Specifications for Efficient Indexing in Spatiotemporal Databases

Yannis Theodoridis, Timos Sellis, Apostolos N. Papadopoulos, Yannis Manolopoulos

October 1, 2007

Computer Science Division
Dept. of Electrical and Computer Engineering
National Technical University of Athens
Zographou 15773, Athens, HELLAS (GREECE)
{theodor, timos}@cs.ntua.gr

Department of Informatics
Aristotle University of Thessaloniki
Thessaloniki 54006, HELLAS (GREECE)
{apapadop, manolopo}@athena.auth.gr

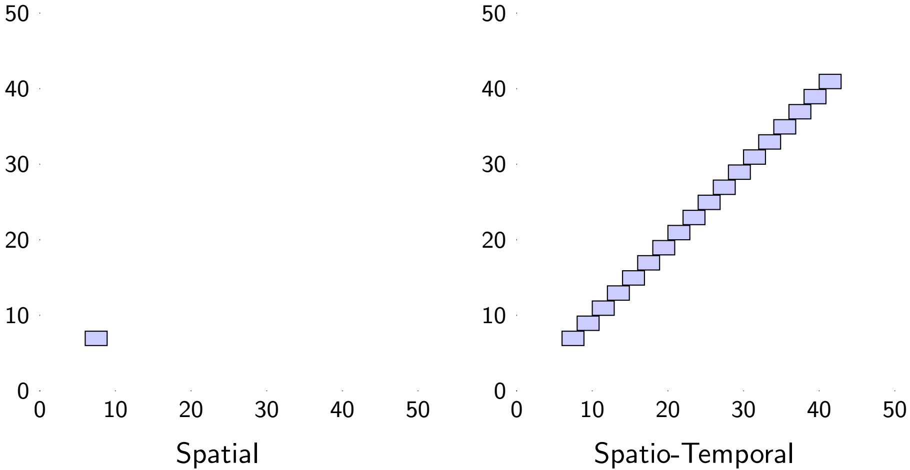
Presented by: Kristian Torp

Outline

1. Motivation
2. Specification
3. Query Support
4. Classification of STAMs
5. Evaluation

Outline

Example



50
40
30
20
10
0
0
10
20
30
40
50
Spatial
50
40
30
20
10
0
0
10
20
30
40
50
Spatio-Temporal

Definition of Spatio-Temporal Object

Definition

A spatio-temporal object o (declared by its identification number o_{id}) is a time-evolving spatial object, i.e., its evolution (or history) is represented by a set of triplets (o_{id}, s_i, t_i), where s_i is the location of object o_{id} at instant t_i (s_i and t_i are called space-stamp and time-stamp, respectively).

Definition of Spatio-Temporal Object

Definition

A spatio-temporal object o (declared by its identification number o_{id}) is a time-evolving spatial object, i.e., its evolution (or history) is represented by a set of triplets (o_{id}, s_i, t_i), where s_i is the location of object o_{id} at instant t_i (s_i and t_i are called space-stamp and time-stamp, respectively).

A merge of two existing research fields

- Spatial databases
- Temporal databases

New Challenges

The Main Challenge

Efficient manipulation of spatial objects and the relationship among them

New Challenges

The Main Challenge

Efficient manipulation of spatial objects and the relationship among them

Details to consider

- Object representation
- Query processing
- Index methods

Terminology

Types of Indices

- Point Access Method (PAM)
- Spatial Access Method (SAM)
- Spatio-Temporal Access Method (STAM)

Outline

Data Set Support

The main discriminator

- Point
- Region

A spatio-temporal DBMS must support both types of data

Temporal Support I

name	dept	VTS	VTE	TTS	TTE
Jim	NY	3	*now*	3	*uc*
Joe	LA	4	*now*	4	11
Joe	LA	4	11	11	*uc*
Joe	UK	11	*now*	11	*uc*
Sam	NY	12	*now*	12	14
Sam	NY	12	14	14	*uc*

Temporal Support II

```
sequenced select *
from emp
as of '2007-09-13';
```

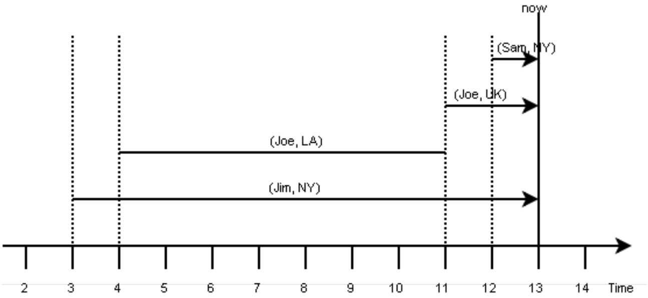
ATSQL

```
select name, dept, vts, vte
from   emp
where  tts <= '2007-09-13'
and    tte >  '2007-09-13'
```

SQL

Name	Dept	VTS	VTE
Jim	NY	3	*now*
Joe	LA	4	11
Joe	UK	11	*now*
Sam	NY	12	*now*

Temporal Support III




```
sequenced validtime
select count(Name)
from emps
```

ATSQL

Count	VTS	VTE
1	3	4
2	4	11
2	11	12
3	12	*now*

Result

Temporal Support IV

```
select count(name), const_period.vts as vts, const_period.vte as vte
from  empat13, (select  t1.vts as vts, t1.vte as vte
                from    empat13 t1
                where   not exists ( select *
                        from   empat13 t2
                        where (t1.vts < t2.vts and t2.vts < t1.vte) or
                              (t1.vts < t2.vte and t2.vte < t1.vte))
                union
                select  t1.vts as vts, t2.vts as vte
                from    empat13 t1, empat13 t2
                where   t1.vts < t2.vts and t2.vts < t1.vte
                and     not exists (select *
                        from   empat13 t3
                        where  (t1.vts < t3.vts and t3.vts < t2.vts) or
                               (t1.vts < t3.vte and t3.vte < t2.vts))
                union
                select  t1.vts as vts, t2.vte as vte
                from    empat13 t1, empat13 t2
                where   t1.vts < t2.vte and t2.vte < t1.vte
                and     not exists (select *
                        from   empat13 t3
                        where  (t1.vts <= t3.vts and t3.vts <= t2.vts) or
                               (t1.vts <= t3.vte and t3.vte <= t2.vts))
               union
               select  t1.vte as vts, t2.vts as vte
               from    empat13 t1, empat13 t2
               where   t1.vte < t2.vts
               and     not exists (select *
                       from   empat13 t3
                       where  (t1.vte < t3.vts and t3.vts < t2.vts) or
```

Temporal Support V

```
                                       (t1.vte < t3.vte and t3.vte < t2.vts))
                    union
                    select  t1.vte as vts, t2.vte as vte
                    from    empat13 t1, empat13 t2
                    where   t2.vts < t1.vte and t1.vte < t2.vte
                    and     not exists (select *
                            from   empat13 t3
                            where  (t1.vte < t3.vts and t3.vts < t2.vte) or
                                   (t1.vte < t3.vte and t3.vte < t2.vte))) const_period
where empat13.vts <= const_period.vts and const_period.vte <= empat13.vte
group by const_period.vts, const_period.vte
order by const_period.vts;
```

Temporal Support VI

- Valid time
- Transaction time

	Transaction-Time: No	Transaction-Time: Yes
Valid Time: No	snapshot	transcation-time
Valid Time: Yes	valid-time	bitemporal

Database Mobility

	Fixed Size	Dynamic Size
Still objects		growing
Moving objects	evolving	full-dynamic

Other Criteria

- Loading of data
- Time-dimension peculiarity
- Support of spatio-temporal queries
  - More on this shortly

Outline

Selection Queries

Example

Find all objects that have lied within a specific area (or at a specific point), during a specific time interval (or at a specific time instant)

Example

Find all spots where it rained yesterday at 12.00 pm

- Filtering on space, within a spatial range
- Filtering on time, at a specific instance or time interval
- Most common type of query

Join Queries

Example

Find all pairs of objects that have lied spatially close (i.e., within distance X), during a specific time interval (or at a specific time instant)

Example

Find all person biking from the railway station to the university staring at the railway station between 12:00 to 14:00 that got wet on the bike ride due to rain.

- Space and time is used in the join condition
- Properly a very expensive operation

Nearest-Neighbor Queries

Example

Find the 5 closest ambulances with respect to the accident place in a time interval of 2 minutes before and after the accident, knowing the directions and velocities of ambulances and the street map

Example

Find the 10 persons that was closest to the rain shower that passed down town Aalborg between 11:12 and 11:17 yesterday.

Additional information needed

- Road network
- Speed and direction of the ambulances

Outline

Evaluation of Four Index Structures

specification / STAM	MR-tree	RT-tree	3D R-tree	HR-tree
Data type support	region	region	region	region
Time support	transaction	transaction	valid	transaction
Data set mobility	full-dynamic	full-dynamic	growing	full-dynamic
Time-stamp update	chronological	chronological	static	chronological
Object approximation	yes	yes	no	yes
Obsolete entries	no	no	no	no
Query processing	timeslice	no	timeslice	timeslice

Outline

1. Motivation
2. Specification
3. Query Support
4. Classification of STAMs
5. Evaluation

Good Points

- Provides an overview of a new field (at the time of publication)
- Overall good criteria
- Compares four specific index structures
- Paper is very readable

Could be Improved

- Missing more details on timeslicing and obsolete entries
- Missing 2D, 3D, nD discussion
- Not clear (to me) why loading of data is interesting
- Only R-trees in evaluation no quad-trees