Information Systems 56 (2016) 19–35

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

# Formalism for a multiresolution time series database model

Aleix Llusà Serra*, Sebastià Vila-Marta, Teresa Escobet Canal

*Department of Electronic System Design and Programming, Universitat Politècnica de Catalunya, Av. Bases de Manresa 61–73, 08242 Manresa, Catalonia, Spain*

ARTICLE INFO

*Article history:*
Received 29 July 2014
Received in revised form
24 August 2015
Accepted 25 August 2015
Recommended by: L. Wong
Available online 6 September 2015

*Keywords:*
Time series
Database systems
Multiresolution
Lossy compression
Approximate queries

ABSTRACT

We formalise a specialised database management system model for time series using a multiresolution approach. These special purpose database systems store time series lossy compressed in a space-bounded storage. Time series can be stored at multiple resolutions, using distinct attribute aggregations and keeping its temporal attribute managed in a consistent way.

The model exhibits a generic approach that facilitates its customisation to suit better the actual application requirements in a given context. The elements, the meaning of which depends on a real application, are of generic nature.

Furthermore, we consider some specific time series properties that are a challenge in the multiresolution approach. We also describe a reference implementation of the model and introduce a use case based on real data.

## 1. Introduction

Data collection processes proliferate due to the emergence of embedded systems and sensor networks, providing opportunities to collect large amounts of data. These data should be analysed and processed by information systems to be useful. A typical processing, for instance, is to detect eventual sensor failures or malfunctions and, if it is possible, to reconstruct the faulty data. The acquired data instances hold a timestamp. Therefore, correctness criteria must include both data values and their timestamps. The sequences of data values collected at specific timestamps are formalised as *time series*.

A time series is a collection of chronological observations. In general, we continuously acquire a time series from phenomena monitoring. On the one hand, we can record observations at regular intervals, such as hourly or daily ones, resulting in equally spaced time data. On the other hand, we can record observations at irregular intervals, such as recording when a pump is open or closed, resulting in unequally spaced time data. Time series data are often voluminous [1,2], thus efficiently storing and accessing them can be complex. Moreover, this is especially critical when developing small embedded systems with constrained resources (capacity, energy or processing power) [3]. Additionally, unequally spaced time data increases the difficulty of processing.

The literature describes several attempts to build systems devoted to managing and store time series data. These systems are generically known as *Time Series Database Management Systems* (TSMS) [4,5]. However, as shown below, most of them exhibit some drawbacks when trying to solve the challenging issues of time series in the temporal data domain.

Time series can be stored and managed by *relational database management systems* that are usually queried using *Structured Query Language* (SQL). Nonetheless, some authors [4,6–8] notice that the use of SQL systems as a time series backend suffers from some drawbacks.

* Corresponding author.
*E-mail addresses:* aleix@dipse.upc.edu (A. Llusà Serra), sebastia.vila@upc.edu (S. Vila-Marta), teresa.escobet@upc.edu (T. Escobet Canal).

http://dx.doi.org/10.1016/j.is.2015.08.006
0306-4379/© 2015 Elsevier Ltd. All rights reserved.

NOSQL or NEWSQL products are being developed to increase the performance and flexibility of SQL systems [7–10]. It is natural to consider them to store time series data. Indeed, the continuous acquisition nature of the time series poses an issue when trying to store and analyse all the data [11].

We can apply compression techniques in two distinct styles to face the challenges posed by time series data. First, to get an approximation to the original signal that facilitates to do pattern search analysis or finding similarities [1,5,12]. Second, as a compression and aggregation approach that leverages the storage of massive *data streams* [13,14]. Nonetheless, handling time series like data streams neither considers adequately the time dimension nor computes the evolution of aggregated parameters along the time, which is interesting for monitoring purposes.

*RRDtool* [15] is a system that stores time series aggregated using different resolutions. These characteristics allow to compact the data and facilitates faster visualisations. In spite of this, because *RRDtool* is a particular application, aggregation operations are limited to network monitoring.

### 1.1. Contributions

This paper formalises a model for TSMS that stores and manages time series data. This model exhibits several unusual characteristics:

- It organises the data in an aggregated way and it allows to store time series using different time resolutions. We name this feature *multiresolution*. Thus, being multiresolution the most salient characteristic of our model, we call the formalised system *Multiresolution Time Series Database Management System* (MTSMS). The model is designed to satisfy the requirements of bounded storage computers such as sensor systems.
- It is a *lossy storage* solution. Multiresolution allows for a lossy storage solution that selects only the relevant data. In some sense, multiresolution is close to the lossy compression methods used in multimedia applications, that discard meaningless data in favour of size.
- It considers the time sampling irregularities of time series and operates coherently with the time dimension of time series.
- It offers a degree of genericity to cope with the semantic characteristics of the actual data.
- It is soundly formalised using set algebra and, particularly, relational algebra.

Multiresolution requires aggregating several data instances into a single one. We abstracted this through an *aggregation function* bound to the precise semantics of the actual data. Because of this, aggregation functions are set as an independent object of the main model. Users can define new aggregation methods better suited for particular fields.

The model also formalises the concept of time series *representation function*. This concept allows users to define different operators considering the behaviour of time series in different contexts. This issue is important to manage the precise semantics of the stored time series.

Our model shares some of its characteristics with other known approaches. The analysis of *RRDtool* [15] inspired the multiresolution. However, we provide a sound formalisation that lacks in [15] and a degree of genericity unavailable in *RRDtool*. Based on this facility, in our model we can define particular time series aggregations like those of *RRDtool*. To formalise time series we follow the same approach used to formalise bitemporal data for a relational DBMS. The model favours more recent data over the older one, which is in common with some other methods, like that of Cormode et al. [13].

We remark that the only goal of the model formalised here is to manage time series data. In practical applications, it would be usual to complement this model with a standard database system to handle all the remaining data if needed. For instance, time series metadata such as units of values, sensor localisation or classification tags would be stored in a standard DBMS.

### 1.2. Outline

This paper is organised as follows. Section 2 introduces previous work that concerns TSMS and MTSMS. The motivation for multiresolution is set out in Section 3. We describe the model in two steps. First, in Section 4 we formalise a TSMS model devoted to the basic elements and operations of time series. Second, in Section 5 we formalise a MTSMS model that extends the previous one with multiresolution capabilities. In Section 6 we describe an implementation of the integrated TSMS and MTSMS model. Section 7 is devoted to a real data multiresolution database example. Finally, Section 8 offers some conclusions.

## 2. Previous work

For the sake of completeness here we describe some previous work related to time series storage. We organise this in three subsections. First, we introduce some previous approaches to database management systems for time series. Second, we explain how some authors applied compression techniques to leverage time series storage. Third, we review time series storage systems based on the data streams paradigm.

### 2.1. Database approaches

According to some authors, TSMS should be considered as a specialised relational DBMS [5]. Segev and Shoshani [16] propose a structured language for querying TSMS. Their time series structures include the notion of regularity and temporal representation and their operations are SQL-like. Dreyer et al. [4] suggest the requirements of a particular purpose TSMS and base the model on five basic structural elements: events, time series, groups, metadata and time series basis. They implement a TSMS named *Calanda* which includes calendar operations, it allows grouping of time series, and it operates with simple queries. They exemplify