Moving objects database models

A COMPARISON

ELIAS AL-TAI, SIMON ORVARSSON
Moving objects database models

A comparison

Elias Al-Tai, Simon Orvarsson

Degree Project in Computer Science, DD143X
Mentor: Michael Minock
Examinator: Örjan Ekeberg

CSC, KTH, 2015-05-08
Abstract

Spatiotemporal databases is a domain that in recent years has gained a lot of attention. Spatiotemporal databases combines two types of databases: spatial databases and temporal databases. Spatial databases are designed for storing and processing spatial information efficiently. Temporal databases have a built-in time concept incorporated to handle queries related to temporal aspects. A spatiotemporal database designed for managing geometric objects that move and change through time is called a moving object database. This report will focus on moving object databases and the different data models that exist to enable them. The moving objects in this study will be represented by flies, moving around randomly in a simulation.

This study evaluates two different data models for storing spatiotemporal data: the Snapshot Model and Sliced Representation. The Snapshot Model is very simplistic. An entry in the database is a snapshot of an object at a specific time. Sliced Representation is a bit more complicated. Temporal development of a value is decomposed into fragments called slices. Within the slice the temporal development of a value can be described by a mathematical function. To evaluate the Snapshot Model, PostgreSQL with an extension called PostGIS which supports spatial data will be used. The temporal aspect will consist of timestamps which will be added to every entry in the database. To evaluate the slicing model, an application called Secondo, developed at the Hagen University in Germany, will be used. There is a set of conditions to determine which method is the most appropriate for the given problem: the creation of the flies and their movement should not exceed 10 seconds. Retrieval of flies that occur in a specified space during a specified moment should not exceed 10 seconds.

The conclusion of this report indicates that the favored data model for the objective is Sliced Representation. The application which uses Sliced Representation, Secondo, could manage 178 moving objects while satisfying the conditions. When the amount of moving objects was increased to a breaking point the application became unstable and started crashing. The breaking point lies between 200 and 500 flies. The application which uses the Snapshot Model, PostGIS will still function at this point but up until this point it performs worse than Secondo. With the conditions in mind, the implementation of the Snapshot Model can handle a maximum of 92 flies and the implementation of the slicing model can handle a maximum of 178 flies correspondingly. Thus, the slicing model is best suited for the objective.
Sammanfattning


Table of contents

1 Introduction ......................................................................................................................... 1
2 Objective ............................................................................................................................... 2
3 Background ............................................................................................................................ 3
  3.1 Databases .......................................................................................................................... 3
  3.2 Geographic Information System (GIS) .............................................................................. 3
  3.3 Spatial Databases .............................................................................................................. 4
  3.4 Temporal Databases ......................................................................................................... 5
  3.5 Spatiotemporal Databases ............................................................................................... 5
  3.6 Spatiotemporal Database Models .................................................................................... 6
    3.6.1 The Snapshot Model ................................................................................................. 6
    3.6.2 The Space-Time Composite Data Model .................................................................. 7
    3.6.3 Simple Time Stamping ............................................................................................. 7
    3.6.4 Sliced Representation ............................................................................................... 7
    3.6.5 Event-Oriented model ............................................................................................... 8
    3.6.6 Three domain model ............................................................................................... 8
    3.6.7 Spatiotemporal Entity-Relationship (STER) Model ................................................... 8
  3.7 PostgreSQL ....................................................................................................................... 8
  3.8 PostGIS ............................................................................................................................ 8
  3.9 Well Known Text (WKT) ................................................................................................. 8
  3.10 QGIS .............................................................................................................................. 8
  3.11 Secondo .......................................................................................................................... 9
4 Hypothesis ............................................................................................................................ 10
5 Method .................................................................................................................................. 11
  5.1 PostGIS with timestamps ............................................................................................... 11
  5.2 Secondo .......................................................................................................................... 15
6 Results ................................................................................................................................... 20
7 Discussion ............................................................................................................................. 21
  7.1 Conclusion ....................................................................................................................... 21
8 References ............................................................................................................................. 22
1 Introduction

The general subject of this report is spatiotemporal databases, i.e databases that manages information about time and space. According to Revesz (1), spatiotemporal databases have generated much interest in recent years. The reason is that many real world applications such as location based services, geographic information systems etc. need to store data that have attributes related to space and time. Traditional database management systems are geared towards providing efficient support for simple objects which have discrete values attached to them. Managing attributes related to space and time using existing Relational Database Management Systems (RDBMS) is complex and inefficient (2). Database Management Systems (DBMS) that manage spatiotemporal objects where spatial and time attributes can continuously change require new techniques.

There are several different purposes and objectives for spatiotemporal databases, for example to create databases that store wireless communication networks which may exist only for a short timespan within a geographic region, databases that store historical tracking of plate tectonic activity and databases that tracks moving objects and their position at a given time (1). Specifically this report will focus on moving objects databases.

The moving objects database domain started to draw attention during the second half of the 90’s (2). The general idea of moving objects databases is that it should be able to represent and store moving entities in a database and handle queries about them. An example of a query that the database should be able to handle is if the user request which objects is contained in a certain spatial volume at a specific time. The subject has generated a lot interest because of its applicability to modern day problems, for example a database that contains moving birds in a geographic region, or a taxi service that tracks the location of several cars.

Spatial and temporal aspects of data have been studied separately, the synergy between spatial and temporal aspect have not been extensively studied. Only a few complete spatiotemporal DBMS systems exist to support spatiotemporal data types and operations (3).
2 Objective

We will review literature on moving objects databases and compare different approaches in terms of how many objects can be tracked and caught in a real time simulation. The report will use a concrete problem called “Fly catcher”. If flies (the moving objects) would stay put, it would be easy to catch them. If you wanted to determine which flies you would catch in a n-dimensional rectangle, you could just index fly positions using an R-Tree. But flies move about randomly. How could you determine how many flies are inside a rectangular area at a specific time? Our objective is to explore two methods to solve this problem, and then compare them to each other. Above all we want to see how many flies can be handled (i.e created and found) at the same time with each method.
3 Background

3.1 Databases
A database is a collection of data which is organized in such a way, that it is easy to access, add and update it. To use a database, a database management system (DBMS) is required. Databases is very frequently used on the internet. A very common use is in user login systems, where you store the username and pair it with a password. When the user enters it in the database, you simply check if that pair exist. The most common database model is the relational one (4). A problem with relational databases is that it lacks a good way to represent continuous time. In order to represent these attributes extensions are required to existing models.

A moving objects database is an instance of a spatiotemporal database, which evolved from the spatial database: geometry in databases, and the temporal database: data over time in databases. When combining the two, it results in a spatiotemporal database. The properties of the spatiotemporal database is a combination of the properties in a spatial database and a temporal database. This means that it supports both geometries and data over time (1).

3.2 Geographic Information System (GIS)
A GIS is a system that is used for handling geographic data. The data can be represented by three different types of objects: points, lines and polygons/regions as seen below.

![Figure 1 Three examples of spatial objects](image)

Each type of object has a set of attributes that describes it. For example, a point usually has a pair of coordinates, latitude and longitude. A line has two pairs of coordinates, latitude and longitude for the starting point of the line, and latitude and longitude for the end point of the line. A polygon is built up by several points, each of these has a pair of coordinates, latitude and longitude. This is the new way describing the objects in a GIS. When GIS was first introduced, a grid system was used. Each cell of the grid could describe an object, or a part of an object. This meant that the accuracy of the grid depends on the size of the cells. This is not very efficient (5).
There is a great correlation between GIS and spatial databases. There is a file format called shapefile (.shp), that can be used to import a GIS to most of the spatial database systems on the market, for example PostGIS. The shapefile contains all the data inside the GIS, every object and their coordinates.

3.3 Spatial Databases

A spatial database is a type of database that extends the regular type of database to also represent and query geometrical objects. The implementation of a DBMS needs to be extended by a new syntax regarding spatial queries, corresponding data structures for geometric shapes (spatial data types (SDTs)), a new set of algorithms to calculate geometric computations (ROSE algebra), e.g. the distance between two objects, and indexing techniques for multidimensional space.

ROSE algebra, which has closure properties, offers data types called points, lines and regions, similar to GIS. Different types of objects can represent different things. For example on a map a point can be a city, a coffee shop or an intersection. A line can be a street or a power line. A region may have holes or consist of several disjoint parts, a region can represent actual region in a country, a part of a city or a forest. These data types can be used in any kind of DBMS data model; relational, object-oriented or something else.

The data types have fundamental relationships like intersects, lies within, lies outside, touches and properties like area, surface area, volume etc. ROSE algebra is essentially a method to see how objects correlate to each other.
3.4 Temporal Databases
A temporal database is a type of database that extends the regular type of database to also represent time. In the regular type of database time can be expressed with timestamps. Normally, a change will be made later in some update to the database after which the previous state is lost. For example, if you want to add a task to a database, or a meeting, you can add a start and an end field. For many applications it is not sufficient to maintain just the current state, the database needs to keep track of history. The objective of temporal database research has been to integrate temporal concepts deeply into the query language and DBMS data model to achieve efficient execution.

3.5 Spatiotemporal Databases
Spatiotemporal databases aim to provide database support for applications which show both spatial- and temporal characteristics. It provides extensions to existing models of databases to include temporal and spatial aspects.
3.6 Spatiotemporal Database Models

Spatiotemporal data models define object data types, relationships, and rules to maintain database integrity for spatiotemporal entities. It should have efficient support for storing spatiotemporal objects and handle spatiotemporal queries. There are several different models used to manage spatiotemporal data, some models will be mentioned in this report.

3.6.1 The Snapshot Model

The Snapshot Model uses time-stamped layers on top of a spatial data model to represent temporal aspects. Time is an attribute of the location. This is the simplest model, however it has some disadvantages:

- Since it only describes the state of an object at a time \( t \), this model is not optimal for working with changes through time. This is because each snapshot must be compared to each other thoroughly.
- A lot of unchanged data will be duplicated. This is because a system of determining whether two snapshots and their attributes are identical are non-existent.
- It is not continuous. Meaning that if you want to find an entry between two timestamps, it will not work, since it doesn't exist.

The snapshot approach generally result in a large amount of data duplication with unchanged properties in space and time. The major drawback of the model is data redundancy and the risk of data inconsistency.
3.6.2 The Space-Time Composite Data Model
This principle was first presented by Langran in 1988 (11). The idea is that every line in both time and space is projected to only the spatial plane. Then an intersection is done between them both, which creates a polygon mesh. Every polygon that this mesh consists of, has an attribute that it is associated with. Changes to attributes are recorded at discrete instances of time. The Space-Time Composite (STC) describes the change of a spatial object through a period of time. STC fails to capture temporality among attributes across space (movement) hence it is not a suitable data model for spatiotemporal databases that handle moving objects. Updating databases of STC requires reconstruction of STC units.

3.6.3 Simple Time Stamping
Each object has a pair of timestamps representing for example creation and deletion time of the object or a start time and a finish time. A downside of the model is that an object can’t have a more complex temporal history than a start and a finish attribute (13).

3.6.4 Sliced Representation
When using Sliced Representation, temporal development of a value is decomposed into fragments called slices. Within the slice the temporal development of a value can be described by a “simple” mathematical function (2).
3.6.5 Event-Oriented model
A log of events and changes made to the objects are logged in a transaction log. Historical information is obtained by combining data from the transaction logs (13).

3.6.6 Three domain model
Space, time and semantics are kept separately and link is provided between them to describe spatiotemporal objects (13).

3.6.7 Spatiotemporal Entity-Relationship (STER) Model
The STER-model is a variation of the classic ER-model that was presented by in a paper in 1976 by Peter Chen (14). The ER-model is a data model which, in a very simplistic way, can represent entities, their relations and their attributes. The STER-model extends the ER-model to also handle spatiotemporal data. This is done by adding attributes that represents time and space, such as a timestamp and a geometric type (13).

3.7 PostgreSQL
PostgreSQL is a DMBS that is based on open source code. It is a very popular implementation of a SQL database, and it has a lot of extensions, PostGIS being one of them (4).

3.8 PostGIS
PostGIS is a popular extension to the PostgreSQL database. It enables support for spatial objects. It adds a lot of operators to calculate e.g. area, length, perimeter, and union of objects. To describe geometries in PostGIS, the markup language Well-known text (WKT) is used.

3.9 Well Known Text (WKT)
WKT is a markup language to represent geometric objects in maps or other spatial objects. It is a standard that is widely used in for example GIS and SQL. WKT supports up to 18 different spatial objects, but the ones used in this study is Point and Polygon. Point(x y) describes a point and Polygon(x1 y1 x2, y2, ..., xn yn, x1 y1) describes a polygon with n vertices, where x and y is latitude and longitude (15).

3.10 QGIS
QGIS is a GIS application that supports most of the GIS formats. It is one of the most popular GIS applications on the market and has great functionality. One of QGIS most important functions for us, is that it interacts very well with PostGIS. You can import a spatial database from PostGIS and get a
visual representation of the database. QGIS started out at Quantum GIS in 2002, but changed its name to only QGIS in 2013 (16).

3.11 Secondo
Secondo is developed at the Hagen University in Germany. It is an implementation of a spatiotemporal database. It currently supports standard algebra including integer- bool- and string operations. In addition to this, it also supports relational algebra, spatial algebra and temporal algebra. This includes operations to see e.g. if a point is inside a rectangle. Secondo can store complete histories of movement, and this can also be queried.

For example, Secondo supports the query: “has this fly ever crossed the point (x, y)”, where x, y is coordinates. It uses spatiotemporal data types as moving regions, moving lines and moving points. A fly and its movement can be represented by one object: a moving point (or mpoint for short). A swarm of flies can be represented by one object also: a moving region (or mregion for short). An mpoint or an mregion is a function from time into point- or region values. A visualization of the Sliced Representation of a spatiotemporal object can be seen below.

![Figure 8 Visualization a line and a region in Secondo](image)

In Secondo, the Sliced Representation model that is explained earlier is used. It represents a time dependent value as a sequence of slices, or temporal units. In each slice, the value is represented by a temporal function. This can be a constant function, or a quadratic polynomial function, depending by the type of object (17).
4 Hypothesis

The hypothesis is that the Sliced Representation model, used by Secondo, should be faster and more memory effective. This is because it is a system specifically built for this kind of problem. PostGIS with a temporal aspect added (i.e. the Snapshot Model) should take up a lot of unnecessary memory, and also be slower while generating the flies.
5 Method

Two different spatiotemporal models will be compared. The first model that will be evaluated is the Snapshot Model with the help of PostGIS with timestamps. The second model that will be evaluated is Sliced Representation with the help of Secondo. First, the models will be tested to evaluate if they can fulfill the objective, if it can handle queries on a database with randomly moving flies. This will be done by using a small sample size. If successful, bigger volumes will be tested, until a breaking point is reached. One condition set is that the creation of the database should not exceed ten seconds. Querying flies within a rectangle at a certain time should also take a maximum of ten seconds. The tests are performed on the same computer, which makes it a fair comparison.

5.1 PostGIS with timestamps

PostGIS is used and the temporal aspect is implemented by adding timestamps. QGIS is used to visualize it. The initial test consists of 25 flies. To make it more realistic, an existing map from Geofabrik.de is downloaded. Geofabrik.de is a website that contains large volumes of GIS-data. This comes in the form of a shapefile, which is easily converted to a spatial database, which the flies will be inserted into. The flies are represented by points. A map of Monaco is chosen because of its small size.

![Figure 9 Monaco represented with a map and with GIS data](image)

The map is represented as a table in a spatial database with nine columns, one for every type of object. There is a column for roads, a column for railways and so on. To solve the problem, first the flies and their random movement orbit must be generated. This is done by adding a table named flies to the database, and adding four columns to it: ID, geom, timestamp and posnumber. The ID is the ID of the fly. Geom is the geometry of the fly, the coordinates and the type of geometry (e.g. point, polygon, line) is stored here. Timestamp is the time when the fly is located in this specific location, and posnumber is the number of the recorded position. The id, timestamp and postnumber in the figure below are self-explanatory. The geometry column is the hexadecimal representation of the geometric object and its coordinates.
To actually generate the flies and make them move randomly, a set of flies is created in the Principality of Monaco, and assigned random coordinates inside the country. Then random orbits based on the previous coordinates is generated. A loop is required for this. This is done by assigning a number of flies random coordinates, then subtracting the x and y coordinates for every fly. The timestamp is updated: for every iteration 500 milliseconds is added to the timestamp. To add a point to a spatial database in PostGIS, the constructor ST_MakePoint is used.

\[
\text{geometry ST\_MakePoint}(\text{double precision } x, \text{ double precision } y);
\]

The arguments passed to the constructor is the latitude and longitude of the point. The latitude and longitude will be a point in central Monaco. To place the flies randomly, a randomly generated number between 0 and 0.01 is subtracted. The query to create the initial positions of 25 flies can be seen below. The Truncate Table query deletes all entries from a table, this is done prior to the insertion.

```sql
DO
$do$
BEGIN
TRUNCATE TABLE flies;
FOR i in 1..25 LOOP
    INSERT INTO flies(id, geom, timestamp, posnumber)
    VALUES (i, ST_MakePoint(7.42461579999997 - random() :: decimal /100,
        43.738417600000001 - random() :: decimal /100), '2015-03-10', 1);
END LOOP;
$do$
```

The result is a large table filled with the locations and timestamps for every fly, the graphic representation of this can be seen in Figure 12. Every color represents a unique fly and all of its locations under a time period of 50 seconds, more specifically, from 00:00:00 to 00:00:49 at the 10th of March 2015.
To find all the flies which are inside a rectangle, defined by four coordinates, at a specific time, a relevant SQL query is executed. The query in words is “find all the flies where its position is inside the specified rectangle, and its timestamp is ‘2015-03-10 00:00:43’, in SQL, see Figure 13. To achieve this, the method ST_Within is used.

```
boolean ST_Within(geometry A, geometry B);
```

ST_Within returns true if geometry A is inside geometry B, and false otherwise. ST_GeomFromText is a constructor which creates a geometry from WKT, in this case a polygon, the rectangle as seen in Figure 14.

```
SELECT geom, id FROM flies
WHERE ST_Within(geom, ST_GeomFromText('POLYGON((7.422144 43.736704,
7.429569 43.736704, 7.429569 43.732784,
7.422144 43.732784,7.422144 43.736704))'))
AND timestamp = '2015-03-10 00:00:43'
```

Figure 13 SQL query to find all flies inside a rectangle

A tiny part of Monaco is chosen, more exactly the harbor as seen below. From Figure 12, it is concluded that there are three flies that moves around in the harbor, thus three results should be retrieved at a given time. The result will be a list of all the rows that satisfies the above specified requirements. By using QGIS, a graphic representation of the result can be acquired. The result can be seen in Figure 15.
It is concluded from the figure above that three flies with the timestamp 00:00:43 is found, this method clearly works. But how many flies can it handle (with a reasonable time)? This part will not have any graphic representation, this is because the best performance possible is desired.
<table>
<thead>
<tr>
<th>Number of flies</th>
<th>Time to create (seconds)</th>
<th>Time to find (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.221</td>
<td>0.011</td>
</tr>
<tr>
<td>20</td>
<td>0.602</td>
<td>0.011</td>
</tr>
<tr>
<td>50</td>
<td>2.959</td>
<td>0.011</td>
</tr>
<tr>
<td>100</td>
<td>11.047</td>
<td>0.011</td>
</tr>
<tr>
<td>200</td>
<td>43.873</td>
<td>0.011</td>
</tr>
<tr>
<td>500</td>
<td>295.920</td>
<td>0.011</td>
</tr>
</tbody>
</table>

Table 1 Table of execution times of each operation

It is concluded from the table that the query to create flies and generate their orbit is not linear. The reason for this is that inside the query, the previous position of the fly is selected. The previous position is used to calculate the new position. A select operation in SQL is not performed in constant time, it is performed in \( \log(n) \) time where \( n \) is the size of the table (18). This also means that in the beginning of the loop, this operation is done twice for every fly and for every position. For example, when the number of flies is 500, there is \( 2 \times 500 \times 100 = 100,000 \) select operations made. As seen in the table retrieving flies is accomplished in constant time.

The condition was that the creation and moving of the flies should not exceed 10 seconds for it to be feasible. With more testing, it is concluded that at most 92 flies at a time can be managed with this method while the condition is fulfilled.

5.2 Secondo

Secondo uses Sliced Representation to store moving objects. All spatiotemporal values \((<\text{moving}_\text{type}>)<\text{unit1}>...<\text{unitn}>\) where \(<\text{unit1}>\) is represented as:

\((<\text{interval}><\text{map}_\text{type}_\text{value}>))\)

An interval value is represented as:

\((<\text{start}><\text{end}><\text{leftclosed}><\text{rightclosed}>))\) where \(<\text{start}>\) and \(<\text{end}>\) are defined instances of type instant representing the start and end instant of the time interval. \(<\text{leftclosed}>\) and \(<\text{rightclosed}>\) are defined Boolean values defining if the interval is open \((false)\) or closed \((true)\) at the start or end time instant. The representation of a \(<\text{map}_\text{type}_\text{value}>\) depends on which type of object is desired; point, line or region.

A moving point \((mpoint)\), an existing abstract datatype in secondo is chosen to represent a single fly. The \((mpoint)\) datatype is implemented so that it is a set of \(<\text{map}_\text{point}_\text{value}>\). \(mpoint = \{(<\text{map}_\text{point}_\text{value1}>,...,<\text{map}_\text{point}_\text{value}_n>\}\) with \(n > 0\).

The representation of a \(<\text{map}_\text{point}_\text{value}>\) is: \((<x_1,y_1,x_2,y_2>)\) where \(x_i\) and \(y_i\) are numeric values. In a given Unit \((<\text{start}><\text{end}><\text{leftclosed}><\text{rightclosed}>)(<x_1,y_1,x_2,y_2>))\) the point moves in the given interval form \((x_1,y_1)\) to \((x_2,y_2)\). The position of a moving point can be computed for a single unit using the following function:

\[ pos : unit(point) \times instant \rightarrow point \]

\[ pos(((s e l r)(x_1,y_1,x_2,y_2),l) := \begin{cases} undefined & \text{if } l \text{ not } \in (s e l r) \\ (x_1,y_1) & \text{if } s = e \text{ and } l \in (s e l r) \\ (x_1,y_1) & \text{otherwise} \end{cases} \]

where:
\[ x_1 = x_1 + \frac{1 - s}{e - s}(x_2 - x_1) \]
\[ y_1 = y_1 + \frac{1 - s}{e - s}(y_2 - y_1) \]

The flies initial positions are randomized, which can be seen in Figure 16. Then a new position based on the initial position is randomized with credible x and y coordinates that a fly could move in 500 milliseconds. Every fly has 100 positions and like the previous solution with PostGIS the first try includes 25 flies. All positions are randomized in an external program.

```javascript
let fly = [const mpoint value {
    {
        ("2003-11-20-06:06" "2003-11-20-06:06:23.156" TRUE FALSE)
        (16821.0 1252.0 16673.0 1387.0)
    }
}]
```

*Figure 16 The initial positions of the flies*
Figure 17 The final positions of the flies after they have moved according to their orbit. The trajectories of the flies has been drawn as lines.

To find out which flies appear in a specific region at a specific time a non-moving region is created:

let reg = [const region value(((4000.0 12000.0)(20000.0 12000)(20000.0 0.0)(4000.0 0.0)))]
A query is then possible that retrieves all flies that appear inside the region at the specific instant. First an instant object is created:

\[ t = \text{theInstant}(2003,11,20,6,56) \]

The query used:

\[ \text{select [id val(orbit atinstant } t) \text{ as pos] from flies where [orbit passes reg, orbit present } t] \]

<table>
<thead>
<tr>
<th>Number of flies</th>
<th>Time to create (seconds)</th>
<th>Time to find (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.3</td>
<td>0.23</td>
</tr>
<tr>
<td>20</td>
<td>0.755</td>
<td>0.23</td>
</tr>
<tr>
<td>50</td>
<td>1.228</td>
<td>0.23</td>
</tr>
<tr>
<td>100</td>
<td>5.27</td>
<td>0.23</td>
</tr>
<tr>
<td>200</td>
<td>11.534</td>
<td>0.23</td>
</tr>
<tr>
<td>500</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

*Table 2 Table of execution times of each operation*
When the amount of moving objects was increased to 500, the application became unstable and started crashing. The condition was that the creation and moving of the flies should not take more than 10 seconds for it to be feasible. The maximum numbers of moving objects that satisfy the condition with Sliced Representation and Secondo is 178 flies.
6 Results

It is possible to complete the objective using both models. However there are performance differences when comparing the different methods. When using the Timestamp Model with the application PostGIS, the maximum amount of moving objects handled while satisfying the conditions was 92 flies. Sliced Representation using the application Secondo could at most handle 178 moving objects while satisfying the conditions. Ignoring the conditions, as the number of flies was increased, PostGIS perform better than Secondo since the latter was unable to handle a large number of flies without crashing. The results will be represented with a chart.

Figure 19 Diagram of the execution times of creating 25 flies with each method
7 Discussion

The hypothesis was that the Sliced Representation model, used by Secondo, would be faster and more memory effective than the Snapshot Model, used by PostGIS. This is true, to some extent. When using lower volumes of flies, below 200, Secondo turned out to be a lot faster than PostGIS. When 500 flies were created, Secondo crashed. PostGIS on the other hand, did not, although it was very slow. It took almost five minutes to generate 500 flies with PostGIS.

A great disadvantage with the Snapshot Model, is that only queries using the exact timestamp can be easily found. The model does not define what happens between two snapshots, you cannot find a fly with a timestamp that doesn't exist without using a more advanced query. For example, you could try to find the timestamp closest to the timestamp you use in the query (if it does not exist to begin with). Secondo uses the slicing representation model which is a continuous model. In this case that means that queries can fetch data about an object from any timestamps if it exist during that time.

The comparison of the different models are highly dependent on how the implementation and optimization of both PostGIS and Secondo. This means that one model may be more memory efficient in theory but the implementation is not well executed. It is not certain that the methods used to generate the flies in each model was the optimal method. Thus it is not certain that the results are correct. It was very interesting to see that Secondo ran out of memory before PostGIS did while trying to create a very large database of moving points, this was something we did not predict. This could mean that PostGIS is more memory efficient than Secondo.

7.1 Conclusion

The conclusion of this study is that there are effective methods to solve the objective concerning managing a database which includes a large collection of moving objects. The Sliced Representation is the preferred model for this instance of the problem regarding moving objects. The main reasons for this is:

- It is continuous. Every timestamp between the first and the last exists and can be queried. This is not the case with the Snapshot Model.
- It is faster. As derived from Figure 19 Secondo is superior to PostGIS with timestamps, especially when the amount of flies was increased. However after a certain threshold of increasing the number of objects, Secondo becomes unstable while PostGIS functions. This doesn't necessarily indicate the model Secondo is using is worse but could just be that the implementation is not as stable.
8 References

1. Revesz P. Introduction to Databases: From Biological to Spatio-Temporal; 2010.

2. L. Forlizzi RHGENMS. A Data Model and Data Structures for Moving Objects Databases. ACM Transactions on Database Systems. 1999 March; 25(1).


