한국지형공간정보학회지 제20권 제2호 2012년 6월 pp. 81-91
연구논문
ISSN: 1598-2955
http://dx.doi.org/10.7319/kogsis.2012.20.2.081

모바일 장치용 시공간 질의 처리 시스템의 개발

Development of a Spatio-Temporal Query Processing System for Mobile Devices

신인수* · 양형식** · 김정준*** · 한기준****
Shin, In Su · Yang, Hyeong Sik · Kim, Joung Joon · Han, Ki Joon

要 旨

최근 유비쿼터스 컴퓨팅 환경이 발전함에 따라 u-GIS는 유비쿼터스 컴퓨팅 환경의 핵심 요소 기술로 대두되고 있고 이에 따른 시공간 데이터의 효율적인 관리에 대한 연구가 활발히 이루어지고 있다. 이러한 u-GIS 환경에서 모바일 장치는 나날이 증가하고 있는 대용량 시공간 데이터를 효율적으로 관리하기에는 어려움을 가지고 있다. 따라서 본 논문은 모바일 장치용 시공간 질의 처리 시스템을 개발하였다. 모바일 장치용 시공간 질의 처리 시스템은 시공간 데이터에 대한 삽입/삭제/갱신/검색을 위한 다양한 시공간 연산자를 제공한다. 그리고 플래시 메모리에 최적화된 시공간 인덱스와 질의 수행 속도를 보장하기 위해 시공간 히스토그램을 이용한 질의 최적화 기능을 제공한다. 마지막으로, 모바일 장치용 시공간 질의 처리 시스템을 가상 시나리오에 적용함으로써 본 시스템이 모바일 환경에서 다양한 u-GIS 응용 분야에 유용하게 활용될 수 있다는 것을 검증하였다.

핵심용어 : 시공간 데이터, 질의 처리, 질의 최적화, 히스토그램, 모바일 장치

Abstract

As the recent development of the ubiquitous computing environment, u-GIS is being highlighted as the core technology of the ubiquitous computing environment, and thereby, studies on spatio-temporal data are being actively conducted. In this u-GIS environment, it is still difficult for existing mobile devices to efficiently manage the massive spatio-temporal data of u-GIS that are increasing day by day. Therefore, this paper develops a spatio-temporal query processing system for mobile devices in order to solve the problem. The system provides various spatio-temporal operators to insert/delete/update/search spatio-temporal data and supports a query optimization function that uses a spatio-temporal index for the flash memory and a spatio-temporal histogram for guaranteeing query execution speed. Lastly, by applying the spatio-temporal query processing system developed in this paper to the virtual scenario, this paper has proved that the system can be utilized in various application fields necessary to process spatio-temporal data in the mobile environment.

Keywords : Spatio-temporal Data, Query Processing, Query Optimizing, Histogram, Mobile Device

1. 서 론

최근 유비쿼터스 컴퓨팅 환경이 발전함에 따라 교통(u-Transport), 복지(u-Care), 문화(u-Culture), 환경(u-Green), 산업(u-Business), 도시(u-City), 행정(u-Government) 분야에서 다양한 시공간 정보를 제공하는 u-GIS가 유비쿼터스 컴퓨팅 환경의 핵심 요소 기술로 대두되고 있다(서용철 등, 2008; 김은형, 2009). 특히, 이러한 u-GIS 환경에서는 언제 어디서나 모바일 장치에서 대용량의 시공간 데이터를 수집하고 가공 및 처리하는 것이 매우 중요한 현안이 되고 있다(민경욱, 2008; 양형식 등, 2011; Wang et al., 2001).

시공간 데이터는 일반적으로 그 크기가 매우 큰 특징을 가지고 있지만, 현재 모바일 장치는 기존 PC에 비해 작은 용량의 저장 공간과 낮은 성능의 프로세서를 사용하고 있고, 자체 파일 시스템을 기반으로 데이터를 처

2012년 5월 15일 접수, 2012년 6월 12일 채택
* 정회원 · 건국대학교 컴퓨터공학부 박사과정(isshin@db.konkuk.ac.kr)
** 알티베이스 사원(hsyang@altibase.com)
*** 건국대학교 컴퓨터공학부 강의교수(jjkim9@db.konkuk.ac.kr)
**** 교신저자 · 정회원 · 건국대학교 컴퓨터공학부 교수(kjhan@db.konkuk.ac.kr)

리 및 관리하고 있다. 그러므로 이러한 파일 시스템은 작은 용량의 데이터를 처리 및 관리하기는 편리할지 모르나 나날이 증가하고 있는 u-GIS의 대용량 시공간 데이터를 관리하거나 매우 복잡하고 다양한 시공간 질의를 처리하기에는 큰 문제가 있다(김정준 등, 2007;김정준 등, 2009).

또한, 대용량의 시공간 데이터를 낮은 성능의 모바일 장치에서 효율적으로 처리하기 위해 질의 최적화에 대한 연구가 활발히 진행되고 있다. 모바일 장치에서 질의 최적화를 위한 기존의 여러 가지 기법 중 질의 크기 추정을 위한 히스토그램 기법은 실행 시간에 대한 부담이 없고 오차율이 적다는 장점으로 인해 가장 활발하게 사용되고 있다(신병철, 이종연, 2006; Theodoridis et al., 1996; Wang et al., 2001). 하지만, 기존의 다양한 히스토그램 기법은 히스토그램 생성 시 고정된 버킷 개수로 공간을 분할하기 때문에 데이터 분포를 제대로 반영하지 못하는 문제점이 있다(황환규, 2004; Acharya et al., 2008; Piatesky, 1984). 그러므로 최근에 시공간 질의 최적화를 위해 인덱스 성능에 영향을 미치는 데이터 비율을 기반으로 버킷 개수를 동적으로 분할하는 시공간 히스토그램 기법이 제안되었다(양형식 등, 2011).

본 논문은 버킷 개수를 동적으로 분할하는 시공간 히스토그램 기법을 적용하여 모바일 장치에서 시공간 질의의 효율적인 처리를 지원하는 모바일 장치용 시공간 질의 처리 시스템을 개발하였다. 모바일 장치용 시공간 질의 처리 시스템은 모바일 장치용 공간 DBMS인 FUNs(ETRI, 2008; Min et al., 2008)를 기반으로 확장하여 시공간 데이터 타입과 시공간 연산자를 지원하고, 모바일 장치의 플래시 메모리를 최적화하기 위해 MBR 압축 기법(김정준 등, 2007)과 버퍼링 기법(김정준 등, 2009)을 적용한 시공간 인덱스 제공한다. 마지막으로 본 논문에서 개발한 모바일 장치용 시공간 질의 처리 시스템을 가상 시나리오에 적용함으로써 본 시스템이 모바일 환경에서 시공간 데이터 처리가 필요한 다양한 u-GIS 응용 분야에서 유용하다는 것을 검증하였다.

본 논문의 구성은 다음과 같다. 제1장의 서론에 이어 제2장에서는 관련 연구로 FUNs에서 제공하는 기능과 질의 최적화를 위한 시공간 히스토그램 기법에 대해 분석한다. 그리고 OGC의 'Simple Feature Specification for SQL' 명세에 대해 설명한다. 제3장에서는 모바일 장치용 시공간 질의 처리 시스템의 전체 구조와 각 관리자에 대해 상세히 설명한다. 제4장에서는 모바일 장치용 시공간 질의 처리 시스템을 가상 시나리오를 적용하여 그 효용성을 검증한다. 마지막으로 제5장에서는 결론에 대하여 언급한다.

2. 관련 연구

본 장에서는 모바일 장치용 공간 DBMS인 FUNs와 질의 최적화를 위한 시공간 히스토그램 기법에 대해 분석한다. 그리고 OGC의 'Simple Feature Specification for SQL' 명세에 대해서 간략히 기술한다.

2.1 FUNs

FUNs는 속성 데이터, 공간 데이터, 네트워크 데이터를 저장하고 질의할 수 있는 모바일 장치용 공간 DBMS이다(ETRI, 2008; Min et al., 2008). FUNs는 일반적인 RDBMS와 유사한 기능을 제공하며, DBMS와 응용프로그램 간의 인터페이스로는 SQL이 아닌 API의 형태를 지원한다. 그리고 데이터를 테이블 스페이스 단위의 파일로 관리하고, 테이블 스페이스에는 다수의 테이블이 포함된다.

FUNs에서는 공간 데이터 타입으로 Point, Line String, Polygon이 지원되며 속성 데이터 검색의 경우는 일반 DBMS에서 제공하는 검색 기능 이외에 한글 초성검색 기능이 지원된다. 공간 데이터 검색으로는 Windows 검색, KNN 검색, 공간 위상 관계 연산 기능을 제공한다. 그리고 빠른 경로 검색을 위해 물리적 레코드 ID 검색을 제공하며 트랜잭션 처리를 위해 logging 기법을 이용한 회복 기능도 제공한다. 특히 신속한 데이터 접근을 위한 인덱스로는 속성 검색을 위한 B+-Tree와 공간 검색을 위한 GRID, R*-Tree를 지원한다.

Figure 1은 FUNs의 전체 구조도를 보여준다.

Figure 1에서 보는 바와 같이 FUNs는 Storage Manager, Buffer Manager, Record Manager, Access Path Manager, Catalog Manager, Transaction Manager, Log Manager, Query Processor로 구성되어 있다.

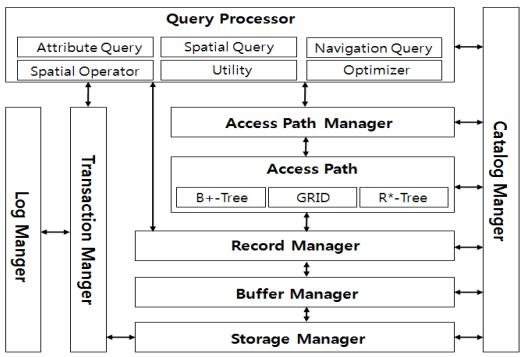

Figure 1. FUNs architecture

Storage Manager는 데이터베이스를 생성 및 삭제하고 테이블 공간과 테이블을 페이지 단위로 관리하는 모듈이고, Buffer Manager는 캐시 데이터를 페이지 단위로 메모리에 적재하여 저장 공간의 I/O 속도를 향상시키는 모듈이다. Record Manager는 페이지 내의 레코드를 관리하는 모듈이고, Access Path Manager는 일반 속성 데이터의 신속한 검색을 위한 B+-Tree 인덱스와 2차원 공간 데이터 검색을 위한 GRID 인덱스와 R*-Tree 인덱스를 제공하는 모듈이다. Catalog Manager는 데이터베이스의 테이블 공간과 테이블, 인덱스 등의 메타데이터를 관리하는 모듈이고, Transaction Manger와 Log Manager는 트랜잭션에 관한 장애와 복구를 관리하는 모듈이다. 마지막으로 Query Processor는 데이터베이스의 생성 및 삭제, 테이블 공간 할당, 기록, 검색, 삭제 등의 질의 처리를 위한 API를 제공하는 모듈이다.

2.2 시공간 히스토그램 기법

공간 데이터베이스 시스템에서 질의 실행 계획의 비용은 전체 테이블에서 질의 조건을 만족하는 질의 결과 크기에 의하여 결정되며, 공간 질의 결과 크기를 근사 계산하기 위해서는 공간 객체들의 배치에 근거하여 전체 공간을 버킷이라 불리는 작은 공간으로 분할하여 히스토그램을 생성하는 히스토그램 기법이 사용되고 있다. 기존 히스토그램 기법은 분할된 각 버킷 내에 존재하는 MBR(Minimum Bounding Rectangle)의 개수 정보를 유지하므로 공간 데이터베이스 시스템은 질의 수행 시 히스토그램을 참조해서 선택율을 계산하여 질의 결과 크기를 추정한다.

그러나 질의 최적화를 위한 기존의 다양한 히스토그램 기법은 히스토그램 생성 시 고정된 버킷 개수로 공간을 분할하기 때문에 데이터의 분포를 제대로 반영하지 못하여 잘못된 질의 실행 계획 생성으로 인한 성능 저하의 원인이 되고 있다. 그러므로, 최근에 이러한 문제점을 해결하기 위해서 시공간 질의 최적화를 위해 데이터 비율을 기반으로 버킷 개수를 동적으로 분할하는 시공간 히스토그램 기법이 제시되었다(양형식 등, 2011).

시공간 히스토그램 기법은 인덱스 성능에 영향을 미치는 데이터 비율을 기반으로 버킷을 동적으로 분할한다. 즉, 버킷을 분할하기 전에 생성될 초기 버킷 개수를 결정한다. 다음은 초기 버킷 개수를 결정하는 식이다.

$$MaxB_inum = Dnum \times Irate \quad (1)$$

$$FirstBnum = Dnum \div MaxB_inum \quad (2)$$

식(1)에서 MaxBinum은 i번째 버킷에 허용되는 최대 MBR 개수이다. 또한 여기서 Dnum은 전체 데이터의 개수를 의미하며, Irate는 인덱스 성능에 영향을 미치는 데이터 비율이다. 식(2)에서 FirstBnum은 초기 버킷 개수를 의미한다.

초기 버킷 개수가 결정되면 버킷 개수에 맞게 편중도와 가중치를 이용하여 버킷을 분할한다. 버킷을 분할할 때 편중도와 가중치를 구하는 식은 MinSkew 히스토그램(Acharya et al., 1999)에서 편중도와 가중치를 구하는 식을 사용한다. 그리고 초기 버킷 개수에 맞게 버킷이 분할되면 각 버킷이 가지는 MBR의 개수를 확인한다. 만약, i번째 버킷에 해당하는 MBR의 개수가 버킷에 허용되는 최대 MBR 개수 보다 크면 해당 버킷을 버킷에 허용되는 최대 MBR 개수 이하가 될 때까지 재귀적으로 분할하여 시공간 히스토그램을 생성한다.

마지막으로 이렇게 최종 분할된 히스토그램은 시공간 데이터 특성에 맞게 시간에 따른 데이터 변화율을 기반으로 재구축되어 여러 개의 시공간 히스토그램으로 생성된다. 시공간 히스토그램 기법은 특정 시간에 질의 횟수 합을 임계값과 비교하여 히스토그램 재구축 여부를 판단한다. 이때, 질의 횟수 합은 검색 질의를 제외한 삽입/삭제/갱신 질의 횟수 합을 의미하며, 임계값은 MaxBinum이다. 즉, 검색 질의를 제외한 삽입/삭제/갱신 질의에 대해 각 버킷의 질의 횟수 합을 구한 후, 만약 질의 횟수의 합이 MaxBinum을 초과하면 시공간 히스토그램이 재구축된다.

시공간 질의 최적화 시스템은 이러한 시공간 히스토그램을 이용하여 계산된 선택율을 기반으로 질의 실행 계획을 생성하고, 가장 최소의 비용을 가진 질의 실행 계획을 선택하여 실행하게 된다.

2.3 Simple Feature Specification for SQL

OGC에서 제안한 “Simple Feature Specification for SQL”은 ODBC API를 경유하는 심플 피쳐 집합의 저장, 검색, 질의, 그리고 갱신을 지원하는 표준 SQL 스키마를 정의하고 있다(Open Geospatial Consortium, 2010). 심플 피쳐는 공간과 비공간 속성을 모두 지원하기 위해 OGC의 OpenGIS Abstract Specification에 정의되어 있다. 공간 속성은 Geometry값이며, Simple Feature는 점 사이의 선형 보간법을 이용한 2D Geometry를 기반으로 한다. 표준 공간 데이터 타입은 Figure 2와 같은 계층 구조로 구성되어 있다.

Figure 2에서 보는 바와 같이 계층 구조의 최상위 타입인 Geometry는 Point, Curve, Surface, Geometry Collection의 하위 타입을 갖고, 이 중 Curve는 LineString

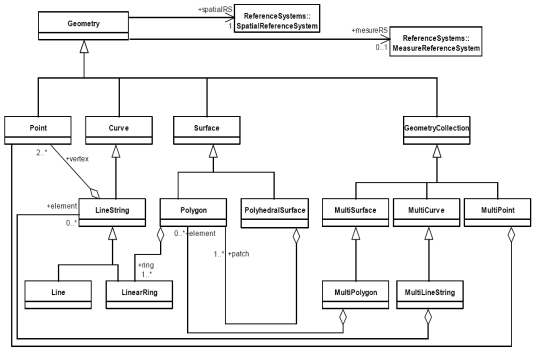

Figure 2. A hierarchy structure of standard spatial data types

타입을, Surface는 Polygon과 Polyhedral Surface 타입을 갖는다. 여기서 GeometryCollection은 이종의 Geometry들의 집합이다. GeometryCollection의 하위 타입인 MultiPoint, MultiCurve, Multi Surface는 동종의 Point, Curve, Surface들의 집합을 다루기 위해 사용된다. 계층 구조에서 보이는 15개의 SQL Geometry 타입 중 Geometry, Curve, Surface, MultiCurve, Line, LinearRing, MultiSurface는 인스턴스를 생성할 수 없는 공간 데이터 타입이다. 따라서 SQL에서 사용 가능한 인스턴스를 생성할 수 있는 공간 데이터 타입은 위의 7개 타입을 제외한 나머지 8가지이다. Table 1은 이러한 인스턴스 생성이 가능한 타입들의 WKT(Well-Known Text) 표현을 보여준다.‘

Table 1에서 보는 바와 같이 인스턴스 생성이 가능한 타입들은 Point, LineString, Polygon, Polyhedral Surface, Multipoint, MultiLineString, MultiPolygon, GeometryCollection이다.

Table 1. WKT expressions of spatial data types

공간 데이터 타입	WKT 표현
Point	POINT (10 10)
LineString	LINESTIRNG (10 10, 20 20, 30 40)
Polygon	POLYGON (10 10, 10 20, 20 20, 20 15, 10 10)
Polyhedral Surface	POLYHEDRALSURFACE ((10 10, 10 20, 20 20, 20 10, 10 10), (20 10, 40 20, 20 20, 20 10))
MultiPoint	MULTIPOINT (10 10, 20 20)
MultiLine String	MULTILINESTRING ((10 10, 20 20), (15 15, 30 15))
MultiPolygon	MULTIPOLYGON (((10 10, 10 20, 20 20, 20 15, 10 10)), ((60 60, 70 70, 80 60, 60 60)))
Geometry Collection	GEOMETRYCOLLECTION (POINT (10 10), POINT (30 30), LINESTRING (10 10, 20 20, 30 40))

3. 시스템 개발

3.1 시스템 전체 구조도

모바일 장치용 시공간 질의 처리 시스템은 시공간 데이터에 대해 삽입/갱신/삭제/검색 기능을 제공하는 모바일 시공간 DBMS이다. Figure 3은 본 논문에서 개발한 모바일 장치용 시공간 질의 처리 시스템의 전체 구조를 보여준다.

Figure 3에서 보는 바와 같이 본 시스템은 FUNs에서 지원하는 공간 데이터 타입 이외에 시공간 데이터 타입을 추가하여 확장하고, 이를 기반으로 모바일 장치용 시공간 질의 처리 시스템을 개발하였다. 모바일 장치용 시공간 질의 처리 시스템은 시공간 질의 처리 관리자, 시공간 질의 최적화 관리자, 시공간 SQL 관리자로 구성된다.

시공간 질의 처리 관리자는 모바일 장치에서 플래시 메모리를 최적화하기 위해 노드 크기를 줄일 수 있는 MBR 압축 기법과 플래시 메모리 쓰기 연산/소거 연산을 줄일 수 있는 버퍼링 기법을 적용한 시공간 인덱스를 제공하고 시공간/시간 연산자를 제공한다. 시공간 질의 최적화 관리자는 시공간 질의를 재작성하고 질의 수행 시간을 단축하기 위해 시공간 히스토그램을 기반으로 선택율을 추정하여 실행 계획을 생성한다.
마지막으로 시공간 SQL 관리자는 시스템에서 지원하는 시공간 타입 및 시공간/시간 SQL을 지원하고 시공간 질의를 분석 및 파싱(parsing)한다.

3.2 시공간 질의 처리 관리자

본 절에서는 시공간 질의 처리 관리자를 구성하는 시공간 연산자 모듈, 시간 연산자 모듈, 시공간 인덱스 모

Figure 3. An architecture of spatio-temporal query processing system

듈에 대해 설명한다.

3.2.1 시공간 연산자 모듈

시공간 연산자 모듈은 OGC의 “Simple Features Specification for SQL”(Open Geospatial Consortium, 2010)에서 제시한 공간 연산자를 기반으로 확장하여 시공간 연산자를 제공한다. 시공간 연산자는 시간 특성에 맞게 타임스탬프 연산자와 인터벌 연산자로 구분되고, 또한 시공간 연산자는 결과값으로 True 혹은 False를 반환하는 시공간 관계 연산자와 시공간 객체를 반환하는 시공간 분석 연산자로 구분된다.

시공간 연산자 모듈은 시공간 데이터 타입으로 ST_POINT, ST_LINESTRING, ST_POLYGON 타입을 지원한다. ST_POINT 타입은 포인트 좌표의 X값, Y값을 가지는 포인트 좌표값과 시작 시간과 끝 시간을 가진다. ST_LINESTRING 타입은 라인을 구성하는 포인트 개수와 포인트 좌표, 시작 시간, 끝 시간을 가진다. ST_POLYGON 타입은 다각형 객체를 구성하는 포인트 개수와 포인트 좌표, 시작 시간, 끝 시간을 가진다.

모바일 장치용 시공간 질의 처리 시스템은 시공간 데이터에 대한 삽입/삭제/갱신/검색 연산 수행을 위해 다양한 시공간 연산자를 제공한다. Table 2는 타임스탬프/인터벌 질의를 위한 시공간 관계 연산자를 보여준다.

Table 2에서 보는 바와 같이 타임스탬프/인터벌 질의를 위한 시공간 관계 연산자는 입력값으로 두 개의 시공간 객체 ST_Geometry1, ST_Geometry2를 입력받고, 특정 시간 time 혹은 특정 인터벌 [stTime, edTime]을 기준으로 시공간 연산을 수행한다. 또한 각 연산에 대한 결과값으로는 True 혹은 False를 반환한다. Table 3은 타임스탬프 질의를 위한 시공간 분석 연산자를 보여준다.

Table 3에서 보는 바와 같이 타임스탬프 질의를 위한 시공간 분석 연산자는 입력값으로 두 개의 시공간 객체 ST_Geometry1, ST_Geometry2를 입력받고, 특정 시간 time을 기준으로 시공간 연산을 수행한다. ST_Union, ST_Difference, ST_Intersection는 결과값으로 시공간 Geometry 객체를 반환하고, ST_Distance는 결과값으로 가장 짧은 거리를 반환한다.

시간 연산자도 시간 특성에 맞게 타임스탬프(Timestamp) 연산자와 인터벌(Interval) 연산자로 구분되고, 또한 결과값으로 True 혹은 False를 반환하는 시간 관계 연산자와 인터벌을 반환하는 시간 분석 연산자로 구분된다(민경욱, 2008). Table 4는 타임스탬프/인터벌 질의를 위한 시간 관계 연산자를 보여준다.

Table 2. Spatio-temporal relation operators for timestamp/interval queries

타임스탬프 시공간 관계 연산자	설 명
ST_Equals(ST_Geometry1, ST_Geometry2, time)	시간 time에 ST_Geometry1과 ST_Geometry2가 동일한지 여부 반환
ST_Disjoint(ST_Geometry1, ST_Geometry2, time)	시간 time에 ST_Geometry1과 ST_Geometry2가 떨어져 있는지 여부 반환
ST_Touches(ST_Geometry1, ST_Geometry2, time)	시간 time에 ST_Geometry1과 ST_Geometry2의 경계가 만나는지 여부 반환
ST_Overlaps(ST_Geometry1, ST_Geometry2, time)	시간 time에 ST_Geometry1과 ST_Geometry2가 겹치는지 여부 반환
ST_Crosses(ST_Geometry1, ST_Geometry2, time)	시간 time에 ST_Geometry1과 ST_Geometry2가 교차하는지 여부 반환
ST_Contains(ST_Geometry1, ST_Geometry2, time)	시간 time에 ST_Geometry1이 ST_Geometry2를 포함하는지 여부 반환
인터벌 시공간 관계 연산자	설 명
ST_Disjoint(ST_Geometry1, ST_Geometry2, stTime, edTime)	인터벌 stTime와 edTime사이에 ST_Geometry1과 ST_Geometry2가 떨어져 있는지 여부 반환
ST_Touches(ST_Geometry1, ST_Geometry2, stTime, edTime)	인터벌 stTime와 edTime사이에 ST_Geometry1과 ST_Geometry2의 경계가 만나는지 여부 반환
ST_Overlaps(ST_Geometry1, ST_Geometry2, stTime, edTime)	인터벌 stTime와 edTime사이에 ST_Geometry1과 ST_Geometry2가 겹치는지 여부 반환
ST_Crosses(ST_Geometry1, ST_Geometry2, stTime, edTime)	인터벌 stTime와 edTime사이에 ST_Geometry1과 ST_Geometry2가 교차하는지 여부 반환
ST_Contains(ST_Geometry1, ST_Geometry2, stTime, edTime)	인터벌 stTime와 edTime사이에 ST_Geometry1이 ST_Geometry2를 포함하는지 여부 반환

Table 3. Spatio-temporal analysis operators for timestamp queries

타임스탬프 시공간 분석 연산자	설 명
ST_Union(ST_Geometry1, ST_Geometry1, time)	시간 time에 ST_Geometry1과 ST_Geometry2를 합친 객체를 반환
ST_Difference(ST_Geometry1, ST_Geometry1, time)	시간 time에 ST_Geometry1에서 ST_Geometry2를 제외한 객체를 반환
ST_Intersection(ST_Geometry1, ST_Geometry1, time)	시간 time에 ST_Geometry1과 ST_Geometry 2 사이에 교차하는 객체를 반환
ST_Distance(ST_Geometry1, ST_Geometry1, time)	시간 time에 ST_Geometry1과 ST_Geometry2 사이의 가장 짧은 거리를 반환

3.2.2 시간 연산자 모듈

Table 4에서 보는 바와 같이 타임스탬프/인터벌 시간 관계 연산자는 입력값으로 ST_Geometry, time 혹은 ST_Geometry, stTime, edTime을 입력받고, 특정 시간 time 혹은 특정 인터벌 [stTime, edTime]을 기준으로 시간 연산을 수행한다. 또한 각 연산자에 대한 결과값으로는 True 혹은 False를 반환한다. Table 5는 인터벌 질의를 위한 시간 분석 연산자를 보여준다.

Table 5에서 보는 바와 같이 인터벌 질의를 위한 시간 분석 연산자는 입력값으로 ST_Geometry, stTime, edTime을 입력받고, 결과값으로 인터벌을 반환한다.

3.2.3 시공간 인덱스 모듈

시공간 인덱스 모듈은 모바일 장치에서 검색 속도 향상을 위해 플래시 메모리를 최적화하기 위해 노드 크기를 줄일 수 있는 MBR 압축 기법과 플래시 메모리 쓰기 연산과 소거 연산을 줄일 수 있는 버퍼링 기법을 적용한 시공간 인덱스를 제공한다.

Table 4. Temporal relation operators for timestamp/interval queries

타임스탬프 시간 관계 연산자	설 명
T_Before(ST_Geometry, time)	시간 time이 ST_Geometry의 time에 선행하는지 여부 반환
T_After(ST_Geometry, time)	시간 time이 ST_Geometry의 time에 후행하는지 여부 반환
T_Equals(ST_Geometry, time)	시간 time과 ST_Geometry의 time이 동일하는지 여부 반환
인터벌 시간 관계 연산자	설 명
T_Contains(ST_Geometry, stTime, edTime)	인터벌 stTime와 edTime사이에 ST_Geometry의 time을 포함하는지 여부 반환
T_Disjoint(ST_Geometry, stTime, edTime)	인터벌 stTime와 edTime사이에 ST_Geometry의 time이 포함되지 않는지 여부 반환

Table 5. Temporal analysis operators for interval queries

인터벌 시간 분석 연산자	설 명
T_Union(ST_Geometry, stTime, edTime)	인터벌 stTime와 edTime사이에 ST_Geometry의 time과 합집합을 반환
T_Intersection(ST_Geometry, stTime, edTime)	인터벌 stTime와 edTime사이에 ST_Geometry의 time과 교집합을 반환
T_Difference(ST_Geometry, stTime, edTime)	인터벌 stTime와 edTime사이에 ST_Geometry의 time과 차집합을 반환

2byte	3bit	4-16bit
좌하점의 상대 좌표	우상점의 길이 플래그	우상점의 실제 값
124	010	10001100

Figure 4. An example of RSMBR structure

가. MBR 압축 기법

시공간 인덱스 모듈은 저장 공간 활용도를 높이기 위해 RSMBR(Relative-Sized MBR) 기법(김정준 등, 2007)을 사용한다. 이 기법은 MBR의 정확도를 유지하면서 MBR 좌표값 크기를 줄여서 데이터의 양을 줄일 수 있는 기법이다. Figure 4는 RSMBR의 구조 및 실제 좌표값이 저장된 예를 보여준다.

Figure 4에서 보는 바와 같이 RSMBR은 좌하점의 상대 좌표, 우상점의 길이 플래그, 우상점의 실제 값을 가진다. 우상점의 길이에 대한 플래그는 실제 값이 차지하는 bit의 크기를 나타낸다. 즉, 1, 2, 3, 4는 각각 실제 값이 4bit, 8bit, 12bit, 16bit의 크기를 가지고 있다는 것을 의미한다. 그러므로 각 bit 단위마다 값은 15, 255, 4095, 65535의 크기까지 표현할 수 있다. 따라서, RSMBR에서 각 (X, Y) 좌표값은 최소 6byte, 최대 10byte로 표현 가능하기 때문에 전체적인 MBR의 저장 공간이 줄어들게 된다.

나. 버퍼링 기법

시공간 인덱스 모듈은 대용량의 시공간 데이터를 효율적으로 관리하기 위해 버퍼링 기법(김정준 등, 2009)을 사용한다. 버퍼링 기법에서 시공간 데이터가 삽입되는 경우에는 시공간 데이터의 엔트리를 삽입 버퍼에 임시로 저장하였다가 누적된 대량의 엔트리들을 일시에 플래시 메모리에 저장된 대상 트리에 삽입한다. 또한 시공간 데이터가 갱신 또는 삭제되는 경우에는 대상 트리에 바로 반영하지 않고 해당 엔트리를 갱신 버퍼 또는 삭제 버퍼에 임시로 저장한 후에 갱신 버퍼 및 삭제 버퍼에 저장된 리프 노드의 주소를 이용해서 직접 리프 노드로 접근하여 시공간 데이터를 갱신 또는 삭제한다.

3.3 시공간 질의 최적화 관리자

본 절에서는 시공간 질의 최적화 관리자를 구성하는 질의 변환 모듈, 비용 산정 모듈, 실행 계획 생성 모듈에 대해 설명한다.

3.3.1 질의 변환 모듈

질의 변환 모듈은 시공간/시간 질의에 대해 보다 빠른 실행 계획을 얻기 위해 주어진 시공간/시간 질의를 질의 변환 규칙에 따라 논리적으로 동등한 형태로 변환하여 재작성한다. Figure 5는 질의 변화 모듈에서 제공하는 질의 변환 규칙을 보여준다.

시공간/시간 질의를 수행하기 전에 WHERE 조건절을 분석하여 Figure 5에서 주어진 질의 변환 규칙을 적용해 주어진 질의를 변환한다. 이러한 질의 변환은 검색하는 레코드의 수를 감소시켜 보다 빠른 질의 수행 속도 향상을 야기할 수 있다. 즉, 주어진 시공간 질의에 대해서 질의 변환한 후 재작성하고, 재작성된 질의를 수행함으로써 질의 수행 속도를 향상시킬 수 있다.

3.3.2 비용 산정 모듈

비용 산정 모듈은 시공간 히스토그램 기법(양형식 등, 2011)을 이용한 선택율을 계산하여 근사적인 시공간 SELECT 질의의 결과 크기를 추정한다. 시공간 히스토그램 기법은 인덱스 성능에 영향을 미치는 데이터 비율을 기반으로 버킷을 동적으로 분할한다. 본 논문에서 사용한 시공간 히스토그램 기법은 버킷을 분할하기

Figure 5. Query transformation rules

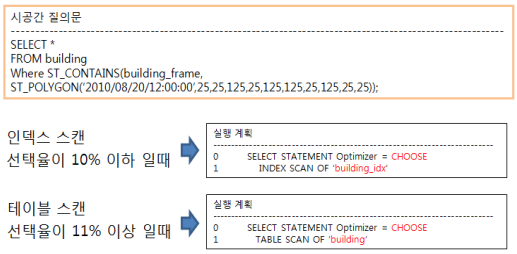

Figure 6. An example of query execution plan

전에 생성될 초기 버킷 개수를 결정하고, 초기 버킷 개수가 결정되면 버킷 개수에 맞게 편중도와 가중치를 이용하여 버킷을 재귀적으로 분할하여 히스토그램을 생성한다. 또한 이렇게 최종 분할된 히스토그램은 시공간 데이터 특성에 맞게 시간에 따른 데이터 변화율을 기반으로 재구축되어 여러 개의 시공간 히스토그램으로 생성된다.

3.3.3 실행 계획 생성 모듈

실행 계획 생성 모듈은 시공간 히스토그램을 이용하여 계산된 선택율을 기반으로 질의 실행 계획을 생성하고, 가장 최소의 비용을 가진 질의 실행 계획을 선택한다. 이 때, 일반적으로 컬럼의 선택율이 10~15%를 넘지 않을 경우에 인덱스가 검색 성능에 최적의 효율을 제공하므로 본 논문에서는 이를 적용하였다(Guttman, 1984). Figure 6은 시공간 질의에 대해 생성된 질의 실행 계획의 예를 보여준다.

Figure 6에서와 같이 본 논문에서는 인덱스 성능에 영향을 미치는 선택율을 10%라고 가정하고, 주어진 시공간 질의에 대해 선택율을 계산하여 선택율이 10%이하는 시공간 인덱스 스캔 방식의 질의 실행 계획을 생성하고 선택율이 11%이상은 전체 테이블 스캔 방식의 질의 실행 계획을 생성한다.

3.4 시공간 SQL 관리자

본 절에서는 시공간 SQL 관리자를 구성하는 시공간 SQL 모듈, 시간 SQL 모듈, 시공간 파서(parser) 모듈에 대해 설명한다.

3.4.1 시공간 SQL 모듈

시공간 SQL 모듈은 모바일 장치용 시공간 질의 처리 시스템이 지원하는 시공간 질의를 위한 시공간 SQL을 제공하는데, 시공간 SQL 문은 시공간 데이터 정의어와 시공간 데이터 조작어로 구분된다. Table 6은 시공간 SQL 모듈에서 지원하는 시공간 데이터 정의어 예를 보여준다.

Table 6에서 보는 바와 같이 테이블을 생성하기 위해 CREATE TABLE 문과 테이블을 삭제하기 위해서 DROP TABLE 문을 제공한다. 또한, 인덱스를 생성하기 위한 CREATE 문과 인덱스를 삭제하기 위한 DROP 문을 제공한다.

Table 7은 시공간 SQL 모듈에서 지원하는 시공간 데이터 조작어 예를 보여준다.

Table 7에서와 같이 시공간 데이터 조작어 예에서 검색은 시간 '2011/05/08/12:00:00'에 ST_Polygon('2011/

Table 6. An example of spatio-temporal data definition language

구분	예제
테이블 생성	CREATE TABLE building (building_id INTEGER, building_name CHAR(25) NOT NULL, building_frame ST_Polygon, CONSTRAINTS c_pk_building_id PRIMARY KEY(building_id));
테이블 삭제	DROP TABLE building;
인덱스 생성	CREATE STtree Building_idx ON Building (building_frame);
인덱스 삭제	DROP STtree Building_idx;

Table 7. An example of spatio-temporal data manipulation language

구분	예제
레코드 검색	SELECT building_id, building_frame FROM building WHERE ST_Touches(ST_Polygon('2011/05/07/15:30', 10, 10, 20, 10, 20, 20, 10, 20, 10, 10), building_frame, '2011/05/08/12:00:00');
레코드 삽입	INSERT INTO building(building_id, building_name, building_frame) VALUES(9001201, '새천년관', ST_Polygon('2011/05/07/15:30', 10, 10, 20, 10, 20, 20, 10, 20, 10, 10));
레코드 삭제	DELETE FROM building WHERE ST_Contains(ST_Polygon('2011/05/07/15:30', 10, 10, 20, 10, 20, 20, 10, 20, 10, 10), building_frame, '2011/09/09/12:00');
레코드 갱신	UPDATE building SET building_frame = ST_Polygon('2011/05/07/15:50', 10, 10, 20, 10, 20, 20, 10, 20, 10, 10) WHERE building_name = '새천년관';

05/07/15:30', 10, 10, 20, 10, 20, 20, 10, 20, 10, 10)과 경계가 만나는 건물의 building_id, building_frame을 검색하는 예이며, 삽입은 building_id, building_name, building_frame이 각각 '9001201', '새천년관', ST_Polygon('2011/05/07/15: 30',10,10,20,10,20,20,10,20,10,10)인 데이터를 삽입하는 예이다. 그리고 삭제는 시간 '2011/09/09/12:00'에 ST_Polygon('2011/05/07/15:30',10,10,20,10,20,20,10,20,10,10)에 포함되는 건물을 삭제하는 예이며, 갱신은 building_name이 '새천년관'인 건물의 building_frame을 ST_Polygon ('2011/ 05/07/15:50', 10, 10, 20, 10, 20, 20, 10, 20, 10, 10)으로 갱신하는 예이다.

3.4.2 시간 SQL 모듈

시간 SQL 모듈은 모바일 장치용 시공간 질의 처리 시스템이 지원하는 시간 질의를 위한 시간 SQL을 제공한다. 시간 SQL 문은 시간 데이터 정의어와 시간 데이터 조작어로 구분되는데, 시간 데이터 정의어는 시공간 데이터 정의어와 동일하게 사용된다. Table 8은 시간 SQL 모듈에서 지원하는 시간 데이터 조작어 예를 보여준다.

Table 8에서 보는 바와 같이 시간 데이터 조작어 예에서 검색은 인터벌 '2001/01/01/00:00:00'에서 '2011/05/08/12:00:00' 사이에 building_frame 시간이 포함되는 건물의 building_name, building_frame을 검색하는 예이며, 삽입은 building_frame이 ST_Polygon ('2011/05/08/12:00:00',10,10,20,10,20,20,10,20,10,10)이고, building_name이 '본관'인 데이터를 삽입하는 예이다. 그리고 삭제는 시간 '2011/05/08/12:00:00'과 building_frame의 시간이 일치하는 건물을 삭제하는 예이며, 갱신은 인터벌 '2008/01/01/00:00:00'과 '2008/ 12/31/24:00:00' 사이에 building_frame의 시간이 포함되는 않는 건물의 building_frame을 ST_Polygon('2009/05/07/15:50:08',10,10,20,10,20,20,10,20,10,10)으로 갱신하는 예이다.

Table 8. An example of temporal data manipulation language

구분	예제
레코드 검색	SELECT building_name, building_frame FROM building WHERE T_Contains(building_frame, '2001/01/01/00:00:00', '2011/05/08/12:00:00');
레코드 삽입	INSERT INTO building(building_frame, building_name) VALUES(ST_Polygon('2011/05/08/12:00:00',10,10 ,20,10,20,20,10,20,10,10), '본관');
레코드 삭제	DELETE FROM building WHERE T_Equals(building_frame,'2011/05/08/12:00:00');
레코드 갱신	UPDATE building SET building_frame = ST_Polygon('2009/05/07/15:50:08',10,10,20,10,20, 20,10,20,10,10) WHERE T_Disjoint(building_frame, '2008/01/01/00:00:00','2008/12/31/24:00:00');

3.4.3 시공간 파서 모듈

시공간 파서 모듈은 시공간/시간 질의에 대한 어휘 분석을 통해 토큰을 생성하고, 오류를 검사하기 위해 시공간 SQL 키워드와 연산자에 대한 키워드를 사용한다. 또한 생성된 토큰을 기반으로 시공간/시간 질의에 대한 구문 분석을 통해 오류를 검사하고 파스 트리(Parse Tree)의 실행 가능 여부를 검사한다. 즉, 구성된 파스 트리를 기반으로 정의된 시공간 SQL 구문과 비교 분석하여 오류가 없으면 주어진 시공간/시간 질의를 실행한다.

4. 시스템 구현

4.1 구현 환경

본 논문에서는 모바일 장치용 시공간 질의 처리 시스템을 구현하기 위해 운영체제는 Microsoft Windows XP Professional Service Pack3을 사용하였고, 도구는 Microsoft Visual Studio 2005를 사용하였으며, 언어는 C++를 사용하였다. 또한 SDK로는 Window Mobile 5.0 Pocket PC SDK를 사용하였다. 그리고 실험을 위해 Intel PXA 270 520MHz, RAM 128MB, Microsoft Windows Mobile 5.0의 사양의 블루버드 503 PDA를 사용하였다.

4.2 가상 시나리오

본 논문에서 개발한 모바일 장치용 시공간 질의 처리 시스템의 효용성을 검증하기 위해 다양한 모바일 GIS 응용 분야 중 GPS를 이용한 건물 위치 측량 시나리오에 적용해 보았다. GPS를 이용한 건물 위치 측량 시나리오는 사용자가 언제 어디서나 실시간으로 GPS 정보를 받아 건물의 위치 정보를 측량하여 보다 정확한 건물의 위치 정보를 삽입/삭제/갱신/검색하는 시나리오이다.

4.2.1 건물 정보 삽입

건물 정보를 삽입하기 위해 GPS를 연결하여 현재 위치를 기반으로 건물의 위치 정보 등을 입력받아 저장한다. Figure 7은 GPS를 이용하여 건물 정보를 삽입하는 예를 보여준다.

Figure 7에서 보는 바와 같이 ①에서는 INPUT 메뉴를 사용하여 건물 위치 정보를 입력받고, ②에서는 입력받은 위치 정보를 INSERT 메뉴를 사용하여 저장한다. ③에서는 MODIFY 메뉴를 사용하여 건물의 부가적인 시간과 건물 이름을 입력받아 위치 정보와 함께 삽입한다. ④에서는 삽입된 건물을 화면에서 보여주고,

Figure 7. Insert of building location information

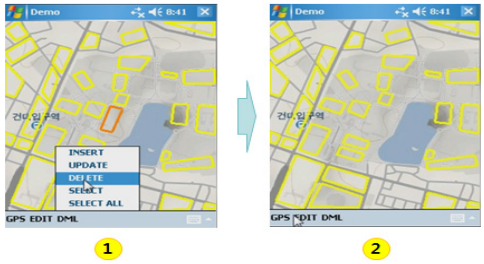

Figure 8. Delete of building location information

⑤에서는 삽입된 건물에 대한 전체 정보(즉, ID, 건물 이름, 시간, 위치)를 보여준다.

4.2.2 건물 정보 삭제

건물 정보를 삭제하기 위해 선택된 건물의 위치 정보 등을 삭제한다. Figure 8은 선택된 건물을 삭제하는 예를 보여준다.

Figure 8에서 보는 바와 같이 ①에서는 선택된 건물을 DELETE 메뉴를 사용하여 삭제하고, ②에서는 선택된 건물이 삭제된 화면을 보여준다.

4.2.3 건물 정보 갱신

건물 정보를 갱신하기 위해 선택된 건물에 대한 건물의 위치 정보 등을 갱신한다. Figure 9는 선택된 건물에 대해서 건물 정보를 갱신하는 예를 보여준다.

Figure 9에서 보는 바와 같이 ①에서는 선택된 건물에 대해서 UPDATE POINT 메뉴를 사용하여 건물의 위치 정보를 입력받고, ②에서는 입력받은 건물의 위치 정보를 기반으로 UPDATE 메뉴를 사용하여 갱신한다. ③에서는 MODIFY 메뉴를 사용하여 건물의 부가적인 시간과 건물 이름을 입력받아 위치 정보와 함께 갱신한다. ④에서는 갱신된 건물을 화면에서 보여주고, ⑤에서는 갱신된 건물에 대한 전체 정보를 보여준다.

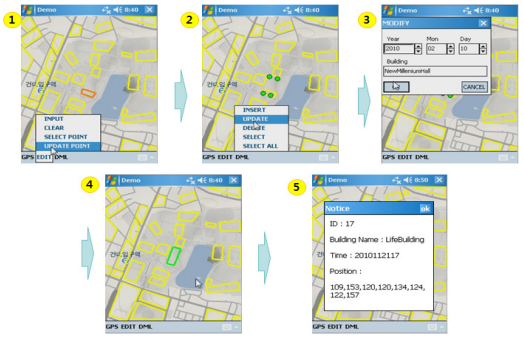

Figure 9. Update of building location information

Figure 10. Search of building location information

4.2.4 건물 정보 검색

건물 정보를 검색하기 위해 입력받은 범위에 포함되는 건물의 위치 정보 등을 검색한다. Figure 10은 입력받은 범위에 포함되는 건물의 정보를 검색하는 예를 보여준다.

Figure 10에서 보는 바와 같이 ①에서는 SELECT POINT 메뉴를 사용하여 검색할 범위를 입력하고, ②에서는 입력받은 검색 범위를 기반으로 SELECT 메뉴를 사용하여 검색한다. ③에서는 건물에 대한 부가적인 정보인 시간과 수행할 시공간 연산자를 입력하여 검색한다. ④에서는 검색된 건물을 화면에서 보여주고, ⑤에서는 검색된 건물에 대한 전체 정보를 보여준다.

5. 결 론

본 논문에서는 모바일 장치에서 시공간 데이터에 대한 질의 처리 및 관리를 지원하는 모바일 장치용 시공간 질의 처리 시스템을 개발하였다. 모바일 장치용 시공간 질의 처리 시스템은 시공간 데이터 타입과 시공간 연산자를 지원하고, 모바일 저장 장치인 플래시 메모리를 최적화하기 위해 MBR 압축 기법과 버퍼링 기법을 적용한 시공간 인덱스 제공한다. 또한, 시공간 질의 최적화를 위해 인덱스 성능에 영향을 미치는 데이터 비율을 기반으로 버킷 개수를 동적으로 분할하는 시공간 히스토그램 기법을 사용한다.

마지막으로, 본 논문에서 구현한 모바일 장치용 시공간 질의 처리 시스템을 GPS를 이용한 건물 위치 측량 시나리오에 적용하여 검증함으로써 본 시스템이 모바일 환경에서 시공간 데이터 처리가 필요한 다양한 응용분야에 유용하게 사용될 수 있음도 확인하였다.

감사의 글

본 연구는 국토해양부 첨단도시기술개발사업-지능형 국토정보기술혁신 사업과제의 연구비지원(10국토정보 J71)에 의해 수행되었습니다.

참고문헌

1. 김은형, 2009, 유비쿼터스 지리정보 표준화 동향, 2009 GIS 공동추계학술대회, 한국지형공간정보학회, pp. 329-335.
2. 김정준, 강홍구, 김동오, 한기준, 2007, 메인 메모리 다차원 인덱스를 위한 효율적인 MBR 압축 기법, 한국공간정보시스템학회, 9권 2호, pp.13-23.
3. 김정준, 심희정, 강홍구, 이기영, 한기준, 2009, 플래시 메모리 기반 효율적인 공간 인덱스, 한국공간정보시스템학회, 11권 2호, pp.133-142.
4. 민경욱, 2008, 실시간 맵 업데이트를 위한 모바일 공간 DBMS 개발, 2008 공동춘계학술대회, 한국GIS학회, pp.37-40.
5. 서용철, 사공호상, 이영주, 2008, 우리나라와 일본의 국가 GIS 추진전략 및 유비쿼터스 정책 비교분석, 한국지형공간정보학회, 16권 2호, pp.3-8.
6. 신병철, 이종연, 2006, 시점 질의를 위한 선택율 추정, 한국정보과학회 데이터베이스, 33권 2호, pp. 214-223.
7. 양형식, 김정준, 한기준, 2011, 시공간 모바일 DBMS에서 질의 최적화를 위한 히스토그램 기법, 건국기술연구논문, 건국대학교 산업기술연구원, 제36집, pp.1-10.
8. 황환규, 2004, 공간 데이터베이스에서 질의 결과 크기 추정을 위한 공간 분할, 대한전자공학회, CI편 41권 2호, pp.23-32.
9. Acharya, S., Poosala, V. and Ramaswamy, S., “Selectivity Estimation in Spatio Databases,” Proc. of the ACM SIGMOD Int. Conf. on Management of Data, 1999, pp.13-24.
10. ETRI, 2008, FUNs Specification (Core) - (Flash-aware

Ubiquitous Navigation system), Version: 1.0.
11. Guttman, A., “R-trees : A Dynamic Index Structure for Spatio Searching,” ACM SIGMOD Int. Conf. on Management of Data, 1984, pp.47-57.
12. Min, K.W., An, K.H., Kim, J.W. and Jin, S.I., “The Mobile spatio DBMS for the Partial Map Air Update in the Navigation,” Proc. of the 11th IEEE Int. Conf. on Intelligent Transportation Systems, 2008, pp.476-481.
13. Open Geospatial Consortium, Inc, OpenGIS Implementation Specification for Geographic Information-Simple Feature Access-Part 2:SQL Option, Version 1.2.1, 2010.
14. Piatesky, S.P. and Connell, G., “Accurate Estimation of the Number of Tuples Satisfying a Condition,” Proc. of the ACM SIGMOD Int. Conf. on Management of Data, 1984, pp.256-275.
15. Tao, Y. and Papadias, D., “The MV3R-tree: A Spatio-Temporal Access Method for Timestamp and Interval Queries,” Proc. of the Int. Conf. on Very Large Data Bases, 2001, pp.431-440.
16. Theodoridis, Y. and Sellis, T.K., “A Model for the Prediction of R-tree Performance,” Proc. of the ACM SIGMOD Int. Conf. on Management of Data, 1996, pp.161-171.
17. Wang, M., Vitter, S.J., Lim, L. and Padmanabhan, S., “Wavelet-Based Cost Estimation for Spatio Queries,” Lecture Notes in Computer Science, Volume 2121, 2001, pp.175-193.