의 분류 시에는 실체화 뷰의 사용률을 높일 수 있도록 메모리에 존재하는 데이터를 우선적으로 사용하도록 고려한다. 또한, 메모리에 저장된 실체화 뷰 테이블의 일부분을 사용할 수 있도록 질의의 비공간 필터 조건과 공간 필터조건을 분할하는 메커니즘을 사용한다. 또한, 제안 기법은 최적화를 위해 분류된 질의에 따른 데이터 접근 경로 중, 질의 별로 소요되는 비용 요소들을 분석하여 최소 비용을 가지는 경로를 선택하여 수행한다.

본 논문은 2장에서 관련 연구로 질의분류 기법과 질의 최적화 기법의 하나인 접근 경로 선택 기법을 소개하고, 3장은 본 논문에서 제안하고 있는 질의 분류를 이용한 최적화 기법을 설명한다. 4장의 성능평가에서는 질의 간에 성능 비교를 통하여 제안 기법이 시스템에 미치는 영향을 평가한다. 5장에서는 결론을 맺는다.

II. 관련연구

1. 질의 분류 기법

질의 분류 방법은 지식 기반(Knowledge-based) 질의 분류와 비지식 기반(Nonknowledge-based) 질의 분류가 있다. 지식 기반 질의 분류는 데이터베이스 스키마, 키의 정보, 질의의 형태 정보(영역질의, 점질의)등의 지식을 질의 분류 요소로 사용하는 방법이다. 대표적인 지식 기반 방법으로는 질의를 클러스터된 인덱스 질의, 클러스터 되지 않은 인덱스 질의, 비 인덱스 질의로 분류하는 ZL의 질의 분류 방법이 있다[15]. 비지식 기반 질의 분류는 데이터베이스를 구성하고 있는 릴레이션의 속성들을 기준으로 질의를 샘플링하여 분류하는 방법이다. 멀티데이터베이스에서는 이와 같은 방식을 적용하여 시스템 최적화에 이용하였다[3-5].

하이브리드 공간 DBMS는 레벨간의 성능차이와 각각에 저장된 데이터의 특징을 이용하여 시스템을 최적화할 수 있는 방향으로 질의가 분류되어야 한다. 본 논문에서는 메모리에 존재하는 데이터만을 요구하는 메모리 질의, 디스크에 존재하는 데이터만을 요구하는 디스크 질의, 메모리와 디스크에서 동시에 데이터를 요구하는 하이브리드 질의로 분류한다.

2. 질의 최적화 기법

본 절에서는 질의 최적화 기법의 하나인 접근 경로 선택 기법에 대하여 살펴본다. 접근 경로 선택 기법은 질의 최적화기가 최소 비용의 실행 계획을 선택하는 기법이다[9][11][12]. 전통적인 디스크 기반 관계형 DBMS에서의 질의 비용은 수식 (1)과 같다. 질의 비용은 예상되는 디스크 I/O의 횟수에 비례하는 디스크 비용과 예상되는 레코드 검사 횟수에 디스크와 메모리간의 처리 변환률 $W1$을 곱한 값으로 계산되었다[12].

$$QueryCost = expected(I/Os) + W1 * expected(records\ examined) \quad (1)$$

하이브리드 DBMS에서의 질의 비용 공식은 수식 (2)과 같다[13]. 하이브리드 DBMS의 저장 구조는 메모리와 디스크 그리고 아카이브를 저장 장치로 사용한다[20]. 따라서 이 시스템에서는 레벨의 특성에 따른 요소(Factor)

들을 선정하고, 그 비용의 계산을 통하여 데이터 접근 경로를 선택하였다.

$$QueryCost = expected\ CPU\ time + W1*(DR+B1/DT)*expected\ disk\ I/Os + W2*(AR+B2/AT)*expected\ archive\ I/Os + W2*P*expected\ platter\ changes \quad (2)$$

수식 (2)에서 *expected CPU time*은 질의 처리에 예상되는 래코드를 메모리에서 검사하는 비용이며, "*W1*(DR - B1/DT)*expected disk I/Os*"은 질의 처리에 예상되는 비용 중에서 디스크 처리 시간을 의미하고, "*W2*(AR - B2/AT)*expected archive I/Os*"은 아카이브에서의 예상 처리 시간이다. *W1*과 *W2*는 각각 시스템 의존적인 값으로써 CPU와 디스크 그리고 아카이브 간의 처리시간 변환률이다. *DR*은 자기 디스크 드라이브에서 임의의 블록으로 이동하는 시간이며, *AR*은 적재된 플래터의 임의의 블록으로 이동하는 시간이다. *B1*과 *B2*는 각각 디스크와 아카이브의 블록 크기를 의미하며, *DT*와 *AT*는 각각 디스크와 아카이브 장치의 전송률이다.

본 논문의 환경인 하이브리드 공간 DBMS에서는 두 개의 레벨에 저장되어 있는 데이터를 효율적으로 사용하도록 질의를 최적화하기 위해서 질의 분류 결과와 질의의 비용을 계산하여야 한다.

III. 질의 분류를 이용한 최적화 기법

1. 시스템 환경

본 논문의 환경인 하이브리드 공간 DBMS는 메모리 레벨의 데이터가 독립적이지 않으며, 디스크 기반의 데이터베이스를 백업 저장장치로 사용하기 때문에 상위 레벨에 저장된 실체화 뷰 처리의 경우 동기화를 통한 회복으로 빠른 메인메모리 알고리즘을 사용할 수 있다. 따라서 기존의 하이브리드 DBMS의 메모리 레벨에서 데이터를 처리하는 것 보다 빠르다는 장점이 있다.

[그림 1]에서 하이브리드 공간 DBMS는 디스크 데이터베이스를 관리하는 GMS[17][19] 저장 관리자와 메모리 데이터베이스를 관리하는 메인메모리 저장 관리자가 있다. 질의 처리기는 각각의 저장 관리자와 통신하는 GMS 질의 처리기와 메인메모리 질의 처리기로 구성되며 이들은 질의 분류 관리자(Classification Manager)와 함께 하이브리드 질의 처리기(Hybrid Query Processor)를 구성한다. 하이브리드 질의 처리기는 사용자로부터 입력된 질의를 두 개의 데이터베이스에 분포되어 있는 데이터를 사용하도록 질의를 처리한다.

또한 하이브리드 공간 DBMS는 실체화 뷰의 동기화를 담당하는 리프레시 관리자(Refresh Manager), 자주 접근되는 데이터에 대하여 실체화 뷰로 자동으로 생성하게 하는 자동화 관리자(Automation Manager), 시스템 종료나 붕괴 후에도 기존에 생성되어 있던 실체화 뷰들을 자동 회복 하게하는 실체화 뷰 재생성 관리자(Materialized View Reconstruction Manager) 등으로 구성된다.

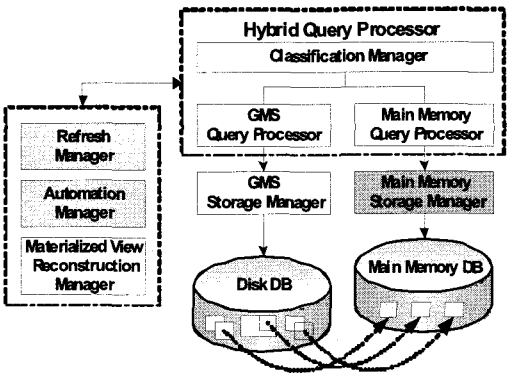

그림 1. 하이브리드 공간 DBMS의 시스템 구조

2. 하이브리드의 특성에 따른 질의 분류

하이브리드 공간 DBMS[2]는 접근 속도가 빠른 실체화 뷰를 최대한 이용하기 위해 질의 분류 매커니즘이 필요하다. 이에 본 연구는 효율적인 질의 분류를 수행하도록 질의의 특성에 따른 질의 분류 방법[21]을 활용한다.

2.1 질의 종류

질의는 데이터의 위치를 고려해 [그림 2]와 같이 메인메모리에 존재하는 실체화 뷰 데이터를 사용하는 메모리 질의(TQ_MM), 디스크에 존재하는 데이터를 사용하는 디스크 질의(TQ_DK), 그리고 메인메모리에 존재하는 데이터와 디스크에 존재하는 데이터를 동시에 사용하는

하이브리드 질의(TQ_ML)로 분류된다.

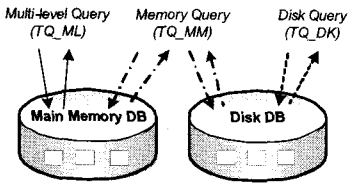

그림 2. 질의 종류

그리고 질의를 처리할 때 실행계획이 단일 테이블에 존재하는 데이터만을 요구하는 경우와 2개 이상의 멀티 테이블에 존재하는 데이터를 요구하는 경우로 나누어 볼 수가 있다. 이 때, 메모리 질의와 디스크 질의는 각각 요구되는 모든 데이터의 테이블 위치가 메모리 또는 디스크에 위치해야만 한다. 그러나 하이브리드 질의의 경우 단일 테이블에서는 메모리에 일부, 디스크에 일부만 존재하는 것을 고려할 수 있으며, 멀티 테이블에서는 질의가 요구하는 테이블들의 개수에 따라 경우의 수가 다양하게 존재할 수 있다.

2.2 메타 데이터

메타 데이터는 디스크 테이블의 정보와 메모리 테이블의 생성 정보 등을 수록하고 있다. 메타데이터의 레코드는 실체화 뷰 생성시 하나씩 증가하며, 메모리로 로딩된 실체화 뷰 테이블 기반 디스크 테이블 이름과 함께 실체화 뷰 생성에 관한 정보인 필드 리스트와 해당 테이블의 비공간 데이터와 공간 데이터에 대한 필터 조건을 포함한다. 또한 실체화 뷰가 존재하는 메모리 데이터베이스에서의 논리적 주소인 OID(Object Identifier)와 생성 기반 테이블이 된 디스크 테이블의 이름과 OID를 함께 제공한다. 이는 입력된 질의에서 요구하는 테이블의 이름을 보고 실체화 뷰로 존재하는 지의 여부를 판단하기 위한 정보로 활용된다[21].

2.3 질의 분류

질의 분류는 사용자 질의로부터 유도된 통합 질의 처리기에서 생성한 질의의 실행 계획을 입력으로 메모리 및 디스크 레벨의 술어를 가지는 실행 계획과 질의의 종류(type)를 출력한다.

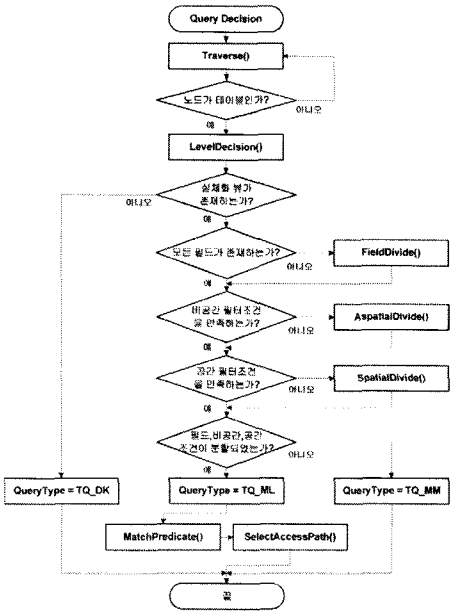

그림 3. 질의 분류 순서도

[그림 3]에서 질의 분류는 QueryDecision 알고리즘을 시작으로 Traverse를 호출하여 실행 계획을 순회한다. 이때 순회 중인 노드가 테이블 노드이면 데이터에 접근해야 할 곳이 디스크 레벨인지 메모리 레벨인지를 판단하기 위한 LevelDecision을 호출한다. LevelDecision에서는 메타데이터를 활용하여 해당 테이블 노드의 접근 위치를 판단하게 되는데, 이때 실체화 뷰가 존재하는 지의 여부를 먼저 비교한다. 해당 테이블의 이름을 가진 실체화 뷰가 존재하지 않으면 질의 종류는 디스크 질의(TQ_DK)로 판정되며 그렇지 않은 경우 필드목록, 비공간 필터조건, 공간 필터조건 등을 추가로 분석한다.

세 가지 조건 중 하나의 조건도 빼놓지 않고 모두 만족하는 경우에는 해당 테이블의 요구되는 모든 데이터가 메모리에 존재한다고 판단되어 질의는 메모리 질의(TQ_MM)으로 판정된다. 그러나, 필드목록의 일부만이 메모리에 존재한다면 메모리에서 접근해야 할 필드목록과 디스크에서 접근해야 할 필드목록을 분할하는 FieldDivide를 수행하고, 비공간 필터조건, 공간 필터조건 등의 일부만이 만족하는 경우에는 해당 조건을 메모리 부분과 디스크 부분으로 분할하는 AspatialDivide와 SpatialDivide를 수행한다. 하나의 분할 알고리즘이라도

수행되었다면 이 질의는 하이브리드 질의(TQ_ML)로 판정되고 디스크와 메모리에 적합한 각각의 술어를 생성한다. 마지막으로 SelectAccessPath 알고리즘은 하이브리드 질의의 경우 디스크 질의로 전환하여 처리하는 것이 더 적은 비용이 소요되는 지를 검사한다.

3. 질의 비용 기반의 최적화

질의 최적화는 각 질의에 대한 비용의 산출을 통해 최소 비용을 갖는 접근 경로를 선택하도록 한다.

3.1 질의의 비용 계산

질의의 비용 계산을 위해 분류된 질의의 형태에 따라 이용하는 데이터베이스의 레벨에 따른 접근에 소요되는 비용만을 고려한다. 메모리 질의(TQ_MM), 디스크 질의(TQ_DK), 그리고 하이브리드 질의(TQ_ML)는 모두 질의 분류 시 질의 결정(QueryDecision)을 위해 요구되는 비용인 *CPU time*(QD)이 공통으로 요구된다.

메모리 질의 비용인 *Cost*(TQ_MM)은 수식 (3)과 같이 표현되며, 수식에서 EMT는 예상되는 메모리 접근 시간을 의미한다. 질의 분류 비용(*CPU time*(QD))과 메모리에 저장되어 있는 실체화 뷰 테이블의 레코드를 읽어 들이는 비용만을 필요로 한다. 예상되는 메모리 레코드 검사 비용은 읽어야 하는 레코드의 개수에 비례한다.

$$Cost(TQ\_MM) = CPU\ time(QD) + EMT(records\ examined) \quad (3)$$

디스크 질의의 비용 *Cost*(TQ_DK)는 수식 (4)와 같이 표현되며, 수식에서 *EDT*는 예상되는 디스크 접근 시간을 의미한다. 질의 분류 비용(*CPU time*(QD))과 함께 디스크에 저장되어 있는 테이블들의 페이지를 버퍼 관리기로 읽어 들이는 DISK I/O 비용, 그리고 메모리 버퍼에 존재하는 레코드를 읽는 비용이 요구된다. 또한 이 비용은 멀티 테이블의 경우 참조되는 디스크 테이블의 개수에 따라 비례하게 된다. 수식에서 $n_d(TQ_DK)$는 TQ_DK의 질의에서 디스크 테이블의 개수를 의미하고, *DK(I/Os)*는 디스크 질의에서 I/O 횟수를 나타낸다.

$$Cost(TQ\_DK) = CPU\ time(QD) + n_d(TQ\_DK) \times (EDT(DK(I/Os)) + EMT(records\ examined\ in\ buffer)) \quad (4)$$

하이브리드 질의의 비용 Cost(TQ_ML)은 수식 (5)에서 질의 분류 비용(*CPU time*(QD))과 실체화 뷰 테이블을 위한 메모리 레코드 검사비용, 디스크 테이블을 위한 디스크 I/O비용, 그리고 버퍼에 존재하는 레코드를 읽는 비용까지 포함한다. 수식 (4)와 마찬가지로 디스크에서 소요되는 비용은 참조하는 디스크 테이블의 개수에 따라 비례하게 된다. 수식에서 $n_d(TQ_ML)$는 TQ_ML의 질의에서 디스크 테이블의 개수이고, *ML(I/Os)*는 하이브리드 질의에서 I/O횟수이다.

$$Cost(TQ\_ML) = CPU\ time(QD) + EMT(records\ examined) + n_d(TQ\_ML) \times (EDT(ML(I/Os)) + EMT(records\ examined\ in\ buffer)) \quad (5)$$

하이브리드 질의 비용은 디스크 질의 비용에 비해 메모리 접근 시간을 더 포함하지만, DK(I/Os)가 ML(I/Os)보다 크므로 디스크 I/O 비용이 상대적으로 작다. 따라서, 디스크 I/O를 고려한 경우에, 하이브리드 질의 전체 비용은 디스크 질의 비용보다 작고, 메모리 질의 처리 비용은 하이브리드 질의 비용보다 작기 때문에 이 관계는 수식 (6)과 같이 유도된다.

$$Cost(TQ\_DK) > Cost(TQ\_ML) > Cost(TQ\_MM) \quad (6)$$

해당 질의가 메모리 질의로 판정되면 메모리 질의 접근 계획에 따라서 질의를 수행하고, 디스크 질의로 판정되면 디스크 질의 접근 계획에 따라 처리한다. 그러나 하이브리드 질의의 경우 메모리에 존재하는 데이터는 디스크에도 존재한다는 저장 구조의 특징 때문에 디스크 질의로 전환하여 처리할 수 있다. 그러나 질의를 전환하여 처리하는 것이 비용 측면에서 효과가 있는지를 고려해야 하므로, 하이브리드 질의와 디스크 질의의 비용 차이를 고려해야 한다. 디스크 질의가 하이브리드 질의보다 한 번이라도 디스크 I/O가 더 많아지면 하이브리드 질의를 디스크 질의로 변환하여 처리하는 것이 이득이지만, 디

스크 I/O가 발생 하지 않는 경우라면 디스크 질의 비용과 하이브리드 질의 비용에서 디스크 I/O 비용을 제외한 메모리 버퍼에서 레코드를 읽어 들이는 비용과 메모리에 저장되어 있는 실체화 뷰 테이블의 레코드를 읽어 들이는 비용의 차이점에 대하여 비교가 필요하다.

먼저 메모리 버퍼에서 레코드를 읽어 들이는 비용는 디스크 질의나 하이브리드 질의에서 디스크 데이터베이스에 존재하는 데이터를 접근할 때 메모리 버퍼에 이미 로딩되어 있는 페이지를 접근할 때 소요되는 비용으로, 버퍼 내에 존재하는 디스크 페이지를 가져오는 비용(α), 가져온 데이터가 조건에 부합되는지를 판단하기 위하여 비교하는 비용(β), 가져온 데이터를 CPU 연산을 위해 메인 메모리에 복사하는 비용(λ), 사용한 페이지를 릴리즈 하는 비용(δ), 그리고 이 연산을 수행하기 위하여 함수를 호출하는 비용(ε)으로 구성된다.

다음으로 메모리에 저장된 실체화 뷰 테이블의 레코드를 읽어 들이는 비용은 메모리 포인터를 이용하여 한번에 접근이 가능한 비용이다. 이 비용을 계산하기 위해서는 해당 데이터가 요구되는 조건에 부합되는지를 비교하기 위한 비용(β)과 함수 호출에 소요되는 비용(ε)만이 요구된다.

메모리 데이터베이스에서 데이터를 접근하는 것과 메모리 버퍼에서 데이터를 접근하는 비용은 수식 (7)과 수식 (8)로 각각 표현된다.

$$EMT(\text{records examined in buffer}) = \text{number of required records} \times (\alpha + \beta + \lambda + \delta + \varepsilon) \quad (7)$$

$$EMT(\text{records examined}) = \text{number of required records} \times (\beta + \varepsilon) \quad (8)$$

따라서 하이브리드 질의의 데이터 접근 경로를 선택하기 위해서는 수식 (4)와 수식 (5)에 수식 (7)과 수식 (8)을 대입하여 계산한다. 따라서 모든 디스크 데이터가 메모리 버퍼에 로딩되어 있다고 가정하고, 디스크 I/O를 제외한 상황에서 하이브리드 질의를 디스크 질의로 전환하여 처리하는 것을 고려한다.

3.2 접근 경로 선택

하이브리드 질의에 대해 질의 분류 과정에서 각각의 디스크와 메모리 술어를 조합하여 산출된 비용을 통해 하이브리드 질의 접근 경로 또는 디스크 질의 데이터 접근 경로를 결정하게 된다. 질의가 메모리 질의 또는 디스크 질의로 판정된 경우 술어의 분할 없이 원래의 술어를 그대로 사용하여 메모리 또는 디스크 데이터베이스에서 접근한다.

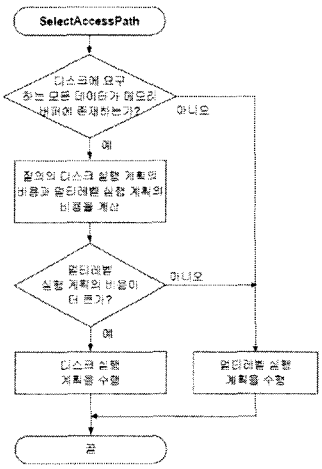

그림 4. 접근 경로 선택의 순서도

접근 경로 선택 절차는 [그림 3]의 MatchPredicate 다음에 SelectAccessPath 과정으로 수행되며 [그림 4]로 표현된다. 하이브리드 질의의 데이터 접근 계획을 수행할 때에는 디스크의 모든 데이터가 메모리 버퍼 영역에 로딩되어 있을 경우 비용 계산을 통하여 디스크 질의의 데이터 접근 계획으로 전환하여 처리할 것인지를 판단한다. 다음으로 하이브리드 질의 비용과 메모리 질의 비용을 각각 계산한 후 하이브리드 질의 비용이 더 크다고 판단되면 디스크 질의로 전환하여 처리한다. 그러나 하이브리드 질의의 비용이 더 작다면 그대로 하이브리드 질의 데이터 접근 계획을 선택하여 질의를 수행한다.

IV. 성능평가

본 장에서는 하이브리드 공간 DBMS에서 질의 분류를 이용한 최적화 기법이 시스템에 미치는 영향을 평가를 위하여 각각 분류된 질의의 성능을 실험한다.

1. 평가 환경

실험을 위하여 City Simulator[16]에 의해 생성된 1500×1200의 지도에 10만개의 객체를 정규 분포시킨 2차원 데이터 집합을 이용하였다. City Simulator는 3차원 위치를 모델링하여 객체들의 행동 패턴을 모의 실험할 수 있도록 가상의 데이터를 생성해주는 프로그램이다. 디스크 데이터의 검색을 위해서 한 번의 전체 검색을 수행하여 메모리 버퍼로 모두 로딩하였고, 메모리 데이터의 검색을 위해서는 검색할 데이터의 10%에서부터 100%까지 10%의 비율로 실체화 뷰를 생성해 두었다.

실험에 사용된 DBMS는 디스크 기반 GMS와 메인메모리 DBMS를 통합하여 자체 개발한 하이브리드 공간 DBMS를 사용하였다. GMS 데이터베이스 크기와 메모리 버퍼 크기는 각각 200Mbyte, 메인메모리 DBMS의 메모리 데이터베이스 크기 역시 200Mbyte로 설정하였다. 그리고 실험은 Pentium IV 3GHz, 1GB 메모리, 200GB 하드디스크의 하드웨어를 기반으로 Windows XP에서 C를 개발언어로 사용하였다.

2. 성능 평가

성능 평가는 데이터의 검색 레코드 수를 변화시키면서 질의간의 성능을 비교를 수행하고, 질의 분류를 이용한 데이터 접근 경로의 선택 기법이 시스템의 성능에 미치는 영향을 평가한다.

실험에 사용되는 질의는 모든 레코드를 디스크 데이터베이스에서 검색하는 질의인 TQ_DK, 전체 레코드 개수에서 10*n%(n=1,2,…,9)는 메모리 데이터베이스에서 나머지는 디스크 데이터베이스에서 검색하는 하이브리드 질의인TQ_ML(10*n%), 모든 레코드를 메모리 데이터베이스에서 검색하는 질의인 TQ_MM, 질의 분류 알고리즘을 거치지 않은 TQ_DK인 TQ_DK without QC, 질의 분류 알고리즘을 거치지 않은 TQ_MM인 TQ_MM without QC, 질의 분류를 적용한 검색 질의인 Applied QC, 질의 분류를 적용하지 않은 검색 질의인 Non-applied QC를 이용하였다.

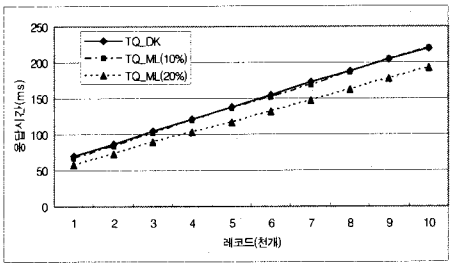

그림 5. 응답시간: TQ_DK vs. TQ_ML

[그림 5]는 레코드 검색 개수에 따른 디스크 질의(TQ_DK)와 하이브리드 질의(TQ_ML)의 응답시간을 나타낸다. 하이브리드 질의(TQ_ML(10%))는 디스크 질의와 유사한데, 이는 TQ_ML(10%)의 질의 비용이 질의 분류 알고리즘의 오버헤드로 인해 TQ_DK보다 커짐에 따라 디스크 질의로 전환되어 처리되기 때문이다. 이는 제안한 기법을 적용함으로써 하이브리드 질의 데이터 접근 경로를 처리함으로써 나타나는 성능 저하 현상을 해소할 수 있음을 의미한다.

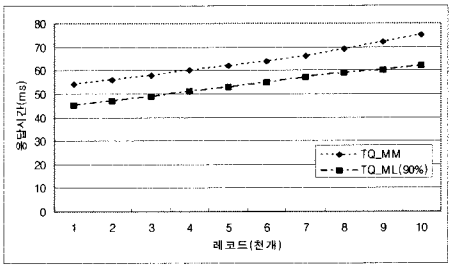

그림 6. 응답시간: TQ_MM vs. TQ_ML

[그림 6]은 레코드 검색 개수에 따른 메모리 질의(TQ_MM)와 하이브리드 질의(TQ_ML)의 응답시간을 표현한다. 하이브리드 질의는 90%의 메모리 데이터를 이용하는 질의(TQ_ML(90%))를 사용하였다.

그 결과 TQ_ML(90%) 질의는 메모리 질의보다 더 많은 응답 시간이 걸린 것을 알 수가 있다. 이것은 메모리 버퍼 공간을 참조하는 연산이 메모리 데이터베이스를 참조하는 연산보다 더 많은 비용이 소요된다는 것을 알 수가 있다([수식6, 7]). 즉 하이브리드 질의에서 메모리의 사용 비율이 거의 100% 가깝다 할지라도 메모리 질의의 비용이 더 적다는 것을 보여준다.

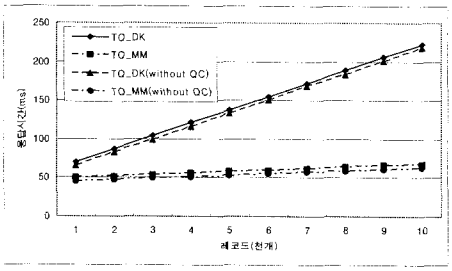

그림 7. 응답시간: TQ_DK vs. TQ_MM

[그림 7]은 레코드 검색 개수에 따른 디스크 질의(TQ_DK)와 메모리 질의 (TQ_MM)의 응답시간을 비교한 것으로, 두 개의 질의 모두 읽는 레코드의 개수가 많아짐에 따라 응답시간이 증가하는 것을 알 수 있으며, TQ_DK의 기울기는 TQ_MM의 기울기보다 페이지 패치 비용(α), 페이지 릴리즈 비용(λ), 그리고 메모리 복사 비용(δ)의 요소들을 더 포함한다.

TQ_DK와 TQ_MM의 기울기는 [수식 3-6]과 [수식 3-7]의 비용 요소이다. 또한 질의 분류를 수행하지 않은 메모리 질의(TQ_MM without QC)와 질의 분류를 수행하지 않은 디스크 질의(TQ_DK without QC)의 응답시간 평가를 통해 질의 분류를 수행한 후의 응답시간은 질의 분류를 수행하지 않은 응답시간과 큰 차이가 나지 않는다는 것을 알 수 있다.

[그림 8]은 질의 분류 기능이 시스템에 미치는 영향을 판단하기 위하여 레코드 검색 개수에 따른 질의 분류를 수행한 질의(Applied QC)와 수행하지 않은 질의(Nonapplied QC)의 응답시간을 비교한 그래프이다. 이 실험은 검색 레코드의 개수에 따라 전체 검색 개수의 50%는 실체화 뷰에서 검색하는 환경에서 실험하였다.

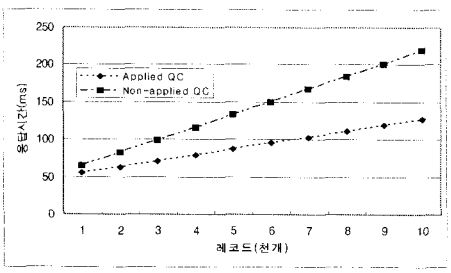

그림 8. 응답시간: Applied QC vs. Non-Applied QC

Applied QC는 질의 분류 알고리즘이 작동하여 레코드를 검색하는 질의를 수행할 때 마다 전체 검색 개수의 50%, 즉 천개의 레코드를 검색하였을 때에는 500개의 실체화 뷰의 레코드와 500개의 디스크 레코드를 읽는 질의이다. Non-applied QC는 질의 분류가 작동하지 않기 때문에 전체 검색개수의 절반이 실체화 뷰로 생성되어 있다고 하더라도 모든 레코드를 디스크에서 읽게 된다.

V. 결론

본 논문에서는 하이브리드 공간 DBMS에서 메모리 데이터베이스에 생성되어 있는 실체화 뷰를 최대한 사용하도록 질의를 분류하여 최소의 비용을 갖는 데이터 접근 경로를 선택하는 최적화 기법을 제안하였다. 질의를 분류하기 위하여 하이브리드의 특성을 이용한 지식 기반 분류 방법을 적용하였고, 사용되는 데이터의 위치에 따라 디스크 질의, 메모리 질의, 하이브리드 질의로 분류하였다. 특히, 하이브리드 질의는 메모리 데이터베이스의 실체화 뷰를 최대한 이용하도록 하기 위하여, 입력된 질의의 술어를 메모리 부분과 디스크 부분으로 분할하여 처리함으로써 실체화 뷰의 사용률을 높였다.

최적화를 위한 데이터 접근 경로 선택은 질의의 비용 계산을 통하여 최소 비용을 가지는 질의의 접근 경로를 선택하게 하였다. 하이브리드 질의의 경우 메모리와 디스크의 데이터를 동시에 가져와야 하는 특징 때문에 비용 계산을 통하여 디스크 질의로 전환하여 처리하는 것을 고려하였다. 따라서 제안 기법에서는 질의 분류 알고리즘을 적용하는 데에 따르는 부하 때문에 하이브리드 질의의 성능이 디스크 질의의 성능보다 떨어지는 것을 해소하였다. 질의 분류를 이용하여 최소의 데이터 접근 경로를 선택하는 것은 시스템 성능을 높일 수 있는 결정하는 주요 기준으로, 시스템 환경에 따라 하이브리드 질의를 선택적으로 사용하는 것은 중요하다.

향후 연구로는 질의 비용을 산정할 때 공간 질의에 대한 추가적인 고려 및 관련 기법과의 추가적인 성능 비교가 필요하며, 또한 하이브리드 공간 DBMS로 이루어진 클러스터 시스템에서의 질의 분류를 이용한 최적화 방법

을 고려한다.

참 고 문 헌

[1] H. Choi, K. Kim, and J. Lee, "Design and implementation of open GIS component software," Proceedings of the Geoscience and Remote Sensing Symposium, Vol.5, pp.2105-2107, 2000(7).

[2] S. H. Eo, S. K. Jang, J. D. Lee, and H. Y. Bae, "Multi-Level SDBMS with Snapshots," Proceedings of the 3rd ASGIS symposium, pp.283-294, 2005(6).

[3] B. Harangsri, J. Shepherd, and A. Ngu, "Query Classification in Multidatabase Systems," Proceedings of 7th Australasian Database Conference, pp.147-159, 1996(1).

[4] B. Harangsri, J. Shepherd, and A. Ngu, "Query Optimisation in Multidatabase Systems using Query Classification," Proc. of the ACM symposium on Applied Computing, pp.173-177, 1996(2).

[5] C. Hsu and C. A. Knoblock, "Semantic Query Optimization for Query Plans of Heterogeneous Multidatabase Systems," IEEE Transactions on Knowledge and Data Engineering, Vol.12, Issue.6, pp.959-978, 2000(11).

[6] S. K. Jang, S. H. Eo, H. S. Kim, and H. Y. Bae, "Query Classification Method for Performance Enhancement in Multi-Level SDBMS with Snapshots," 3rd ASGIS symposium, pp.295-304, 2005(6).

[7] C. S. Jensen, A. F. Christensen, T. B. Pedersen, D. Pfoser, S. Saltenis, and N. Tryfona, "Location-Based Services - A Database Perspective," Proc. of the 18th Scandinavian Research Conference on Geographical Information Science, As, pp.59-68, 2001(6).

[8] Y. Kawahara, N. Kawanishi, H. Morikawa, and T. Aoyama, "Top-down approach toward building ubiquitous sensor network applications," Proc of the Software Engineering Conference, pp.695-702, 2004(11).

[9] V. Kumar and J.Mullins, "An integrated data structure with multiple access paths for database and its performance," Proc. of the COMPSAC 93, Proceedings of the 17th Annual International Conference, pp.241-247, 1993(11).

[10] M. McMorrow, "Telematics - exploiting its potential," Manufacturing Engineer, Vol.83, No.1, pp.46-48. 2004(2).

[11] T. Moulder, "Access Path to Performance," Technical Support, pp.18-21, 2005(3).

[12] P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie and T. G. Price, "Access Path Selection in a Relational Database Management System," Proc. of the ACM SIGMOD Conf. on Management of data, pp.23-34, 1979.

[13] M. Stonebraker, "Managing Persistent Objects in a Multi-Level Store," Proceedings of the ACM SIGMOD international conference on Management of data, pp.2-11, 1991.

[14] K. Virrantaus, J. Markkula, A. Garmash, V. Terziyan, J. Veijalainen, A. Katanosov, and H. Tirri, "Developing GIS- supported Location-based Services," Proceedings of the Second International Conference on Web Information Systems Engineering, Vol.2, pp.66-75, 2001(12).

[15] Zhu and P. A. Larson, "A Query Sampling Method for Estimating Local Cost Parameters in a Multidatabase System," In Data Engineering, pp.144-153, 1994.

[16] http://www.alphaworks.ibm.com/tech/citysimulator

[17] 박상근, 박순영, 정원일, 김명근, 배해영, "GMS: 공간 데이터베이스 관리 시스템", 2003 공동 춘계 학술대회 논문집, pp.217-224, 2003.

[18] 이봉규, 송지영, "NGIS 기반하의 Business GIS 발전방안", 한국공간정보 시스템 학회 논문지, Vol.7, No.2, pp.3-14, 2005.

[19] 이환재, 안준순, 강동재, 이경모, 정보홍, 박동선, 배해영, "GEO/Millenium: 클라이언트-서버 공간 데이터베이스 시스템", 한국정보과학회 2000년 춘계학술대회, pp.48-50, 2000.

[20] 장석규, 어상훈, 김명근, 배해영, "스냅샷 데이터를 갖는 다중레벨 저장 DBMS에서 성능향상을 위한 질의 분류 방법", 데이터베이스 연구회 학술대회, pp.121-126, 2005.

[21] 장석규, 어상훈, 김명근, 배해영, "위치기반 서비스를 위한 다중레벨 DBMS에서 질의 분류 컴포넌트의 설계 및 구현", 한국정보처리학회 논문지 D, Vol.12-D, No.5, pp.689-698, 2005.

저 자 소 개

정 원 일(Weonil Chung) 정회원

- 1998년 2월 : 인하대학교 전자계산공학과(공학사)
- 2004년 8월 : 인하대학교 컴퓨터정보공학과(공학박사)
- 2004년 7월 ~ 2006년 7월 : 한국전자통신연구원 선임연구원
- 2007년 3월 ~ 현재 : 호서대학교 정보보호학과 전임강사

<관심분야> : 데이터스트림, 이동객체, 데이터 보안

장 석 규(Seok-Kyu Jang) 정회원

- 2004년 2월 : 인하대학교 컴퓨터공학부(공학사)
- 2006년 2월 : 인하대학교 컴퓨터정보공학과(공학석사)
- 2006년 3월 ~ 현재 : 삼성전자 연구원

<관심분야> : 하이브리드 데이터베이스, LBS/텔레메틱스, 유비쿼터스 컴퓨팅