An OLAP-Based Approach to Modeling and Querying Granular Temporal Trends

Alberto Sabaini[1], Esteban Zimányi[2], and Carlo Combi[1]

[1] Department of Computer Science, University of Verona, Italy
{alberto.sabaini,carlo.combi}@univr.it
[2] Department of Computer and Decision Engineering,
Université Libre de Bruxelles, Belgium
ezimanyi@ulb.ac.be

Abstract. Data warehouses contain valuable information for decision-making purposes, they can be queried and visualised with Online Analytical Processing (OLAP) tools. They contain time-related information and thus representing and reasoning on temporal data is important both to guarantee the efficacy and the quality of decision-making processes, and to detect any emergency situation as soon as possible. Several proposals deal with temporal data models and query languages for data warehouses, allowing one to use different time granularities both when storing and when querying data. In this paper we focus on two aspects pertaining to temporal data in data warehouses, namely, *temporal patterns* and *temporal granularities*. We first motivate the need for discovering granular trends in an OLAP context. Then, we propose a model for analyzing granular temporal trends in time series by taking advantage of the hierarchical structure of the time dimension.

1 Introduction

Data warehouses contain valuable information for decision-making purposes. They can be queried and visualised with Online Analytical Processing (OLAP) tools. Data warehouses are different from usual databases, as they contain aggregated data described by a multidimensional data model. They contain time-related information and thus representing and reasoning on temporal data is important both to guarantee the efficacy and the quality of decision-making processes, and to detect any emergency situation as soon as possible. Several proposals deal with temporal data models and query languages for data warehouses, allowing one to use different time granularities when storing and querying data [2,5,7].

In this paper, we focus on two general aspects of these data, namely, temporal patterns and temporal granularites. Temporal patterns provide a significant source of information for decision-makers. They represent specific sequences of data values relevant to the considered domain. Typically, temporal patterns are made of some basic *temporal trends* (i.e., increase and decrease). As an example, negative trends of articles sold could induce a manager to promote or to lower

L. Bellatreche and M.K. Mohania (Eds.): DaWaK 2014, LNCS 8646, pp. 69–77, 2014.

their prices. On the other hand, temporal granularities are used for representing time in the modelled reality. They can be used for describing time units at a finer or coarser level of detail. Applications have to deal with various granularities even for the same domain, and they need to be carefully considered both when representing, storing, and querying temporal data. As an example, purchases may be registered in minutes or hours, while suppliers' deliveries may be scheduled in weeks or months.

In the following, we address the problem of finding *granular temporal trends* in data warehouses by proposing two new OLAP operators, namely, *Trend* and *TrendAggregator*. The first operator exploits the hierarchical structure of dimensions in order to find trends of possibly aggregated data at some granularity level. It allows users to use standard OLAP operations for retrieving time series at the desired level of detail, as a mean to find trends that yield decision-making, e.g., "Display for each product, the positive trends in days for each quarter of 2013." The second operator allows users to further investigate the discovered trends, represented by means of a multidimensional model. This operator aggregates trends on the dimension hierarchies, e.g., "Display for each city, the longest trends between days of 2013."

This paper is organized as follows. In Sect. 2 we describe the notion of *granular temporal trends* introduced in [1]. Then, we briefly discuss some main contributions from the literature in the area of trend analysis on time series. In Sect. 3 we describe the novel trend operators for time series in OLAP and the aggregation of the discovered trends. Lastly, in Sect. 4 we draw some conclusions.

2 Related Work

In this section we start by discussing a logic-based taxonomy for the description of granular temporal trends, presented in [1], on which our work is based. The authors presented a graphical representation of the dimensions related to granular trends, called *Trend Dimension Tree*. It systematically describes the temporal features related to a granular trend. The main feature we are going to consider is the *granular type* one: it allows one to distinguish two kinds of trends, called *intragranule* and *intergranule*, respectively. In intragranular trends, temporal properties expressed by means of a trend must be satisfied inside a given granule; in intergranular trends, properties must be satisfied over different granules.

Now we briefly discuss some contributions from the literature regarding time series and temporal trends. In [8], the author introduced trend dependencies, which allow one to express significant temporal trends, e.g., *Salaries of employees should never decrease across months*. The time dimension is captured by trend dependencies through the concept of time accessibility relation, which can express also time granularities in a simple and elegant way. Trend dependencies can compare attribute values by some comparison operator.

The authors of [4] developed a method for extracting trends by means of neural networks, used as a tool to discover all existing hidden trends in several different types of crimes in US cities.

The authors of [6] argue that traditional temporal data models are mainly focused on describing temporal data based on versioning of objects, tuples, or attributes. Time series data are frequently found in real-world applications, such as sales, economics, and scientific data, but they are not effectively managed by this approach. They propose a temporal query language that treats both version-based and time series data in an uniform manner.

3 Granular Temporal Trends in OLAP

In this section, we recall the definition of the multidimensional model and describe how it may be queried by means of Online Analytical Processing (OLAP) tools. We discuss how OLAP queries may be interpreted as *time series*. At last, we define trends and we describe two new operators for discovering granular temporal trends on time series.

A multidimensional model is based on the notion of dimensions, measures, and facts. Each dimension is organized in a hierarchy of levels, corresponding to data domains at different granularities. A multidimensional schema describes numerical facts defined with respect to a particular combination of levels and quantified by measures. An example of a multidimensional schema is depicted in Fig. 1.

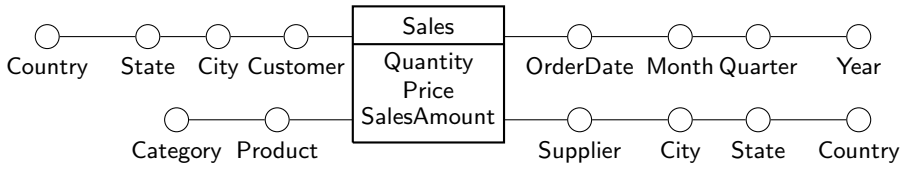


Fig. 1. An example of a multidimensional fact with four dimensions and three measures represented by means of the dimensional fact model (DFM) [3]

Data may be retrieved from instances of multidimensional schemas by means of OLAP queries. They consist of three basic analytical operations: roll-up for aggregating data (hence, showing data at a coarser level), drill-down for showning data at a finer level, and slicing and dicing for filtering data. Consider the following OLAP query:

Example 1. Display for each Day *the total* Quantity *of items sold.* This query may be represented in a more compact way as:

Sales(OrderDate:Day, Qty)

The schema name is Sales, all the omitted atemporal dimensions are aggregated to the All level, the level Day has been selected for the time dimension OrderDate, and the selected measure is Qty.

Now consider the following OLAP query:

Example 2. Display for each City *and for each* Day *the total* Quantity *of items sold.*

Sales(Customer:City, OrderDate:Day, Qty)

The City level of the Customer dimension has been selected, the omitted atemporal dimensions are aggregated to the All level, and the time dimension OrderDate is still at the Day level.

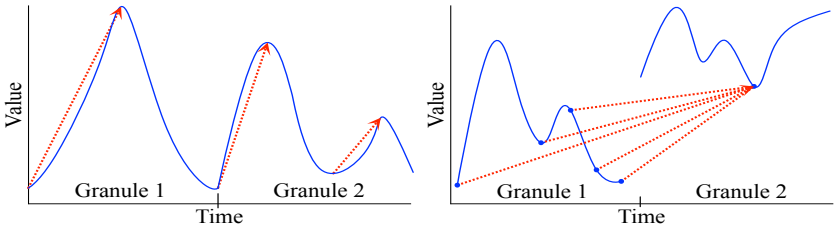
The results of both queries may be interpreted as *time series*. In an OLAP context, they are ordered sets of triples $\langle \bar{d}, t, v \rangle$, where $\bar{d}$ represents the grouping values for the atemporal dimensions, t is a time point at a particular granularity, and v is its possibly aggregated value. The grouping component $\bar{d}$ contains all the level members for each atemporal dimension considered in a query. It resembles the grouping clause of SQL queries, since it identifies one time series among the selected ones. A fact aggregated to the All level in each dimension is represented by a time series in which $\bar{d}$ is empty.

The result of the query in Example 1 may be interpreted as triples $\langle \emptyset$, Day, *value*$\rangle$, ordered according to the time dimension OrderDate. *value* is the value of the measure Qty, aggregated on each dimension to the level All. Values are not aggregated on the selected temporal dimension OrderDate, since the level Day corresponds to the bottom one.

On the other hand, the result of the query in Example 2 may be represented as triples $\langle$City, Day, *value*$\rangle$, ordered according to the time dimension OrderDate. *value* refers to the measure Qty, aggregated on each dimension to the level All, and to the level City in the Customer dimension.

Trends point out peculiar behaviours in time series. They are essential for decision making-processes, since they abstract the evolution of a measure across time as increasing or decreasing according to a fixed threshold. Users need analysis tools for discovering this information on time series in OLAP context. We are interested in finding *granular temporal trends* on time series of data that could possibly be aggregated. According to the classification presented in [1], there are eight different characterizations of granular trend. We are going to consider two of them, namely, *intragranular trend* and *intergranular trend*, depicted in Fig. 2.

In the first case, one is interested in finding time points within one granule of the specified granularity in which a trend holds. An example of a query is "Find positive trends of items sold for each product category between days in months." In the second case, one is interested in finding trends between some time points within a granule and a representative in the consecutive one. An example of a query is "Find positive trends that hold in at least 90% of days for each city across months." In both cases a base level and a grouping level have to be selected on the temporal dimension for discovering granular temporal trends. Time points will be considered at the base level and granules will be considered at the grouping level (e.g., Day as base level and Quarter as grouping level). Intragranular trends express a relationship between members of the base level, while intergranular trends express a relationship between members of the grouping level.



(a) Intragranular trend (b) Intergranular trend

Fig. 2. Trend types: (a) *intragranular* case (b) *intergranular*

In the following, we propose a new OLAP operator, *Trend*, that discovers *intragranular* or *intergranular* trends on time series of possibly be aggregated data. It takes as input six parameters: *Trend(GranuleType, TrendType, BaseLevel, GroupingLevel, Threshold, [Percentage])*. (1) *GranuleType* may be either *intragranular* or *intergranular*. (2) *TrendType* may be either ↑ for increasing or ↓ for decreasing. (3) *BaseLevel* and *GroupingLevel* are the levels of the temporal dimension to be used for discovering the trends. (4) *Threshold* is a numerical number that quantifies the minimum difference between two time points. (5) *Percentage* is a numerical value between 0 and 100, that can be specified only in the *intergranular* case: it is the minimum of time points in a granule that needs to participate in the trend. The trends discovered by the operator can be represented a multidimensional way, allowing users to perform more in-depth analysis. To this purpose, we also propose a new aggregation operator, namely, *TrendAggregator*, that summarizes the previously discovered trends.

3.1 Intragranular *Trend* Operator

An *intragranular trend* expresses a relationship between time points in the same granule within a time series. Two (or more) consecutive time points are considered a trend if they belong to the same granule, and the specified pattern (increase, or decrease) holds according to a threshold.

The *Trend* operator analyses OLAP queries results interpreted as time series. An *intragranular trend* is a subset of time points of a time series. Given three consecutive time points t_i, t_i, t_{i+1} within the same grouping granule, t_i belongs to a trend if the pattern between t_i and t_{i+1} holds. The trend starts in t_i if the pattern does not hold between t_{i-1} and t_i.

An intragranular trend is characterized by its starting point, its measure value at the trend start, its length expressed in number of time points, and its total steepness. The latter is the sum of the differences between all values in it.

The result of the *Trend* operator is represented as a multidimensional instance of a schema Fact′, derived from its source schema Fact. It contains two new measures, the trend Length and its Steepness. The dimensions used for grouping

time series are kept, while the others are dropped. Furthermore, the dimensions hierarchies do not contain the levels below the one used in a query. As an example, consider a query on the schema depicted in Fig. 1, in which only the City level of the Customer dimension is used. All the other atemporal dimensions are not used, and hence they are dropped in the derived schema. The Customer dimension loses the bottom level Customer, since it is below the City one.

Trends are maximal within a time series (i.e., each point may belong to only one trend), and only the starting points are kept in the resulting fact instance, as representative of the discovered trends.

Example 3. Display for customer cities, the positive trends of quantities of items sold between days in months of 2013.

$$\mathsf{SalesIncrease} \leftarrow \mathit{Trend}_{\mathit{Intragranular},\uparrow,\mathsf{Day},\mathsf{Month},\,0}(\mathsf{Sales}(\mathsf{Customer{:}City},\ \mathsf{OrderDate{:}Day},\ \mathsf{OrderDate{:}Year{=}2013},\ \mathsf{Qty}))$$

First, data contained in the Sales cube are aggregated to the City level in the Customer dimension, and to the All one in the other dimensions. Then, the aggregated data are sliced, and only the orders placed in 2013 are considered.

Each day t_i is compared with the next one t_{i+1} to check if it belongs to a positive trend. Table 1 shows an example of a time series and the discovered trends. Specific days do not appear within the table, indicating that no sales occurred that day. Thus, July 22^{nd} and 31^{st} are considered subsequent days within the month granule. According to that time series, two trends have been discovered. The first one starts on July 21^{st}, lasts for 10 days, and has a total steepness of 156. The second one starts on August 1^{st}, it lasts 10 days, and it has a steepness of 20.

The resulting trends are stored in a new fact named SalesIncrease. Its schema consists of the dimension Customer, from the City level and above, and the OrderDate dimension that starts from the Day level, as shown in Fig. 3.

Table 1. Example of a time series regarding quantities of items sold in 2013 and the discovered trends

Date	Quantity
2013-07-21	9
2013-07-22	17
2013-07-31	165
2013-08-01	250
2013-08-10	270

OrderDate	Quantity	Length	Steepness
2013-07-21	9	10	156
2013-08-01	250	10	20
...	...	...	...
...	...	...	...
...	...	...	...

3.2 Intergranular Trend4BI Operator

An *intergranular trend* expresses a relationship between time granules. This kind of trend requires that a given pattern holds in some of the consecutive granules

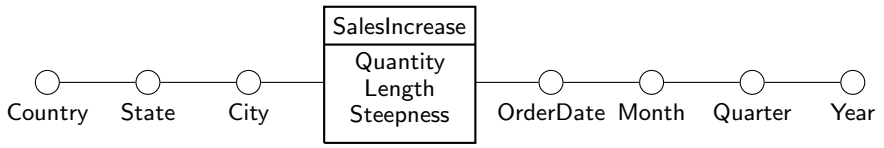


Fig. 3. Example of application of *Trend* on the schema depicted in Fig. 1, for finding intragranular trends. The derived schema consists of the dimension Customer, from the City level and above, and the OrderDate dimension.

of the specified grouping granularity level. Two consecutive granules g_1, g_2 are considered a trend if the specified pattern holds between their time points: the pattern must hold among a specified percentage of time points in g_1, ranging from 0% to 100%, and a representative in g_2. A different representative is chosen depending on the trend type. In the case of a positive trend, all the points of g_1 will be compared to the minimum of g_2 (in case of negative trend, the representative is the maximum). Figure 2 shows a positive trend between almost all the time points in g_1 and the minimum of g_2. As an example, two months belong to a positive increasing trend with threshold equal to 30 for the measure and 100% threshold for the subgranules, if the difference between the measure within all the days in the first month and the minimum in the second month is equal or greater than the 30.

The *Trend* operator analyses OLAP queries interpreted as time series. A trend is a subset of grouping time granules of a series. Each time point t_i in the first granule g_i is compared with the representative t_j in the consecutive granule, for assessing if g_i belongs to a trend or not. The starting point of a trend is the granule for which the pattern does not hold with the previous one. An intergranular trend is characterized by its starting point, the length of the trend (expressed in number of granules), and the total steepness. The steepness between two consecutive granules, is the maximal difference between all points in the first granule, and the representative in the second one. The result of the *Trend* operator may be represented by the multidimensional schema Fact′, derived from its source schema Fact. It contains two new measures, the trend Length and its Steepness. The dimensions used for grouping time series are kept, while the others are dropped. Furthermore, the dimensions hierarchies do not contain the levels below the one used in a query.

Example 4. Display for customer cities, the positive trends of quantities of items sold across months with a 100% threshold. The set of time series to be analyzed is the result of the following query:

$$\mathsf{SalesIncrease} \leftarrow \mathit{Trend}_{\mathit{Intergranular},\uparrow,\mathsf{Day},\mathsf{Month},0,100}(\mathsf{Sales}(\mathsf{Customer{:}City}, \mathsf{Supplier{:}All}, \mathsf{Product{:}All}, \mathsf{OrderDate{:}Day}, \mathsf{Qty}))$$

First, data contained in the Sales cube is aggregated to the City level in the Customer dimension, and to the All in the other dimensions. The result is interpreted as a set of time series, one for each product city.

Each day t_i in the month g_i is compared to the minimum day t_j in the subsequent month g_j, to check whether g_i belongs to a positive trend. The resulting trends are stored in a new fact Sales′. Its schema consists of the dimension Customer, with the City level and above, and the dimension OrderDate, that starts from the Month level, as shown in Fig. 4.

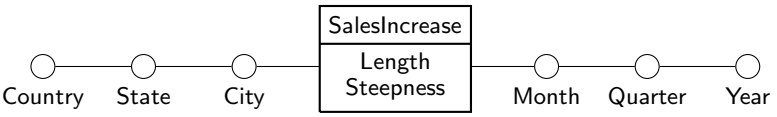


Fig. 4. Example of application of the *Trend* operator on the schema depicted in Fig. 1, for finding intergranular trends. The derived schema consists of the dimension Customer, from the City level and above, and the OrderDate dimension, from the Month and above.

3.3 Trend Aggregator

The trends discovered by the *Trend* operator are represented in a multidimensional way, where only the time dimension and the atemporal dimensions used for grouping purposes are kept, as shown in Figures 3 and 4. The derived schema allows users to continue their analysis by means of OLAP operators. Thus, aggregating the discovered trends is a problem that needs to be addressed: they are characterized by two measures, the length and the steepness, which are strictly linked. Usual aggregation operators would consider them separately, leading to erroneous results. Consider the example in Table 2, in which several trends have been discovered. The temporal dimension may be aggregated to the Quarter

Table 2. Example of discovered trends. The *TrendAggregator* operator may be applied for aggregating them to the Quarter level on the time dimension. The highlighted trend is the result of this aggregation.

OrderDate	Length	Steepness
2013-07-21	*10*	*156*
2013-08-01	10	20
2013-08-15	3	190

level, in order to discover its longest trend. The standard *max* operator would return 10 as maximum for the Length measure, and 190 for the Steepness. This could mislead the user, since that trend does not exists. An ad hoc operator for preserving this link is needed.

TrendAggregator allows one to perform a *prioritized maximum or minimum aggregation*: it consists of choosing the longest trend, with the maximal or minimal steepness, depending on the user's choice. In the example shown in Table 2,

the trend aggregated to the Quarter level is the first one, since the maximum Length is 10, and the maximal Steepness associated to it is 156.

4 Conclusions and Future Work

Users need to perform more in-depth analysis for enhancing their decision-making processes. Temporal patterns provide a significant source of information for decision-makers, but they are usually made of some basic temporal trends. To address this problem, we merged two general aspects of temporal data, namely, temporal patterns and temporal granularities, with analysis on aggregated data. We proposed two new operators, we called, *Trend* and *TrendAggregator*, for finding *granular temporal trends* in Online Analytical Processing context. *Trend* points out peculiar behaviour in data, by exploiting the hierarchical structure of dimensions in order to find trends of possibly aggregated data at some granularity levels. *TrendAggregator* allows users to further investigate and summarize the discovered trends, represented by means of a multidimensional model.

We are extending our approach to consider time series with multiple values associated to each time point. We are also working on how to temporally summarize the discovered trends. We are interested in finding temporal trends in medical data, and in particular in reports of unexpected adverse reactions induced by drug administrations.

References

1. Combi, C., Pozzi, G., Rossato, R.: Querying temporal clinical databases on granular trends. Journal of Biomedical Informatics 45(2), 273–291 (2012)
2. Eder, J., Koncilia, C., Morzy, T.: The COMET metamodel for temporal data warehouses. In: Pidduck, A.B., Mylopoulos, J., Woo, C.C., Ozsu, M.T. (eds.) CAiSE 2002. LNCS, vol. 2348, pp. 83–99. Springer, Heidelberg (2002)
3. Golfarelli, M., Maio, D., Rizzi, S.: The dimensional fact model: A conceptual model for data warehouses. Int. J. Cooperative Inf. Syst. 7(2-3), 215–247 (1998)
4. Kaikhah, K., Doddameti, S.: Discovering trends in large datasets using neural networks. Appl. Intell. 24(1), 51–60 (2006)
5. Khatri, V., Ram, S., Snodgrass, R.T., Terenziani, P.: Capturing telic/atelic temporal data semantics: Generalizing conventional conceptual models. IEEE Trans. Knowl. Data Eng. 26(3), 528–548 (2014)
6. Lee, J.Y., Elmasri, R.: An EER-based conceptual model and query language for time-series data. In: Ling, T.-W., Ram, S., Li Lee, M. (eds.) ER 1998. LNCS, vol. 1507, pp. 21–34. Springer, Heidelberg (1998)
7. Malinowski, E., Zimányi, E.: A conceptual model for temporal data warehouses and its transformation to the ER and the object-relational models. Data Knowl. Eng. 64(1), 101–133 (2008)
8. Wijsen, J.: Reasoning about qualitative trends in databases. Inf. Syst. 23(7), 463–487 (1998)