ULB UNIVERSITÉ LIBRE DE BRUXELLES

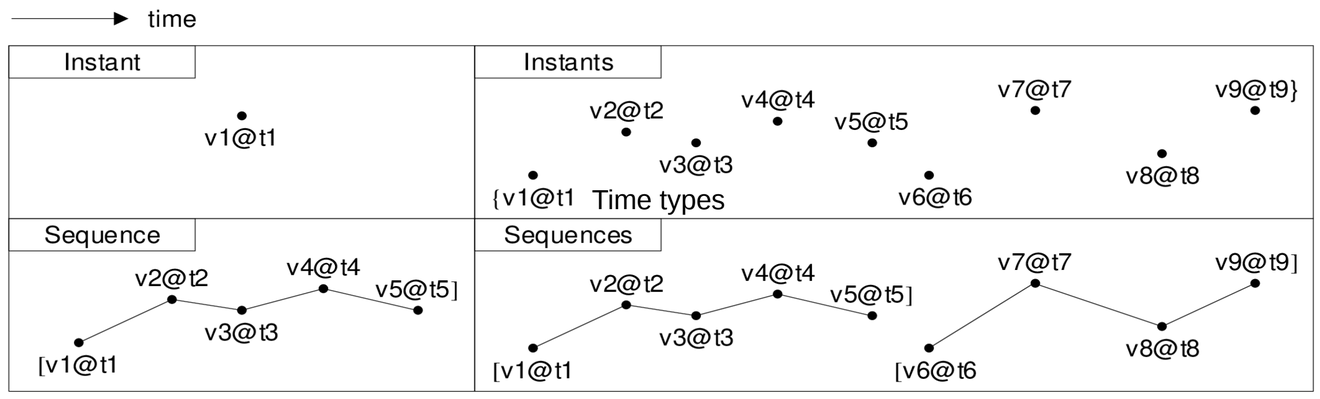
MobilityDB: A Mainstream Moving Object Database System

Esteban Zimányi, Mahmoud Sakr, Arthur Lesuisse, Mohamed Bakli
CoDE. Université Libre de Bruxelles, Belgium

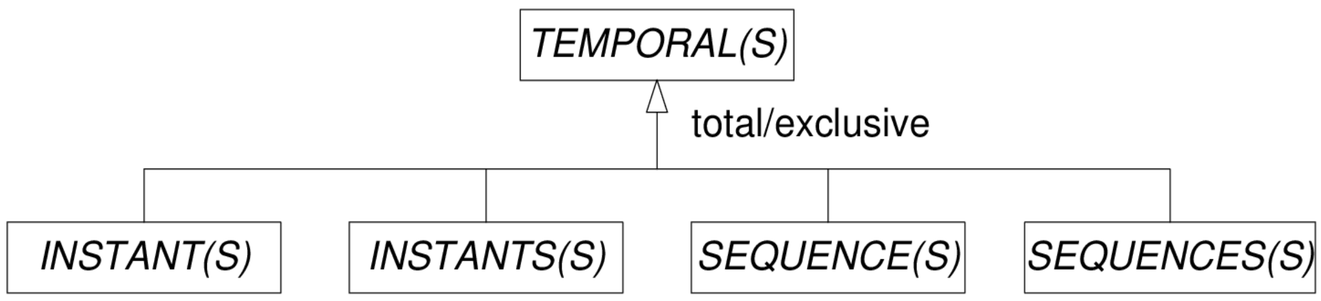
MobilityDB

It is an extensive MOD implementation on top of PostgreSQL and PostGIS with multiple novel aspects. MobilityDB features:

- Spatiotemporal types for moving geometry and geography points.
- Temporal integers, reals, Booleans, and strings.
- A rich set of operations on these types.
- The types are supported with spatiotemporal index access methods by extending GiST (Generalized Search Tree) and SP-GiST (Space Partitioning GiST).
- The query interface is SQL.
- MobilityDB thus extends the PostgreSQL optimizer with statistics collectors and selectivity estimation functions.
- It is available as open source.
- It integrates well with the PostgreSQL eco-system (e.g., Citus, PipelineDB).
- MobilityDB aims at implementing the OGC standard of Moving Features.



Time types




```
CREATE TABLE User (
    id int, checkIns tgeompoint(instants, 4326),
    numFriends tint(sequence),
    Trajectories tgeompoint(sequences, 4326) );
```

Q. 4 Which vehicles have passed the points from Points?

```
1  SELECT DISTINCT P.PointId, P.geom, T.CarId
2  FROM Trips T, Points P
3  WHERE st_intersects(trajectory(T.Trip), P.geom)
4  ORDER BY P.PointId, T.CarId
```

Q.10 List the minimum temporal distance between pairs of cars that ever come closer than 100 meters to one another.

```
1  SELECT T1.CarId AS Car1Id, T2.CarId AS Car2Id,
2      tmin(distance(T1.Trip, T2.Trip )) AS MinDistance
3  FROM Trips T1, Trips100 T2
4  WHERE T1.CarId < T2.CarId
5    AND tdwithin(T1.Trip, T2.Trip, 100.0) &= true
6    AND T1.Trip && expandSpatial(T2.Trip, 100)
7  GROUP BY T1.CarId, T2.CarId
```

```
1  GroupAggregate
2    Group Key: t1.carid, t2.carid"
3    -> Sort
4      Sort Key: t1.carid, t2.carid"
5      Sort Method: quicksort
6      -> Nested Loop
7        -> Seq Scan on trips100 t2
8        -> Index Scan using trips_spgist_idx on trips t1
9          Index Cond: (trip && expandspatial(t2.trip, 100))
10         Filter: ((carid < t2.carid) AND
11             (tdwithin(trip, t2.trip, 100) &= true))
```

Q. 8 What are the overall traveled distances of the vehicles with license plate numbers from QueryLicences1 during the periods from QueryPeriods1?

```
1  SELECT L.Licence, P.PeriodId, P.Period,
2      SUM(length(atPeriod(T.Trip, P.Period))) AS Dist
3  FROM Trips T, Licences1 L, Periods1 P
4  WHERE T.CarId = L.CarId AND T.Trip && P.Period
5  GROUP BY L.Licence, P.PeriodId, P.Period
6  ORDER BY L.Licence, P.PeriodId
```

Q. 17 Which points from Points have been visited by a maximum number of different vehicles?

```
1  WITH PointCount AS (
2      SELECT P.PointId, COUNT(DISTINCT T.CarId) AS Hits
3      FROM Trips T, Points P
4      WHERE st_intersects( trajectory( T.Trip ), P.geom )
5      GROUP BY P.PointId )
6  SELECT PointId, Hits
7  FROM PointCount AS P
8  WHERE P.Hits = ( SELECT MAX(Hits) FROM Point Count )
```

MobilityDB Cloud		MobilityDB Distributed	MobilityDB Python	MobilityDB JDBC
MobilityDB MapMatch		MobilityDB ETL	MobilityDB Stream	MobilityDB View
Citus	MobilityDB Network	MobilityDB	Psycopg 2.8	PostgreSQL JDBC 42.2.6
PostgreSQL 11/ PostGIS 2.5		Python 3.7		Java 11
Ubuntu 18.04.2 LTS				

Benchmark

- 17 BerlinMOD/R queries V.S. SECONDO, 4 data sets.
- MobilityDB came faster in 43 out of 68 queries.
- Total runtime:
  - MobilityDB 1700 seconds,
  - MobilityDB partitioned 1590 seconds,
  - and SECONDO 11900 seconds.

ECOLE POLYTECHNIQUE DE BRUXELLES