시공간 데이터베이스 시스템을 위한 질의어의 설계

박 동 선*, 박 춘 명**
* 국립 충주대학교 전기전자정보공학부
** 국립 충주대학교 컴퓨터공학과

Design of Query Language for Spatio-Temporal Database System

Park Dong-Seon* and Park Chun-Myoung**
* School of Electrical Electronic Information Engineering Faculty, Chung-Ju National University.
** Dept. of Computer Engineering, Chung-Ju National University.

요 약

90년대 초반부터 공간정보와 함께 시간정보를 포함하는 시공간 데이터베이스 시스템에 대한 연구가 활발히 진행되고 있지만, 대부분의 연구가 시공간 데이터모델에 치중하고 있으며 시공간 질의어에 대한 연구는 최근에 시작되고 있다.

본 논문에서는 시간 정보의 포함 여부에 따라 다양한 형태로 구분되는 시공간 테이블을 효율적으로 관리할 수 있으며, 비공간정보에 대한 조건과 공간정보에 대한 조건 및 시간정보에 대한 조건으로 구성되는 시공간 질의를 간결하고 효과적으로 표현할 수 있는 시공간 질의어를 설계한다.

1. 서론

공간 데이터베이스 시스템의 활용 분야가 다양해짐에 따라 환경영향평가, 도시계획, 재난방지, 이동체 추적, 시설물 관리 등과 같은 분야에서도 이를 이용하려는 노력이 진행되고 있다. 이러한 분야에서는 객체의 현재 상태와 함께 과거 상태를 이용하여 공간 객체의 변화 과정에 대한 분석을 통한 보다 효율적인 공간 객체의 관리를 필요로 한다[1]. 이를 위하여 지난 90년대 초반부터 시공간 데이터베이스 시스템에 대한 연구가 활발히 진행중이지만 대부분의 연구가 시공간 데이터 모델[2,3,4,5]에 치중되어 있으며, 시공간 데이터베이스 시스템을 대상으로 발생하는 다양한 질의를 효과적으로 표현하고 실행할 수 있는 질의어에 대한 연구는 많지 않다.

본 논문에서는 시간 정보의 포함에 따라 다양한 형태로 구성되는 시공간 데이터베이스를 효율적으로 관리할 수 있고, 다양한 조건을 포함하는 질의 구문을 쉽고 명확하게 작성할 수 있는 시공간 질의어를 설계한다. 본 논문에서 설계하는 시공간 질의어는 MBFA 기법과 분리된 저장구조[6]를 기반으로 하며, 현재 표준화 작업이 진행중인 SQL3[7]의 규격을 참조하였다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로 기존에 연구된 시공간 질의어의 장·단점을 살펴보고, 분리된 저장구조와 MBFA기법에 대하여 간단히 살펴본다. 3장에서는 데이터 정의어와 데이터 조작어로 구성되는 시공간 질의어를 설계하며, 4장에서는 설계한 질의어를 이용하여 실세계에서 발생할 수 있는 질의에 대한 구문 작성 예를 보인다. 마지막 5장에서는 결론과 함께 향후 연구과제에 대하여 알아본다.

2. 관련연구

2.1 기존의 시공간 질의어

시공간 질의어에 대한 연구는 최근에 시작되어 GUI(Graphic User Interface)를 이용한 질의 방식에 대한 연구와[8], SQL을 확장한 질의어인 STSQL[9]과 STQL[10]등이 발표되었다. STSQL과 STQL의 특징과 문제점은 다음과 같다.

STSQL(Spatio-Temporal SQL)[9]은 기존의 DBMS에 시간 정보를 추가하여 시공간 DBMS로 변경할 때, 기존의 DBMS를 기반으로 작성된 응용 프로그램과 응용 프로그램에서 이용하는 데이터들을 수정하지 않고 그대로 유지할 수 있는 방법과, 시공간 DBMS에서 개발된 응용 프로그램이 기존의 응용프로그램과 조화를 이룰 수 있게 하는 방법을 제공하는 것을 목적으로 연구되었다.

STSQL에서는 시간 정보나 공간 정보를 포함하는 다차원 테이블(dimensional table)을 1차원 스냅 테이블(snapshot table)의 집합으로 간주하며, 질의어에서

시간 정보나 공간 정보를 저장하고 있는 다차원 어트리뷰트(dimension attribute)를 다루는 방법을 지시하기 위하여 다음의 두 가지의 플래그(flag)를 제공한다.

- REDUCIBLE : 다차원 테이블을 스냅 테이블로 일반화하여 기존의 SQL구문을 이용하여 작성한 스냅 질의를 수행할 수 있도록 한다.
- NONREDUCIBLE : REDUCIBLE을 이용하여 스냅 어트리뷰트로 변환하면 질의의 수행이 불가능하게 되는 어트리뷰트를 지정하여 일반적인 어트리뷰트와 동일하게 질의에서 다루어져야 함을 나타낸다.

이와 같이 STSQL은 기존의 스냅 데이터베이스를 대상으로 하는 질의와, 다차원 데이터베이스를 대상으로 하는 질의간의 호환성에 중점을 두어 설계된 질의어이다. 따라서, 다양한 조건으로 구성되는 시공간 질의를 효과적으로 표현할 수 있는 방법을 제공하지 못하고 있다.

STQL(Spatio-Temporal Query Language)[10]은 시간의 흐름에 따라 동적으로 변하는 시공간 위상 관계에 대한 질의를 효율적으로 표현하기 위한 질의어로, 기존의 공간 위상 술어를 이용하여 시공간 술어를 정의하는 방법에 제공한다. 즉, 여러 개의 공간 위상 술어를 이용하여 표현하는 질의 구문을 간결하게 표현하기 위하여 STQL에서는 DEFINE 구문을 이용하여 새로운 시공간 위상 술어를 정의할 수 있는 방법을 제공한다.

```
DEFINE Enters AS Disjoint>>Meet>>Inside
SELECT flights.id
FROM   flights, weather
WHERE weather.kind = "hurricane"
AND    Route Enters Extent
```

STQL은 시공간 위상 관계에 대한 술어를 쉽게 정의할 수 있으며, 이를 이용하여 질의 구문을 단순화할 수 있다는 장점이 있으나, STSQL과 동일하게 속성 정보와 공간 정보, 시간 정보와 관련된 다양한 조건들로 구성되는 질의를 표현할 수 있는 방법을 제공하지 않는다.

2.2 분리된 저장구조와 MBFA기법

[6]에서는 객체 정보의 유효 시간을 기준으로 현재 시점에 유효한 데이터는 현재 데이터베이스에 저장하고, 과거시점에 유효한 데이터는 과거 데이터베이스에 저장하도록 하는 분리된 저장구조를 설계하였으며, 과거데이터의 중복을 최소로 하고, 현재데이터 및 과거데이터에 대한 검색 연산을 빠르게 수행할 수 있는 저장기법인 MBFA기법을 설계하였다.

현재 데이터베이스는 공간객체의 상태를 표현하는 모든 공간정보를 저장하는 현재테이블로 구성되며, 과거 데이터베이스는 공간 객체의 상태를 표현하는 데이터들을 저장하는 과거테이블과, 각 공간 객체의 기준상태(base-state)에 대한 정보를 저장하는 기준상태 테이블로 구성된다. 과거테이블에 저장되는 과거 데이터는 기준상태와 비교하여 추가되거나 삭제된 데이터를 저장하여 데이터베이스의 크기를 줄일 수 있도록 하였다.

3. 시공간 질의어의 설계

본 장에서는 공간 객체의 현재 상태와 함께 과거의 상태를 저장하기 위한 데이터베이스를 효율적으로 관리할 수 있으며, 비공간정보에 대한 조건과 공간정보에 대한 조건 및 시간정보에 대한 조건을 포함하는 시공간 질의를 효과적으로 표현할 수 있는 시공간 질의어를 설계한다.

3.1 데이터 정의어의 설계

시공간 데이터 정의어에는 시간정보를 포함하는 테이블, 뷰 및 색인의 생성문과 변경문 및 삭제문이 포함되며, 이들 문장은 기존의 문자기반 속성과 함께 공간 속성 및 시간 속성에 대한 표현 방법을 포함한다. 본 논문에서는 테이블 생성문과 변경문에 대해서만 정의한다.

3.1.1 시공간 테이블 생성문

시간 정보와 공간 정보를 포함하는 테이블의 생성은 CREATE TABLE문에 의하여 수행되며, 형식은 다음 그림1과 같다.

```
CREATE TABLE <table name>
   [SHAPE <spatial data type>]
   [AS <table type>]
   [TIME_GRANULE <temporal granularity>]
   <table contents source>
   [ON COMMIT <table commit action> ROWS]
```

(그림 1) 시공간 테이블 생성문

① SHAPE <spatial type>

<spatial data type>에는 레이어를 구성하는 공간 객체들의 공간적 특징인 점(POINT), 선(LINE), 다각형(POLYGON), 다각선(POLYLINE)을 기술할 수 있다. SHAPE절은 생략할 수 있으며, 이 경우에는 문자나 숫자로만 구성된 테이블을 생성함을 의미한다.

② AS <table type>
생성하고자 하는 테이블의 종류를 기술한다. <table type>에 기술할 수 있는 테이블의 종류는 SNAPSHOT, ROLLBACK, HISTORICAL, TEMPORAL의 네 가지이며, HISTORICAL이나 TEMPORAL타입의 테이블을 생성하는 경우는 아래와 같이 <time represent>에 유효시간의 표현 방법을 기술하여야 한다. <time represent>에는 사건 타입의 EVENT와 기간 타입의 INTERVAL이 올 수 있다.

```
<table type> ::=
      SNAPSHOT
    | ROLLBACK
    | HISTORICAL <time represent>
    | TEMPORAL <time represent>
<time represent> ::= EVENT | INTERVAL
```

AS절은 생략할 수 있으며, 이 경우 유효시간과 거래시간을 모두 포함하지 않는 SNAPSHOT타입의 테이블을 생성하게 되고, TIME_GRANULE절은 기술할 수 없다.

③ TIME_GRANULE <temporal granularity>
HISTORICAL과 TEMPORAL타입의 테이블을 생성하는 경우 테이블에 저장하는 유효시간의 시간 단위를 지정하기 위하여 사용한다. <temporal granularity>에는 'YEAR', 'MONTH', 'DAY', 'TIME', 'MINUTE', 'SECOND'의 여섯 가지 종류의 시간 단위를 지정할 수 있다. 이 절도 생략할 수 있으며, 이 경우는 초 단위까지의 시간을 저장하는 SECOND가 기본 설정값이 된다.

3.1.2 시공간 테이블 변경문

```
ALTER TABLE <table name>
   ADD [COLUMN] <column definition>
       ...
   [ALTER TABLE_TYPE <table type>]
   [ALTER TIME_REPRESENT
                        <time represent>]
   [ALTER TIME_GRANULE
                    <temporal granularity>]
```

(그림 2) 시공간 테이블 변경문

시간지원 시스템에서의 테이블 변경은 테이블 타입의 변경과, 시간 표현 방법의 변경 및 시간 단위의 변경이 가능해야 한다. 본 논문에서 설계한 테이블 변경문의 형태는 (그림 2)와 같다.

① ALTER TABLE_TYPE <table type>
테이블의 타입을 변경하기 위하여 사용한다. 즉, 유효시간이나 거래시간에 대한 어트리뷰트를 추가하거나 삭제하여 테이블에서 저장하는 시간의 종류를 변경하는 경우에 사용한다. 유효시간이나 거래시간이 추가되어 테이블 타입을 변경하고자 하는 경우는 두 시간이 모두 널을 허용하지 않기 때문에 테이블이 비어있는 경우만 가능하며, 이러한 경우 데이터베이스에는 현재 테이블만 존재하기 때문에 과거 테이블에 대한 변경은 고려하지 않아도 된다.

② ALTER TIME_REPRESENT <time represent>
유효시간의 표현방법을 변경하고자 할 때 사용한다. 변경 대상 테이블이 SNAPSHOT테이블이거나 ROLLBACK테이블일 경우는 이 구문을 사용할 수 없다.

③ ALTER TIME_GRANULE <temporal granularity>
테이블에 저장되는 유효시간의 시간 단위를 변경하기 위하여 사용한다.

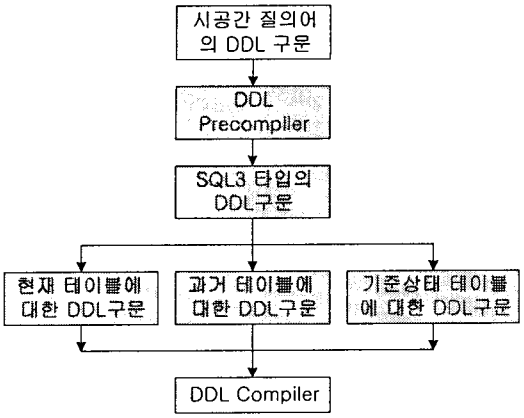
본 논문에서는 기존의 DDL 컴파일러를 수정하지 않고 앞에서 정의한 데이터 정의어를 이용한 질의 구문을 실행하기 위하여 (그림 3)과 같이 DDL 전 처리기(DDL Precompiler)를 둔다. DDL 전 처리기에서는 두 단계의 질의 변형을 수행한다. 첫 번째 단계에서는 TGSQL의 데이터 정의어를 이용한 구문을 기존 SQL의 DDL구문으로 변경하며, 두 번째 단계에서는 첫 번째 단계의 결과를 현재 테이블과 과거 테이블 및 기준상태 테이블에 대한 DDL구문으로 변형하여 DDL 컴파일러에게 전달한다.

3.2 데이터 조작어의 설계

시공간 데이터 조작어는 SELECT, UPDATE, DELETE 및 INSERT 명령어로 구성되지만, 본 논문에서는 데이터 검색문에 대해서만 정의한다.

3.2.1 시공간 데이터 검색문

시공간 데이터베이스 시스템을 대상으로 발생하는



(그림 3) 시공간 DDL구문에 대한 질의 변형 과정

질의의 검색 조건은 비공간정보에 대한 조건과 공간정보 및 시간정보에 대한 조건으로 구성된다. 따라서 본 논문에서는 이들을 명확하게 기술할 수 있도록 하기 위하여 공간정보에 대한 검색 조건을 기술하기 위한 SPATIAL절과 시간정보에 대한 검색 조건을 기술하기 위한 VALID, TRANS, TEMPORAL절을 추가하였다. 유효시간을 이용하는 조건은 VALID절에 기술하며, 거래시간과 관련된 조건은 TRANS절에 기술하고, 유효시간과 거래시간을 동시에 포함하는 조건은 TEMPORAL절에 기술한다. 또한 질의의 수행결과에서 시간정보를 제외한 모든 어트리뷰트의 값이 동일하고 튜플의 시간정보간에 OVERLAP이나 MEET, INCLUDE 또는 EXCLUDE의 시간 위상 관계가 존재하는 튜플들을 처리하는 방법을 지정하기 위하여 <temporal set quantifier>을 추가하였다. <temporal set quantifier>에는 'COALESCE'나 'TEMP_ALL'이 올 수 있다. 본 논문에서 설계한 검색문의 구문 형식은 다음과 같다.

```
SELECT [<set quantifier>]
       [<temporal set quantifier>] <select list>
FROM <table reference list>
[WHERE <search condition>]
[SPATIAL <spatial search condition>]
[VALID <valid time search condition>]
[TRANS <transaction time search condition>]
[TEMPORAL
  <valid & transaction time search condition>]
[GROUP BY <grouping specifications>]
[HAVING <search condition>]
[ORDER BY <sort specification list>]
```

(그림 4) 시공간 데이터 검색문

또한, SELECT절에는 기존의 속성정보와 함께 공간정보 및 시간정보를 검색 연산의 결과로 출력할 수 있도록 하기 위하여 (그림 5)와 같이 <select list>를 확장하였으며, 검색의 결과를 유효시간이나 거래시간을 기준으로 순서화 할 수 있도록 하기 위하여 (그림 6)과 같이 ORDER BY절의 <sort specification list>를 확장하였다. VALID_START와 TRANS_START는 각각 검색된 결과의 유효시간과 거래시간의 시작시간을 기준으로 순서화하며, VALID_END와 TRANS_END는 유효시간과 거래시간의 종료시간을 기준으로 순서화 한다.

```
<select list> ::=
    <asterisk>
  | <select sublist> [{<comma> <select sublist>}...]
<select sublist> ::=
    <derived column>
  | <item qualifier> <period> <asterisk>
<derived column> ::=
    <value expression> [ <as clause> ]
<value expression> ::=
                ...
  | <spatial numerical value expression>
  | <spatial geometrical value expression>
  | <spatial translation value expression>
  | <event type time value expression>
  | <interval type time value expression>
  | <duration type time value expression>
```

(그림 5) SELECT절의 <select list>의 확장

```
<sort specification list> ::=
  <sort specification>
          [{<comma> <sort specification>}...]
<sort specification> ::=
    <sort key> [<collate clause>]
          [<ordering specification>]
<sort key> ::=
      <value expression>
    | <temporal sort condition>
<temporal sort condition> ::=
      VALID_START | VALID_END
    | TRANS_START | TRANS_END
```

(그림 6) ORDER BY절의 확장

4. 질의 예

4.1 테이블 생성

(그림 7)은 건물을 관리하기 위한 테이블의 생성문

의 예이다. 본 장에서는 (그림 7)의 Building테이블을 이용하여 3장에서 설계한 질의어의 예를 제시한다.

```
CREATE TABLE Building
    SHAPE POLYGON
    AS TEMPORAL INTERVAL
    TIME_GRANULE DAY
    ( ID        CHAR(4) NOT NULL,
      Owner     CHAR(20),
      Address   CHAR(60),
      PRIMARY KEY (ID) )
```

(그림 7) 예제 테이블 생성

【테이블 생성문의 질의 변형】

- 1단계 질의 변형 : 테이블 생성문의 SHAPE절과 AS절, TIME_GRANULE절에 기술된 내용에 의하여 공간 정보를 저장하기 위한 어트리뷰트와 시간 정보를 저장하기 위한 어트리뷰트가 첨가된 생성문으로 변형한다. (그림 8)은 (그림 7)의 생성문에 대한 1단계 질의 변형 결과이다.

```
CREATE TABLE Building
( ID        CHAR(4) NOT NULL,
  Owner     CHAR(20),
  Address   CHAR(60),
  Spatial   POLYGON,
  VS        DAY_TIMESTAMP NOT NULL,
  VE        DAY_TIMESTAMP NOT NULL,
  TS        SECOND_TIMESTAMP NOT NULL,
  TE        SECOND_TIMESTAMP NOT NULL,
  PRIMARY KEY (ID) )
```

(그림 8) (그림 7)의 1단계 질의 변형 결과

- 2단계 질의 변형 : 1단계에서 SQL3 구문으로 변형된 테이블 생성문을 이용하여 분리된 저장구조의 현재 테이블과 과거 테이블 및 기준상태 테이블을 생성하는 구문과 함께 각 테이블의 기본 색인을 생성하는 구문으로 변형된다.

```
CREATE TABLE Building_Current
( ID        CHAR(4) NOT NULL,
  Owner     CHAR(20),
  Address   CHAR(60),
  Node      POLYGON,
  VS        DAY_TIMESTAMP  NOT NULL,
  TS        SECOND_TIMESTAMP NOT NULL,
  PRIMARY KEY (ID) )
```

(a) 현재 테이블 생성 구문

```
CREATE TABLE Building_Past
( ID        CHAR(4) NOT NULL,
  Owner     CHAR(20),
  Address   CHAR(60),
  U_Node    POLYLINE,
  D_Node    POLYLINE,
  VS        DAY_TIMESTAMP NOT NULL,
  VE        DAY_TIMESTAMP NOT NULL,
  TS        SECOND_TIMESTAMP NOT NULL,
  TE        SECOND_TIMESTAMP NOT NULL,
  PRIMARY KEY (ID, VS) )
CREATE INDEX Building_Past_Idx2
                    ON Past (ID, VE)
```

(b) 과거 테이블 생성 구문

```
CREATE TABLE Building_Basestate
( ID      CHAR(4) NOT NULL,
  VS      DAY_TIMESTAMP NOT NULL,
  VE      DAY_TIMESTAMP NOT NULL,
  TS      SECOND_TIMESTAMP NOT NULL,
  TE      SECOND_TIMESTAMP NOT NULL,
  PRIMARY KEY (ID, VS) )
```

(c) 기준상태 테이블 생성 구문

(그림 9) (그림 7)의 2단계 질의 변형 결과

4.2 데이터 검색

다음은 속성조건과 공간조건 및 시간조건을 모두 포함하고 있는 질의에 대하여 확장된 SQL의 구문을 이용한 검색문이다.

(질의-1) 유효시간의 간격이 2년 이상인 건물의 주소와 넓이, 유효시간을 검색하여라.

```
SELECT Address, AREA(), VALIDTIME()
FROM Building
VALID TEMP_DURATION(Land, YEAR) >= (YEAR, 2);
```

(질의-2) 건물의 넓이가 300m²이상이고, 속성조건 λ를 만족하는 건물과 유효기간이 동일한 건물의 소유주와 주소, 유효시간을 검색하여 유효시작시간 순서대로 출력하여라.

```
SELECT Owner, Address, VALIDTIME(),
FROM Building
SPATIAL AREA() >= 300
VALID Land TEMP_EQUAL ( SELECT VALIDTIME()
                        WHERE λ )
ORDER BY VALID_START ASC;
```

(질의-3) 건물의 완공일자와 시스템에 기록된 시간의 차이가 2달 이상인 건물의 주소와 완공일자와 시스

템에 기록된 일자를 검색하라.

```
SELECT Address, START_OF(VALIDTIME()),
       START_OF(TRANSACTIONTIME())
FROM Building
TEMPORAL
TEMP_DURATION(START_OF(VALIDTIME
    (Building)), START_OF(TRANSACTIONTIME
    (Building)), MONTH) >= (MONTH, 2);
```

5. 결론 및 향후 연구과제

본 논문에서는 시간 정보의 포함 여부에 따라 다양한 형태로 구분되는 시공간 테이블에 대한 정의어와, 다양한 조건으로 구성되는 시공간 질의를 효과적으로 표현할 수 있는 데이터 조작어를 설계하였다. 본 논문에서 설계한 데이터 정의어는 DDL 전 처리기를 통하여 SQL3의 구문 형태로 질의 변형을 수행함으로써 기존의 DDL컴파일러를 수정하지 않고도 시공간 테이블에 대한 관리를 효과적으로 수행할 수 있도록 하였다. 또한 데이터 조작어는 비공간정보와 공간정보 및 시간정보에 대한 다양한 조건으로 구성되는 시공간 질의어를 간결하고 효과적으로 기술할 수 있도록 설계하여 기존의 SQL사용자들이 쉽게 이용할 수 있도록 하였다.

시공간 데이터베이스는 시간의 흐름에 따라 데이터베이스의 크기가 지속적으로 증가한다. 따라서, 방대한 양의 데이터베이스를 대상으로 하여 보다 효율적인 연산의 수행을 위해서는 시간과 공간의 개념이 모두 포함된 시공간 색인에 대한 연구가 필요하다.

참고문헌

[1] D. Peuquet, L. Qian, "A Integrated Database Design for Temporal GIS", SDH, 1996.

[2] C. Claramunt, M. Theiault, "Toward semantics for modeling Spatio-Temporal processes within GIS", SDH, 1996.

[3] M. Erwig, R. H. Guting, M. Schneider, M. Vazirgiannis, "Spatio-Temporal Data Types: An Approach to Modeling and Querying Moving Objects in Databases", GeoInformatica, 1999.

[4] S. Ram, J.S. Park, "Modeling Spatial and Temporal Semantics in a Large Heterogeneous GIS Database Environment", Proceedings of the 2nd Americas Conference on Information Systems(AIS '96), Phoenix, Arizona, August 16-18, 1996, pp. 683-685.

[5] Toshimi Tagashira, Toshiyuki Amagasa, Masayoshi Aritsugi, Yoshinari Kanamori, "Interval-Based Representation of Spatio-Temporal Concepts", Advanced Information Systems Engineering, 9th International Conference, CAiSE'97, LNCS 1250, 1997, pp. 231-244.

[6] 박동선, 배해영, "다차원 지리정보시스템을 위한 저장기법 및 분리된 저장구조", 정보처리논문지, Vol. 7, No. 1, 2000.

[7] P. Fortier, "SQL3 Implementing the SQL foundation Standard", McGraw-Hill, 1999.

[8] R. Edsall, D. Peuquet, "Graphical query techniques for temporal GIS", 13th Auto-carto Conf., 1997.

[9] M. Böhlen, C. S. Jensen, B. Skjellaug, "Spatio-Temporal Database support for Legacy Applications", ACM Symposium on Applied Computing(SAC '98), 1998.

[10] M. Erwig, M. Schneider, "Developments in Spatio-Temporal Query Languages", 10th International Workshop on Database & Expert Systems Applications, 1999, pp. 441-449.