Modelling Multidimensional Data in a Dataflow-Based Visual Data Analysis Environment

Frank Wietek

University of Oldenburg, Department of Computer Science
Escherweg 2, D–26121 Oldenburg, Germany
wietek@informatik.uni-oldenburg.de

Abstract. Multidimensional data analysis is currently being discussed in terms like *OLAP*, *data warehousing*, or *decision support*, mainly concentrating on business applications. Numerous *OLAP-tools* providing flexible query facilities for *datacubes* are being designed and distributed. Typical analysis sessions with these kind of systems comprise long and branching sequences of exploratory analysis steps which base upon each other. While concentrating on single functions and processing steps, management of this analysis process as a whole is scarcely supported.
This paper proposes a dataflow-based visual programming environment for multidimensional data analysis (*VIOLA*) as an approach to deal with this problem. Providing a foundation of basic operations, data processing, navigation, and user interaction, an appropriate data model (*MADEIRA*) is developed. Epidemiological studies, i. e. investigations of aggregate data on populations, their state of health, and potential risk factors, will serve as a leading example of a typical application area.

1 Introduction

Especially in the field of business applications, flexible and systematic analysis of fast growing databases has gained more and more general interest in the last few years. Based on *data warehouses*, which collect data from different sources, *OLAP-tools* are oriented towards decision support [1]. Databases analysed by these tools are modelled as a set of multidimensional *dataspaces* or *cubes*. These contain aggregated *fact data* (*measures*, described by *quantifying* or *summary attributes*), e. g. sales or profit, which are classified by a number of *dimensions* (*parameters*, described by *qualifying* or *category attributes*) like product, shop, or time. Instances of parameter values are usually classified in a *category hierarchy*.

Each dimension of a datacube corresponds to a criterion for partitioning base data in different subgroups. A *cell* of a cube describes the group of base data entities defined by the corresponding instances of the cube's category attributes. Since all cell values relate to groups of entities, this kind of data is also called *macrodata*. This contrasts with typically relational *microdata* mostly constituting the base data from which aggregated measures are calculated by *aggregation functions* like count, sum, or average over certain attributes.

M. Jarke, A. Oberweis (Eds.): CAiSE'99, LNCS 1626, pp. 149–163, 1999.

Multidimensional data analysis is certainly not restricted to the business domain. In this paper, epidemiological studies (as for example carried out in disease registries) are supposed to serve as an application example. A typical multidimensional dataspace in this domain represents incidence or mortality counts or rates by age, sex, study region, time, and type of disease. Studies typically handle base data of up to some million cases classified in 5–50 dimensions – each of them with two to some thousand different values on usually not much more than ten different aggregation levels. As well as enterprise managers in the "original" OLAP domain introduced above, who navigate through business databases while searching for interesting information, also in the area of epidemiology specialists of different disciplines (social scientists, doctors, health data administrators) are to be provided with intuitive facilities for comfortable, exploratory data analysis.

Most available OLAP-tools (cf. [2]) concentrate on powerful statistical and data management operations and facilities for data visualization, but do hardly support management of this analysis process as a whole by providing the user with an overview of what he has done so far and by offering possibilities to manipulate this exploration history. In this paper, the idea of a dataflow-based visual analysis environment called *VIOLA* (*VIsual On-Line data Analysis environment*) will be proposed to deal with this subject (see Sect. 2).

Corresponding to available tools, existing multidimensional data models (see [3] for a survey) restrict themselves to the database point of view. Additionally, in many cases their design is strongly influenced by the idea of a relational implementation. They provide powerful operations to construct complex database queries, but do not take further steps of data analysis into account.

What we claim is to model the datacube as a data structure for both database queries *and* further processing. Thus, this paper introduces the data model *MADEIRA* (*Modelling Analyses of Data in Epidemiological InteRActive studies*) which

- is independent of a physical (relational) database implementation,
- provides enough information about measures and categories to facilitate intelligent and sensible selection and application of analysis methods to given datasets (especially for, but not restricted to epidemiology), and
- builds a framework for a data analysis environment, which combines different analysis procedures visually and thus keeps track of the whole process of an analysis session – both for complex pre-designed reports and (even more important) for ad-hoc data exploration.

In Sect. 3, some basic ideas, structures and operations of *MADEIRA* will be defined in order to give a solid foundation of the subsequent two sections 4 and 5. These are supposed to show how the idea of dataflow-programming based on a formal logical data model is able to enhance the power and usability of existing OLAP-tools in an "intelligent" way. Different aspects of implementing basic data analysis operations are discussed: execution of visual queries, selection of methods, examples of visual control structures, and interactive data visualization. Section 6 discusses some related approaches to multidimensional data modelling

and visual programming in data analysis. Finally, Sect. 7 gives a short summary and points out some ideas for future work.

2 A Dataflow-Based Visual Data Analysis Environment

What is a suitable data analysis environment and what are efficient tools and user interfaces for "good" data analysis? Hand [4] considers a sensible distribution of subtasks between human user and computer-based tools the main goal of performing "intelligent" data analyses. Each one of these two should be able to concentrate on his respective strength: the data analyst on his creativeness, his ability to recognize complex structures and to develop new hypotheses – the computer on management and preparation of data and methods and on processing complex calculations efficiently and systematically.

Visualization of information provides the basic opportunity of integrating the human analyst into the process of data analysis. Especially combined with flexible facilities of modifying parameters and presentation modes as well as selection of data and calculated measures, suitable graphics reveal hidden structures of information. Insights gained in this way lead to new hypotheses and further investigations under new points of view – resulting in an refining analysis cycle.

Many existing tools for data analysis (like SAS, SPSS, or also OLAP-tools) leave the management of this process itself to the user. Thus, he might easily get lost in the course of calculations and produced datasets, not knowing exactly how we got his results and how they relate to the base data. An intelligent system for data analysis should not only provide the user with routines for calculation and visualization, but should also give access to the complete course of a data analysis session to make results interpretable and reproducible and to simplify comparative modifications and repetitions of an analysis sequence.

According to the outstanding role of visualization, dataflow-based visual programming [5] is a suitable paradigm to fulfill these requirements by visualizing the whole process of an analysis. The analysis system *VIOLA* is supposed to implement this concept in the context of OLAP. Different building-blocks, representing data sources and methods of data analysis, management, and visualization, are interconnected in a dataflow-chart corresponding to calculations of intermediate results. Exploratory data analysis is performed by changing datasets or selecting subsets, by interactively modifying parameters of methods, exchanging methods for similar ones, or expanding the flowchart by further analysis steps [6]. Processing multiple datasets "in parallel" in *one* node facilitates flexible reusage and modification of common analysis sequences on multiple datasets and ensures comparability of the respective results. This might additionally be supported by introducing hierarchical subnets of analysis steps.

In our application domain, data exchanged between nodes of an analysis network are multidimensional datacubes. The next section will define some parts of a formal logical data model for datacubes and their processing in order to continue the idea of defining steps of a data analysis process explicitly and making results of an analysis session exactly interpretable.

3 The Multidimensional Data Model *MADEIRA*

The following considerations have been of great importance for the design of the data model *MADEIRA* as a framework for a visual data analysis environment and discriminate this model from existing ones:

- In order to guarantee intuitive usability of *VIOLA*, *MADEIRA* is restricted to *one* main data structure, namely a multidimensional datacube, and its components. Microdata do not play a significant role in *MADEIRA*.
- The most important operation for navigation in datacubes is aggregation. In order to support aggregation as good as possible,
  - semantics, especially "disjointness" and "completeness" of categories (e. g. all cities of a state do *not* cover the whole state),
  - aggregation levels and hierarchies of categories, and
  - aggregation functions being applicable to a datacube

  need to be modelled explicitly. This is also of great importance for integration of data from different data sources and for consideration of set-valued category instances as is sometimes necessary.
- Various metadata, especially object domains described by macrodata as well as detailed type descriptions of measures have to be integrated into the model to allow exact operator definitions and dataset descriptions for user information. Thus, *MADEIRA* stresses the aspect of *semantic* data definition and usage of this knowledge in data processing, whereas most existing multidimensional data models are restricted to the mere *syntax* of datacubes.
- By applying statistical functions to datacubes, complex measures are calculated, which often cannot be further aggregated to higher levels without accessing the base data. In order to be still able to facilitate efficient and interactive data navigation and also for reasons of flexibility, data of different aggregation levels must be combinable in *one* datacube.

In the following, we will at first define categories and dimensions as basic elements for classifying base data; afterwards dataspaces representing macrodata, which are described by summary and category attributes, are introduced. Finally, aggregation and restriction of dataspaces are formally modelled.

For a set M, let 2^M denote the set of all subsets of M and $\mathbb{N}^M$ the set of all multisets on M. Let $a.a_i$ denote a component of a structure $a = (a_1, \ldots, a_n)$.

3.1 Categories and Dimensions

Let $\mathcal{O}$ be a set of OBJECTS to be considered in a data analysis (e. g. persons, tumours, etc.) and $\mathcal{F}$ be a set of FEATURES, such that each $o \in \mathcal{O}$ is described by a subset of $\mathcal{F}$ (e. g. persons by sex, age, residence or tumours by some medical attributes). In the following, a subset $O \subseteq \mathcal{O}$ generally represents a kind of object-oriented concept (e. g. "all persons"), whereas $\mathrm{inst}(O) \subseteq O$ denotes a particular instance of analysed objects.

Instances of features are described by CATEGORIES (e. g. *Hamburg*, *1998*, or *male*), which are elements of a set $\mathcal{C}$. $\mathcal{C}$ is partitioned into a set $\mathcal{DO} \subset 2^{\mathcal{C}}$ of

pairwise disjoint DOMAINS (e. g. *region, time, age*, etc.), such that each feature $f \in \mathcal{F}$ is described by categories of one unique domain $D_f \in \mathfrak{D}\mathcal{O}$ – let $\mathcal{F}_D$ denote the set of all features described by categories of domain D. We will consider categories some kind of logical statements or predicates over features and objects.

$\mathcal{O}_f$ and $\mathcal{O}_{f,c}$ denote the sets of objects in $\mathcal{O}$ which are describable by feature f or for which category $c \in D_f$ holds for f, respectively. We write "$o \vdash_f c$" if $o \in \mathcal{O}_{f,c}$; e. g. if a person x lives in Germany, "$x \vdash_{residence}$ *Germany*" holds.

Two categories c_1, c_2 of one domain D are called EQUIVALENT ("$c_1 \equiv c_2$") if $\forall f \in \mathcal{F}_D : \mathcal{O}_{f,c_1} = \mathcal{O}_{f,c_2}$. Analogously, SUBSUMPTION of categories ("c_1 is finer than (subsumed by) c_2", "$c_1 \preceq c_2$") is defined as $\forall f \in \mathcal{F}_D : \mathcal{O}_{f,c_1} \subseteq \mathcal{O}_{f,c_2}$. c_1 and c_2 are said to be RELATED ("$c_1 \| c_2$") if $c_1 \preceq c_2$ or vice versa. In a similar way, also disjunction ("$c_1 \vee c_2$" satisfying $\mathcal{O}_{f,c_1 \vee c_2} = \mathcal{O}_{f,c_1} \cup \mathcal{O}_{f,c_2}$), conjunction ("$c_1 \wedge c_2$"), and negation ("$\neg c$") of categories are defined intuitively. Finally, we call two categories DISJOINT with respect to a feature f and a set of objects O ("$c_1 \perp_{f,O} c_2$") if $O \cap \mathcal{O}_{f,c_1} \cap \mathcal{O}_{f,c_2} = \emptyset$.

Whereas subsumption is intended to consider general finer-coarser-relationships between categories (e. g. "*Hamburg* $\preceq$ *Germany*" in the region domain), disjointness needs to be related to single features and object-sets to enable also modelling of set-valued features (with almost no categories being disjoint).

Aggregation levels define groups of categories being typically used together in an analysis for classifying objects described by them. If modelled explicitly, most existing data models (e. g. [7]) consider categories of one level being "of the same granularity". Although covering typical cases (e. g. "all counties of Germany") this approach is to restrictive, e. g. when modelling five-year age-groups with one group "75 and older". Rather often single categories belong to *different* levels of one domain, e. g. cities belong both to a community and a ward level of the region domain. Thus, we only claim a level not to contain *related* categories. (Furthermore, for single-valued features all categories of a level are disjoint.)

Definition 1 (Aggregation levels). *An* (AGGREGATION) LEVEL $le = (D, C)$ *on domain* D *is given by a finite, non-empty set* $\mathrm{dom}(le) = C \subseteq D$ *satisfying* $\forall c_1 \neq c_2 \in \mathrm{dom}(le) : c_1 \nparallel c_2$. $\mathcal{L}_D$ *is the set of all levels on domain* D.

A relation on levels of one domain can easily be derived from subsumption on categories. This relation focuses on providing sensible aggregation paths from finer to coarser levels. Thus, categories of the higher level must in each case be *completely* disaggregated on the lower one. Furthermore, we distinguish cases in which not a complete level, but just a single category of a level is partitioned (as also proposed in [8]):

Definition 2 (A relation on levels). *A level* le *is said to be* FINER *than level* le' *(*$le \lhd le'$*,* cf. Fig. 1*) if*

1. $\exists c' \in le' : c' \equiv \bigvee \mathrm{dom}(le)$ *(just one category disaggregated) or*
2. $\forall c' \in \mathrm{dom}(le') : c' \equiv \bigvee \{c \in \mathrm{dom}(le) | c \preceq c'\}$ *(complete level* le' *disaggregated) – distinguishing two subcases:*

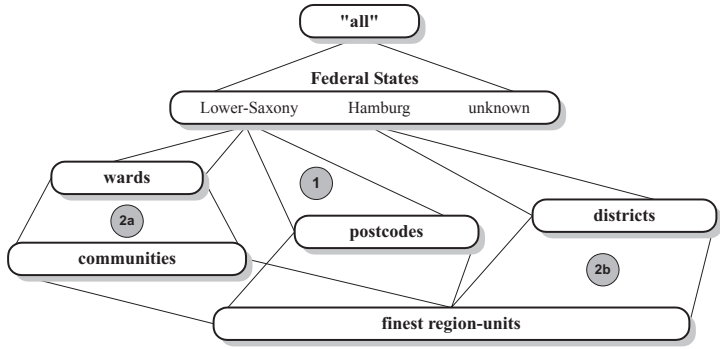

Fig. 1. Example of a category hierarchy on regions

(a) $\bigvee \mathrm{dom}(le) \equiv \bigvee \mathrm{dom}(le')$*, i. e. the levels are "equivalent", or*
(b) this is not the case, i. e. the finer level covers a larger range.

To ensure completeness of disaggregation for a category c', categories subsumed by c' with semantics "c', not otherwise specified" will frequently be introduced in the finer level.

A dimension on domain D primarily defines a category hierarchy of levels covering that part of D which is of interest for the respective application domain.

Definition 3 (Dimensions and category hierarchies). *A* DIMENSION $d = (D, L, le_0)$ *on domain* D *is defined by*

- *a finite set* $L \subseteq \mathcal{L}_D$ *of levels and*
- *a* ROOT LEVEL $le_0 \in L$ *satisfying* $\mathrm{dom}(le_0) = \{c_0\}$ *and* $\forall le \in L : \forall c \in \mathrm{dom}(le) : c \preceq c_0$ *(c_0 corresponds to a value "all").*

Relation "$\lhd$" restricted to L *is called* CATEGORY HIERARCHY *of* d.

Figure 1 outlines a graph-based representation of a category hierarchy on regions showing the respective categories only on state-level. Furthermore, Cases (1), (2a), and (2b) in Definition 2 are marked – with "Hamburg" and "Lower-Saxony" being subdivided in different ways and "finest region units" existing for both of these states.

The sense in defining category hierarchies shall be summarized in the context of using them in a dataflow-based analysis environment: They

1. define typical aggregation paths in multidimensional databases with respect to one category attribute,
2. provide groups of categories aggregating *completely* to another category, and
3. guarantee disjointness of categories (for single-valued features).

Thus, they facilitate correct handling of macrodata – many approaches to summarizability (cf. [9]) and automatic aggregation are made much easier by considering categories statements on features.

3.2 Multidimensional Dataspaces

Macrodata define multidimensional dataspaces, the dimensionality of which is described by category attributes and which contain fact data described by summary attributes in their cells.

Similar to dimensions defining possible values for category attributes, measures define types of summary attributes:

Definition 4 (Measures). *A* MEASURE $m = (T, F)$ *is defined by a set* T *of possible measure values (e. g. the natural or real numbers, often supplemented by some null-values) and a number of calculation functions defining how* m *can be derived from different measures. In detail,* F *is a set of pairs, each consisting of a set of source measures* $\{m_1, \ldots, m_n\}$ *and a function* $f\colon \mathbb{N}^{m_1.T \times \ldots \times m_n.T} \to m.T$.

An example of a measure is the incidence rate with type "real" and calculation from "case count" and "person years" by division and weighting with $\frac{1}{100{,}000}$ – in this case, function f is just defined for value-sets with only *one* element pair. Another example would be the age-standardized rate, which aggregates over a *set* of age-specific case and population data in a weighted sum.

Summary attributes are "instances" of measures related to a specific object-set and defining aggregation functions for aggregating summary data to coarser levels or categories.

Definition 5 (Summary attributes). *A* SUMMARY ATTRIBUTE $sa = (m, O, f^{\text{sum}}, f^{\text{aggr}})$ *on measure* m *is defined by*

- *the set of objects* O *described by* sa,
- *a (partial)* SUMMARY FUNCTION $f^{\text{sum}}\colon 2^O \to m.T$ *defining the calculation of* sa*-values for subsets of* O,
- *a family of (partial)* AGGREGATION FUNCTIONS $(f^{\text{aggr}}_{f,O})_{f \in \mathcal{F}, O \subseteq \mathcal{O}}$ *with* $f^{\text{aggr}}_{f,O}\colon \mathbb{N}^{m.T \times D_f} \to m.T$, *i. e. measure values for given categories* $c_1, \ldots, c_n$ *are aggregated to a value related to the category* $c = c_1 \vee \ldots \vee c_n$.

sa *is called* SUMMARIZABLE *w. r. t. feature* f *and object-set* O *if* $f^{\text{aggr}}_{f,O}$ *is total, it is called* DISJOINT SUMMARIZABLE *if this function is defined for all sets of pairwise disjoint categories.*

Obviously, f^{sum} and f^{aggr} have to be consistent – we do without a formal definition of this relation here.

Definition 6 (Category attributes). *A* CATEGORY ATTRIBUTE $ca = (O, f, C)$ *is defined by a set of objects* O *described by a feature* f *and a set* $\text{dom}(ca) = C \subset D_f$ *of categories describing* f. *Let* $\mathcal{CA}$ *denote the set of all category attributes.*

Note that categories of an attribute are not restricted to one aggregation level as is often the case in existing data models.

Now we are able to define dataspaces describing macrodata:

Definition 7 (Dataspaces). *A* DATASPACE $ds = (O, CA, sa)$ *is defined by the underlying object-set* O*, a set* $CA = \{ca_1, \ldots ca_n\}$ *of category attributes describing different features and a summary attribute* sa *satisfying* $sa.O = O = ca.O$ *for all* $ca \in CA$*.*

The instance of a dataspace is given by an object-set $\mathrm{inst}(O) \subseteq O$ *and a total function* $f^{\mathrm{inst}} : \mathrm{dom}(ca_1) \times \ldots \times \mathrm{dom}(ca_n) \to sa.m.T$*. This function is derived from the summary function* f^{sum} *of* sa *– we leave out the details here.*

Let $\mathcal{DS}$ *denote the set of all dataspaces.*

An example of a dataspace are population data (summary attribute "average yearly population count") by time and region (category attributes) with population count being disjoint summarizable (by addition) over all features except for those relating to the time dimension.

3.3 Aggregation and Restriction of Dataspaces

Aggregation and restriction are the two most important operations on multidimensional dataspaces. Due to preparing definitions in the previous sections, we are able to unite these two neatly in one single operation, namely derivation.

At first, derivation of categories will be introduced, describing how categories are composed of other ones. Based on this composition of categories along one dimension, complete dataspaces can be aggregated from finer ones by grouping values of the respective measure using the associated aggregation function.

Definition 8 (Derivation of categories). *A category* c' *in domain* D *is said to be* DERIVABLE *from a set* $C \subseteq D$ *of categories if* $\exists C_{c'} \subseteq C : c' \equiv \bigvee C_{c'}$*. If categories of* $C_{c'}$ *are even pairwise disjoint, then* c' *is called* DISJOINT DERIVABLE *from* C *(w. r. t. a feature* $f \in \mathcal{F}_D$ *and an object-set* O*).*

Definition 9 (Derivation of dataspaces). DERIVATION *of one dataspace from another is defined by a (partial) function* $f^{\mathrm{der}} : \mathcal{DS} \times \mathcal{CA} \to \mathcal{DS}$*. Given a dataspace* $ds = (O, CA, sa)$ *and a category attribute* ca' *describing feature* f*,* $f^{\mathrm{der}}(ds, ca') = ds'$ *is defined ("*ds' *is* DERIVABLE *from* ds*") if*

1. $ca'.O = O$*,*
2. $\exists ca \in CA : ca.f = f$*,*
3. $\mathrm{dom}(ca') \subseteq \mathrm{dom}(ca)$ *(only restriction, no aggregation needed)* ***or*** sa *is summarizable over* f *and* O*, and*
4. *all categories in* ca' *are derivable*[1] *from* $\mathrm{dom}(ca)$*.*

In this case, $ds' = (O, (CA \setminus \{ca\}) \cup \{ca'\}, sa)$ *with its instance satisfying*

$$\forall c' \in \mathrm{dom}(ca') : f^{\mathrm{inst}\prime}(\ldots, c', \ldots) = sa.f^{\mathrm{aggr}}(\{(f^{\mathrm{inst}}(\ldots, c, \ldots), c) \mid c \in C_{c'}\})$$

over the same object-set $\mathrm{inst}(O)$ *as* ds *(with* $C_{c'}$ *– not necessarily unique – as in Definition 8 and* f^{inst} *denoting the instance of* ds*).*

[1] If all categories are disjoint derivable from $\mathrm{dom}(ca)$, disjoint summarizability of sa is sufficient in (3).

Derivation of dataspaces is the basic operation in navigating through multidimensional databases and will be investigated in detail in the next section.

In this section, fundamental concepts of *MADEIRA* have been introduced. We had to skip over many interesting details of the model, as for example

- more than one summary attribute per dataspace,
- more than one object-set (and feature) described by an attribute,
- multiple object-sets, playing certain *roles* for summary attributes,
- null-values, set-valued features, and
- further basic operations on dataspaces (union, join, etc.),

but the ideas described so far will be sufficient to show in the remainder of this paper how this underlying data model can be used to define and improve basic data analysis facilities of *VIOLA*.

4 Interactive Data Processing Based on *MADEIRA*

Interplay of the data model *MADEIRA* and visual dataflow-based analysis in *VIOLA* is essential for describing analysis sessions to the user: The latter provides him with the exact calculation history of (intermediate) results and the former describes the respective meaning of data values by measure definitions, represented object-sets, and their classification by categories modelled independently of a specific dataspace in dimensions and category hierarchies.

But also the definition of the visual programming language provided by *VIOLA* and internal data processing are significantly influenced and supported by the design of *MADEIRA*. In the following, some examples and corresponding ideas will be outlined without formalizing all details but using and motivating the concepts of *MADEIRA*.

4.1 Flexible Data Management

In view of the fact, that navigation in dataspaces, searching for "interesting" subspaces, and comparing subsets of a dataspace amounts to a substantial part of exploratory data analyses, flexible data management support is of crucial importance. Different useful variants of derivation as introduced in Definition 9 should be used to define operators (i. e. types of derivation nodes) of *VIOLA* supporting typical navigation steps in a kind of visual query language.

Figure 2 shows some examples of deriving dataspaces from a one-dimensional spatial dataspace and visualizing the respective results in different ways. The numbers of derivation operators correspond to the types of derivation introduced below. Three spatial aggregation levels are used: *communities* $\lhd$ *wards* $\lhd$ *regional prosperity* (defining groups of wards with similar average income).

A *derivation node* (parameterized with a category attribute ca') applies $f^{\mathrm{der}}(\cdot, ca')$ to any dataspace *ds* to be processed, with ca' being defined by

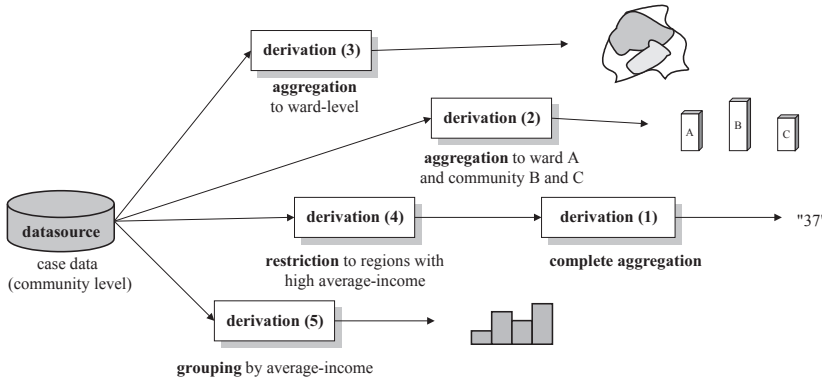

Fig. 2. Instances of derivation in visual queries

- a feature f of an object-set O,
- the corresponding category attribute ca of ds (if existing),
- one of the following five node types (cf. Fig. 2), and
- (in Case 2 to 5) a set of categories $C' \subset D_f$ (e. g. a complete level),

such that $\mathrm{dom}(ca') =$

1. $\bigvee \mathrm{dom}(ca)$ (complete aggregation of one attribute to a total).
2. C' ("pure" derivation).
3. $\{c' \in C' \mid c' \text{ derivable from } \mathrm{dom}(ca)\}$ (only (disjoint) derivable categories).
4. $\mathrm{dom}(ca) \cap \{c \in D_f \mid c \preceq \bigvee C'\}$ (restriction to categories subsumed by C').
5. $\{\bigvee\{c \in \mathrm{dom}(ca) \mid c \preceq c'\} \mid c' \in C'\}$ (non-complete (as opposed to Case 2) grouping to categories in C').

This list and the corresponding implementation is, of course, open for further instances of derivation.

Thus, we see how explicit modelling of levels and categories (their disjointness and equivalence) on the one hand and summary attributes with differentiated aggregation functions on the other allow intelligent and – above all – automatic aggregation of macrodata.

4.2 Selection of Analysis Methods

Graph nodes in *VIOLA* representing statistical analysis functions are determined by the calculated measure m (cf. Definition 4). Thus, selection of necessary and suitable input data for a node is given by the calculation functions $m.F$.

Correspondingly given a dataspace with a summary attribute for measure m_1, the out-port of the respective graph node may be connected to a node representing measure m if $m.F$ contains a tuple $(M, \cdot)$ satisfying $m_1 \in M$. Additionally, further members of M define further input measures, i. e. dataspaces

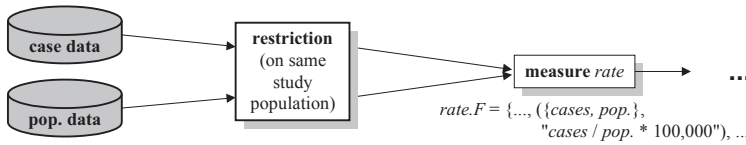

Fig. 3. Calculation of rates from case and population data

needed to calculate m (see the example in Fig. 3). *VIOLA* should help the user to find suitable corresponding graph nodes.

This concept of method selection might be improved by defining a type hierarchy on measures which would allow to group elements of $m.F$. Furthermore, weighting of applicability distinguishing more levels than just "yes" or "no" might be introduced. This extends *VIOLA* to a simple knowledge-based system giving advice which methods are better applicable to a given dataspace than others. For example, some statistical procedures perform "better" on continuous data (measures) than on discrete data or vice versa.

Selection of visualization functions might be supported in a similar way. Additional inclusion of a dataspace's category attributes (their features and dimensions) in rating applicability would be very valuable here, e. g. for specifying that maps are only applicable to spatial data or that certain charts afford one-, two- or three-dimensional data. Finally, *VIOLA* is easily extended with new analysis components by defining a new type of graph node, a new measure and the ways of calculating it from existing measures.

4.3 Caching and Further Query Optimization

Data processing in *VIOLA* is demand-driven. After determining the descriptions (given by summary and category attributes) of requested dataspaces for all nodes of a dataflow-net by propagating descriptions of queries from data sources through all network branches, the calculation of dataspace instances is controlled by output nodes propagating data requests back to the data sources.

Each graph node corresponds to a cache entry in working memory which makes available dataspaces for more efficient processing of further calculations in different parts of a network or even different analysis sessions. Definition 7 provides a semantic description of a cache entry. Before requesting input data due to the network specification and afterwards calculating the result of a node, the cache is searched for dataspaces from which the desired dataspace is derivable (cf. Definition 9). Thus, also cache admission and replacement strategy base on a measure of "generality" of dataspaces, describing how many other (typically requested) dataspaces are derivable from a cache entry.

Definition and types of derivation as introduced in Sect. 4.1 also permit combining subsequent data management procedures in a single derivation step (and dataspace traversal) even w. r. t. different features. Moreover, data source nodes might incorporate subsequent data management steps in a single query unless intermediate results are needed.

Different data source nodes relating to the same database can be processed in a single database query which extends all requested dataspaces to a common "superspace" from which single results can be derived, if necessary.

Summing up, cache management and optimization lead to a transparent transformation of a data analysis graph specified by a user into another network more efficiently processed. This facilitates efficient query processing as multiple queries can be processed in combination, but also poses new questions of how to handle interactive network modifications. Their translation into transformations and extensions of the cache-based derivation graph in an "intelligent" way must combine the paradigm of "dataflow-based visual queries" with known techniques for query optimization.

4.4 Control Structures

MADEIRA also supports control structures allowing for more complex sequences of data analyses. In this paper, we just want to mention loops describing repetitive processing of groups of calculations on different subspaces of a dataspace, e. g. for automatic report generation or incidence monitoring. A "loop counter" can be defined by

- a set of features and respective dimensions,
- an aggregation path consisting of a sequence $le_1 \lhd \ldots \lhd le_n$ of related levels,
- or a level le specifying a set of categories,

the elements of which are used as parameters of a derivation operator in different loop iterations. Besides conditional branching, this defines a simple (especially always terminating), but useful operator supporting typical routine tasks and enhancing the strictly sequential data processing in existing OLAP-tools.

5 Interactive Graphics

Section 4.1 emphasized the importance of data management and navigation in multidimensional dataspaces for performing data analysis flexibly and effectively. This section will propose a different way of implementing these operations aiming at active integration of the user into the exploration process (cf. [10]).

Similar to the idea of visual programming centering around interactive manipulation of the whole process of data analysis, special concepts of interacting with analysis results are needed. Tables and graphics must not only constitute final products, but should also serve as a starting point for further comparative and deepening analyses.

Dynamic queries [11] provide a simple, but powerful example to meet these requirements. They enable the user to restrict dataspaces *interactively* during inspection of results, e. g. using sliders over certain dimensions (or features). This is simply implemented by data management nodes as introduced in Sect. 4.1 (typically of, but not restricted to type (4) or – using appropriate visual grouping operators – (5)) which propagate their results in real-time. This kind of *really*

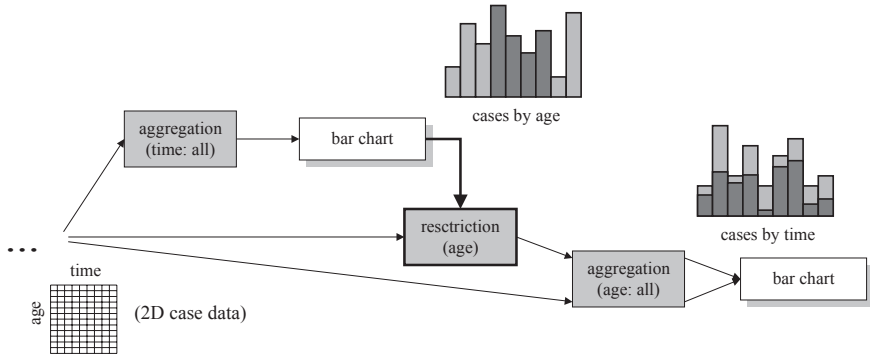

Fig. 4. Linking two charts in a dataflow-programme

interactive navigation facilitates flexible search for interesting subspaces and thus adds a further "dimension" to usually only one- or two-dimensional charts.

Based upon dynamic queries, *linking* of different charts dynamically relates two or more visualization results with each other instead of relating data management and visualization (see e. g. [12]). A typical instance of linking is *linked highlighting*: Subspaces of a dataspace selected in one chart (usually by a subset of one category attribute's elements) are highlighted in another chart that shows a dataspace derived from the former (or a common ancestor). Figure 4 outlines an exemplary implementation in a dataflow-chart: A dataset with case data classified by year of diagnosis and age is visualized in two charts, one after aggregation over each dimension. Selection of age-groups in the first chart highlights the respective shares of cases in all age-groups in the time chart.

Implementations of this concept in existing data analysis systems (e. g. S-Plus or SAS-Insight) are often restricted to linking between charts that show parts of the same dataspace. Besides, they allow no intermediate statistical calculations. Modelling linking in a dataflow-based visual analysis environment as outlined in Fig. 4 provides much more flexibility and can be defined neatly. In terms of Sect. 4.1, the chart serving as the "source" of linking just provides a "normal" data management node with the set C' of categories to define the category attribute for derivation. This node anew can process an *arbitrary* dataspace with a category attribute related to the respective feature. Furthermore, the user is free to carry out further calculations before visualizing the linked dataspace.

6 Related Work

During the last few years, several approaches to modelling multidimensional data have been proposed (cf. [3]). Most of them are based on a relational implementation of macrodata in *star schemes*. Powerful operations, also incorporating microdata, especially support queries on very large business databases which make use of various statistical analysis functions.

In contrast to those, this contribution is intended to define a formal framework of an interactive environment for data analysis which is closely oriented to the "pure" multidimensional model and implements statistical functions by separate operations outside (but via types of calculated measures related to) the core data model. This tool primarily aims at supporting epidemiological studies on "medium-sized" databases with up to some million cases.

One focus of this work is to neatly define categories and category hierarchies as an indispensable foundation of structured and semantically definite navigation on dataspaces. A similar modelling approach (yet not considering interactive navigation) is found in [7].

Tools implementing dataflow-based visual programming in data analysis are mainly found in the domains of image processing and scientific information visualization (e. g. IBM Data Explorer [13] or AVS [14]). However, interactive exploration and intelligent processing of multidimensional databases is possible with restrictions only: typical OLAP-operations are not explicitly provided, immediate database support including caching mechanisms is missing, and interactive graphics are not available.

Another similar approach is the project IDEA [15], which also designs an environment for database-supported data exploration, but without taking special requirements of handling multidimensional data into account.

7 Summary and Future Work

In this paper, some basic elements of a visual data analysis environment based on a formal multidimensional data model have been introduced.

Visual language and formal model cooperate and complete one another in making data analyses exactly comprehensible. The dataflow-view provides the user with an appropriate overview of an analysis session and aims at easy operation and intuitive exploration, whereas the data model (especially the formal representation of category hierarchies, measures and aggregation functions) serves an exact definition of data semantics as well as correct and partially automatic application of analysis methods.

However, several tasks are still subject to future work:

- incorporation of even more semantic metadata for better user support,
- complete implementation of *MADEIRA*, possibly based on an existing data warehousing or OLAP tool, which provides efficient management of multidimensional data,
- integration of *MADEIRA* into a dataflow programming language using some kind of formal workflow modelling,
- elaboration and implementation of the ideas introduced in Sect. 4 and 5, and
- evaluation of all concepts within an epidemiological information system for the cancer registry of Lower-Saxony[2] and (later on) in similar domains — especially w. r. t. acceptance among typical user groups, e. g. epidemiologists.

[2] Parts of the data model are already successfully implemented and used in routine analysis, processing relational base data stored in an ORACLE database. Further-

The crucial task in system design will be to provide an environment which helps the user to concentrate on the exploratory data analysis process itself, to interact with data and methods *directly*, and *not* primarily to operate a tool.

References

[1] S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. *SIGMOD Record*, 26(1):65–74, 1997.

[2] E. Woods, E. Kyral, et al. *OVUM evaluates OLAP*. OVUM Ltd, 1996.

[3] M. Blaschka, C. Sapia, G. Höfling, and B. Dinter. Finding your way through multidimensional data models. In *Proc. Int. Workshop on Data Warehouse Design and OLAP Technology (DWDOT), Wien*, 1998.

[4] D. J. Hand. Intelligent data analysis: Issues and opportunities. In *Advances in Intelligent Data Analysis (Proc. IDA'97)*, pages 1–14. Springer Verlag, 1997.

[5] D. D. Hils. Visual languages and computing survey: Data flow visual programming languages. *Journal on Visual Languages and Computing*, 3(1):69–101, 1993.

[6] F. Wietek. Die Epi-Workbench – ein graphischer Editor zur Modellierung deskriptiver epidemiologischer Studien. *KI–Journal. Schwerpunkt KI und Medizin*, 3:27–31, 1997.

[7] L. Cabibbo and R. Torlone. A logical approach to multidimensional databases. In *6th Int. Conf. on Extending Database Technology (EDBT)*, pages 183–197. Springer Verlag, 1998.

[8] W. Lehner, J. Albrecht, and H. Wedekind. Normal forms for multidimensional databases. In *10th Int. Conf. on Scientific and Statistical Database Management (SSDBM)*, pages 63–72. IEEE Press, 1998.

[9] H.-J. Lenz and A. Shoshani. Summarizability in OLAP and statistical data bases. In *9th Int. Conf. on Scientific and Statistical Database Management (SSDBM)*, pages 132–143. IEEE Press, 1997.

[10] R. R. Springmeyer, M. M. Blattner, and N. L. Max. A characterization of the scientific data analysis process. In *Proc. IEEE Visualization 1992*, pages 235–242.

[11] B. Shneiderman. Dynamic queries for visual information seeking. *IEEE Software*, 11(6):70–77, 1994.

[12] W. S. Cleveland and M. E. McGill, editors. *Dynamic Graphics for Statistics*. Wadsworth & Brooks / Cole Advanced Books and Software, Belmont, CA, 1988.

[13] B. Lucas, G. D. Abram, et al. An architecture for a scientific visualization system. In *Proc. IEEE Visualization 1992*, pages 107–114.

[14] C. Upson, J. Faulhaber, et al. The application visualization system: A computational environment for scientific visualization. *IEEE Computer Graphics and Applications*, 9(4):30–42, 1989.

[15] P. G. Selfridge, D. Srivastava, and L. O. Wilson. IDEA: Interactive data exploration and analysis. In *Proc. SIGMOD'96*, pages 24–35, 1996.

more, a prototypical implementation of a visual dataflow-based environment has been applied in evaluating a study of avoidable mortality in Lower-Saxony.