Real-Time Spatiotemporal Data Indexing Structure

Guillaume Noël, Sylvie Servigne, Robert Laurini
Liris
INSA-Lyon, Bat B. Pascal, 20 av. A. Einstein,
69622 Villeurbanne Cedex FRANCE
{gnoel, servigne}@if.insa-lyon.fr, laurini@insa-lyon.fr

SUMMARY

This document describes the Po-tree, a new indexing structure for spatiotemporal databases with soft real time constraints. This is crucial for risk management, particularly for the monitoring of volcanic risk. For example, the Popocatepetl, a Mexican volcano, is closely monitored through an network of spatially referenced sensors. They all send back their measurements to a central spatiotemporal database. The Po-tree shall therefore be able to index huge quantities of data in a restricted amount of time. It shall also be able to favor the newer data, which are queried more frequently. While most structures tend to favor the temporal aspect over the spatial, or tend to consider time as another spatial dimension, the Po-tree offers an alternative. It does so by combining two different structures, for spatial and temporal dimensions respectively, stressing the spatial aspect, and linking every source of information to a new temporal sub-tree. While mobility is not yet supported, experimentations and comparisons have shown interesting potential in the structure.

KEYWORDS: *Spatiotemporal, Database indexing, Real-time, Natural disaster prevention*

INTRODUCTION

Geographic Information Systems provide solutions to a wide panel of problems, from agronomy to urban planning, and much more. The databases linked to such systems are usually very large and cumbersome. They have to keep tracks of numerous data of different kinds. Solutions exist to face this kind of needs. Yet, a particular aspect remains partially untouched: the specific needs of spatiotemporal indexing with real time constraints. More specifically, we intend to study the indexing of data issued from an array of spatially referenced sensors. While the application case is linked to natural disasters prevention, we shall show that the structure we propose can be extended to cover different needs. We currently propose a new database index to deal with these sensors, the Po-tree.

This paper is divided into three chapters. First of all, we shall define more precisely the problem we intend to address, followed by an introduction to our application case. Next, a state-of-the-art will present the different approaches to data structuring and indexing methods. Finally, we shall introduce our solution, the Po-tree, and some test results to study its usefulness.

APPLICATION CASE

Natural disaster management cover a wide range of topics, from floods to earthquakes. Many of these potential disaster are monitored through the use of networks of sensors. In many cases these sensors are linked to a central database, in some cases collecting data from an entire country. From here on, specialists study the values measured, usually focusing on the evolution of data and on the ratio between

key values. In all of these cases when sensors are used spatio-temporal constraints, can be defined, along with real-time ones.

More specifically, our solution has been designed so as to provide volcanologists with a tool to help them in their study of a Mexican volcano, the Popocatepetl.

The CENAPRED (Cenapred, 2004) is an official mexican organization in charge of the monitoring of the volcano and of the coordinating of the different teams of scientists . A network of sensors has been set up. These different sensors measure heterogeneous data, from seismic activity to gases composition. Furthermore, they are gathered in spatially referenced fixed stations. Every sensor has its own frequency, going as high as 100 Hz. The number and diversity of sensors provide the database with important quantities of data to work with. See figure 1 for a representation of the case.

All of the collected data are later sent to a central database, linked to a datawarehouse. Computation following the reception of the measurements can also lead to newer data to index. Aggregate data, be it the number of times a threshold is reached or the trend of evolution on the last minutes, are usually considered highly valuable to the volcanologists.

These scientists are usually not directly interested in the raw data issued from the sensors. They value more the evolution of the data, with a stress on the most recent data. As such, chemists would be more interested in the change of ratio of two compounds for a gas than by the raw concentrations of the compounds.

As for now, querying the database implies having a preliminary knowledge of the position of every sensor, as they are still referenced through an identifier. Scientists have asked for improvements on this aspect, with the addition of spatial informations. The current system is based on earthworm, a widespread system for volcanic monotoring (usgs, 2004).

Among the most common queries are spatial point / time interval lookups. The volcanologists use such queries on specific locations, pertinent sensors so as to forge a preliminary understanding of the volcano activity and mechanics. Later on, other queries help in adjusting these suppositions.

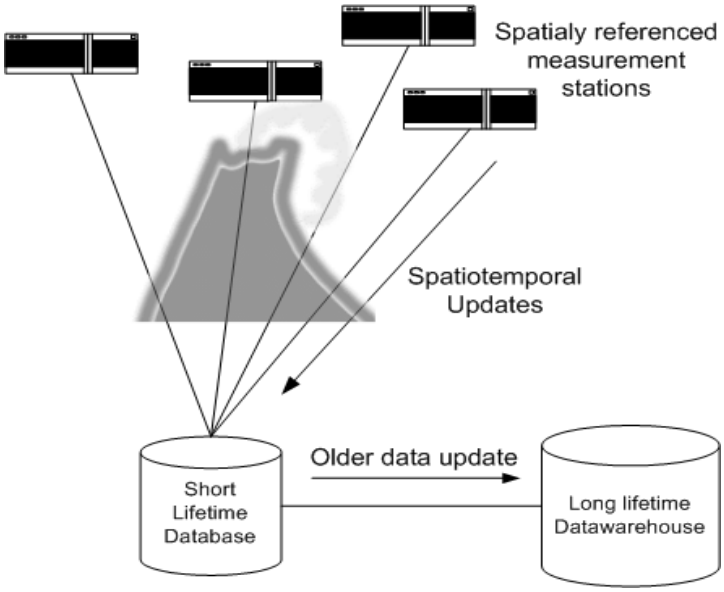

Figure 1: application case

A study of this case makes arise a few particularities. The data are issued from a fixed number of sensors, information sources characterized by their geographical positions. The temporal aspect is given by the timestamps of the actual measurement of data. From this fact we can infer spatio-temporal properties.

A huge number of data, akin to points in a spatio-temporal space, must be indexed by the database. Due to this aspect, we can add time constraints to the different queries the base has to process. As it is reasonable to lose one measure for a sensor at 100 Hz without truly harming the whole system, we can add soft real-time constraints respects to our solution.

Most of the lookups are linked to spatial points and temporal intervals, with a focus placed on the most recent data. Therefore, while keeping tracks of all the measures is needed, the solution has to focus on working with the newer ones.

Moreover, due to the actual use of the database, it is possible to define a notion of reference information sources, that could lead to a hierarchy of stations.

Lastly, the link to a data warehouse implies that the database must take into account the possibility to replicate the oldest data to this second system.

This application case, the indexing of data issued from a network of spatially referenced fixed sensors brings different needs to the indexing techniques used in the database. This index has to take into account spatio-temporal and soft real-time particularities. Furthermore, it also has to put the stress on the newest data as they are considered as more interesting for the volcanologists. While the different domains linked to these specificities have already been studied on separated grounds, comparatively little work has been performed on the integration in a common case .

STATE OF THE ART

A state of the art on the different existing methods is essential to better understand the pros and the cons, the main concepts of the situation.

Real-time

For the real-time approach, the main idea is to answer queries within time constraints. It is possible to separate three kinds of constraints (Lam & Kuo, 2001).

The soft one, used in our case, imply that transactions should be fulfilled within time limits, yet it is understandable that some transaction can not comply with the limits.

The firm constraints , more restrictive, allow some transaction not to be fulfilled within the time limits, yet in this case the whole system can be slightly impaired.

The hard constraints, finally impose that under no circumstances a transaction should miss a deadline. Otherwise, the system could come to a halt..

Priorities are generally used to define which transaction is more important than another. It is not to define which one should occur before another... Different techniques can be used so as to assign priorities: Earliest Deadline First, Rate Monotonic,... (Lam & Kuo, 2001)

Real-time computing is not similar to Fast-computing. Fast-computing does not prevent a low priority transaction to block high priority transactions (priority inversion) because they have already locked the access to some resources, data,...

However, for databases, the current paradigm is to keep the index, and even the whole base in main memory so as to reduce the number of slow disk access. Index Consistency Control (ICC) methods can then be used to make sure no priority inversion occur while accessing the index (Haritsa & Seshadri, 2001).

Spatial approach

The spatial approaches for such cases usually tend to linearize the data so as to use known « fast » structures. Such is the case for kd-trees (Ooi & Tan, 1997) or other methods for spatial objects, or more accurately spatial points.

Kd-trees are related to binary trees. A reference point is taken, along with a reference dimension. Every other point that falls below the reference point for this dimension shall branch to the left, all points with higher values branch to the right. At the next level, a new reference point is taken in each branch and the next dimension is used as a reference.

Another widely accepted approach is to use rectangles, bounding structures to match the position of objects. The bounding rectangles can then be regrouped within bigger rectangles so as to create a balanced tree. The R-tree, and its sibling R*-tree (Ooi & Tan, 1997) are examples of this.

While the R-trees allow to work with complex objects (approximated as rectangles and not points), their higher building and querying time make the use of lighter structures appealing to index points.

One important point to note is that these structures are linked to the oder of insertion of data. Two sets of identical data arranged in different orders shall be referenced in differently shaped trees.

Temporal approach

For the temporal approach, it is important to note that different notions of time can be used for databases (Ooi & Tan, 1997*).

The Transaction Time allows users to perform « rollbacks » so as to find past-values. It does not allow to modify previously entered values, nor to enter future values. One can only append new data issued at the present moment.

The Valid Time represents the time when a fact is considered true. It allows users to modify past data, and to enter future data. However, it does not allow rollbacks.

The Bi Temporal Time is a mix between the two others, allowing rollbacks, post-modifications and future updates.

There mainly are two ways of considering temporality. The latter is to consider that time is monotonous (time goes in one direction) and to use B-trees as index structures. An intersting variation of this is to consider that the data flow constantly, therefore it becomes possible to link the root of the tree more closely the last leaf. This leaf containing the most recent data. Such an idea has been developed in AP-trees (Gunahdi & Segev, 1993).

The other way of considering temporality is to consider time just as a spatial dimension and to use R-trees, with on one dimension the timestamps and on the other dimension the validity duration.

Spatiotemporal approach

Spatiotemporal approaches have to face the variety of possible types of data: points, ranges, intervals,... (Wang, Zhou & Lu, 2000) This leads to a distinction between three families of indexing trees. Those that work with objects in continuous movement, those for discrete changes and finaly those for continuous changes of movements.

Another way of differentiating the families of index has been brought by (Mockbel, Ghanem & Aref, 2003). They have focused on approaches aiming at indexing past positions, present ones and future ones.

Many trees have been developed to answer specific needs. Some trees tend to consider the temporal aspect as yet another spatial dimension, which has led to 3DR-trees (Theodoridis, Vazirgiannis & Sellis, 1996). However, these trees doesn't take into account the monotonicity of time and usually need to have a previous knowledge of the data to index.

Another family of trees make a difference between spatial and temporal dimensions. In HR-tree (Nascimento, 1998), typical of this case, snapshots of spatial R-trees are linked in a time indexed balanced tree. The main problem of this kind of trees is the size of the tree. While nodes that do not change between two snapshots are shared among the R-trees, only minor changes force to duplicate some of the data. Furthermore, they tend not to be optimal for interval queries.

Yet, from these aspects we can take some ideas. First of all, the differentiation of temporal and spatial data can be used to segment the tree. Then, a variation of B+-tree, the AP tree , offers the idea of a direct link between the root of the temporal data and the latest node. All of these ideas have been associated to provide our solution, the Po-tree.

PO-TREE

Our solution, the Po-tree, is based on the differentiation of temporal and spatial data, with a focus given to the latter. This way the notion of information sources is linked to a specific spatial location.

The spatial aspect is indexed through a Kd-tree, while the temporal aspect uses modified B+-trees (see figure 2). As for now, mobility is not yet managed by the structure. However the specificities of both of these trees allow on-line and batches updates: it is possible to update the structure on real-time or using batch files.

The stations being immobile, the current structure does not yet allow mobile sources of information. This way, every spatial location, akin to spatial object (sensor) is directly linked to a specific temporal tree. Requests shall first determine the spatial nodes concerned and later on determine the temporal nodes.

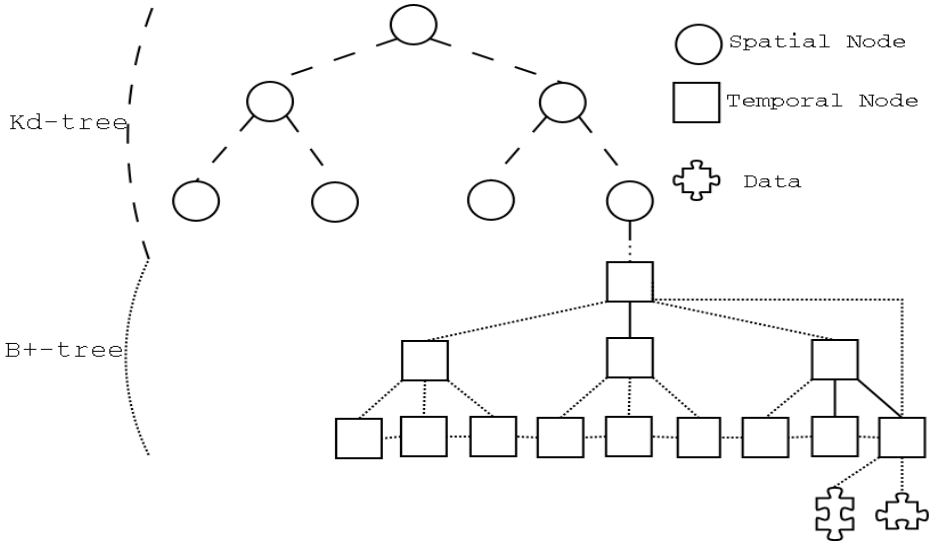

Figure 2. Po-tree structure

Po-tree Structure

Kd-trees are simple structures, as seen in figure 2. Yet they are not perfect structures. One of their main problem being the fact that they rely on the order of inserted data. If the data are entered in different orders, the final trees may have different shapes. Another issue is the fact that they are not perfectly suited for mobility, which is not part of our needs so far.

However, Index Concurrency Control methods, originally designed for B-trees can easily be adapted to cover Kd-trees. The basics of ICC methods usually implie the use of latches, fast locks that restrict the access of certain nodes of the tree, depending of the kind of access required. While the root node can appear as a bottleneck for such use, the deletion of nodes, completely blocking a node for the duration of the transaction, are non-existants. As for insertions, they block the access on the node just over the insertion point but they do not forbide the lookups on other parts of the tree.

Different tests have that these trees fared reasonably well compared to R-trees for small number of datas (Paspalis, 2003). As a matter of fact the R*-trees use more resources than the kd-tree to create or update minimum bounding boxes. Such boxes are heavier to deal with for points than binary trees. Yet future development shall work toward R*-trees as spatial access methods as they also provide solutions for more specific queries (nearest neigbourgs,...).

As each B+-tree is linked to an object, it is possible to develop a secondary structure so as to access directly the temporal data of specific objects, without the need to first determine their position. This can be useful for the notion of hierarchy of information sources.

Furthermore, it has been noticed that the most recent data are considered of higher interest than the older one. It has also been noticed that inserts are generally held at rightmost of the structure, where are found the newest nodes. Therefore, the temporal tree has been modified to add a direct link between the root and the latest node. While maintaining this link requires minimum work for the system, a simple test prevents being forced to traverse the whole tree so as to append or to find the requested data. This direct link is useful to save processing time.

As most, if not all, of the updates take place to the rightmost part of the temporal tree, the fill factor of leaf nodes can be placed higher than usual. Deletes should be somehow rare under normal conditions, and updates that does not concern the newest data should be even rarer, unless the systems experiences lag time due to network problems between the sensors and the database. Therefore the split and merge procedures can be changed so that the nodes can be filled almost at their maximum capacity.

This configuration implies that this tree is more specifically designed for queries on the most recent data. Spatial range / temporal interval requests that does not ends at the present time does not take any advantage of the specificities of the tree.

Tests

Different tests have been conducted between the Po-tree and R*-tree structures (thanks to Hadjieleftheriou's implementation, Hadjieleftheriou 2004). Randomly generated data have been generated and sequentially issued to a fixed number of random points acting as information sources. Tests have been conducted changing the total number of data to index (1000-200000), the number of information sources (10-100) used and the portion of the base to scan for interval queries. The tests have been conducted on a 1.6 G Hz, 128 Mo RAM computer, running Linux. The programing language used was Java.

Due to the differentiation of spatial and temporal component, and due to the fact that the data were coming from a finite set of spatial points, the Po-tree built time has been greatly reduced compared to the R*-tree. While 25 000 points stemming from 100 different locations were indexed in less than one second with the Po-tree, it took nearly 45 seconds with a R*-tree.

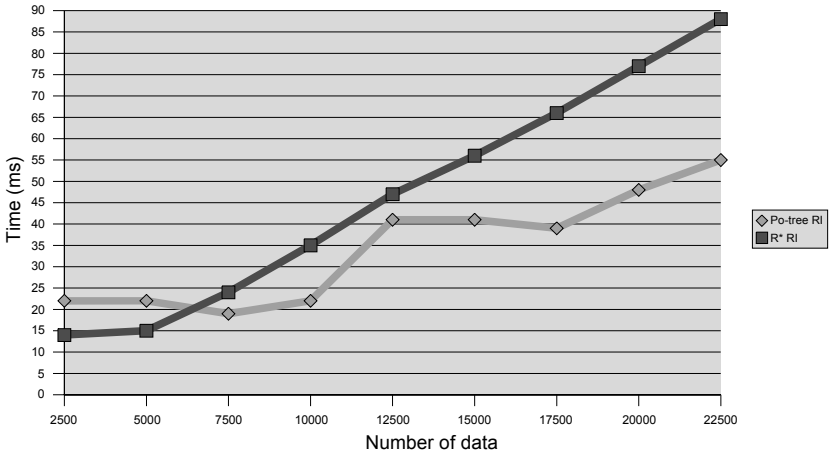

Figure 3: Range-Interval queries

Other tests have shown that the construction time of the Po-tree evolved linearly with the number of stations, the number of different spatial locations.

The different queries have shown interesting properties as well. The interval queries took an advantage of the linking of the temporal nodes of the Po-tree. For point-interval queries, the Po-tree can be up to 8 times faster the the R*-tree. While for interval queries the difference has shown much lighter, it still remains in favor of our solution, as shown in figure 3. On this figure, the last 10% of the entered data where fetched. The spatial range covered the whole possible locations. It is visible than for a low number of data the R*-tree fares better, yet when the amount of data rises past 6000, it is the Po-tree that gains the advantage. These results can be explained by the fact that R*-trees need to consider the whole set of data so as to perform a query. Therefore, the more data are indexed, the longer it takes to find specific values.

The results obtained have shown that the Po-tree was compatible with the constraints set by our application case: favoring the newest data, processing of big quantities of data in a given time, fixed set of spatial sources, possibility to use in a real-time system. Even though the mobility is not yet easily managed, the Po-tree meets the initial specifications.

CONCLUSION

The Po-tree aims at indexing spatio-temporal data issued from a network of spatially referenced sensors, with a focus given to the newest data. Our goal was to accelerate answering time to real-time queries. Our application case is linked to volcanic monitoring, yet it can be extended so as to include other natural disaster prevention scenarios, or scenarios where a set of fixed sensors sends huge quantities of data to a central database.

The structure of the Po-tree uses two parts. The main difference with the existing solutions stemming from this devision: it does not favor the temporal dimension over the spatial but does the reverse... A spatial sub-tree references the positions of fixed sensors, sources of informations. Each position, each sensor, is then linked to a temporal sub-tree that shall points to the actual data. The spatial sub-tree is based on the kd-tree, compatible with ICC protocols, yet sensible to the insertion order while the the temporal tree is a modified B+-tree, akin to an AP-tree.

Different tests have shown that this solution can be used with ease in cases where updates from a given set of sensor are frequent. The lookups strategy has been designed to favor the newest data. The notion of information source allow fast interval queries as the data from a given source are linked through the temporal sub-tree.

Further developments of the structure will include mobility and the use of R*-trees to replace the actual spatial sub-tree. Another point that shall be studied will be the linkage of the database to the data warehouse. The Po-tree has been designed with specific needs in mind, but it can be easily adapted to cover a majority of natural disaster cases where fixed sensors are the main source of information.

Acknowledgments should be made to the Universidad de las Americas, Puebla, for their work on the Popocatepetl monitoring. They coordinate and greatly help the different researches based on this volcano.

BIBLIOGRAPHY

CENAPRED, Monitoreo y Vigilancia del Volcan Popocatepetl,[Online], viewable at: http://tornado.cenapred.unam.mx/mvolcan.html (last consulted 20/02/04)

Gunadhi H & Segev A, 1993, Efficient indexing methods for temporal relation, In IEEE Transactions on knowledge and Data Engineering, 5(3), 496-509

Hadjieleftheriou M. Spatial Index Library [Online], viewable at: http://www.cs.ucr.edu/~marioh/spatialindex/, (last consulted 20/02/04)

Haritsa J.R., Seshadri S., 2001, Real- time index concurrency control. In Real Time Database System – Architecture and Techniques, Kluwer Academic Publishers, Boston, ISBN: 0-7923-7218-2, 60-74

Lam K.Y., Kuo T.W., 2001, Real time database systems: an overview of systems characteristics and issues. In Real Time Database System – Architecture and Techniques, Kluwer Academic Publishers, Boston, ISBN: 0-7923-7218-2, 4-16

Mokbel M., Ghanem T.M., Aref W.G., "Spatio-temporal Access Methods", *IEEE Data Engineering Bulletin*, Vol 26, n°2, 2003, pp. 40-49

Nascimento,M., Silva, J., 1998, Towards Historical R-trees. In Proceedings of ACM Symposium on Applied Computing (ACM-SAC), 235—240

Ooi B.C., Tan K.L., 1997, spatial databases. In Indexing Techniques for Advanced Database Systems, Kluwer Academic Publishers, Boston, ISBN 0-7923-9985-4, 39-75

Ooi B.C., Tan K.L., 1997*, temporal databases. In Indexing Techniques for Advanced Database Systems, Kluwer Academic Publishers, Boston, ISBN 0-7923-9985-4, 113-150

Paspalis N. Implementation of Range searching Data-Structures and Algorithms [Online], viewable at: http://www.cs.ucsb.edu/~nearchos/cs235/cs235.html, (last consulted 20/11/03)

Theodoridis Y., Vazirgiannis M. & Sellis T., 1996, Spatio-temporal indexing for large multimedia application, In Proceedings of the 3rd IEEE conference on multimedia computing and systems (ICMCS)

Usgs, Earthworm documentation [on line], viewable at: gldbrick.cr.usgs.gov/, (last consulted 15/01/04)

Wang X., Zhou X., Lu S., 2000, Spatiotemporal Data Modeling and Management: A Survey, In Proceedings of the 36th International Conference on Technology of Object-Oriented languages and Systems, Xi'an, China. IEEE Computer Society Press, 202-221