A Foundation for Representing and Querying Moving Objects

Ralf Hartmut Güting, Michael Böhlen, Martin Erwig, Christian Jensen, Nikos Lorentzos, Markus Schneider and Michalis Vazirgiannis

Speaker: Maurizio Atzori, University of Pisa

Spatio-temporal databases

- DBMS data model and query language capable of handling time dependent geometries
  - Moving objects
- An abstract data type extension to a “standard” DBMS type system

Moving objects

- Two fundamental abstractions:
  - Moving point
  - Moving region

Intro

- Objects (at any point of time)
  - Position
  - Extent
- Ex: counties, taxis, air planes, lakes...
- So-called spatio-temporal objects

Example: forest-harvesting machines

- GPS device
- During the process:
  - Cuts down trees, strips off the branches
  - Measure the amount and properties of the harvested wood and send information
    - Handling orders for wood
    - Scheduling the pickup of already harvested wood as well as further harvesting

Types of spatio-temporal objects

- Discretely moving objects
  - we keep track of objects' changing positions and extents
  - Ex: Land parcels
- Continuously moving objects
  - Discrete changing tracks are not enough
  - A new framework is needed

Basic types used

- The framework takes at its outset a set of basic types:
  - Integer
  - Boolean
  - Spatial data types
    - Point
    - Region
  - Temporal type instant

Framework's properties

- Closure
- Simplicity
- Expressiveness

Language embedding

- The language is not important
- A relational model and an SQL-like language will be used for examples

Spatio-temporal data type: signature describing the type system

Type constructor	Signature	
int, real, string, bool		→BASE
point, points, line, region		→SPATIAL
instant		→TIME
moving, intime	BASE U SPATIAL	→TEMPORAL
range	BASE U TIME	→RANGE

Signature describing the type system: notes

- Constant types
- Points: a finite set of point
- Regions can contain holes
- moving(point)
- moving(region)

Classes of operations on Nontemporal types

- Predicates
  - Isempty, = , != , intersects, inside, <,> <=, >=, before, touches, attached, overlaps, on_border, in_interior
- Set operations
  - Intersection, union, minus, crossing, touch_points, common_border
- Aggregation
  - Min, max, avg, center, single

Classes of operations on Nontemporal types (2)

- Numeric
  - No_components, size, perimeter, duration, length, area
- Distance and Direction
  - Distance, direction
- Base Type Specific
  - And, or, not

Classes of operations on Temporal types

- Projection to Domain/Range
  - Deftime, rangevalues, locations, trajectory, routes, traversed, inst, val
- Interaction with Domain/Range
  - Atinstant, atperiods, initial, final, present, at, atmin, atmax, passes
- When
  - When
- Rate of Change
  - Derivative, speed, turn, velocity

An example

country(name: *string*, area: *region*)

- "Determine the region of Europe from the regions of its countries"

```
LET sum = AGGREGATE(union,
   TheEmptyRegion);
LET Europe = sum(area) FROM country
```

Another one

- “Find the point where highway A1 crosses the river Rhine”

```
LET RhineA1 = ELEMENT(
    SELECT single(crossing(R.route,H.route))
    FROM river R, highway H
    WHERE R.name = “Rhine” and H.name =
    “A1” and R.route intersects H.route)
```

Distance and Direction

city(name: string, pop: int, center: point)

- “Find all cities north of and within 200 kms of Munich”

```
LET Munich = ELEMENT(SELECT center FROM city WHERE
    name =“Munich”);
SELECT name FROM city
WHERE distance(center, Munich) <200 and
Direction(Munich,center) >= 45 and
Direction(Munich,center) <= 135
```

Operations on Temporal types

- “How long is the part of the route of flight LH 257 that lies within France”

```
LET route257 = ELEMENT(
    SELECT route FROM flight WHERE airline
= “LH” and no= 257);
length(intersection(France,
trajectory(route257)))
```

Operations on Temporal types (2)

- “What are the departure and arrival times of flight LH 257?”

  min(deftime(route257));max(deftime(route257));

Time and distance

Closest: mpoint x point → intime(point)

- "At what time and distance does flight 257 pass the Eiffel tower?"

```
LET EiffelTower = ELEMENT(SELECT pos
  FROM site WHERE name="Eiffel Tower");
LET pass = closest(route257, EiffelTower);
Inst(pass); distance(EiffelTower,val(pass));
```

When

- “Restrict a moving region mr to the times when its area was greater than 1000”

  mr when [FUN (r:region) area(r) > 1000]