First publ. in: 12. GI-Fachtagung für Datenbanksysteme in Business, Technologie und Web (BTW 2007), Aachen, Germany, 2007

Hierarchy-driven Visual Exploration of Multidimensional Data Cubes

Svetlana Mansmann Florian Mansmann Marc H. Scholl
Daniel A. Keim

University of Konstanz (Germany)
Department of Computer & Information Science
Fach D188, 78457 Konstanz

Firstname.Lastname@uni-konstanz.de

Abstract: Analysts interact with OLAP data in a predominantly "drill-down" fashion, i.e. gradually descending from a coarsely grained overview towards the desired level of detail. Analysis tools enable visual exploration as a sequence of navigation steps in the data cubes and their dimensional hierarchies. However, most state-of-the-art solutions are limited either in their capacity to handle complex multidimensional data or in the ability of their visual metaphors to provide an overview+details context.

This work proposes an explorative framework for OLAP data based on a simple but powerful approach to analyzing data cubes of virtually arbitrary complexity. The data is queried using an intuitive navigation in which each dimension is represented by its hierarchy schema. Any granularity level can be dragged into the visualization to serve as an disaggregation axis. The results of the iterative exploration are mapped to a specified visualization technique. We favor hierarchical layouts for their natural ability to show step-wise decomposition of aggregate values. The power of the tool to support various application scenarios is demonstrated by presenting use cases from different domains and the visualization techniques suitable for solving specific analysis tasks.

1 Introduction

The last decade has witnessed an explosion of visual interfaces for OLAP (OnLine Analytical Processing) - dashboards, charts, maps and scatterplots - which have impacted the business intelligence modern analytics. Interactive features, such as zooming, slicing, brushing and filtering, are becoming a commonplace in analysis software. With ever-growing volumes of accumulated data visualization becomes indispensable for extracting useful knowledge from data by a human expert. Adequate visual presentation helps to rapidly reveal patterns, recognize trends or anomalies. Especially the ad hoc queries, when an expert is guided by a mere guess or a hypothesis about the knowledge hidden in the "raw" data, benefit from the ability to visually specify the data set of interest and interact with it.

Analysis tools that abound the market offer a multitude of features and functions that

require training and skill to understand and use. Feature overload and usability deficiencies often lead to loss of orientation and discourage users from using novel techniques. The work reported in this paper is an attempt to design an OLAP interface advanced in terms of its exploration and visualization capabilities, but simple and intuitive in its usability with minimum training requirements. The applicability of a particular visualization technique depends on multiple criteria, such as the type of the analytical task, data volume and complexity, user preferences and skills. Our approach is to account for a variety of tasks by designing a comprehensive framework, in which users can experiment with various layouts and techniques to find satisfactory solutions to specific problems.

OLAP systems employ multidimensional data model to structure "raw" data into multidimensional cubes in which each data entry is shaped into a *fact* with associated numerical *measure(s)* and descriptive *dimensions* that characterize the fact [PJ01]. The values within a dimension can be further organized in a containment type hierarchy to support multiple granularities. The table containing the facts is referred to as *fact table*; other tables, called *dimensional tables*, store the dimensional values and the hierarchical relationship among them. Analytical queries aggregate measure values over a range of dimension values to provide the view of the desired dimensionality and granularity. The most common OLAP operations are *slice-and-dice* to define a sub-cube, *drill-down* and *roll-up* to perform aggregation and disaggregation, respectively, along a hierarchical dimension, *drill-across* to combine multiple cubes, *ranking* to find the outlier values, and *rotating* to see the data grouped by other dimensions.

While analytical queries aggregate over detailed data, visual exploration evolves in the inverse direction, i.e. descending from coarsely grained aggregates to more detailed views via stepwise decomposition along selected dimensions. The prevailing "from-overview-to-details" query direction is reflected in the structure of a typical OLAP data browser: each data cube is presented as a hierarchy of its dimensions where each dimension is a recursive nesting of its granularity levels, starting from the coarsest granularity and continuing with each finer level as a child of its predecessor. Users proceed by selecting the measure(s) and the aggregation function to invoke, the dimensions to use as decomposition axes, by filtering the selected data set and manipulating the visual representation of the result. These query steps are performed irrespective of the type of query and the chosen visualization technique. Therefore, we see a great usability improvement potential in designing a common uniform navigation framework for satisfying any type of analytical query.

The rest of the paper is organized as follows. Some outstanding related work on OLAP visualization is discussed in Section 2. In Section 3 we describe the data navigation as the core component of our proposed visual OLAP interface for visual exploration of data. Section 4 identifies the visualization techniques which best fit the needs of explorative analysis. Section 5 introduces the explorative framework as an abstraction of the mapping between user interaction and visual data representations. Finally, the concluding remarks are found in Section 6.

2 Related Work

An attempt to address the related work on designing visual interfaces for exploring multidimensional data in its entirety would explode the scope of this work. We limit ourselves to naming a few solutions which offer distinguished features relevant for our work.

Eick at al. presented a survey of common visual metaphors and associated interaction techniques with improved visual scalability, i.e. the capability to effectively display large volumes of multidimensional data [EK02]. The proposed techniques were implemented in the ADVIZOR system, in which each visual metaphor has a single zoom path. Rather than displaying multiple granularities within the same visualization, multiple *perspectives*, i.e. linked views displayed on the same screen, present various projections of data.

Stolte et al. proposed a class of multiscale, i.e. supporting multiple zoom paths, visualization techniques for data cubes in [STH03]. Implementation of these techniques is a part of an advanced system called Polaris [STH02], as well as its commercial successor Tableau Software [T06], which extend the Pivot Table interface by offering a combination of displays and tools for visual specification of analysis tasks.

Techapichetvanich et al. proposed a visualization framework for OLAP based on the Hierarchical Dimensional Visualization (HDDV) technique [TD05] which uses colored stacked bars. Each bar shows the results of decomposing the root aggregate along a chose dimension, whereas the color is used to mark the portion of values satisfying the specified range condition. Bars are not explicitly linked to each other, allowing to split the same aggregate along multiple dimensions.

A work by Hellerstein et al. [HAC+99] elaborates on the mechanisms to improve human-computer interaction and user control when analyzing massive data sets. Their proposed framework combines visualization and data mining techniques to guide the user through the data discovery process.

ProClarity was the first to introduce a hierarchical drill-down visualization called *Decomposition Tree* [P06], which places each aggregate into a node and its constituent subaggregates, obtained by drilling down into any dimension, to its child nodes. Child nodes display the next level of detail from left to right by value and percentage of the total. ProClarity's technique is designed to expand node by node for the "speed-of-thought" analysis, is limited to decomposing a single measure and has no visual formatting of the values. Besides, it is rather wasteful in terms of display utilization and is thus infeasible for exploring large data volumes.

Further enhancements to the hierarchical disaggregation are found in the OLAP web client Report Portal 2.1 released by XMLA Consulting [R06]. Report Portal offers graphical decomposition trees, such as BarChart and PieChart Trees, which arrange the entire drill-down view into a chart hierarchy. Moreover, drill-down steps are not aligned into dimensional levels, thus allowing to expand "sibling" values along different dimensions.

In a previous work we introduced an explorative framework based on enhanced decomposition trees [VM06] which implement various tree layouts and allow to choose between different visual presentations within the nodes. Schema-based navigation approach en-

ables efficient browsing and fast generation of visual hierarchies, support of complex dimensional hierarchies and parallel exploration on multiple cubes along shared dimensions. Classification of supported dimensional patterns can be found in [MS06]. In this work we concentrate on the database aspects of transforming the cube schema into frontend navigation, translating user interactions into OLAP queries and arranging the results of successive query steps into a specified visual format.

Whenever a data cube contains spatio-temporal characteristics, the analysis may benefit from specialized exploration techniques for space-time patterns. A synopsis of techniques for spatio-temporal exploration arranged according to the data and task types is produced in [AAG03]. Kuchar et al. [KHHP06] point out that time dimension is not an ordinary data attribute and that, therefore, to ensure satisfactory analysis, interaction and visualization techniques have to incorporate explicit awareness of temporal characteristics.

3 Datacube as a Navigational Hierarchy

The entire explorative framework can be considered as composed of an input and an output area for specifying queries and presenting query results, respectively. The input element is the navigation which displays the structure and the contents of the data in form of a browser for visual querying. The main area of the graphical interface serves for outputting the results of user interaction in a selected visual format.

To be visually navigable, the facts are modeled according to the multidimensional data model.Thereby, a straightforward spreadsheet approach with no distinction between structure and contents is inadequate for managing complex data where explicit knowledge of the structure is needed to ensure the validity of generated OLAP operations.

3.1 Conceptual and Logical Design

Conceptual design is the initial phase of building a data cube. At this stage, the developer investigates the user requirements as well as the available data and defines the structure of the fact tables and of the associated dimensional hierarchies using modeling techniques such as ER or Multidimensional ER model, UML or Dimensional Fact model.

The challenges of designing a data cube are demonstrated at an example drawn from an existing university data warehouse. Decision-makers expect the data from decentralized procurement systems to be extracted into a single fact table *Expenditures* with measures *amount* and *items* for the amount paid and the number of ordered items, respectively. The facts are determined by dimensions *category* for cost type, *project* as the order's destination, *funding* as the funding source, *institution* as the origin of the order, and *period* for tracing the date of the purcgaethe expenditure was registered. Figure 1 shows the expected structure of the cube as a dimensional fact schema. The schema is designed using the Dimensional Fact model, introduced by Golfarelli et al. in [GMR98] and extended by super-/subclass relationships to account for heterogeneity in dimension *institution*.

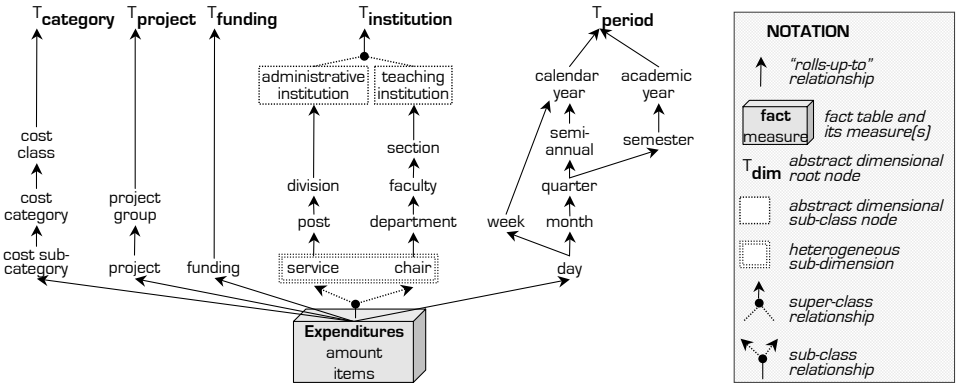


Figure 1: Dimensional fact schema of data cube **Expenditures**

With the exception of *funding*, dimensional values are further arranged into hierarchies. Dimensions are presented as directed graphs with aggregation levels as nodes and "rolls-up-to" relationships between them as edges. A dimension's graph is rooted at an abstract node with a single value $\top$, interpreted as all. The bottom-level node of each dimension is linked directly to the fact table. Temporal dimension *period* is a classical example of a multiple hierarchy: days can be aggregated along three distinct aggregation paths.

A dimension worthwhile special treatment is *institution*. It is typical for institutional hierarchies to be unbalanced and consist of multiple sub-classes, each with its own attributes and hierarchical levels. However, an analysis task may require a heterogeneous hierarchy to be "fetched" into a single dimension by means of generalization, i.e. superclass/subclass relationships. Standard OLAP tools, incapable of supporting generalization hierarchies, would only work on separate data marts, one for each homogeneous sub-class. Our approach admits heterogeneity within a dimension but enforces the hierarchy's subdivision into homogeneous disjunct subclasses to ensure correct aggregation (further details can be found in [MS06]). Back to the example at hand, all institutional instances are subdivided into sub-classes of administrative and teaching institutions, each with its own hierarchy.

Relational OLAP employs star or snowflake schema, both based on arranging the data in fact and dimensional tables. Fact table is constrained to have one column per measure, as well as one column per dimension for storing the references to the values in the respective dimensional table. Star schema places the entire dimension, however complex it might be, into a single relation by pre-joining all aggregation levels, similarly to a spreadsheet. Snowflake schema normalizes hierarchical dimensions by placing each granularity into a separate table, thus eliminating redundant storage and facilitating consistency of updates.

Snowflake schema becomes the only option when dimensional hierarchies are prone to irregularities, such as heterogeneity, non-strictness, missing values, mixed granularity etc. Figure 2 shows the resulting snowflake schema *institution*. An auxiliary relation *subclass* stores the IDs and the names of all defined subclass categories. The constraint of being referenced by just a single attribute in the fact table is satisfied by the "trick" of merging the primary keys from *service* and *chair* into a single relation *institution*.

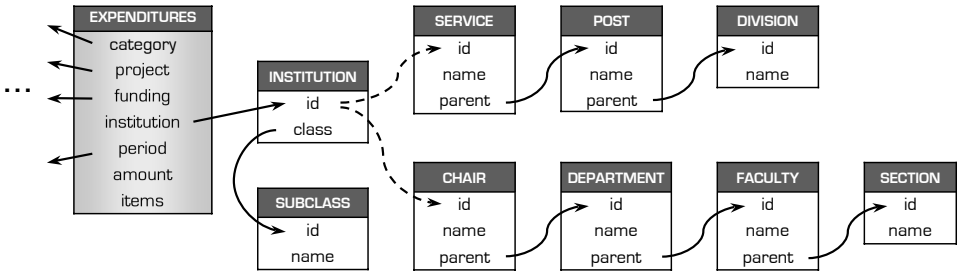
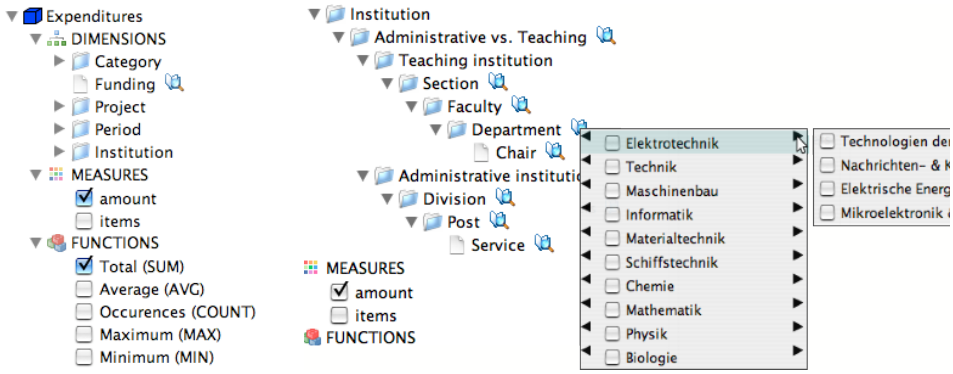


Figure 2: Logical design of the complex dimension *institution*

Upon the completion of the normalized snowflake schema, a star-schema view of the each homogeneous (sub-)dimension is generated. The view is defined as a join between all dimensional levels and its purpose is simply to facilitate the generation of the OLAP query from the visual navigation events (further explanation is given in the next subsection).

3.2 Navigation Hierarchy

Users access the data via a file-browser-like navigation hierarchy. The top-level navigation objects are the data cubes, consisting of three types of objects, namely the dimensional hierarchies, the measures, and the aggregation functions applicable to the measures, as shown in Figure 3(a). Dimensions are represented by their hierarchy schemata in a top-down fashion: the top level is the abstract root node $\top$ (with the exception of non-hierarchical dimensions which do not need an extra root node); each further hierarchy level is a sub-folder nested in the folder of its parent level. Non-hierarchical dimension levels, i.e. those with the finest granularity, are presented by non-folder icons.



(a) Data cube navigation (b) Navigation paths of a generalization hierarchy

Figure 3: Schema-based data browser for querying OLAP cubes

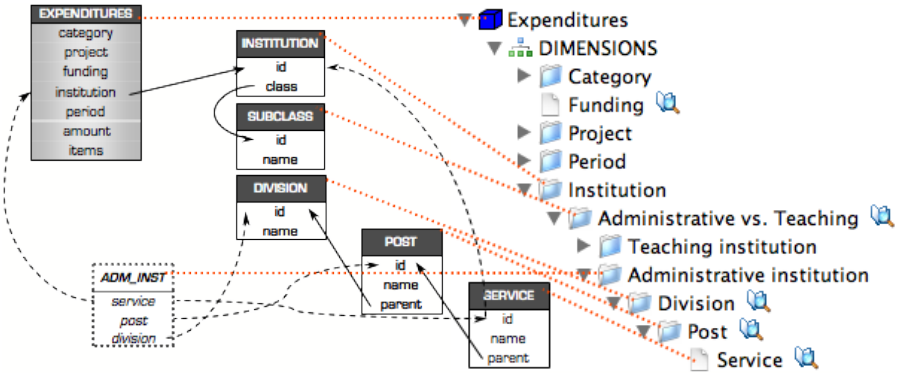


Figure 4: Navigation-to-database mapping fragment

Figure 3(b) shows the browser hierarchy of *institution*. Notice, that unlike the dominating OLAP navigation approaches, which directly display the data hierarchy of a dimension, our *schema-based* approach explicitly presents the hierarchy schema. Each schema node has a dimensional data table related to it; the values at each level are accessed via a "lens" icon, as illustrated in Figure 3(b) for the node *department*. Dimensional data can be further explored in the context of its hierarchy: data entities are supplied with "back" and "forward" arrows to navigate to that entry's parent and child values, respectively.

Once the user has chosen the measure m and the aggregation function a from fact table f, the query can be initialized as `SELECT a(m) FROM f`. Selection of any sub-dimension d_i in dimension d is interpreted as a drill-down along d:
`SELECT` d_i`.id, a(m) FROM f JOIN ... JOIN` d_i `GROUP BY` d_i`.id`.
Filtering along any dimension is done by setting/unsetting values displayed in the dimension's popup data view and is reflected in the `WHERE` clause.

Since dimensions in the snowflake schema are linked to the fact table only via the bottom granularity level, all higher levels referenced by a query need to be joined all the way down in order to reach the fact data. Figure 4 shows the correspondence between the navigation nodes and the data tables for the subtree of *administrative institution*. Black arrows show the corresponding join attributes of all tables. Notice the beneficial effect of the joined view *adm_inst*: any dimensional node irrespective of its depth can be joined with the fact table directly via this view, without having to traverse the navigation tree. Materialization of the joined views may be considered for improving the query execution time due to spared intra-dimensional joins.

Information about the structure of the data cubes, necessary for generating the data browser and mapping navigation events to database queries, is stored as metadata. The metadata has been modeled in accordance with the specification of the Common Warehouse Metamodel (CWM) [Obj06]. In the recent years, CWM has evolved into a widely adopted standard for metadata interchange in the data warehouse environment. *OLAP* package of this model provides all necessary concepts (e.g., multiple hierarchy, generalization relationship, etc.) for describing data cubes as navigation objects in an explorative interface.

4 Visualization of OLAP Queries

Confronted with evaluation of thousands of tuples returned by an OLAP query, users tend to focus on a few outstanding values and discard the remaining data, or run another query to retrieve the next-level aggregates (drill-down or roll-up) w.r.t. the ones obtained previously. While this strategy can be successfully applied for reporting tasks, the exploratory analysis is driven by finding information hidden in the data via ad hoc queries. In such scenarios, abstract visual representations (i.e., overviews) facilitate the comprehension.

As data volumes for analysis are huge and a lot of insight is hidden at lower granularity levels, coarsely grained aggregates play rather an auxiliary role in explorative queries. The analyst needs to assess the data at different aggregation levels and compare the aggregates at the same level as well as across levels. As an example, one might want to compare expenditures of departments with each other, but comparison of a small department with a project team of approximately the same size would also make sense.

Multiscale visualization techniques are effective for facilitating the exploration process because they change the visual representation to show the data at different levels of abstraction: at a high level, there are just few coarsely grained values; as the user zooms, the data density decreases, allowing to show more detailed representations of individual data points [STH03]. Preservation of the overview throughout the course of interaction (zooming and panning) becomes crucial as otherwise the user might easily lose the context.

Eventually, scalability of the visualization techniques, i.e. the ability to display ever growing number of items in a limited screen while avoiding clutter and overlap, becomes a challenge [EK02]. This challenge turned display space into a scarce resource, which in turn lead to the development of space-filling visualization techniques that attempt to utilize the whole available display space.

4.1 Hierarchical Visualization Techniques for OLAP

There are many factors to evaluate when identifying visualization techniques adequate for a specific application. Visualization encompasses anything from simple charts to highly unique representations, from static diagrams to complex interactive views. In the analysis tools of the current state of the art , traditional visualization techniques such as charts, plots, or pivot tables, prevail. Their main advantages are simplicity, straightforward interpretability, and familiarity. However, the applicability of these techniques is rather limited. For example, the charts only allow uniform granularity, which makes it impossible to compare aggregates across granularity levels within the same view. Furthermore, previously visualized values get lost each time a drill-down or a roll-up step is performed.

The standard interface for a series of successive disaggregation steps is a pivot table, which nests multiple granularities along its two axes. Evidently, pivot tables are not effective for exploratory analysis because of textual list-based representation of the data. For some query types, the resulting pivot table can be replaced by a visual spreadsheet technique. Consider applying a non-cumulative aggregation function, such as `MIN`, `MAX`, or `AVG`, in

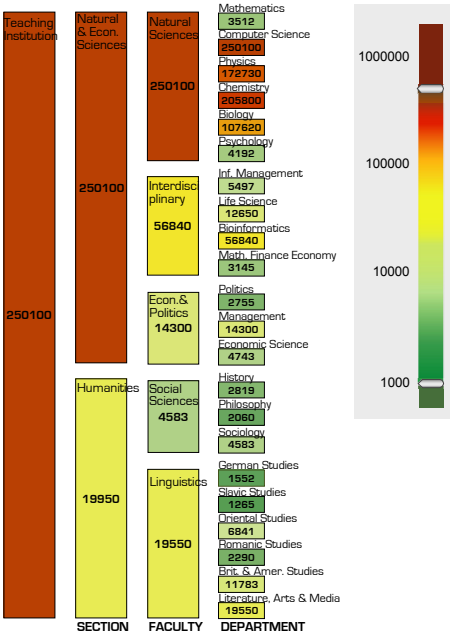


Figure 5: Hierarchical HeatMaps for exploring non-cumulative aggregates

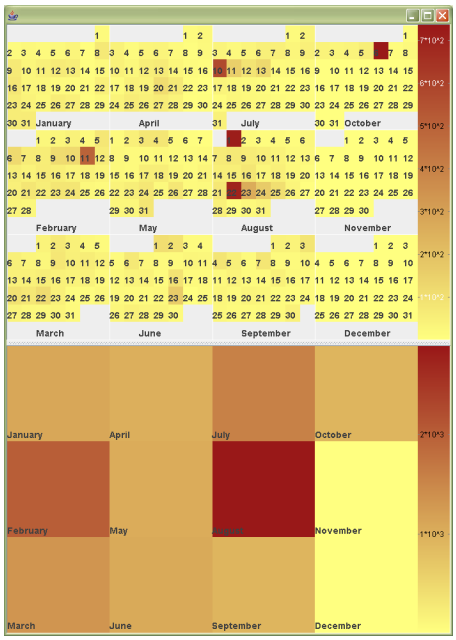


Figure 6: Multiscale *Recursive Pattern* for (dis-)aggregating along time dimension

which case the aggregates at all levels remain within comparable ranges. This common value range opens up an opportunity to map the values to attribute *color* and transform the pivot table into a hierarchical HeatMap. Figure 5 contains an example of such HeatMap: bottom-level nodes are represented as cells in an array; higher level nodes are shaped as rectangles spanning the width of their subtrees, similarly to the cells in a pivot table. The visualization is obtained by a 3-step drill-down along the subtree of *teaching institution* for the measure *amount*'s maximum value. A logarithmic scale of the colormap copes with the skewness of the resulting distribution; sliders at the colormap's poles are useful for dynamically adjusting the visualization's sensitivity to outliers.

Due to the hierarchical characteristics found in many dimensions of data cubes, hierarchical visualization techniques lend themselves to be utilized in the analytical process. In general, the analyst follows a top-down approach starting at the higher aggregation levels, because the limited amount of aggregates makes it easier to retain an overview. Step-wise, he either decides to apply slice-dice or drill-down operations to reduce the subset or to retrieve more details.

Traditionally, node link diagrams are used to represent hierarchical data in a compact yet intuitive way. Given a good spatial embedding of the nodes of such a tree, it is easy to determine connectivity characteristics of individual nodes. In the context of visualizing OLAP data, a technique, called Decomposition Tree, proposed by ProClarity [P06], and their visually enhanced variants [VM06] bridge the gap from spreadsheet or chart views

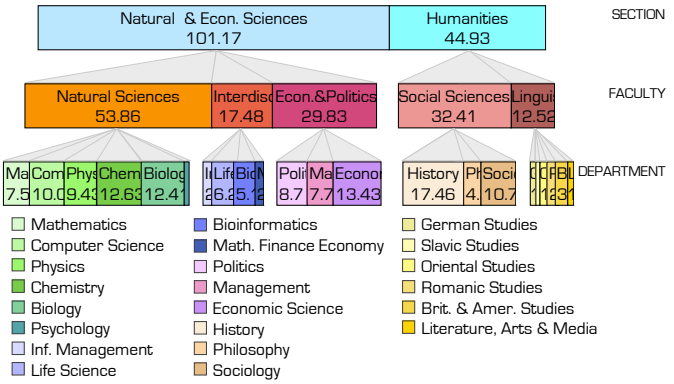


Figure 7: Decomposition Tree with space-filling bars

to hierarchical views. Decomposition tree preserves the aggregates computed in previous steps, similarly to pivot tables, but employs a node-link tree layout to show the parent-child relationships between the sub-aggregates. Figure 7 gives an example of a decomposition tree, which arranges the child sub-aggregates of each value into a space-filling bar chart. The values at each level and across levels are visually comparable to each other via the size of their rectangle areas.

The *connection* layouts, based on node-link diagram supports immediate perception of child-parent relationships, however, it is not efficient in terms of display utilization because most of the pixels are wasted as background. Moreover, the nodes at lower levels quickly run out of space for displaying the labeling information. Here, the *enclosure*, also known as "value-by-area", visualization techniques offer a space-optimized solution. The most commonly used enclosure techniques are TreeMaps [Shn92] and their variations, employing rectangular node shape, and radial layouts, such as the SolarPlot [Chu98] and InterRing [YWRP03], in which the aggregates are laid out radially, with the most aggregated values at the center and finer granularity levels farther away from the center.

Figure 8 shows the results of querying a data cube containing student enrollments from a university data warehouse, visualized using a SolarPlot. The aggregated measure (number of enrollments) values are mapped to the size of the circle segments in each ring. For examining the distribution of foreign students over departments, the analyst filtered out Germany in the dimension *country* (German: Land), retrieved the sub-aggregates at *subcontinent* (Ger.: Subkontinent) and, subsequently, at country level, and, in the final step, added the dimension *department* (Ger.: Institut) as the outer ring. SolarPlot technique overcomes the problem of nested rectangle layouts (TreeMaps) that allocate progressively less space to the nodes at lower tree levels. This is due to the fact that outer rings intrinsically occupy more space than the inner ones. One drawback is that large proportions of the usually rectangular screen stay unused.

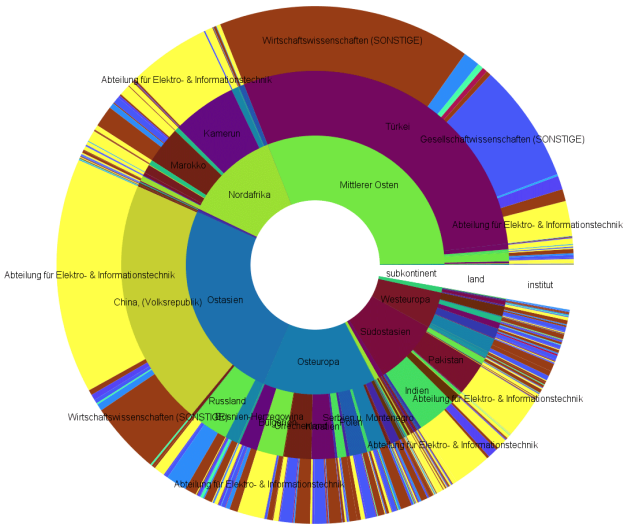


Figure 8: Radial space-filling visualization (solar plot)

4.2 Temporal and Spatial Visualization Techniques

In the multidimensional data model, spatial and temporal components of the data are treated just like any other dimensions. However, to fully exploit the potential and the richness of spatio-temporal data attributes, the awareness of their specific characteristics has to be incorporated into the data warehouse and the analysis tools on top of it.

The outstanding role of the temporal dimension in OLAP is reflected in the data warehouse definition: "...a subject-oriented, integrated, time-variant, non-volatile collection of data in support of management's decision making process" [Inm02]. Rivest et al. [RBM01] point out that space is the other of the two analytical components, needed to take full advantage of a data warehouse, and that spatial dimensions, just like temporal ones, should be considered standard for any data warehouse implementation. Indeed, there is an estimation that about 80% of all data stored in corporate databases has a spatial component [Fra92].

The requirements towards a visual interface in handling spatio-temporal data is twofold: 1) to communicate spatial or temporal patterns hidden within the data, and 2) to facilitate query specification by building up a mental image of the data distribution which helps in restricting too generally formulated queries to data regions of interest.

Example 1. To visualize the temporal evolution of a certain measure within a data cube, a recursive pattern visualization [AKK95] can be adapted to fit into a calendar view, similar to the calendar-based visualization [VWVS99]. Figure 6 demonstrates the results of analyzing the volumes of email communication along time dimension using the above approach. As the information can be displayed at several granularities without changing position of the underlying data items, this technique has proven to be very scalable. Recursive pattern calendar view can be used as an exploration technique in its own right,

or as a node-level view in a decomposition tree, to combine temporal exploration with dis-aggregation along other dimensions.

Example 2. Spatial attributes, such as address, location coordinates, position, orientation, or size, are a frequently encountered dimension in many data cubes. Geography is often used as an exploration axis due to the fact that other data characteristics (i.e., consumer behavior) may strongly vary in space. However, in straightforward techniques, such as highlighting a map or a cartogram according to the measure's value, large countries - which do not necessarily represent more data elements than small countries - dominate the visual impression and a lot of display space remains unused.

Distortion approaches are employed to overcome the disadvantages of classical maps. The measure under analysis is used as a distortion parameter as demonstrated in [KMP+05]. In case of a gap-free rectangular distortion, the resulting map degrades to a TreeMap, as in the *Hierarchical Network Map* technique [MV06]. Figure 9 reveals the geographical distribution of the network traffic at a network gateway. The approach simultaneously displays two measures: one value is mapped to the size of the rectangular areas whereas the other is shown by color. In this example, the size of the country's or the internet backbone system's node is proportional to the number of IP addresses they represent, and color shows for the actual traffic volumes registered at each node.

5 Exploratory Framework

Interactive visualization is defined by Colin Ware in [War04] as "a process made up of a number of interlocking feedback loops that fall into three broad classes. At the lowest level is the data manipulation loop, through which objects are selected and moved using the basic skills of eye-hand coordination. ... At an intermediate level is an exploration and navigation loop, through which an analyst finds his or her way in a large visual data space... At the highest level is a problem-solving loop through which the analyst forms hypotheses about the data and refines them through an augmented visualization process."

The exploratory framework introduced in this work supports the interactive visualization process, described above, by defining an appropriate interface for 1) data navigation, i.e. translating visual navigation steps into database queries, 2) visualization, i.e. mapping the retrieved data into a selected visual format, and 3) interaction, i.e. stepwise manipulation of the generated view to solve the defined analytical task. In section 3 we already described our approach to data navigation. Therefore, the focus in this section is on the last two steps that address the generation and iterative refinement of the visualization.

The previous section contains just a limited selection of visualization techniques from a multitude of potentially applicable ones. The challenge of integrating various layouts into a unified framework is to find a common abstraction interface for them all in order to enable their generation by invoking the same navigation events.

As all supported visualization techniques are based on a hierarchical layout, each visualization is a tree, i.e. a directed acyclic graph with all edges oriented away from the root. Any two nodes of the tree connected by an edge are characterized by parent-child relation-

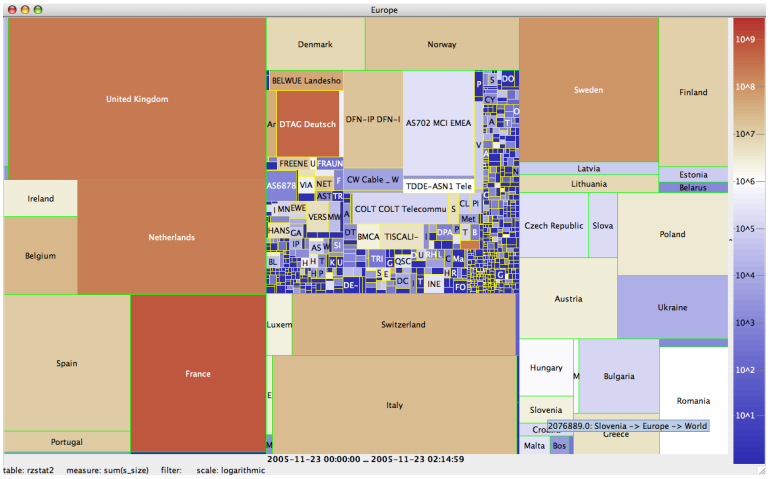


Figure 9: *Hierarchical Network Map* showing aggregated network traffic of the countries with a drill-down into Germany's node showing the level of its internet backbone systems.

ship. In the context of disaggregation the tree abstraction can be narrowed to the notion of a decomposition tree: the nodes contain the aggregate values and the edges are labeled by the respective dimensional characteristics of their nodes. Depending on the layout (e.g., in "edge-less" space-filling approaches), the labels can be placed inside the nodes.

Child level aggregates are obtained by decomposing a node or the entire tree level along a dimensional category. At this point, the visualizations can be classified according to their node properties. In *plain-value-node* techniques, as found in Figures 8 and 5 , each tree node refers to a single aggregate value, mapped to node's area and/or color. *Chart-node* approaches, as in Figures 7 and 6, allow decomposition inside the node, i.e. a node is composed of all child values of an aggregate, obtained by decomposing the latter along a specified dimension and arranged into a chart or other 2-dimensional view.

Further common elements of the presentation layer can be defined as follows:

Dimensional axis corresponds to a dimension category, represented by a node in the navigation hierarchy. Every distinct value of the dimension's values is interpreted as a *coordinate* (e.g. axis *month*, coordinate *March*). Each level of the tree, and – in case of a chart-node layout – also each node has its own dimensional axis.

Decomposition axis is the dimensional axis chosen for the disaggregation step, resulting in populating the chart-nodes' contents or adding a new level to the tree. *Decomposition schema* is the lattice of all decomposition axes in the visual hierarchy.

Core axis is defined for the class of space- and/or time-aware visualization techniques. These approaches treat the respective spatial or temporal dimension differently from any other dimension, for example, in the recursive pattern, the only valid decomposition operation is that along the time axis whereas other dimensions may be only used for filtering.

Tree split generates a new level of the tree by dis-aggregating the values in the bottom-

level nodes along the specified decomposition axis. In terms of OLAP operations, disaggregation corresponds either to a drill-down, i.e. descending in the granularity of the same dimension, or to a nesting, i.e. decomposing along a category from a different dimension. This separation is important for chart-node techniques, which distinguish between a primary, an inner, and an outer split:

- *Primary split* is the first decomposition step which generates the subset of aggregates to populate the chart view of the root node.
- *Inner split* is a drill-down into a category of the primary split. The inner split refines the granularity within the node's chart.
- *Outer split* is a decomposition step along any axis that is not a part of the primary split axis' dimensional hierarchy. Outer split leaves the dimensional axis of the nodes' inner charts unchanged.

The above defined abstractions of the presentation are fundamental for implementing a uniform approach to generating a visualization and interacting with it. Once a visualization technique has been chosen, the view is instantiated with the measure(s) and the aggregation function selected from the cube's navigation hierarchy. Decomposition steps are performed by dragging the dimensional nodes from the navigation into the visualization area. Filtering can be performed in the background by setting the values to hide/display in the data browser, or directly, by deleting nodes or entire sub-trees from the visualization.

Further interaction techniques, such as zooming, rearranging the layout, swapping axes, sorting, brushing, re-scaling etc. are supported. However, these features are not specific to our framework and therefore, we omit further details.

6 Conclusion

In this work, we introduced a framework for interactive hierarchy-driven exploration of OLAP data, in which the data navigation as well as the visualization approaches are based on the hierarchical view of the data. Hierarchical structures enjoy growing popularity in visual analysis. Multidimensional data cubes do not only contain hierarchically structured data but are also explored in a predominantly hierarchical fashion by iteratively refining the subset of interest and its dimensionality and granularity. We identify two key issues in supporting powerful visual analysis, namely, (1) data navigation for specifying iterative queries and (2) minimization of cognitive process, or appropriate presentation.

As the first building-block of the framework, we proposed a schema-based data navigation which facilitates visual specification of OLAP queries. Unlike traditional direct data tree navigation, found in many OLAP interfaces, our approach strictly distinguishes between the schema of a hierarchical dimension and the data that populates it. Representing dimensions by their schema results in a highly compact navigation, admitting multiple and generalization hierarchies. The ability to stay at the schema level allows the user to generate a visualization of a large data set with a small number of drag&drop operations.

In the next step, we identified scalable visualization techniques that are able to cope with the challenge of representing potentially large result sets of OLAP queries. Hierarchical layouts are appreciated for their natural ability to show the data of different granularity in the same view. An explorative query in a data cube can be viewed as a series of disaggregation and filtering steps, therefore the sub-aggregates computed at each step can be displayed in form of a decomposition tree. Special characteristics of time and space dimensions were taken into account as those two dimensions are prominent in the context of data warehouses and full account of their specific requirements is likely to reveal valuable patterns in the course of exploration.

By defining a common abstraction level for mapping the results of user interaction into a visual presentation, we could integrate manifold visualization approaches for solving a variety of exploration and conventional reporting tasks in the data warehouse context.

References

[AAG03] Natalia Andrienko, Gennady Andrienko, and Peter Gatalsky. Exploratory spatio-temporal visualization: an analytical review. *Journal of Visual Languages & Computing*, 14:503–541, December 2003.

[AKK95] M. Ankerst, D. A. Keim, and H.-P. Kriegel. Recursive Pattern: A Technique for Visualizing Very Large Amounts of Data. In *Proc. Visualization '95, Atlanta, GA*, pages 279–286, 1995.

[Chu98] Mei C. Chuah. Dynamic Aggregation with Circular Visual Designs. In *InfoVis '98: Proceedings of the 1998 IEEE Symposium on Information Visualization*, pages 35–43. IEEE Computer Society, 1998.

[EK02] Stephen Eick and Alan Karr. Visual Scalability. *Journal of Computational & Graphical Statistics*, 11(1):22–43, 2002.

[Fra92] Carl Franklin. An introduction to geographic information systems: linking maps to databases. *Database*, 15(2):12–21, 1992.

[GMR98] Matteo Golfarelli, Dario Maio, and Stefano Rizzi. Conceptual Design of Data Warehouses from E/R Schema. In *HICSS '98: Proceedings of the 31st Annual Hawaii International Conference on System Sciences*, volume 7, pages 334–343, 1998.

[HAC+99] Joseph M. Hellerstein, Ron Avnur, Andy Chou, Christian Hidber, Chris Olston, Vijayshankar Raman, Tali Roth, and Peter J. Haas. Interactive Data Analysis: The Control Project. *Computer*, 32(8):51–59, 1999.

[Inm02] William H. Inmon. *Building the Data Warehouse, 3rd Edition*. John Wiley & Sons, Inc., New York, NY, USA, 2002.

[KHHP06] Olga A. Kuchar, Thomas J. Hoeft, Susan Havre, and Kenneth A. Perrine. Isn't It About Time? *IEEE Computer Graphics and Applications*, 26(3):80–83, 2006.

[KMP+05] Daniel A. Keim, Florian Mansmann, Christian Panse, Joern Schneidewind, and Mike Sips. Mail Explorer - Spatial and Temporal Exploration of Electronic Mail. In *EuroVis 2005: Eurographics/IEEE-VGTC Symposium on Visualization*, 2005.

[MS06] Svetlana Mansmann and Marc H. Scholl. Extending Visual OLAP for Handling Irregular Dimensional Hierarchies. In *DaWaK 2006: Proceedings of 8th International Conference on Data Warehousing and Knowledge Discovery*, pages 95–105, 2006.

[MV06] Florian Mansmann and Svetlana Vinnik. Interactive Exploration of Data Traffic with Hierarchical Network Maps. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1440–1449, 2006.

[Obj06] Object Management Group (OMG). Common Warehouse Metamodel (CWM) Specification, V1.1, 2006. Online: `http://www.omg.org/technology/documents/formal/cwm.htm`.

[P06] ProClarity Analytics 6, 2006. Online: `http://www.proclarity.com/products/proclarity_analytics_6.asp`.

[PJ01] Torben Bach Pedersen and Christian S. Jensen. Multidimensional Database Technology. *IEEE Computer*, 34(12):40–46, 2001.

[R06] Report Portal: Zero-footprint OLAP Web Client Solution. *XMLA Consulting*, 2006. Online: `http://www.reportportal.com`.

[RBM01] S. Rivest, Y. Bédard, and P. Marchand. Towards better support for spatial decision-making: Defining the characteristics of Spatial On-Line Analytical Processing (SOLAP). *Geomatica, the journal of the Canadian Institute of Geomatics*, 55(2001):539–555, 2001.

[Shn92] Ben Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99, 1992.

[STH02] Chris Stolte, Diane Tang, and Pat Hanrahan. Query, analysis, and visualization of hierarchically structured data using Polaris. In *ACM SIGKDD '02: Proceedings of 8th International Conference on Knowledge Discovery and Data Mining*, pages 112–122, 2002.

[STH03] Chris Stolte, Diane Tang, and Pat Hanrahan. Multiscale Visualization Using Data Cubes. *IEEE Trans. on Visualization and Computer Graphics*, 9(2):176–187, 2003.

[T06] Tableau Software, 2006. Online: `http://www.tableausoftware.com`.

[TD05] Kesaraporn Techapichetvanich and Amitava Datta. Interactive Visualization for OLAP. In *ICCSA 2005: Proc. of the International Conference on Computational Science and its Applications (Part III)*, pages 206–214, 2005.

[VM06] Svetlana Vinnik and Florian Mansmann. From Analysis to Interactive Exploration: Building Visual Hierarchies from OLAP Cubes. In *EDBT 2006: Proceedings of 10th International Conference on Extending Database Technology*, pages 496–514, 2006.

[VWVS99] Jarke J. V. Van Wijk and Edward R. V. Van Selow. Cluster and Calendar Based Visualization of Time Series Data. In *INFOVIS '99: Proceedings of the 1999 IEEE Symposium on Information Visualization*, pages 4–9, Washington, DC, USA, 1999.

[War04] Colin Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers Inc., US, 2nd edition, 2004.

[YWRP03] Jing Yang, Matthew O. Ward, Elke A. Rundensteiner, and Anilkumar Patro. InterRing: a visual interface for navigating and manipulating hierarchies. *Information Visualization*, 2(1):16–30, 2003.