Eder J., Koncilia C., Mitsche D. (2004). *Analysing Slices of Data Warehouses to Detect Structural Modifications*. Proceedings of the 16th International Conference on Advanced Information Systems Engineering (CAiSE 2004), June 7-11, Riga, Latvia, LNCS 3084, pp. 492-505

Analysing Slices of Data Warehouses to Detect Structural Modifications

Johann Eder* and Christian Koncilia* and Dieter Mitsche§

* University of Klagenfurt
Dep. of Informatics-Systems
{eder,koncilia}@isys.uni-klu.ac.at

§ Swiss Federal Institute of Technology in Zurich (ETHZ)
Institute of Theoretical Computer Science
dmitsche@inf.ethz.ch

Abstract. A source for errors in results of OLAP queries lies in neglecting changes in dimension data of Data Warehouses over time. In addition to the correct representation of such changes in temporal data warehouses it is necessary to discover such changes when they appear in the sources of a data warehouse. We propose to use data mining techniques in form of multidimensional outlier detection to discover unexpected deviations in the fact data suggesting changes in dimension data.

1 Introduction

A data warehouse is an integrated, usually materialized view over several data sources, e.g., data that comes from On-Line Transaction Processing systems, from spreadsheets, from the world wide web or from other sources. Data Warehouses are building blocks for many information systems, in particular systems supporting decision making, controlling, revision, customer relationship management (CRM), etc.[WB97, HLV00]. The most popular architectures are multidimensional data warehouses (data cubes) where facts (transaction data) are "indexed" by several dimensions representing a hierarchical organization of master data.

In this paper, we will address the problem of how to detect changes in these dimensions. To the best of our knowledge only [EKM03] addresses this problem. Whereas the approach presented in [EKM03] was very much trimmed to cope with the important question of performance in data warehouses and focuses on identifying only simple types of structural changes, we will present another approach in this paper that analyses the data in more detail to increase the accuracy of the detected structural changes.

Although data warehouses are typically deployed to analyse data from a longer time period than transactional databases, they are not well prepared for changes in the structure of the dimension data. When analysts query data warehouses with OLAP tools they have to know which dimension data changed. If the analyst wants to analyse population trends of European countries along the last 20 years, he or she has to be aware of the reunification of Germany, the separation of Slovenia, Croatia, etc to come to meaningful conclusions. Although in this example the structural changes are obvious there are also hidden changes. For instance, when data that stems from the web is being analysed the sources might change in a way that effects the structures in the data warehouse without any notification about the changes. Other situations include the passing of transaction data where master data changes are only implicit and probably not easily visible, or changes of the way economic figures are computed (e.g. unemployment rate, price-earnings ratio [Mad03], etc.).

The effects of structural changes in data warehouses and approaches to overcome the problems they cause were already subject of several projects [Yan01, BSH99, Vai01, CS99] including our own efforts [EK01, EKM02] to build a temporal data warehouse structure with means to transform data between structural versions such that OLAP tools work on data cleaned of the effects of structural changes. These proposals allow a correct representation of structural changes once they are known. For practical reasons however, we want to complement these approaches with systems which help to detect such changes.

In this paper we address the following important issue: how can such structural changes be recognized, even if the sources do not notify the data warehouse about the changes. This defensive strategy, of course, can only be an aid to avoid some problems, it is not a replacement for adequate means for managing knowledge about changes. Nevertheless, in several practical situations we trace erroneous results of OLAP queries back to structural changes not known by the analysts and the data warehouse operators. Erroneous in the sense that the resulting data did not correctly represent the state of affairs in the real world.

As means for detecting such changes we propose the use of data mining techniques. In a nutshell, the problem can be described as a multidimensional outlier detection problem. We will discuss two different approaches to solve this problem.

The remainder of this paper is organized as follows: in section 2 we will discuss the different types of structural changes in data warehouse dimensions. In section 3 we will briefly overview different data mining techniques applicable for automatic detection of structural changes. In section 4 we will discuss two different approaches in depth. In section 5 we will present the results of different experiments we conducted. Finally, we conclude in section 6.

2 Types of structural changes

A multidimensional view on data consists of a set of measures arranged by different dimensions [OLA97]. Hence, a cube can also be seen as an n-dimensional array. Measures are numerical values that are referenced by a vector $\nu = (DM_1, DM_2, ..., DM_n)$ where DM_i is a member belonging to dimension D_i [Kur99].

Typical examples of dimensions frequently found in multidimensional databases are *Time*, *Facts* or *Products*. The structure of each dimension is defined

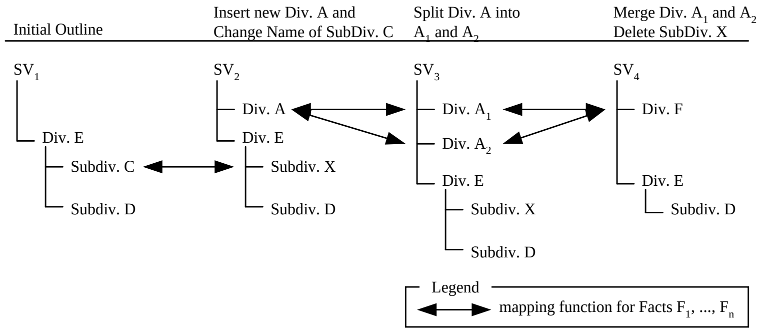
Initial Outline
Insert new Div. A and Change Name of SubDiv. C
Split Div. A into A_1 and A_2
Merge Div. A_1 and A_2 Delete SubDiv. X
SV_1
Div. E
Subdiv. C
Subdiv. D
SV_2
Div. A
Div. E
Subdiv. X
Subdiv. D
SV_3
Div. A_1
Div. A_2
Div. E
Subdiv. X
Subdiv. D
SV_4
Div. F
Div. E
Subdiv. D
Legend
mapping function for Facts F_1, ..., F_n

Fig. 1. An example of structural changes

by a set of categories. For instance, the dimension *Time* could consist of the categories *Year*, *Quarter* and *Month* that are in the hierarchical relation $Year \rightarrow Quarter \rightarrow Month$ ($A \rightarrow B$ means that B rolls-up to A).

Each category consists of a set of dimension members. Dimension members define the instances of a data warehouse schema. For instance, *January*, *February* and *March* are dimension members assigned to the category *Month*.

We will now briefly discuss different types of structural changes. Furthermore, we will argue why some of these structural changes do not need to be detected automatically.

In [EK01] we showed how the basic operations `INSERT`, `UPDATE` and `DELETE` have to be adopted for a temporal data warehouse. With respect to dimension members these basic operations may be combined to represent the following complex operations:

i.) SPLIT: One dimension member M splits into n dimension members, M_1, ..., M_n.
For instance, Figure 1 shows a split operation between the structure versions SV_2 and SV_3 where a division *"Div.A"* splits up into two divisions *"Div.A_1"* and *"Div.A_2"*.

ii.) MERGE: n dimension members $M_1, ..., M_n$ are merged together into one dimension member M.
A merge is the opposite to a split, i. e. a split in one direction of time is always a merge in the opposite direction of time. Consider, for the example given above, that these modifications occur at the timepoint T. For each analysis that requires measures from a timepoint before T for the structure version which is valid at timepoint T we would call these modifications "a split". For each analysis that requires measures from timepoint T for a structure version valid before timepoint T these modifications would be called "a merge".

iii.) CHANGE: An attribute of a dimension member changes, for example, if the product number or the name of a department changes. Such a modification can be carried out by using the update operation defined above.
With respect to dimension members representing facts, CHANGE could mean that the way how to compute measures changes (for example, the

way how to compute the unemployment rate changed in Austria because they joined the European Union in 1995) or that the unit of facts changes (for instance, from Austrian Schillings to EURO).

iv.) MOVE: A dimension member moves from one parent to another, i.e., we modify the hierarchical position of a dimension member.

v.) NEW-MEMBER: A new dimension member is inserted.

vi.) DELETE-MEMBER: A dimension member is deleted.

For two operations, namely NEW-MEMBER and DELETE-MEMBER, there is no need to use data mining techniques to automatically detect these modifications. When loading data from data sources for a dimension member which is new in the data source but does not exist in the warehouse yet, the NEW-MEMBER operation is detected automatically by the ETL-Tool (extraction, transformation and loading tool). On the other hand, the ETL-Tool automatically detects when no fact values are available in the data source for deleted dimension members.

3 Data mining techniques

In this section a short overview of different data mining techniques for automatic detection of structural changes is issued.

The simplest method for detecting structural changes is the calculation of deviation matrices. Absolute and relative differences between consecutive values, and differences in the shares of each dimension member between two chronons can be easily computed - the runtime of this approach is clearly linear in the number of analysed values. Since this method is very fast, it should be used as a first sieve.

A second approach whose runtime complexity is in the same order as the calculation of traditional deviation matrices is the attempt to model a given data set with a stepwise constant differential equation (perhaps with a simple functional equation). This model, however, only makes sense if there exists some rudimentary, basic knowledge about factors that could have caused the development of certain members (but not exact knowledge, since in this case no data mining would have to be done anymore). After having solved the equation (for solution techniques of stepwise differential equations refer to [Dia00]), the relative and absolute differences between the predicted value and the actual value can be considered to detect structural changes.

Other techniques that can be used for detecting structural changes are mostly techniques that are also used for time-series analysis:

- *autoregression* - a significantly high absolute and relative difference between a dimension member's actual value and its predicted value is an indicator for a structural change of that dimension member. Several models to predict data may be used (for instance, the *AutoRegression Moving Average* (p,q)-model or the *AutoRegression Integrated Moving Average*(p,d,q)-model).

- *autocorrelation* - the usage of this method is similar to the method of autoregression. The results of this method, however, can be easily visualized with the help of correlograms.
- *crosscorrelation* and *regression* - these methods can be used to detect significant dependencies between two different members. Especially a very low correlation coefficient (a very inaccurate prediction with a simple regression model, respectively) could lead to the roots of a structural change.
- *discrete fourier transform (DFT)*, *discrete cosine transform (DCT)*, different types of *discrete wavelet transforms* - the maximum difference (scaled by mean of the vector) as well as the overall difference (scaled by mean and length of the vector) of the coefficients of the transforms of two dimension members can be used to detect structural changes.
- *singular value decomposition (SVD)* - unusually high differences in singular values can be used for detecting changes in the measure dimension when analysing the whole data matrix. If single dimension members are compared, the differences of the eigenvalues of the covariance matrices of the dimension members (= *principal component analysis*) can be used in the same way.

In this paper, due to lack of space no detailed explanation of these methods is given, for details refer to [Atk89] (fourier transform), [Vid99] (wavelet transforms), [BD02] (autoregression and -correlation), [Hol02] (SVD and principal component analysis), [Wei85] (linear regression and crosscorrelation).

4 Different data mining approaches

In this section different ways of treating the fact that the data are referenced over a set of n dimensions are presented. Since in data warehouses there is usually a multidimensional view on the data, the techniques shown in the previous section have to be applied carefully. If all structure dimensions are considered simultaneously and a structural change occurred in one structure dimension, it is impossible to detect the dimension that was responsible for this change. Two approaches to solve this problem are presented: the first one analyses matrices that result from grouping all values along one dimension whereas the second one analyses matrices where in all but one structure dimensions a special dimension member is fixed. Finally, a short comparison of the efficiency and runtime complexity of the two approaches is given.

Before the start of one of the two approaches, the whole data matrix should be checked for changes in the measure dimension: differences of the sums of all absolute values of two consecutive chronons are calculated. Under the assumption, that the number of chronons C is very small compared to the number of dimension members in structure dimensions, this step can be neglected for the analysis of overall runtime complexity (only $O(C)$ values have to be analysed). If the differences between two consecutive chronons are substantially bigger than those between other chronons, then this is an indicator for a change in the measure dimension. Changes at this level that are detected must be either corrected or eliminated - otherwise the results in both of the following approaches will be biased by these errors; if one change (e.g. a change in the calculation of the formula) is detected and the data are corrected (e.g. then all values are noted in the same currency), it is recommended to repeat this difference calculation - maybe further changes in the measure dimension can be detected. If no further change in the measure dimension can be recognized, then one of the two following approaches may proceed.

4.1 Approach 1: Grouping along one dimension

This approach consists of different steps of analysing the data and is therefore best explained by the following enumeration:

1.) In the first step of this approach the data are grouped by one structure dimension. The deviation matrices that were described in section 3 can be applied here to detect dimension members that were affected by structural changes.

2.) If the data grouped by one structure dimension can be adequately modelled with a stepwise constant differential equation (or a simple functional equation) then also the deviation matrices that calculate the absolute and relative difference between the model-estimated value and the actual value should be used.

3.) In each structure dimension where one dimension member is known that definitely remained unchanged throughout all chronons (fairly stable so that it can be considered as a dimension member with an average development, mostly a dimension member with rather big absolute values), other data mining techniques such as autocorrelation, autoregression, discrete fourier transform, discrete wavelet transform, principal component analysis, cross-correlation and linear regression can be used to compare this 'average' dimension member with any other dimension member detected in steps 1 and 2. If one of the methods shows big differences between the average dimension member and the previously detected dimension member, then this is an indicator for a structural change of the latter one. Hence, these methods on the one hand are used to make the selection of detected dimension members smaller, on the other hand they are also used to 'prove' the results of the previous steps. However, all these methods should not be applied to a dimension member that is lacking values, whose data are too volatile or whose values are often zero. If no 'average' dimension member is known, the dimension members that were detected in previous steps can also be compared with the sum of the absolute values of all dimension members. In any case, it is for performance reasons recommended to use the method of autocorrelation at first; among all wavelet transforms the Haar method is the fastest.

4.) If in steps 1, 2 and 3 no (or not all) structural changes are detected and one still assumes structural changes, then the values are grouped by $i+1$ structure dimensions, where i ($i = 1 \ldots n-1$, $n =$ number of structure dimensions) is the number of structure dimensions that were used for grouping values in the current step. Again, steps 1, 2 and 3 can be applied.

To make the idea of this approach clear, one small example is given. If you consider the data given in table 1, the whole data set seems to be very volatile at first sight (especially between $year_3$ and $year_4$). Since no changes in the measure dimension can be detected, the approach may proceed by grouping all values along one structure dimension. On the resulting view the differences of shares of dimension members are calculated (this deviation matrix was chosen because it shows the outliers most clearly in this case) - the results are presented in table 2; it is clearly pointed out that there are strong indicators for a structural change in structure dimension SD_2 between $year_1$ and $year_2$, for a structural change in structure dimension SD_3 between $year_2$ and $year_3$ and for a structural change in structure dimension SD_4 between $year_3$ and $year_4$ (in this example with just two dimension members per structure dimension the changes in the one member have to be counted up in the other - it is therefore not known whether between $year_1$ and $year_2$ dimension member SD_{21} or SD_{22} changed. In real-world data warehouses with many more dimension members, however, it usually is clear which dimension member changed). Here, due to lack of space steps 2 and 3 are omitted; if one assumes further structural changes, the detected structural changes have to be corrected, and the above deviation matrix can be calculated once again. In this case, however, all differences of all dimension members between all years are zero - all dimension members stay unchanged throughout the four years. Hence, a further analysis of combined structural changes is useless.

SD_1	SD_2	SD_3	SD_4	$year_1$	$year_2$	$year_3$	$year_4$
SD_{11}	SD_{21}	SD_{31}	SD_{41}	100	20	60	18
SD_{11}	SD_{21}	SD_{31}	SD_{42}	200	40	80	122
SD_{11}	SD_{21}	SD_{32}	SD_{41}	300	60	20	6
SD_{11}	SD_{21}	SD_{32}	SD_{42}	400	80	40	54
SD_{11}	SD_{22}	SD_{31}	SD_{41}	500	500	700	210
SD_{11}	SD_{22}	SD_{31}	SD_{42}	600	600	800	1290
SD_{11}	SD_{22}	SD_{32}	SD_{41}	700	700	500	150
SD_{11}	SD_{22}	SD_{32}	SD_{42}	800	800	600	950
SD_{12}	SD_{21}	SD_{31}	SD_{41}	900	180	220	66
SD_{12}	SD_{21}	SD_{31}	SD_{42}	1000	200	240	394
SD_{12}	SD_{21}	SD_{32}	SD_{41}	1100	220	180	54
SD_{12}	SD_{21}	SD_{32}	SD_{42}	1200	240	200	326
SD_{12}	SD_{22}	SD_{31}	SD_{41}	1300	1300	1500	450
SD_{12}	SD_{22}	SD_{31}	SD_{42}	1400	1400	1600	2650
SD_{12}	SD_{22}	SD_{32}	SD_{41}	1500	1500	1300	390
SD_{12}	SD_{22}	SD_{32}	SD_{42}	1600	1600	1400	2310

SD=structure dimension, SD_{ij}=j-th dimension member in structure dimension i

Table 1. Structural changes in a data warehouse with four structure dimensions

$\Delta(\%)$	$year_{12}$	$year_{23}$	$year_{34}$
SD_{11}	3.19%	0%	0%
SD_{12}	-3.19%	0%	0%
SD_{21}	**-27.22%**	0%	0%
SD_{22}	**27.22%**	0%	0%
SD_{31}	0.8%	**10.17%**	0%
SD_{32}	-0.8%	**-10.17%**	0%
SD_{41}	0.4%	0%	**-33.22%**
SD_{42}	-0.4%	0%	**33.22%**

SD=structure dimension, SD_{ij}=j-th dimension member in structure dimension i,
$\Delta(\%)$=change in share of a dimension member between two consecutive years,
$year_{mn}$=comparison of shares of different dimension members between year m and year n

Table 2. Detection of changes in the structure dimension

4.2 Approach 2: Fixing $n-1$ dimension members

The previous approach might detect many examples in his first step, but not all of them. This approach therefore does not group values along one dimension, but it fixes $n-1$ dimension members and analyses the remaining n-th structure dimension on the matrices addressed by those fixed dimension members.

Consider the following example with two dimensions SD_1 with four dimension members $SD_{11}, \ldots, SD_{14}$ and SD_2 with four dimension members SD_{21}, …, SD_{24}. Their values for four consecutive years for a certain measure are shown in table 3. As boldly pointed out, there has been a structural change in four of the sixteen combinations of dimension members in $year_3$ (marked in bold).

When grouping the values along one structure dimension (as suggested in the previous section), the change cannot be recognized anymore, as table 4 shows. Any data mining technique attempting to find the real change in this matrix will not work.

The solution of this approach to cope with this problem is different from the previous one: in all but one structure dimensions one dimension member is fixed; this combination of fixed dimension members leads to a simple 2-dimensional matrix with the time dimension on the x-axis and the only dimension that was excluded on the y-axis. In other words, instead of aggregating all values along one dimension (as shown in the previous section), the data matrix is cut into different slices that are analysed separately. On this data matrix a simple deviation matrix with relative differences is computed (more sophisticated techniques should not be applied since this might cause a very poor performance): let $data(i, j)$ denote the value of the original 2-dimensional matrix, and let $d(i, j)$ denote the value of the resulting deviation matrix in row i and column j (the value of the i-th dimension member of the excluded dimension between chronons j and $j+1$), then $d(i, j)$ is computed as follows:

$$d(i,j) = \begin{cases} \Delta(i,j)/max(i,j), & if \quad \Delta(i,j)/max(i,j) \leq 0, \\ 1, & \hfill else, \end{cases}$$

SD_1	SD_2	$year_1$	$year_2$	$year_3$	$year_4$
SD_{11}	SD_{21}	200	190	200	205
SD_{11}	SD_{22}	150	155	165	160
SD_{11}	SD_{23}	30	29	**49**	48
SD_{11}	SD_{24}	200	220	215	205
SD_{12}	SD_{21}	200	190	200	205
SD_{12}	SD_{22}	150	155	165	160
SD_{12}	SD_{23}	30	29	**49**	48
SD_{12}	SD_{24}	200	220	215	205
SD_{13}	SD_{21}	200	190	200	205
SD_{13}	SD_{22}	150	155	165	160
SD_{13}	SD_{23}	30	29	**9**	9
SD_{13}	SD_{24}	200	220	215	205
SD_{14}	SD_{21}	200	190	200	205
SD_{14}	SD_{22}	150	155	165	160
SD_{14}	SD_{23}	30	29	**9**	9
SD_{14}	SD_{24}	200	220	215	205

SD_{ij}=j-th dimension member in structure dimension i

Table 3. A small data warehouse with a structural change

SD	$year_1$	$year_2$	$year_3$	$year_4$
SD_{11}	580	594	629	618
SD_{12}	580	594	629	618
SD_{13}	580	594	589	579
SD_{14}	580	594	589	579
SD_{21}	800	760	800	820
SD_{22}	600	620	660	640
SD_{23}	120	116	116	118
SD_{24}	800	880	860	820

SD_{ij}=j-th dimension member in structure dimension i

Table 4. Values of the small data warehouse grouped along one dimension

where $\mathit{\Delta}(i,j) := abs(data(i,j) - data(i,j-1))$, and $max(i,j) := max(abs(data(i,j)), abs(data(i,j{-}1))), \forall i = 1,\ldots,n, \forall j = 2,\ldots,m$. Using this normalization $d(i,j)$ can be interpreted as the probability that a structural change of the i-th dimension member of the dimension on the y-axis has occurred between chronon j and $j+1$. All these 'probabilities' over all possible matrices with the same dimension on the y-axis are summed and divided through the number of possible matrices (since not necessarily all possible combinations of dimension members of different dimensions have to have values in the data warehouse it is recommended to use a counter of occurrences when implementing this approach) - in this way a robust estimate for the probability is computed. Hence, the final result of this approach is a probability matrix P for each structure dimension, where $P_k(i,j)$ indicates the probability of a structural change of the i-th dimension member in structure dimension k between chronon j and $j+1$.

The effectiveness of this approach is illustrated at the data given in table 3. In tables 5 and 6 all possible slices for dimension SD_2 are presented, and in tables 7 and 8 the corresponding deviation matrices are presented. Similarly, all four matrices for dimension SD_1 are computed (matrices not shown here). The final probability matrices for structure dimension SD_1 and SD_2 are presented in tables 9 and 10, respectively. If the probability threshold for a structural change is 10%, then the detected structural changes are all values in tables 9 and 10 that are marked in bold typeface.

$SD_1 = SD_{11}$	$year_1$	$year_2$	$year_3$	$year_4$
SD_{21}	200	190	200	205
SD_{22}	150	155	165	160
SD_{23}	30	29	49	48
SD_{24}	200	220	215	205

SD_{ij}=j-th dimension member in structure dimension i

Table 5. Values of structure dimension SD_2 on slice $SD_1 = SD_{11}$ (same result on slice $SD_1 = SD_{12}$)

$SD_1 = SD_{13}$	$year_1$	$year_2$	$year_3$	$year_4$
SD_{21}	200	190	200	205
SD_{22}	150	155	165	160
SD_{23}	30	29	9	9
SD_{24}	200	220	215	205

SD_{ij}=j-th dimension member in structure dimension i

Table 6. Values of structure dimension SD_2 on slice $SD_1 = SD_{11}$ (same result on slice $SD_1 = SD_{12}$)

$SD_1 = SD_{11}$	$year_{12}$	$year_{23}$	$year_{34}$
SD_{21}	0.05	0.05	0.024
SD_{22}	0.032	0.06	0.03
SD_{23}	0.033	0.408	0.02
SD_{24}	0.09	0.022	0.047

SD_{ij}=j-th dimension member in structure dimension i, $year_{ij}$=difference between $year_i$ and $year_j$

Table 7. Probabilities of a structural change in structure dimension SD_2 on slice $SD_1 = SD_{11}$ (same result on slice $SD_1 = SD_{12}$)

$SD_1 = SD_{13}$	$year_{12}$	$year_{23}$	$year_{34}$
SD_{21}	0.05	0.05	0.024
SD_{22}	0.032	0.06	0.03
SD_{23}	0.033	0.69	0
SD_{24}	0.09	0.022	0.047

SD_{ij}=j-th dimension member in structure dimension i, $year_{ij}$=difference between $year_i$ and $year_j$

Table 8. Probabilities of a structural change in structure dimension SD_2 on slice $SD_1 = SD_{13}$ (same result on slice $SD_1 = SD_{14}$)

P	$year_{12}$	$year_{23}$	$year_{34}$
SD_{11}	0.052	**0.135**	0.03
SD_{12}	0.052	**0.135**	0.03
SD_{13}	0.052	**0.206**	0.025
SD_{14}	0.052	**0.206**	0.025

SD_{ij}=j-th dimension member in structure dimension i, $year_{ij}$=difference between $year_i$ and $year_j$

Table 9. Probability matrix P for structure dimension SD_1

P	$year_{12}$	$year_{23}$	$year_{34}$
SD_{21}	0.05	0.05	0.024
SD_{22}	0.032	0.06	0.03
SD_{23}	0.033	**0.549**	0.01
SD_{24}	0.09	0.022	0.047

SD_{ij}=j-th dimension member in structure dimension i, $year_{ij}$=difference between $year_i$ and $year_j$

Table 10. Probability matrix P for structure dimension SD_2

As can be seen from the probability matrices, there is a relatively high probability (0.549, see table 10) of a structural change of dimension member SD_{23} between $year_2$ and $year_3$; this change even affected the values of all dimension members in structure dimension SD_1 (see table 9): the values of all dimension members between $year_2$ and $year_3$ are higher than the threshold of 10%. In such a case it is useful to increase the threshold to detect the real source of the structural change. However, in real data warehouses, there are usually more than four dimension members in a structure dimension - if in the previous example there were 100 dimension members in structure dimension, then the change of dimension member SD_{23} would only have an insignificant effect on the probability matrix of structure dimension SD_1; in any case, it is important to find the right threshold for the method.

4.3 Comparison of the two approaches

When comparing the two different approaches shown in section 4.1 and 4.2 the differences become clear: whereas the first approach is very much trimmed to cope with the important question of performance in data warehouses and focuses on identifying only simple types of structural changes, the second approach analyses the data in more detail to get a better quality of the detected structural changes. More formally, if D_i denotes the number of dimension members in dimension i, $i = 1, \ldots, n$ (ordered in such a way that $D_1 \geq D_2 \geq \ldots \geq D_n$), then in the first step (which is often the only step when choosing this approach) of the first approach only $O(D_1)$ values have to be analysed, in the i-th step $O(D_1^i)$ (it is again assumed that the number of chronons C is small compared to D_1, therefore it is neglected in the complexity order). In the second approach for each structure dimension $O(D_1 * D_2 * \ldots * D_{n-1})$ matrices have to be computed, since all possible combinations of dimension members in $n-1$ structure dimensions can address a matrix. This is valid for all n structure dimensions, but since only one structure dimension is analysed at once, the overall runtime complexity is also $O(D_1 * D_2 * \ldots * D_{n-1})$ (again under the assumption that C is very small compared to D_i, i=1,. . . ,n). The second approach tries to use the fact that the results of the analysis are not immediately necessary for daily business but are part of long-term strategic planning - therefore it should not cause troubles if the analysis might last for a couple of days.

5 Experiments

Both approaches were tested on different data sets. The running times of the different approaches as well as precision and recall of both approaches are listed in table 11 below.

All running times were measured on a Pentium III 866 MhZ processor with 128 MB SDRAM. In all examples three structure dimensions and one time dimension with the same number of dimension members in each dimension were used, and 4-5 structural changes in certain dimension members were hidden.

As can be seen from table 11, two things are most remarkable: on the one hand the running time of the 'grouping' approach is far lower than the running time of the 'fixing' approach, on the other hand the 'grouping' approach does not detect structural changes when they appear in dimension members with small absolute values - their contribution to the overall sum of all dimension members is vanishingly small; hence these changes remain undetected (in the third data set all changes were designed to happen in such dimension members - and precision and recall are very bad - 0 %!).

$\#d.\ mem$	file size	$time(G)$	$time(F)$	$Precision(G)$	$Recall(G)$	$Precision(F)$	$Recall(F)$
20	1.44 MB	0.11	12.85	100%	100%	100%	100%
35	8.52 MB	0.11	68.66	100%	80%	100%	80%
35	11.1 MB	0.11	67.77	**0%**	**0%**	100%	100%
60	136 MB	0.99	1'521.3	100%	100%	100%	100%

G='Grouping' data along one structure dimension, F='Fixing' $n-1$ structure dimensions; running times for both approaches are measured in seconds; differences in file sizes between second and third data set are due to different data types (integer vs. real) - megabytes represent file sizes of simple textfiles; # d. mem = number of dimension members in each of the three structure dimensions and in the time dimension

Table 11. Running times, precision and recall for both approaches on test data sets

6 Conclusion

We proposed two different approaches for detecting changes in data warehouses. Whereas the 'grouping' approach is faster, the 'fixing' approach may yield better results. In our opinion the 'grouping' approach should be chosen if and only if the results are so urgent that the fixing approach is infeasible due to its high runtime complexity.

As a conclusion it can be said that in principle, the methods work well, but further work is required in the following areas:

- The thresholds for detecting structural changes in the different methods were so far just found by experience and 'trial and error'. Here more sophisticated methods (probably based on sub-sampling the data) for automatic, data-adapted fine-tuning of the thresholds still need to be investigated.
- The performance of the 'fixing' approach may become too bad in huge real-world data warehouses - sophisticated sub-sampling methods might hold the key for dealing with this important issue.

Data mining techniques for discovering changes in the structure or semantics of dimension data of data warehouses should become a standard tool within the data cleaning typically done before data is entered into data warehouses. Either the data production process is designed and operated in way that ensures that the modifications in master data are explicitly reported to the data warehouse or means for detection of such changes are necessary for achieving quality in results of OLAP queries.

References

[Atk89] K. E. Atkinson. *An Introduction to Numerical Analysis.* John Wiley, New York, USA, 1989.

[BD02] P. J. Brockwell and R. A. Davis. *Introduction to Time Series Forecasting.* Springer Verlag, New York, USA, 2002.

[BSH99] M. Blaschka, C. Sapia, and G. Höfling. On Schema Evolution in Multidimensional Databases. In *Proceedings of the DaWak99 Conference*, Florence, Italy, 1999.

[CS99] P. Chamoni and S. Stock. Temporal Structures in Data Warehousing. In *Proceedings of the 1st International Conference on Data Warehousing and Knowledge Discovery (DaWaK'99)*, pages 353–358, Florence, Italy, 1999.

[Dia00] F. Diacu. *An Introduction to Differential Equations - Order and Chaos.* W. H. Freeman, New York, USA, 2000.

[EK01] J. Eder and C. Koncilia. Changes of Dimension Data in Temporal Data Warehouses. In *Proceedings of 3rd International Conference on Data Warehousing and Knowledge Discovery (DaWaK'01)*, Munich, Germany, 2001. Springer Verlag (LNCS 2114).

[EKM02] J. Eder, C. Koncilia, and T. Morzy. The COMET Metamodel for Temporal Data Warehouses. In *Proceedings of the 14th International Conference on Advanced Information Systems Engineering (CAISE'02)*, Toronto, Canada, 2002. Springer Verlag (LNCS 2348).

[EKM03] J. Eder, C. Koncilia, and D. Mitsche. Automatic Detection of Structural Changes in Data Warehouses. In *Proceedings of the 5th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2003)*, Prague, 2003. Springer Verlag (LNCS 2737).

[HLV00] B. Hüsemann, J. Lechtenbörger, and G. Vossen. Conceptual Data Warehouse Design. In *Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW 2000)*, Stockholm, 2000.

[Hol02] J. Hollmen. Principal component analysis, 2002.
URL: http://www.cis.hut.fi/ jhollmen/dippa/node30.html.

[Kur99] A. Kurz. *Data Warehousing - Enabling Technology.* MITP-Verlag, Bonn, 1 edition, 1999.

[Mad03] S. Madnick. Oh, so That Is What You Meant! The Interplay of Data Quality and Data Semantics. In *Proceedings of the 22nd International Conference on Conceptual Modeling (ER 2003)*, Chicago, IL, USA, 2003. Springer Verlag (LNCS 2813).

[OLA97] OLAP Council. *The OLAP Council: OLAP and OLAP Server Definitions.* OLAP Council, 1997.
URL: http://www.olapcouncil.org/research/glossaryly.htm.

[Vai01] A. Vaisman. *Updates, View Maintenance and Time Management in Multidimensional Databases.* Universidad de Buenos Aires, 2001. Ph.D. Thesis.

[Vid99] B. Vidakovic. *Statistical Modeling by Wavelets.* John Wiley, New York, USA, 1999.

[WB97] M. Wu and A. Buchmann. Research Issues in Data Warehousing. In *Datenbanksysteme in Büro, Technik und Wissenschaft (BTW'97) GI-Fachtagung*, Ulm, Germany, 1997.

[Wei85] S. Weisberg. *Applied Linear Regression.* John Wiley, New York, USA, 1985.

[Yan01] J. Yang. *Temporal Data Warehousing.* Stanford University, June 2001. Ph.D. Thesis.