A semi-automatic design methodology for (Big) Data Warehouse transforming facts into dimensions

Lucile Sautot, Sandro Bimonte, Ludovic Journaux

A semi-automatic design methodology for Data Warehouse and Big Data Warehouse transforming facts into dimensions

Lucile Sautot, Sandro Bimonte and Ludovic Journaux

Abstract—A decision support system is used by decision makers for a long time. But, in some cases, the originally designed multidimensional schema does not cover the entire needs of decision makers, which can change over time. One such unfulfilled needs, is using facts to describe dimension members. In this article, we propose a methodology to transform the constellation schema of a data warehouse by integrating factual data into a dimension. The proposed methodology and algorithms enrich a constellation multidimensional schema with new analytical possibilities for decision makers. This enrichment has repercussions for the entire multidimensional schema that are managed by multidimensional modeling, hierarchy calculation and the hierarchy version. In this article, we present a theoretical view of the proposed methodology supported by a case study, an implemented prototype and a complete evaluation based on a standard benchmark.

Index Terms—Data Warehouse, OLAP, Modeling, Hierarchy, Version, Refinement

1 Introduction

Data Warehouses (DWs) and On-Line Analytical Processing (OLAP) systems are first Business Intelligence tools. DWs and OLAP systems are designed to provide analytical views of large data sets. They have been successfully used in several application domains including marketing, health and retail. Warehoused data are stored according to the multidimensional model used, which defines the analytical axes (i.e. dimensions), and subjects (facts). The data are analyzed using OLAP tools that provide a set of operators (Roll-up, Slice, etc.) which make it possible to aggregate measures at different levels of granularities (levels of dimensions hierarchies). OLAP operators results are visualized using pivot tables and interactive graphical displays provided by OLAP clients. The OLAP decision-making process is based on the exploratory, interactive and iterative analysis of warehoused data.

Therefore, the better the implemented multidimensional model matches the analytical needs of the decision makers, the more useful OLAP analysis are for the decision making process. To achieve that goal, much effort has been invested by academic and industrial communities in designing DWs. Several authors have proposed design methodologies for DWs based on data (i.e. data-driven), user requirements (i.e. user-driven), and data and user requirements (mixed-driven) [1]. It is widely accepted that mixed-driven methods are the most suitable for effective DW design. (Semi)-automatic methodologies have also been proposed to (i) speed up the design process, and (ii) enable error free implementation. All these methods usually produce constellation schemes (i.e., a multidimensional model with several facts). The OLAP Drill-across operator was defined to explore different facts according to some common dimensions, which allow the decision makers to "visualize" relationships among the different subjects for analysis.

However, with the advent of "Big Data", and the proliferation of available data in all application domains (e.g. agriculture, ecology, the environment) new design issues have emerged related to the warehousing of these data sources. Indeed, during the modeling process of a DW from real-world data sources, OLAP designers often note that the structure of the data sources is not compatible with the multidimensional model. Some authors therefore investigated dimensions with numerical data and with no hierarchical structure, and factual data with categorical data [2], [3], [4].

But these are not effective when factual data has to be used as the analytical dimension (dimension with hierarchies) in a constellation schema.

In this work, our aim was thus to extend a previous study [5] to provide a semi-automatic mixed-driven design methodology that creates dimension hierarchies originating from factual data in a constellation schema. The main contributions of this paper are:

1) *Transformation*: A formal approach is proposed to automatically calculate dimension hierarchies using factual data. The new hierarchies can be used to analyze other facts in a constellation schema. This work extends that proposed in [5] by using several facts for the creation of hierarchies.
2) *Reduction*: The previous methodology created several hierarchies that are added to the new dimension(s). However, when hierarchies become numer-

- *L. Sautot was with UMR TETIS, AgroParisTech, Maison de la Télédétection, Montpellier, FRANCE. E-mail: lucile.sautot@agroparistech.fr*
- *S. Bimonte is with UR TSCF, IRSTEA, Aubière, FRANCE. E-mail: sandro.bimonte@irstea.fr*
- *L. Journaux is with AgroSup Dijon, Dijon, FRANCE. E-mail: ludovic.journaux@agrosupdijon.fr*

Manuscript received April 19, 2005; revised August 26, 2015.

ous, the multidimensional model becomes too complex to be used by decision makers [6]. We therefore present three algorithms to "simplify" (i.e. reduce) the number of newly created hierarchies in the multidimensional model. The three algorithms are based on two main principles: (i) *the conceptual level* reduces the levels of the multidimensional model; (ii) *the data level* reduces the data volume of the multidimensional model.

3) *Implementation and evaluation*: We describe the implementation and experimentation of our methodology using the standard OLAP benchmark TPC-DS, a benchmark for big DWs. We chose it to validate our approach in the context of big data, since big data is increasingly used by industrial and academic communities. We also validated our methodology on a real DW concerning biodiversity.

The rest of the article is organized as follows: in Section 2 we describe the motivation for our work using an realistic case study.

In Section 3, we provide an overview of the proposed methodology. In Section 4, we provide some theoretical preliminaries. In Section 5, we detail the calculation of new hierarchies. In Section 6, we describe the general refinement of the multidimensional schema and in Section 7, we present the algorithms used to reduce the number of calculated hierarchies .

In Section 8, we described the implementation of the prototype of the methodology, and in Section 9, we detail the results of the experimentation on the de-facto standard benchmark for DWs: the TPC-DS benchmark.

Finally, in Section 10, we provide a literature review of the main topics of this article, before drawing our conclusions.

2 Motivation

The decision makers of an agriculture cooperative wish to analyze:

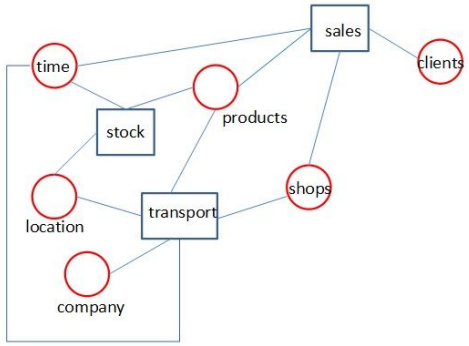
- The cooperative's sales over time, the products sold, the shops, and the clients.
- The stocks available to the cooperative, over time, the products, and the farms that produce them.
- The transport required over time, the products transported, the farms that stock the products, the shops that sell the products and the transporter.

The available hierarchies for the dimensions of this multidimensional schema are:

- {Day, Month, Year} and {Day, Day of Week} for the "time" dimension.
- {Name} and {Owner} for the "farm" dimension.
- {Transporter, Company} for the "transporter" dimension.
- {Client} for the "client" dimension.
- {Shop, Franchise} for the "shop" dimension.
- {Product, Category} for the "product" dimension. An example is shown in Figure 2. Figure 2a is the schema and Figure 2b its instance, with the members of each level.

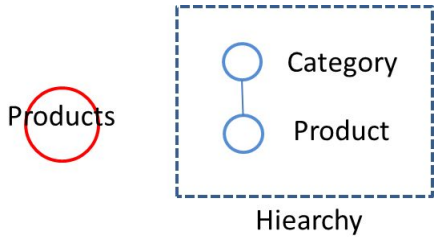
TABLE 1: An extract from the "sales" fact

Sales	*Clients.client*	*Products.products*	*Time.day*	*Shop.shops*
14,500	Rossi	Carrots	9-19-90	Carr1
1,200	Verdi	Carrots	9-19-90	Carr2
12,450	Rossi	Carrots	9-20-90	Carr1
13,540	Verdi	Carrots	9-20-90	Carr2

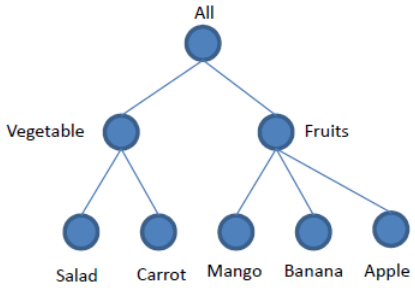


Blue squares: factual nodes; red circles: dimensional nodes

Fig. 1: The multidimensional of the example that motivated our work



(a) Schema of the hierarchy



(b) Instance of the hierarchy

Fig. 2: The hierarchical structure of the "Products" dimension

The constellation schema of the data warehouse associated with this example is shown in Figure 1. Extracts of data for sales and transport facts are shown in Tables 1 and 2.

For the sake of readability, the levels are not represented. This constellation schema allows for several different OLAP queries using:

- *each fact* (such as "What are total sales per product and year?" - Table 3), or using the fact "transport" it is possible to obtain the distance traveled per year for each product and shop. For example, Table 4 shows

TABLE 2: An extract from the "transport" fact

Distance (km)	*Products.products*	*Company.company*	*Time.day*	*Shop.shops*
1,245	Carrots	Trucking company	9-19-90	Carr1
1,503	Carrots	Logistics +	9-19-90	Carr1
12	Carrots	Trucking company	9-19-90	Carr2
1,245	Carrots	Trucking company	9-20-90	Carr1
12	Carrots	Trucking company	9-20-90	Carr2

TABLE 3: The total quantity of products sold per year

Dimension	**Dimension**	**Measure**
Product	*Year*	*Total Sales*
Carrots	2010	14,500
Bananas	2010	45,200
Carrots	2009	15,000
Apples	2010	20,000

TABLE 4: The distance traveled by each product and shop to the company Carrfor per year

Dimension	**Dimension**	**Dimension**	**Measure**
Year	*Shop*	*Product*	*Distance (km)*
2010	Carr1	Carrots	12.5
2010	Carr1	Bananas	4,000
2010	Carr2	Carrots	50
2010	Carr1	Apples	30

the results of the query "What is the distance traveled by each product, shop and year transported by the Company Carrfor in France?"),

- *more facts using the Drill-across operator with common dimensions*. An example of a Drill-across operator between the "transport" and "sales" facts using the "products" and "shops" dimensions is: "What are the total sales margins and distance per product and shop?" - Table 5.

The Drill-across query cited above makes it possible to link transport and sales, but only by simple visualization of the two measures in the OLAP client. In other words, it does not allow the products to be characterized in terms of transport effort, which is more useful for exploration and analysis. According to this new analytical need, the standard Drill-across operator appears to be useless because the transport data need to be integrated in the "product" dimension (or in another dimension) to aggregate sales by transport, as shown in Table 6. The new hierarchy in the "product" dimension is built with the "transport" fact data (Figure 3).

TABLE 5: The sales margin and average distance per product sold and per shop

Dimension	**Dimension**	**Dimension**	**Measure**	**Measure**
Year	*Product*	*Shop*	*Total Sales*	*Distance (km)*
2010	Carrots	Carr1	14,500	12.5
2010	Bananas	Carr1	45,200	4,000
2010	Carrots	Carr2	15,000	50
2010	Apples	Carr1	20,000	30

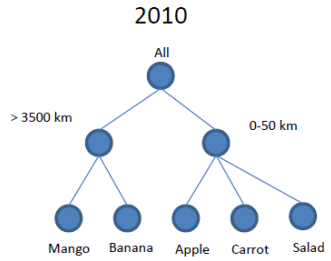


Fig. 3: The new hierarchy of products

TABLE 6: The sales margin per product sold depending on the distance considered as dimensional data

Dimension	**Dimension**	**Measure**
Distance (km)	*Product*	*Total Sales*
0-50	Carrots	29,500
0-50	Apples	20,500
0-50	ALL	49,500
more than 3,500	Bananas	45,200
more than 3,500	ALL	45,200

Existing DW design methodologies do not create hierarchies with factual data. That is why in this work, we propose a new methodology that semi-automatically integrates factual data into a dimension. This new design approach creates several challenges. Indeed, the output constellation schema must be:

- *well-formed*: dimensions are related to facts according to the multidimensional model;
- *usable*: the number of multidimensional elements (i.e. facts, dimensions, hierarchies, etc.) must be small [6];
- *coherent*: facts must be analyzed according to the correct new dimension data. For example, it would be incorrect to analyze sales in 1990 using transport data for 2010.

In conclusion, in some cases, the analytical needs expressed by the decision makers require the integration of factual data into a dimension. But this integration raises two problems:

- factual data have no hierarchical structure. Hierarchies must therefore be calculated with factual data before their integration into a dimension,
- a constellation schema can be represented as a set of interconnected facts and dimensions. This implies taking the multidimensional context of factual data into account during their integration into a dimension.

In this work, we propose a new semi-automatic design methodology integrating factual data into a dimension in a constellation multidimensional model.

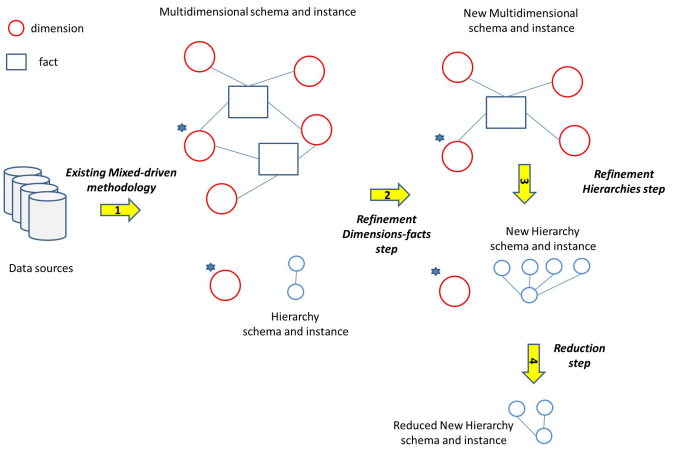


Fig. 4: Overview of our methodology

3 Overview of the Methodology

In this section, we present an overview of our methodology, which involves four main steps (Figure 4):

1) ***DERIVATION***: Using an existing mixed-driven (for example [7]) or data-driven design methodology a multidimensional model (schema and instance) is obtained.
2) ***REFINEMENT***: When the multidimensional model is a constellation schema in which a fact (for example "Transport", called source fact) can be used to enrich a dimension (for example "Products", called target dimension), *our refinement algorithm* (Algorithm 1) is applied. This algorithm transforms the multidimensional model to create a new well-formed and coherent multidimensional model (see Subsection 6 and [5]), where the source fact is eliminated, and the target dimension is duplicated. As shown in Figure 5, the product dimension is duplicated in two dimensions, the fact transport is eliminated with its company dimension.
3) ***CREATION OF NEW HIERARCHIES***: The target dimension is enriched with some new hierarchies calculated with source fact data (see Subsection 6 and Algorithm 2). For example, the new dimension "product(sales)" is enriched with several new hierarchies as shown in Figure 6.
4) ***REDUCTION IN THE NUMBER OF NEW HIERARCHIES***: When the number of the new hierarchies created in step 3 is so high, in regards of the usability of the multidimensional model becomes unusable, one or more of the reduction algorithms can be performed (see Section 7). These algorithms reduce the number of hierarchies of the dimension enriched in step 3.

4 Preliminaries

In this section, we provide some preliminary definitions and then describe a formalization of multidimensional models based on graphs. This formalization is then used throughout the paper.

In detail:

Definition 1 Multidimensional graph.

A multidimensional graph G is a directed graph $G = \langle D; F; A \rangle$ where:

- $D = \{d_1, ..d_\delta\}$ is a set of δ dimensional nodes d_i, which represent dimensions.
- $F = \{f_1, ..f_\zeta\}$ is a set of ζ fact nodes f_i, which represent facts.
- $A = \{a_1, ..a_\alpha\} \vee \forall i \in [1, \alpha], a = (f_j, d_k)$ with $j \in [1, \zeta] \wedge k \in [1, \delta]$ is a set of arcs a_i, from fact node to a dimensional node.
- G does not contains isolated nodes (i.e. a node without arcs).
- G can contain possibly disconnected sub-graphs.
- Each fact node f_i is associated with at least two different dimensional nodes.
- Each fact node f_i contains ν numerical attributes called measures $\{M_1^i, ..., M_\nu^i\}$ and each measure M_k^i is associated to an aggregation function called Ag_k^i.

Example 1

In Figure 1, we present the multidimensional graph associated with our motivating example, with factual nodes shown as blue squares and dimensional nodes as red circles.

Definition 2 Hierarchy.

A dimensional node contains at least one hierarchy H_i.
Each hierarchy is composed of a set of levels $\{L_1, ..., L_j\}$ that form a lattice.
The instance of a hierarchy is a tree of levels' members with the ALL member as the root of the tree.
We assume that all hierarchies in the multidimensional model are strict, balanced and, if a dimension contains several hierarchies, that these hierarchies are independently parallel (see Chapter 4 in [8] for a complete definition of hierarchies in a multidimensional schema).

Example 2

In Figure 2, we present the hierarchical graph associated with the "Products" dimension. Note that the lowest level of this dimension is the "Product" level. This hierarchical graph contains only one hierarchy, which groups the products according to category ("Category" level).

In the rest of this section, we formalize the concepts of the *target dimension* and *source fact* that are the inputs of our methodology.

Definition 3 Target dimension

The target dimension d_t of a multidimensional graph G is a dimension such as: $d_t \in D \wedge \exists (f_1, d_t), ..., (f_u, d_t)$ with $u \in [2, \zeta]$.
This means that d_t is associated with at least two facts. One of these facts, named "source fact", is used to create new hierarchies in the target dimension.

Example 3

In our multidimensional graph, we chose the "Products" dimension as the target dimension. Note that the "Products" dimension is associated with three facts: "Sales", "Stock" and "Transport" (see Figure 1).

The *source fact* is the fact that is used to enrich the *target dimension* and that is eliminated from the multidimensional model.

Definition 4 Source fact

The source fact f_s of a multidimensional graph G with a target dimension d_t is a fact node $f_s \in F \wedge \exists (f_s, d_t) \in A$.
This means that a source fact is a fact that is associated with the target dimension.

Example 4

If we chose "Products" as target dimension, "Sales", "Stock" and "Transport" can be chosen as source fact. In our case, we choose "Transport" as the source fact.

5 Creation of New Hierarchies

In this section, we present the main idea behind the creation of a hierarchy using a data mining algorithm as defined in Step 3 of the methodology "Creation of new hierarchies". The aim is to create a new hierarchy in the target dimension with data from the source fact, using an ascendant hierarchical clustering algorithm: we want perform the clustering algorithm on members of the level of d_t, described by the measures of f_s.

In particular, we use the *hierarchical agglomerative clustering* (more details in [9]). The results of a hierarchical agglomerative clustering can be shown as a tree that represents the distance between the individuals [10]. This tree can be considered as a hierarchy. The aim of this method is to build a hierarchy to find groups in the data. In hierarchical agglomerative clustering, each branch of the created hierarchy is a cluster. This method has several steps [11]. We describe them with respect to our case study using the data in Table 7:

1) Calculation of distances between products.
2) Choice of the two closest products.
3) Aggregation of the two closest products in a cluster. The cluster is now considered as a type of product.
4) Go back to the step 1 and loop if there is more than one product.

The result is shown in Figures 5 and 6. *In the present work, we used hierarchical agglomerative clustering, but any other algorithm that classifies data into a hierarchical tree structure can be used.*

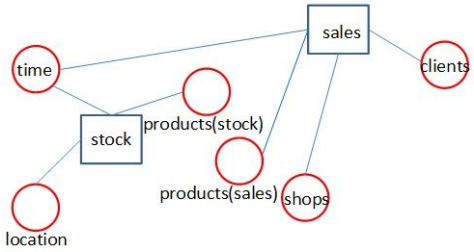
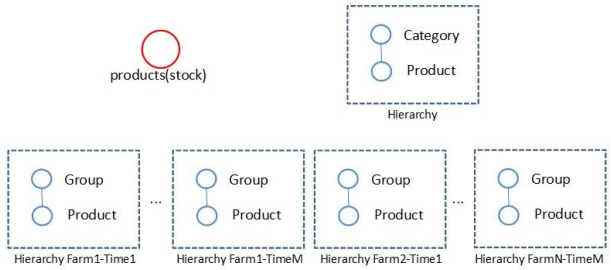
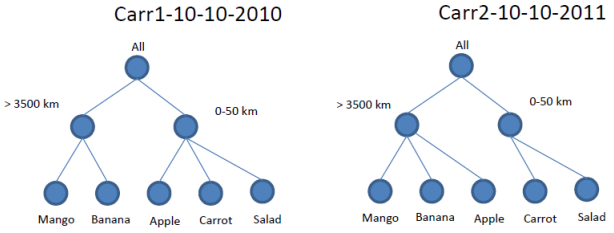


Fig. 5: The multidimensional graph associated with our motivating example after Algorithm 1



(a) Schema of the hierarchy



(b) Instance of the hierarchy

Fig. 6: The hierarchical structure of the new dimension "products(sales)

Let us formalize the data used by the hierarchical agglomerative clustering (i.e. the result of the query).

Definition 5 Query

Let G be a multidimensional graph. Let d_i be a dimension. Let $\{m_b^i \wedge b \in [1, \mu]\}$ the members of d_i.
Let f_z be a fact node with $\{M_1^z, ..., M_\nu^z\}$ the set of measures of this fact.
Then, the result of the Query f_z on $d_1.m_{k1}^1, d_j.*, ..., d_i.*, d_\delta.m_{k\delta}^\delta$ noted as Query $f_z(\{d_1.m_{k1}^1, d_j.*, ..., d_i.*, d_\delta.m_{k\delta}^\delta\})$ with $s \in [1, \nu]$ and $ki \in [1, \mu i]$, is the set of tuples representing facts of f_z aggregated with the members of the dimensions $\{d_1.m_{k1}^1, ..., d_\delta.m_{k\delta}^\delta\}$ for each member of a combination of the members of $d_i, ..., d_j$.

Example 5

An example of Query is:

$$Query(Transport, \{Products.[*], Location.[Jasper'sFarm], Shops.[Carr1], Time.[09/25/2010], Company.ALL\})$$

It corresponds to the distance of each product delivered by Jasper's Farm to "Carr1" shop on February 25, 2010. This result is presented in Table 7.

6 Refinement of dimensions-facts and hierarchies

In this section, we describe the refinement of the multidimensional model obtained in the DERIVATION step moving factual data from the source fact to the new hierarchies of the target dimension. Note that the target dimension and the source fact are chosen by the decision makers based on their analytical needs, since a multidimensional graph can present more candidate nodes as target and source facts. Before presenting the details of the proposed algorithm, we introduce some concepts.

Once the target dimension and the source fact are fixed, we group:

- dimensions in: *contextual dimensions*, and *non-contextual dimensions*.
- facts in: *contextual facts* and *non-contextual facts*.

Informally, a *contextual dimension* is a dimension that can pose problems after its elimination from the multidimensional model, leading to a not well-formed model, contrary to the non-contextual dimensions that can be removed without any problems arising. In other words, since the refinement algorithm eliminates the source fact and enriches the target dimensions, then the elimination of the source fact has consequences for the other dimensions and facts.

Formally,

Definition 6 Contextual fact

Let graph G, d_t be the target dimension and f_s the source fact
Then, $f_i \in F - \{f_s\} \vee \exists (f_i, d_t)$ is a contextual fact

Definition 7 Contextual dimensions

Let graph G, d_t be the target dimension and f_s the source fact Then, $f_i \in F - \{f_s\} \vee \exists (f_i, d_t)$ is a contextual fact
The context of f_i, called $C(f_i)$ is a set of dimensions
$C(f_i) = \{d_j \in D \vee d_j \neq d_t \wedge \exists (f_i, d_j) \wedge \exists (f_s, d_j)\}$
A dimension belonging to the context of a fact node $C(f_i)$ is denoted as contextual dimension.

Example 6

In our case study, we chose "Transport" as the source fact and "Products" as the target dimension. Therefore, sales and stock are contextual facts and contextual dimensions are: $C(sales) = \{time, shops\}$ and $C(stock) = \{time, location\}$.

In the following, we detail these properties of the multidimensional model that have to be granted by the transformation process.

Property 1 A well-formed multidimensional model

The multidimensional model obtained by removing the source fact must be a graph as defined in Definition 1.

Property 2 Coherent hierarchies

The target dimension can be associated with facts that differ from the source fact (i.e. contextual facts). Therefore, the hierarchies created on the target dimensions must be coherent with the dimensions of the contextual fact (i.e. contextual dimensions), since the analyses are conducted on the contextual facts.

Property 3 Coherent dimensions

Several contextual fact nodes can exist but cannot share the same contextual dimensions. Thus, the partitions of the source fact generated by these contexts are not the same, and consequently, the generated hierarchies are not the same.

Example 7

For example, let Table 1 be the Sales facts, and Table 2 the Transports facts.
The product carrot (the target dimension) has different distances (source fact) depending on the time and shops (contextual dimensions). Therefore, it is not logical, for example, to use the distance on September 20, 1990 (contextual dimension) to analyze sales (contextual fact) on September 19, 1989.
More specifically, each contextual fact must be analyzed according to a partition of the source fact, and this partition must be defined by the values of common dimensions. In our example,
for the context $\{9-19-90, Carr1\}$, carrot is analyzed using distance values : 1,245, 1,503;
for the context $\{9-19-90, Carr2\}$, carrot is analyzed using distance values : 12;
and so on.
The "company" dimension is not a contextual dimension (see Definition 7) because this dimension is not shared by the "transport" fact (the source fact) and another fact (see Figure 1). Consequently, there is no risk that inducted by the "company" dimension will cause incoherence. In fact, we do not need detailed data concerning this dimension: in the rest of this article, we aggregate facts for non-contextual dimensions.

TABLE 7: An example of the result of the query

Products.Product	Location.Farm	Shops.store	Time.day	Transport.All	Distance
Apple	Jasper's Farm	Carr1	9-25-2010	All	35
Carrots	Jasper's Farm	Carr1	9-25-2010	All	35
Banana	Jasper's Farm	Carr1	9-25-2010	All	35

In the rest of this section, we present the algorithms that implement the refinement step (Figure 4). Algorithm 1 has a multidimensional model as input and returns a coherent and well-formed multidimensional model, without the source fact, and the target dimension enriched with new hierarchies.

More specifically, the main steps of Algorithm 1 are the following:
The algorithm finds the contextual fact nodes (line 2).

For each contextual fact node f_i , the algorithm adds a new dimension d_{ti} to the multidimensional graph, which is a clone of the target dimension (line 3 in Algorithm 1).

This new dimension d_{ti} is then enriched with contextual hierarchies, deduced from the context of f_i (line 4 in Algorithm 1). The details of this step are presented in Algorithm 2. Next, the algorithm removes isolated nodes, and completes the relationships among the nodes. After which, the algorithm adds an arc between f_i and d_{ti}, and removes the arc that exists between f_i and d_i (lines 5 and 6 in Algorithm 1).

Finally, when all the fact nodes of the multidimensional graph have been treated, the multidimensional graph is cleaned (line 9 in Algorithm 1). This step comprises the following sub-steps:

- The source fact f_s is disconnected from its dimensions, including from the target dimension.
- The source fact is removed from the multidimensional graph.
- The algorithm performs a loop which ends when the multidimensional graph contains no isolated factual or dimensional node. During this loop, the algorithm visits each node: if this node is isolated, it from removed of the multidimensional graph.

In detail, the main steps of Algorithm 2 are the following:
First, the algorithm calculates all possible combinations of members of the contextual dimensions according to the contextual fact $C(f_i)$ (see Definition 6). For each combination (line 1 in Algorithm 2), the algorithm obtains a set of instances of the source fact f_s (line 2 in Algorithm 2). All other non-contextual dimensions are considered at the "ALL" level.

With this set of instances, named I, the algorithm creates the hierarchies as described in the previous section.

After which, the new hierarchy is integrated into d_{ti}, which is a clone of the target dimension. Consequently, d_{ti} has the same lowest level as the target dimension (line 4 in Algorithm 2).

input : G a multidimensional graph, $d_t \in G$ the target dimension, $f_s \in G$ the source fact.

1 **for** ***each*** *contextual fact node* f_i *in* G **do**
2 $d_{ti} = G.addDimension(d_t)$ – a new version of the target dimension is created
3 $G.addArc(f_i, d_{ti})$ – the dimension is linked to the fact
4 $d_{ti} = G.generateContextualHierarchies(d_t, f_s, f_i, d_{ti})$ – new hierarchies are created
5 $G.deleteArc(f_i, d_t)$
6 **end**
7 – clean the rest of the multidimensional graph
8 $G.deleteIsolatedNodes()$
9 $G.deleteArcsWithoutOneEnding()$
output: G

Algorithm 1: Main refinement algorithm

input : G a multidimensional graph, $d_t \in G$ the target dimension, $f_s \in G$ the source fact, f_i a factual node, d_{ti} a clone of the target dimension linked to f_i.

1 Let $\{dnc_1, ..., dnc_k\}$ be the non-contextual dimensions;
2 $C(f_i) = \{dc_1, ..., dc_n\}$ the contextual dimensions;
3 $Combinations = findCombinations(C(f_i))$ – the list of all possible combinations of lowest levels' members of dimensions in $C(f_i)$
4 **for** ***each*** $c = (dc_1.m_{j1}, dc_2.m_{j2}, ..., dc_n.m_{jn})$ *in* $Combinations$ **do**
5 $I = Query(f_s, d_t.*, dc_1.m_{j1}, ..., dc_n.m_{jn}, dnc_1.ALL, ..., dnc_k.ALL);$ – create partitions
6 $H_s = calculateHierarchies(I)$; – apply the data mining method to create hierarchies
7 $d_{ti}.addHierarchies(H_s)$;
8 **end**
output: G

Algorithm 2: generateContextualHierarchies algorithm

In the following example, we describe the overall refinement step preformed by Algorithm 1 on our case study taking the multidimensional model in Figures 1 and 2 as input, and obtaining the multidimensional model in Figures 5 and 6 as result.

Example 8

Algorithm 1 starts with the graph from Figure 1, with f_s = 'Transports' and d_t = 'Products'. With these inputs, the contextual fact nodes are "stock" and "sales". The algorithm generates two new dimensions "products(stock)" and "products(sales)".
Algorithm 2 is applied to these two dimensions. The "Transports" fact is deleted, together with its arcs.
Let us consider "product(sales)" : we apply the *generateContextualHierarchies* algorithm (Algorithm 2) with inputs: this dimension, f_s = 'Transports' and d_t = 'Products', f_i = 'Sales' and d_{ti} = 'product(sales)'. In this situation, "company" is the non-contextual dimension of sales, and "time" and "shops" are the contextual dimensions. The algorithm applies *Query* on Transports (see Example 5 and Table 7). With these data (see Table 8), the hierarchical agglomerative clustering algorithm is applied to define a new hierarchy.

The proposed algorithms grant the coherence property for hierarchies and dimensions as follows:

Coherent hierarchies: Coherent hierarchies: The coherence of the hierarchies is achieved by dividing the source fact into several subsets (a Query result), one per combination of contextual dimension members. Each subset is used to calculate a new hierarchy in the target dimension (see Table 8).

Coherent dimensions: The dimension's coherence property is also granted. Indeed, in our approach, for each contextual fact node, a version of the target dimension is created. For example, in Figures 5 and 6, we present the multidimensional graph associated with our motivating example Algorithm 1. Note that the "Products" dimension has been split into two new dimensions: "products(stock)" and "products(sales)".

Well-formed multidimensional model: The algorithm grants the well-formed property. Indeed, using these restructuring policies, the final multidimensional model presents the hierarchies and dimensions previously described, and the source fact (e.g. "Transport") has been removed of the multidimensional graph. Finally, the dimensions (e.g. "Company"), that are linked only to the source fact (e.g. "Transport") have been removed to avoid isolated nodes in the multidimensional graph.

In conclusion, the refinement step makes it possible to reorganize the multidimensional model by removing the source fact and to obtain a new well-formed multidimensional model. Moreover, it prepares the inputs for the the hierarchies step, in which the new hierarchies of the target dimension are created.

7 Reduction in the number of new hierarchies

In this section, we provide all details for the reduction step of the proposed methodology (see Figure 4). We explain the aims of this step and the three methods proposed to achieve these aims.

A context $C(f_i)$ can contain many dimensions, each of which comprises many members. The number of instances of a context can therefore be huge. A huge number of instances of a context means a huge number of calculated hierarchies.

Thus, if many instances are related to a particular context, the number of instances in this context needs to be reduced in order to ensure the multidimensional model is usable. Indeed, the OLAP decision-making process is an exploration process with which the decision maker navigates the warehoused data by using a set of hierarchies as entry point. A very large number of hierarchies thus makes the data warehouse not suitable for this kind of decisional process. Therefore in our proposal, the refined multidimensional graph has to be more "usable". To reduce the number of instances of a context $C(f_i)$, different strategies can be used based on the schema of the multidimensional model and on the warehoused data:

1) schema of the multidimensional model
   - A Roll-Up operation performed on some dimensions
   - Clustering of calculated hierarchies.
2) warehoused data
   - Clustering of source facts.

We define these strategies in the following subsections.

7.1 Roll-Up of contextual dimensions

A Roll-up operation can be performed in the source fact f_s on dimensions levels in $C(f_i)$. This Roll-up operation moves data from the lowest level members to members of less detailed levels. Since hierarchy data are structured as a tree, the coarser the level is and the fewer the members are (for example see Figure 2b). Therefore, this strategy reduces the number of context instances by reducing the number of dimension members from $C(f_i)$, and hence the number of rows made up of dimension members from $C(f_i)$.

The choice of the Roll-up dimension depends on the analytical needs of the decision makers.

Formally, line 3 of Algorithm 2 can be replaced by:

Definition 8 Roll-up of contextual dimensions

$Combinations = findCombinations(C(f_i))$ – the list of all possible combinations of levels' members of dimensions in $C(f_i)$

TABLE 8: Instances of "Transport" used to generate hierarchies with "Products" lowest members with a partition by "Shop" members

Contextual dimensions		**Target dimension**	**Source fact**	**Hierarchies**
Shops.shop	Time.day	Products.products	Distances (km)	
Carr1	10-10-2010	Pineapples	12.6	1^{st} Hierarchy
		Carrots	2	
		Bananas	55.1	
		...	...	
Carr2	10-11-2010	Pineapples	13	2^{nd} Hierarchy
		Carrots	10.9	
		Bananas	23.7	
		...	...	
...	...	...	...	...
Carr1	10-13-2010	Pineapples	12.6	3^{rd} Hierarchy
		Carrots	8	
		Bananas	3,050	
		...	...	

TABLE 9: Aggregated data along the "shops" dimension.

Contexual dimensions		**Target dimension**	**Source fact**	**Hierarchies**
Shops.All	Time.year	Products.products	Distances (km)	
ALL	2010	Pineapples	38.2	1^{st} Hierarchy
	2010	Carrots	20.9	
	2010	Bananas	3,128.8	

Example 9

For example, in order to reduce members, the decision makers can decide to use yearly data instead of daily data. In this case, line 4 of Algorithm 2 will be : **For each** c in $\{(ALL, 2010), (Carr1, 2010), (Carr2, 2010), ...\}$, a combination of members of dimension "Time", roll-up at the level "year" and all the members of the dimension "Shops".
As an example, Table 9 presents data from Table 8 after a Roll-up.

At the conceptual level, the multidimensional schema is well formed, because this operation does not affect the global structure of the multidimensional schema: it just reduces the number of levels in a hierarchy.

At the physical level, the data are not affected by this Roll-Up operation. In fact, the algorithm does not physically delete the lower levels, but just declares them "disabled".

The second question is "How to manage the granularity in the fact table f_i related to the context $C(f_i)$?". In fact, if the granularity of the context changes, the granularity of f_i should change too, to avoid inconsistencies in OLAP queries.

The simplest solution is to link f_i and f_s to the second level of the dimension considered in the Roll-Up operation. As an example, in Figure 5, "Sales" and "Products" (which is the source fact) will be linked to the "Month" level of the time dimension rather than to the "Day" level. In this case, the granularity of f_i and the granularity of f_s are coherent.

7.2 Clustering source facts

This approach corresponds to "data" reduction: in this approach, we do not consider a conceptual structure, but define a mathematical proximity between multidimensional data. We then propose a reduction based on the clustering of source facts.

The main idea is grouping similar facts originating from the source facts and only using data for the one representative fact in each cluster. In this way, the number of source facts decreases and, the number of context instances decreases too [12].

To perform this reduction, we use the k-medoids algorithm over a Query (we recall that a Query represents aggregated facts data) [13].

The k-medoids algorithm is related to the k-means algorithm and the medoidshift algorithm. K-means tends to put a data set into groups and attempts to minimize the distance between the data points in a group and the centroid of the group. K-medoids uses the same general working process. The main difference is that k-means calculates coordinates of the centroid of each group. This centroid is an artificial data point. K-medoids uses a real data point (named "medoid") as centroid, the closest to the calculated centroid. Once clusters have been obtained, only the medoids are used in the creation of hierarchies. Formally,

Definition 9 Clustering source facts

Let $Clustering(Query\ q)$ be the reduction method that is applied to a Query q, then line 5 of the Algorithm 2 is $H_s = calculateHierarchy(Clustering(I))$

TABLE 10: Instances of "Transport" used to generate hierarchies with "Products" lowest members with a partition by "Shop" members, with medoids and clusters

Cluster	Contextual dimensions		Target dimension	Source fact	Hierarchies
	Shops.shop	Time.day	Products.products	Distances (km)	
MEDOID of Cluster1	Carr1	10-10-2010	Pineapples	12.6	1^{st} Hierarchy
			Carrots	2	
			Bananas	55.1	
			...	...	
Cluster1	Carr2	10-11-2010	Pineapples	13	NONE
			Carrots	10.9	
			Bananas	23.7	
			...	...	
...	...	...	...	...	...
MEDOID of Cluster2	Carr1	10-13-2010	Pineapples	12.6	2^{nd} Hierarchy
			Carrots	8	
			Bananas	3,050	
			...	...	

Example 10

As an example, we present Table 10. This table shows that the hierarchies are calculated only for identified medoids. In this example, two hierarchies are calculated instead of three. For example, for cluster 1, the new hierarchy is built only using facts associated with its medoid.

7.3 Clustering calculated hierarchies

Finally, the number of calculated hierarchies can be reduced *a posteriori*.

First, each calculated hierarchy is represented in its canonical form. After which, the proposed prototype identifies hierarchies that are strictly identical, and groups them together.

"Group hierarchies together" means associate several context instances with one hierarchy.

Formally,

Definition 10 Clustering calculated hierarchies

Let $Clustering(H_s)$ be the reduction method that is applied to a set of calculated hierarchies, named H_s, then line 5 of Algorithm 2 is
$H_s = Clustering(calculateHierarchy(I))$

Example 11

As an example, the hierarchies in Figure 6b are not similar so all the hierarchies are used.

8 Implementation

In this section, we describe the system, Schema Enricher (SE), which implements our methodology. Schema Enricher is a tool coupled with a Relational OLAP architecture composed of

- the relational DBMS PostgreSQL, which stores data
- the Mondrian OLAP server, computes OLAP queries and contains the XML definitions of the multidimensional models.

SE is provided with a simple user interface that allows users to (i) define the inputs of the methodology (i.e. the target dimension and the source fact); and (ii) choose the hierarchy reduction algorithm.

An example of the user interface is shown in Figure 7. After downloading the schema and the warehoused data, the user can choose a source fact and a target dimension. Then SE calculates the contextual hierarchies (see Section 3), and finally the user can choose how to reduce the number of contextual hierarchies using one (or more) of three methods proposed: Roll-up of contextual dimensions, Clustering source facts or Clustering contextual hierarchies (see Section 7).

SE also provides an overview of the multidimensional schema for the user in the form of a graph.

The database structure of the DW does not change, but the prototype changes the XML Mondrian file representing the multidimensional model. An example is shown in Figure 9.

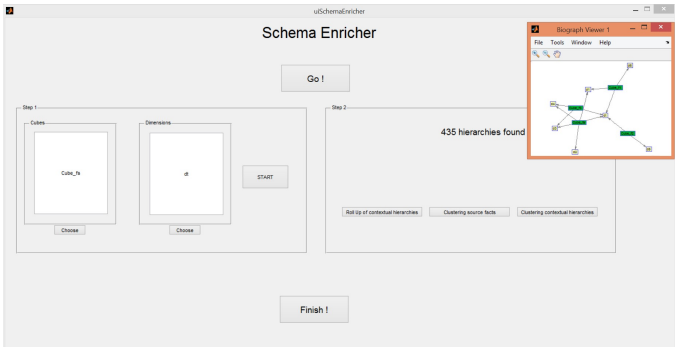


Fig. 7: A screen shot of the developed prototype

9 Experiments

In this section, we present the time and usability performances of our methodology. We applied the method to the de-facto standard for OLAP applications TPC-DS benchmark. TPC-DS is a benchmark for big DWs. We chose it to validate our approach in the context of big data. We also validated our proposal on a real DW concerning biodiversity.

9.1 Settings

Our experiment was performed in the following technical framework:

- Hardware: Intel®Core™i7-4170HQ CPU 2.50 GHz, RAM: 8Go
- Operating system: Windows 8.1
- Data Base Management System: PostgreSQL 9.4 [1]
- OLAP server: Mondrian[2]

9.2 Big Data Warehouse benchmark

Table 11 lists data related to the size of the data warehouse generated with the TPC-DS tool.

For our evaluation, as inputs for the method, we used:

- the "date_dim" dimension as target dimension and
- the "inventory" fact as source fact.

We provided an extract of the TPC-DS benchmark schema as input, centered on the "inventory" fact (see Figure 8a). The output schema of the methodology is shown in Figure 8b.

In particular, the chosen source fact, the "inventory" fact, is associated with three dimensions: "date_dim" (the target dimension), "item" and "warehouse". The chosen target dimension, the "date_dim" dimension, is associated with seven facts: "inventory" (the source fact), "catalog_sales", "catalog_returns", "store_sales", "store_returns", "web_sales" and "web_returns". The source fact shares dimensions with the other six fact nodes. For this situation, we detail the contexts of each fact in the multidimensional graph:

$$
\begin{aligned}
C(catalog\_sales) &= \{item, warehouse\} \\
C(catalog\_returns) &= \{item, warehouse\} \\
C(web\_sales) &= \{item, warehouse\} \\
C(web\_returns) &= \{item, warehouse\} \\
C(store\_sales) &= \{item\} \\
C(store\_returns) &= \{item\}
\end{aligned}
$$

We note that:

$$
\begin{aligned}
C(catalog\_sales) &= C(catalog\_returns) \\
&= C(web\_sales) \\
&= C(web\_returns)
\end{aligned}
$$

$$
C(store\_sales) = C(store\_returns)
$$

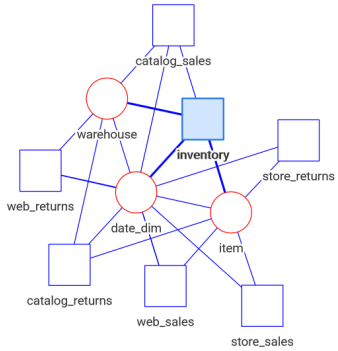
The prototype identified two contexts, i.e. two versions of the enriched target dimension: the first context $\{item, warehouse\}$ is associated with four dimensions, and the second context $\{item\}$ is associated with two dimensions.

After this identification step, the prototype processes the context instances to build hierarchies.

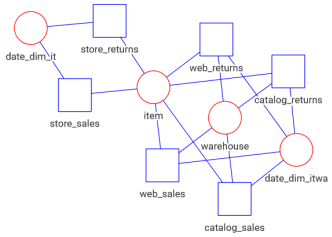
[1]https://www.postgresql.org/
[2]http://community.pentaho.com/projects/mondrian/

TABLE 11: Volume of dimensions and facts, in the TPC-DS data warehouse, used for our experiments

Type	Name	Number of records
Dimension	warehouse	10 members
Dimension	item	73,148 members
Dimension	date_dim	1,140 members
Fact	web_returns	71,763 facts
Fact	catalog_returns	144,067 facts
Fact	store_returns	260,664 facts
Fact	web_sales	629,880 facts
Fact	catalog_sales	1,441,530 facts
Fact	store_sales	2,648,920 facts
Fact	inventory	11,075,400 facts



(a) Before



(b) After

Blue squares: factual nodes; red circles: dimensional nodes

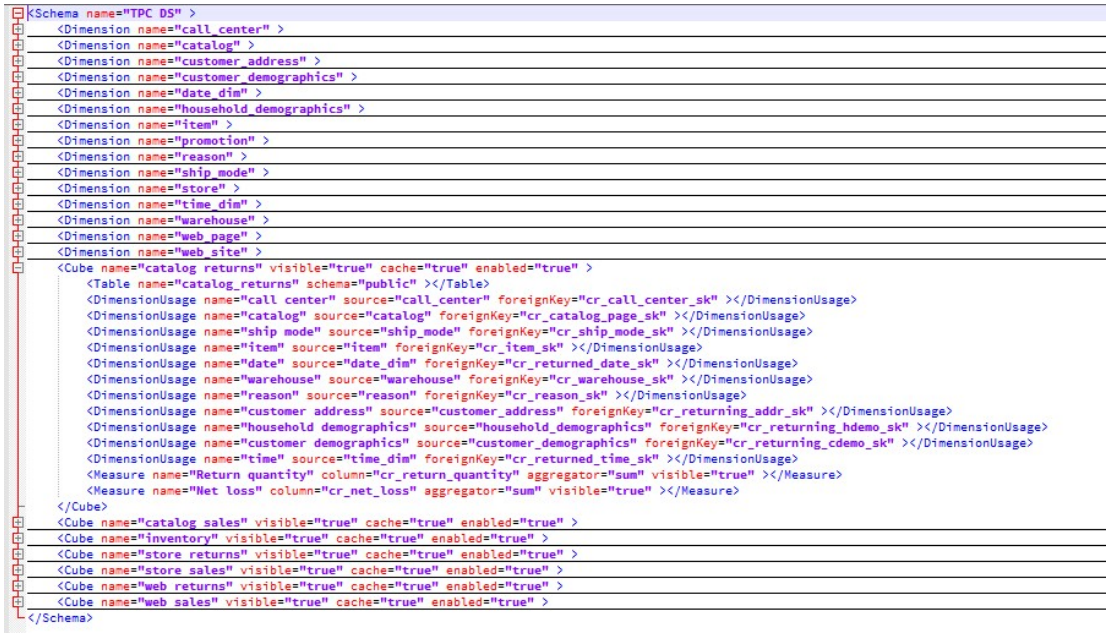
Fig. 8: An extract of the multidimensional graph of the TPC-DS benchmark, centered on the "inventory" fact, before and after the work of the prototype

After identification of the contexts and calculation of new hierarchies, the prototype modifies the multidimensional graph to add the two new dimensions: "date_dim_it" and "date_dim_itwa", which corresponds to the contexts $\{item\}$, and $\{item, warehouse\}$ respectively. The "inventory" fact is deleted by the prototype. Figure 8b shows an extract of the multidimensional graph of the TPC-DS benchmark, modified by the prototype.

Figure 9 presents, at a physical level, the conceptual modifications of the multidimensional schema shown in Figure 8. In Figure 9b, two new dimensions, "context_item" and "context_item_warehouse" have been created, and are associated with cubes. As an example, "context_item_warehouse" is used in the "catalog returns" cube. The "context_item_warehouse" contains 20 calculated hierarchies.

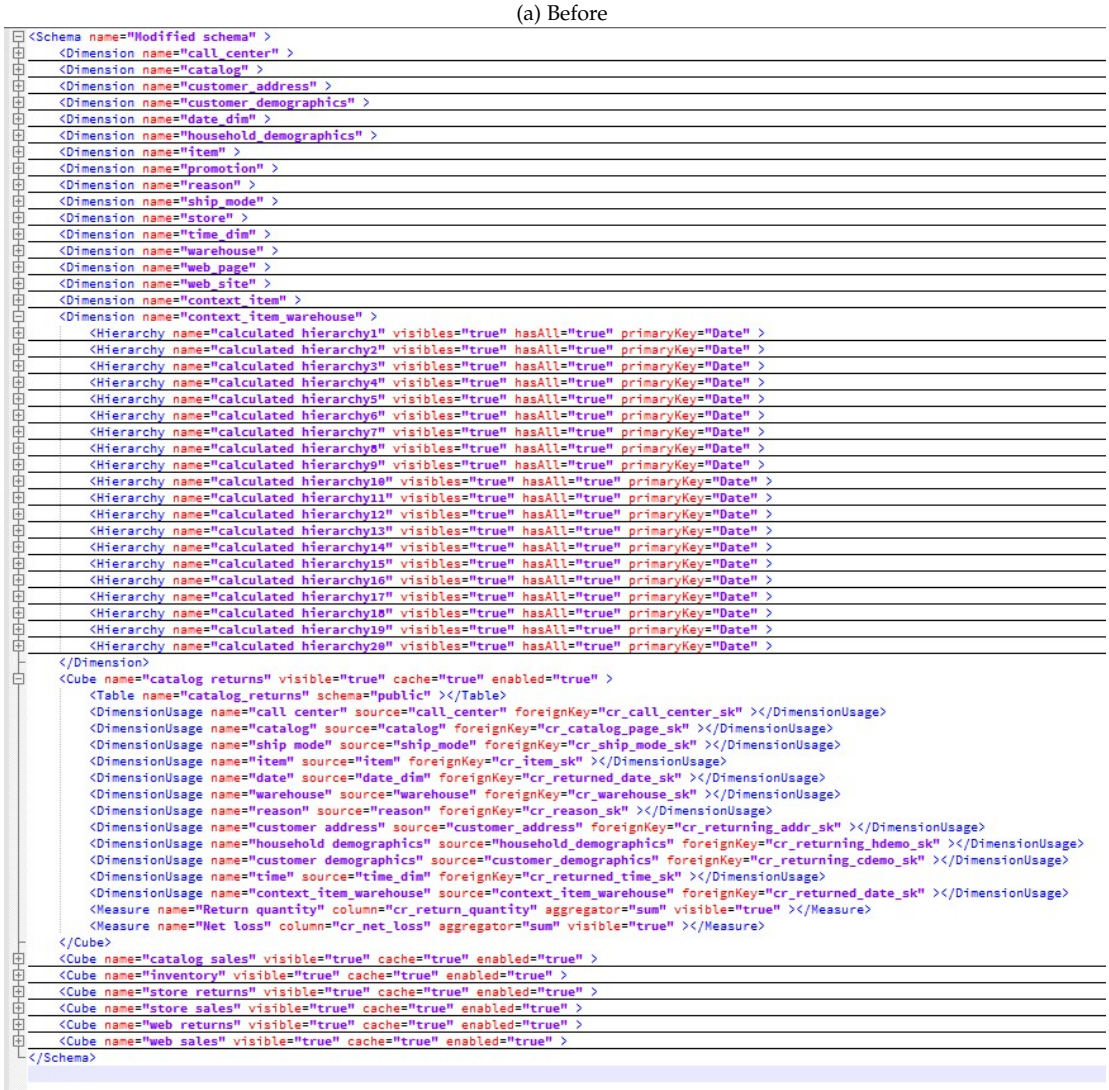
9.2.1 Results: execution time

Table 12 shows the execution times of the prototype measured on the TPC-DS benchmark.



```
<Schema name="TPC DS" >
    <Dimension name="call_center" >
    <Dimension name="catalog" >
    <Dimension name="customer_address" >
    <Dimension name="customer_demographics" >
    <Dimension name="date_dim" >
    <Dimension name="household_demographics" >
    <Dimension name="item" >
    <Dimension name="promotion" >
    <Dimension name="reason" >
    <Dimension name="ship_mode" >
    <Dimension name="store" >
    <Dimension name="time_dim" >
    <Dimension name="warehouse" >
    <Dimension name="web_page" >
    <Dimension name="web_site" >
    <Cube name="catalog returns" visible="true" cache="true" enabled="true" >
        <Table name="catalog_returns" schema="public" ></Table>
        <DimensionUsage name="call center" source="call_center" foreignKey="cr_call_center_sk" ></DimensionUsage>
        <DimensionUsage name="catalog" source="catalog" foreignKey="cr_catalog_page_sk" ></DimensionUsage>
        <DimensionUsage name="ship mode" source="ship_mode" foreignKey="cr_ship_mode_sk" ></DimensionUsage>
        <DimensionUsage name="item" source="item" foreignKey="cr_item_sk" ></DimensionUsage>
        <DimensionUsage name="date" source="date_dim" foreignKey="cr_returned_date_sk" ></DimensionUsage>
        <DimensionUsage name="warehouse" source="warehouse" foreignKey="cr_warehouse_sk" ></DimensionUsage>
        <DimensionUsage name="reason" source="reason" foreignKey="cr_reason_sk" ></DimensionUsage>
        <DimensionUsage name="customer address" source="customer_address" foreignKey="cr_returning_addr_sk" ></DimensionUsage>
        <DimensionUsage name="household demographics" source="household_demographics" foreignKey="cr_returning_hdemo_sk" ></DimensionUsage>
        <DimensionUsage name="customer demographics" source="customer_demographics" foreignKey="cr_returning_cdemo_sk" ></DimensionUsage>
        <DimensionUsage name="time" source="time_dim" foreignKey="cr_returned_time_sk" ></DimensionUsage>
        <Measure name="Return quantity" column="cr_return_quantity" aggregator="sum" visible="true" ></Measure>
        <Measure name="Net loss" column="cr_net_loss" aggregator="sum" visible="true" ></Measure>
    </Cube>
    <Cube name="catalog sales" visible="true" cache="true" enabled="true" >
    <Cube name="inventory" visible="true" cache="true" enabled="true" >
    <Cube name="store returns" visible="true" cache="true" enabled="true" >
    <Cube name="store sales" visible="true" cache="true" enabled="true" >
    <Cube name="web returns" visible="true" cache="true" enabled="true" >
    <Cube name="web sales" visible="true" cache="true" enabled="true" >
</Schema>
```

(a) Before



```
<Schema name="Modified schema" >
    <Dimension name="call_center" >
    <Dimension name="catalog" >
    <Dimension name="customer_address" >
    <Dimension name="customer_demographics" >
    <Dimension name="date_dim" >
    <Dimension name="household_demographics" >
    <Dimension name="item" >
    <Dimension name="promotion" >
    <Dimension name="reason" >
    <Dimension name="ship_mode" >
    <Dimension name="store" >
    <Dimension name="time_dim" >
    <Dimension name="warehouse" >
    <Dimension name="web_page" >
    <Dimension name="web_site" >
    <Dimension name="context_item" >
    <Dimension name="context_item_warehouse" >
        <Hierarchy name="calculated hierarchy1" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy2" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy3" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy4" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy5" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy6" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy7" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy8" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy9" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy10" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy11" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy12" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy13" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy14" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy15" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy16" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy17" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy18" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy19" visibles="true" hasAll="true" primaryKey="Date" >
        <Hierarchy name="calculated hierarchy20" visibles="true" hasAll="true" primaryKey="Date" >
    </Dimension>
    <Cube name="catalog returns" visible="true" cache="true" enabled="true" >
        <Table name="catalog_returns" schema="public" ></Table>
        <DimensionUsage name="call center" source="call_center" foreignKey="cr_call_center_sk" ></DimensionUsage>
        <DimensionUsage name="catalog" source="catalog" foreignKey="cr_catalog_page_sk" ></DimensionUsage>
        <DimensionUsage name="ship mode" source="ship_mode" foreignKey="cr_ship_mode_sk" ></DimensionUsage>
        <DimensionUsage name="item" source="item" foreignKey="cr_item_sk" ></DimensionUsage>
        <DimensionUsage name="date" source="date_dim" foreignKey="cr_returned_date_sk" ></DimensionUsage>
        <DimensionUsage name="warehouse" source="warehouse" foreignKey="cr_warehouse_sk" ></DimensionUsage>
        <DimensionUsage name="reason" source="reason" foreignKey="cr_reason_sk" ></DimensionUsage>
        <DimensionUsage name="customer address" source="customer_address" foreignKey="cr_returning_addr_sk" ></DimensionUsage>
        <DimensionUsage name="household demographics" source="household_demographics" foreignKey="cr_returning_hdemo_sk" ></DimensionUsage>
        <DimensionUsage name="customer demographics" source="customer_demographics" foreignKey="cr_returning_cdemo_sk" ></DimensionUsage>
        <DimensionUsage name="time" source="time_dim" foreignKey="cr_returned_time_sk" ></DimensionUsage>
        <DimensionUsage name="context_item_warehouse" source="context_item_warehouse" foreignKey="cr_returned_date_sk" ></DimensionUsage>
        <Measure name="Return quantity" column="cr_return_quantity" aggregator="sum" visible="true" ></Measure>
        <Measure name="Net loss" column="cr_net_loss" aggregator="sum" visible="true" ></Measure>
    </Cube>
    <Cube name="catalog sales" visible="true" cache="true" enabled="true" >
    <Cube name="inventory" visible="true" cache="true" enabled="true" >
    <Cube name="store returns" visible="true" cache="true" enabled="true" >
    <Cube name="store sales" visible="true" cache="true" enabled="true" >
    <Cube name="web returns" visible="true" cache="true" enabled="true" >
    <Cube name="web sales" visible="true" cache="true" enabled="true" >
</Schema>
```

(b) After

Fig. 9: The physical multidimensional schema of the TPC-DS benchmark before and after the work of the prototype

As mentioned in Table 12, the total execution time of the main algorithm with the proposed prototype is on average 32,957.2 seconds i.e. 9h 9min and 17s. This execution time is compatible with an off-line execution on an OLAP server. The execution times of the reduction steps are similar. The steps of the creation of the new hierarchies creation are longer, since they execute queries on a large dataset. Thus, *we conclude that even with big data sets such as the TPC-DS, in an off-line process, the execution time of our methodology is feasible.*

9.2.2 Results: number of processed context instances

Table 12 also shows the number of context instances (i.e. new hierarchies calculated by the prototype) after each step, measured on the TPC-DS benchmark. The performance parameter evaluated is the number of processed context instances, which corresponds to the number of new hierarchies integrated in the data warehouse.

In the first step, the prototype identified 804,628 context instances, because we used dimensions with large number of members. Of course, a multidimensional schema with more than 800,000 hierarchies is not usable for a decision maker. Therefore, to reduce this number, we applied a cascade of the different reduction methods.

For the first hierarchy reduction method, we performed a Roll-up of the contextual hierarchies (see Subsection 7.1), obtaining 7,386 context instances. This number of context instances was still too high with respect to the usability of the multidimensional schema. We consequently performed the third implemented hierarchy reduction method: clustering of the calculated hierarchies (see Subsection 7.3). After execution of the 7,386 queries that represent the context instances, we obtained 7386 data tables.

Next, a hierarchy was calculated with each data table. We then subjected the hierarchies to hierarchy clustering. This method gathers identical hierarchies in clusters.

After this step, 2,462 context instances were still available which is still too many for a usable multidimensional schema. We thus performed the second implemented hierarchy reduction method: clustering of source facts (see Subsection 7.2). For this step, we chose a small number of clusters ($k = 20$). After this step, we obtained 40 hierarchies, i.e. 20 hierarchies for each new dimension (see Section 6 for the number of new dimensions). As described above, the number of hierarchies obtained depends on the size of the contextual dimensions. In this example, it appears to be huge because we used the TPC-DS big data warehouse. It is also interesting to note that the reduction methods can be applied in an iterative way to reach a "usable" number of hierarchies.

9.2.3 Discussion

As described in the preceding subsections, the execution time and the number of hierarchies remain feasible even when the methodology is applied to a big data warehouse. However, the choice of the hierarchy reduction algorithm needs to be discussed, since the algorithms lead to different hierarchies and hence to different OLAP analyses.

Therefore, in Table 13, we summarize the features of methods to reduce the number of context instances that affect the quality of the resulting multidimensional model.

Each method has typical features. First, concerning the information used to select the calculated hierarchies: the "Roll-up of contextual dimensions" method is based on the multidimensional schema to select context instances and, incidentally the future calculated hierarchies. The "clustering of source facts" method uses the proximity between source facts to select context instances, and consequently, future calculated hierarchies. The "clustering of calculated hierarchies" uses the proximity between structures of the calculated hierarchies to select them.

Second, one of the three methods has an impact on the multidimensional schema. The "Roll-up of contextual dimensions" method changes the granularity of facts. The two other methods have no impact on the multidimensional schema.

Finally, the decision makers may have a different understanding of how each method works. The "Roll-up of contextual dimensions" method allows decision makers to control the process, whereas, with the two other methods, the decision makers cannot control the process.

The "clustering of source facts" strategy can be used by decision makers who are tolerant of approximations, as this strategy represents a group of combinations of dimension members, described by factual data, by the medoid of this group, i.e. the combination closest to the group center. Conversely, the "clustering of calculated hierarchies" strategy directly compares the calculated hierarchies, and group hierarchies with a similar structure together: there is no approximation in the grouping process.

In conclusion, with the three hierarchy reduction methods, we reduced the number of new calculated hierarchies from 804,628 to 40.

A question one might ask is why using three methods, when the k-medoids method can dramatically reduce the number of hierarchies?

First, it should be noted that using k-medoid means having data available from the source facts to perform the clustering algorithm. However, as mentioned in subsection 9.2.1, the execution of queries that retrieve the necessary data from the data warehouse is the most time consuming task run by the proposed prototype. The Roll-Up of contextual hierarchies reduces the number of queries that will be executed, and hence, the execution time of the prototype.

Second, the k-medoid algorithm produces an "imprecise" result: each cluster is represented by its medoid, but each element of the cluster does not have exactly the same properties as the medoid, whereas the hierarchy clustering method groups hierarchies that are identical.

That is why, in our opinion, the method based on k-medoids should be used as a last resort, when the Roll-Up of contextual hierarchies does not reduce the number of context instances sufficiently, to ensure the multidimensional schema is usable.

TABLE 12: Results of performance tests: Number of processed context instances

Step	Description	Number of context instances	Execution time
CREATION OF NEW HIERARCHIES	Initialize and build queries in order to load data from the data warehouse	804,628	659
REDUCTION IN THE NUMBER OF NEW HIERARCHIES with *Roll-up of contextual dimensions*	Execute Roll-Up queries	7,386	3.0
CREATION OF NEW HIERARCHIES	Initialize and build queries in order to load data from the data warehouse	7,386	32,186.2
REDUCTION IN THE NUMBER OF NEW HIERARCHIES with *Clustering of calculated hierarchies*	Execute clustering on hierarchies	2,462	82
REDUCTION IN THE NUMBER OF NEW HIERARCHIES with *Clustering of source facts*	Execute K-medoids on factual data	40	4.9
REFINEMENT	Update data warehouse and multidimensional cube with new hierarchies	40	20.8

TABLE 13: Comparison of methods to reduce the number of contextual hierarchies

Reduction strategy	Selection of hierarchies depends on ...	Schema	Awareness
Roll-up of contextual dimensions	Schema of contextual hierarchies	Granularities of facts changes	The decision maker is aware of which data are left and does not tolerate imprecision
Clustering of source facts	Factual data in source fact	Schema does not change[1]	The decision maker is not aware of which data are left but tolerates imprecision
Clustering of calculated hierarchies	Schema of calculated hierarchies	Schema does not change[1]	The decision maker is not aware of which data are left and does not tolerate imprecision

[1]: Except the target dimension and the source fact, which are modified in all cases.

9.3 Real life DW

In this section, we validate our proposal on a real DW concerning biodiversity. The original multidimensional model is presented in Figure 10.

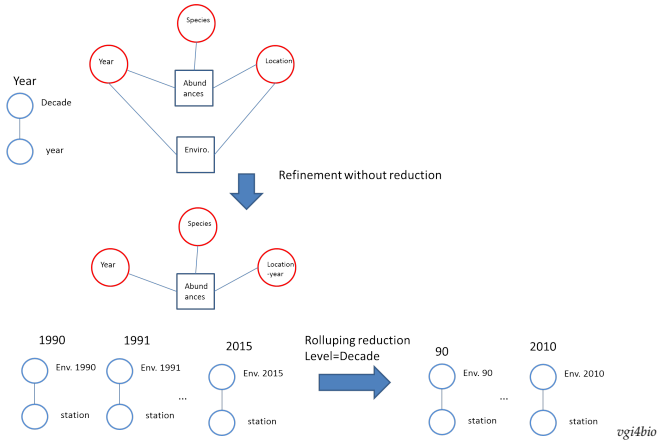


Fig. 10: biodiversity DW

It contains two facts:

- one representing the abundance of species according to the year, the species and the location;
- the other representing the environmental characteristics of the location dimension according to the year (the land use, the altitude, etc.).

Ecologists want to analyze abundances at a given location as a function of the type of environment. Therefore, we apply the methodology using the enviro. fact as source fact and the location as the target dimension. The methodology then removes the enviro. fact and adds 25 hierarchies to the location dimension (one per year). Since 25 hierarchies are too many for our decision makers, we provide the Roll-up reduction method using the "Decade" level of the "Time" dimension. Then, the prototype creates only three new hierarchies (one per decade).

We do not give the execution times here as they are negligible because the DW contains only a few thousand of rows.

This biodiversity case study confirms the feasibility of our methodology in real-life applications in terms of execution time and usability.

10 Related Work

In this section, we review the literature on the topics covered in this article. First, we focus on the integration of data mining processes into OLAP systems. Second, we present articles related to the automatic building of hierarchies and dimensions. Third, we review the literature

about versions in data warehousing. Finally, we draw some conclusions.

Mining warehoused data. Many authors studied the integration of data mining tools into OLAP systems in the last 20 years.

The concept of OLAP Mining (OLAM) was introduced in 1998 by Han [14]. The main contribution of this work is identifying the need for data mining methodologies and algorithms compatible with multidimensional data, and of data mining tools integrated in OLAP systems. These methodologies, algorithms and tools are used in OLAM systems to analyze multidimensional data.

The OLAM systems proposed in the literature can be classified according to the data mining algorithms used to perform analysis on multidimensional data. Several articles offer methodologies and tools that deduce association rules from an OLAP cube: [15], [16], [17], [18], [19], [20]. Other authors provide ways to integrate supervised classification [21] or prediction [22], [23] algorithms into an OLAP system. Clustering algorithms have also been implemented in OLAP systems to group facts with similar values [18], [24], [25], [26]). Note that some contributions integrate different data mining tools, e.g. [27] (several auto-regression methods), [18] (discovery of association rules and clustering algorithm) or [26] (clustering algorithm and principal component analysis). Finally, some authors' contributions guide the choice of one data mining method among the existing methods depending on the business case concerned [28].

These articles suggest mining multidimensional data described by an existing multidimensional schema, whereas other contributions propose using data mining algorithms to build or to update the multidimensional model. As an example, Eder et al. propose using auto-regressive methods to detect structural changes in warehoused data [27].

Automatic building of hierarchies or dimensions. Many authors provide methods to build new hierarchies or dimensions in an OLAP cube with data mining algorithms such as clustering algorithms [2], [25], [29], [30], [31], association rules [32], [33] or time serie analysis [34].

Concerning automatic building of hierarchies or dimensions, some authors provide methodologies that can work on data with no hierarchical structure, continuous data for example [4], [35] or social network data [36], [37]. Other approaches are centered around user requirements [3].

Data mining algorithms are thus used in different articles to analyze multidimensional data or to enrich multidimensional models. But all these works are designed and tested on only one hypercube.

Versioning in a data warehouse. Concerning multiversion data warehouses, several articles offer conceptual definitions and models associated with temporal and multi-version data warehouses [38], [39]. Some authors studied the constraints associated with versioning of a data warehouse [40], [41]. There is an abundant literature on versioning for warehoused data, but, in these works, the versions are always related to the timelapse. Furthermore, to the best of our knowledge, no article provides a methodology to automatically calculate versions of a hierarchy in a multidimensional model.

Conclusion on related work. An abundant literature exists on data mining processes integrated in OLAP systems. These works offer, *inter alia*, to automatically build new hierarchies or dimensions in an OLAP cube. However, these works are limited to one hypercube and are not applied to data warehouses with a constellation schema. Otherwise, multi-version data warehouses and OLAM systems are rarely combined in the same contribution.

CONCLUSION

We offer a theoretical methodology to refine a multidimensional schema in constellation. Our methodology enables the enrichment of a dimension with facts in order to analyze other facts. Moreover, our methodology takes into account potential dependencies between facts that share dimensions, in order to ensure that the queries sent by the decision makers to the refined multidimensional schema are coherent.

The core of our proposals is the calculation of several hierarchies with factual data, which will be integrated into the chosen target dimension, in order to apply a structure to factual data that is compatible with integration into a dimension. This calculation is achieved using hierarchical clustering. The number of calculated hierarchies, and the data used to perform the hierarchy calculation, will depend on the contexts identified by the proposed methodology.

Another important point is the reduction in the number of the calculated hierarchies, in order to ensure the usability of the refined multidimensional schema. In this article, we provide three complementary methods to reduce the number of calculated hierarchies.

We offer an application of the proposed methodology in the form of algorithms to use when applying the proposed methodology to a data warehouse. These algorithms have been implemented in a complete prototype we developed using Matlab®.

The proposed prototype was tested on a standard benchmark: TPC-DS. This benchmark has two major advantages. First, the multidimensional schema of this benchmark is a constellation schema. Second, the benchmark is oriented for big data applications, and is consequently useful to test the scalability of the proposed prototype.

The experiments we conducted demonstrated the efficiency of the proposed methodology and prototype. As a first result, the prototype transformed the multidimensional schema as expected by the proposed methodology. Moreover, the execution times observed on the TPC-DS benchmark are quite satisfactory for off-line use of the proposed algorithm, with a total execution time of approximatively ten hours. Note that the main part of this execution time is spent on the processing MDX queries on the OLAP server: we assume that this execution time could easily be reduced by using a high-performance computer as server. Regarding the reduction of the number of calculated hierarchies, the proposed prototype is very efficient: during our experiment, the number of hierarchies was reduced from 804,628 to 40.

However, despite these satisfying results, some problems remain to be resolved.

First, concerning the proposed implementation using Matlab®. During our tests of the prototype, we observed that the number of cells into a matrix or a table cannot exceed 10^8. This limitation could be problematic for big data applications, which have to deal with massive numbers of data. To overcome this problem, two complementary options can be envisaged. The first solution is optimization of the prototype, which should be able to manage data split into several matrix. A second option is the development of another version of the prototype using another language.

Second, querying of newly created dimensions is not discussed in this article. These new dimensions contain several versions of a hierarchy, each version is related to a particular context instance in particular that corresponds to the relevance domain of the version concerned. Several other articles provide some solution to the versioning issue (see [42] for a survey on versioning in data warehouse). But, a querying process adapted to our methodology needs to be developed.

These two points represent our future practically and theoretically work on the refinement of multidimensional scheme.

ACKNOWLEDGMENTS

This article has received financial support from the French National Agency of Research (ANR), for the project VGI4Bio[3].

This article has received financial support from AgroParisTech.

We gratefully thanks Mrs. Daphne Goodfellow for her corrections.

REFERENCES

[1] S. Rizzi, *Conceptual modeling solutions for the data warehouse*, Data warehouses and OLAP: Concepts, architectures and solutions, pp. 1-26, 2007, Hershey, USA: IRM Press

[2] R. Ben Messaoud, O. Boussaid and S. Rabaséda, *A New OLAP Aggregation Based on the AHC Technique*, ACM Seventh International Workshop on Data Warehousing and OLAP (DOLAP), pp. 65-72, 2004

[3] F. Bentayeb and R. Khemiri, *Adapting OLAP Analysis to User's Constraints through Semantic Hierarchies*, Proceedings of the 15th International Conference on Enterprise Information Systems (ICEIS 2013), pp. 160-167, 2013

[4] M. Ceci, A. Cuzzocrea and D. Malerba, *OLAP over Continuous Domains via Density-Based Hierarchical Clustering*, 15th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (KES 2011), pp. 559-570, 2011

[5] L. Sautot, S. Bimonte, L. Journaux and B. Faivre, *Mixed driven Refinement design of Multidimensional models based on Agglomerative Hierarchical Clustering*, 17th International Conference on Enterprise Information Systems (ICEIS'15), 2015

[6] M. Serrano, J. Trujillo, C. Calero and M. Piattini, *Metrics for data warehouse conceptual models understandability*, Information and Software Technology, 49(8), pp. 851-870, 2007

[7] O. Romero and A. Abello, *Automatic validation of requirements to support multidimensional design*, Data and Knowledge Engineering, 69, pp. 917-942, 2010

[8] A. Vaisman and E. Zimányi, *Data Warehouse Systems: Design and Implementation*, 2014, Springer

[3]https://www.vgi4bio.fr/

[9] L. Sautot, B. Faivre, L. Journaux and P. Molin, *The hierarchical agglomerative clustering with Gower index: A methodology for automatic design of OLAP cube in ecological data processing context*, Ecological Informatics, 26(2), pp. 217 - 230, 2015

[10] A. K. Jain, M. N. Murty and P. J. Flynn, *Data Clustering: A Review*, ACM Computing Survey, 31(3), pp. 264-322, 1999

[11] S. Tufféry, *Data mining and statistics for decision making*, 2011, John Wiley & Sons

[12] L. Sautot, S. Bimonte, L. Journaux and B. Faivre, *A methodology and tool for rapid prototyping of data warehouses using data mining: Application to birds biodiversity*, International Conference on Model and Data Engineering, pp. 250-257, 2014

[13] L. Kaufman, *Clustering by means of medoids*, Statistical data analysis based on the L1-norm and related methods, 1987

[14] J. Han, *Towards On-Line Analytical Mining in Large Databases*, Sigmod Record, 27, pp. 97-107, 1998

[15] G. Bogdanova and T. Georgieva, *Discovering the Association Rules in OLAP Data Cube with Daily Downloads of Folklore Materials*, International Conference on Computer Systems and Technologies (CompSysTech 2005), 2005

[16] M. Kaya and R. Alhajj, *Fuzzy OLAP Association Rules Mining-Based Modular Reinforcement Learning Approach for Multiagent Systems*, IEEE Transactions on Systems, Man, and Cybernetics, 35(2), pp. 326-338, 2005

[17] R. Messaoud, O. Boussaid and S. Loudcher Rabaséda, *A data mining-based OLAP aggregation of complex data: Application on XML documents*, International Journal of Data Warehousing and Mining (IJDWM), 4(2), 2006

[18] A. H.L. Lim and C.-S. Lee, *Processing online analytics with classification and association rule mining*, Knowledge-Based Systems, 23, pp. 248-255, 2010

[19] J. J. Jadav and M. Panchal, *Association Rule Mining Method On OLAP Cube*, International Journal of Engineering Research and Applications (IJERA), 2(2), pp. 1147-1151, 2012

[20] M. Usman, R. Pears and A.C.M. Fong, *Discovering diverse association rules from multidimensional schema*, Expert Systems with Applications, 40, pp. 5975-5996, 2013

[21] H.C.W. Lau, K.S. Chin, K.F. Pun and A. Ning, *Decision supporting functionality in a virtual enterprise network*, Expert Systems with Applications, 19, pp. 261-270, 2000

[22] A. Sair, B. Erraha, M. Elkyal and S. Loudcher, *Prediction in OLAP cube*, International Journal of Computer Science Issues (IJCSI), 9, pp. 449-458, 2012

[23] W. Abdelbaki,S. Yahia, B. Sadok and R. Ben Messaoud, *Modular Neural Networks for Extending OLAP to Prediction*, Transactions on Large-Scale Data-and Knowledge-Centered Systems XXI, pp. 73-93, 2015, Springer

[24] Y. W. Choong, A. Laurent and D. Laurent, *Mining multiple-level fuzzy blocks from multidimensional data*, Fuzzy Sets and Systems, 159, pp. 1535-1553, 2008

[25] B. Leonhardi, B. Mitschang, R. Pulido, C. Sieb and M. Wurst, *Augmenting OLAP Exploration with Dynamic Advanced Analytics*, 13th International Conference on Extending Database Technology (EDBT 2010), 2010

[26] M. Ceci, A. Cuzzocrea and D. Malerba, *Effectively and efficiently supporting roll-up and drill-down OLAP operations over continuous dimensions via hierarchical clustering*, Journal of Intelligent Information Systems, 44(3), pp. 309-333, 2015

[27] J. Eder, C. Koncilia and D. Mitsche, *Automatic Detection of Structural Changes in Data Warehouses*, Proceedings of the 5th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2003), pp. 119-128, 2003

[28] J.-L. Seng and T.C. Chen, *An analytic approach to select data mining for business decision*, Expert Systems with Applications, 37, pp. 8042-8057, 2010

[29] J. Zubcoff, J. Pardillo and J. Trujillo, *Integrating clustering data mining into the multidimensional modeling of data warehouses with UML profiles*, Data Warehousing and Knowledge Discovery, pp. 199-208, 2007

[30] F. Bentayeb, *K-means based approach for OLAP dimension updates*, 10th International Conference on Enterprise Information Systems (ICEIS), 2008

[31] N. Karayannidis and T. Sellis, *Hierarchical clustering for OLAP: the CUBE File approach*, The International Journal on Very Large Data Bases, 17, pp. 621-655, 2008

[32] C. Favre, F. Bentayeb and O. Boussaid, *A Knowledge-Driven Data Warehouse Model for Analysis Evolution.*, Frontiers in Artificial Intelligence and Applications, 143, pp. 2-71, 2006

[33] J. Zubcoff and J. Trujillo, *A UML 2.0 profile to design Association Rule mining models in the multidimensional conceptual modeling of data warehouses*, Data and Knowledge Engineering, 63, pp. 44-62, 2007
[34] J. Zubcoff, J. Pardillo and J. Trujillo, *A UML profile for the conceptual modelling of data-mining with time-series in data warehouses*, Information and Software Technology, 51, pp. 922-977, 2009
[35] S. Palaniappan and T. K. Hong, *Discretization of continuous valued dimensions in OLAP data cubes*, International Journal of Computer Science and Network Security, 8(11), pp. 116-126, 2008
[36] N. U. Rehman, S. Mansmann, A. Weiler and M. H. Scholl, *Discovering Dynamic Classification Hierarchies in OLAP Dimensions*, 20th International Symposium on Methodologies for Intelligent System (ISMIS 2012), pp. 425-434, 2012
[37] J. A. P. Sacenti, F. Salvini, R. Fileto, A. Raffaetà and A. Roncato, *Automatically Tailoring Semantics-Enabled Dimensions for Movement Data Warehouses*, Proceedings of the 17th International Conference on Big Data Analytics and Knowledge Discovery (DaWaK 2015), 2015
[38] B. Bebel, J. Eder, C. Koncilia, T. Morzy and R. Wrembel, *Creation and Management of Versions in Multiversion Data Warehouse*, Proceedings of the ACM Symposium on Applied Computing (SAC), 2004
[39] M. Golfarelli, J. Lechtenborger, S. Rizzi and G. Vossen, *Schema Versioning in Data Warehouses*, Proceedings of the 23rd International Conference on Conceptual Modeling (ER Workshops 2004), LNCS 3289, pp. 415-428, 2004
[40] I. Z. Turki, F. Ghozzi Jedidi and R. Bouaziz, *Multiversion Data Warehouse Constraints*, ACM Thirteenth International Workshop On Data Warehousing and OLAP (DOLAP 2010), 2010
[41] I. Z. Turki, F. Ghozzi Jedidi and R. Bouaziz, *Constraints to manage consistency in multiversion data warehouse*, Proceedings of the 35th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2012
[42] K. Saroha and A. Gosain, *Multi-version data warehouse: A survey*, 5th International Conference - Confluence The Next Generation Information Technology Summit (Confluence), pp. 40-45, 2014

Ludovic Journaux Biography text here.

Lucile Sautot Biography text here.

Sandro Bimonte Biography text here.