Methods for Heterogeneity Detection During Multi-Dimensional Data Mart Integration

Geno Stefanov

**Summary:**

The paper focuses on the detection of heterogeneities between multi-dimensional data marts. In many cases, data which resides in multiple and independently developed data marts is needed for decision-making. The multi-dimensional model introduces, in addition to the ER data model, dimension and fact entity. As a result of the multi-dimensional model elements, two groups of heterogeneities have been identified – dimension and fact. The former depends on differences between the dimensions’ hierarchies, their members, the names of the members, their levels and dimensions. The latter kind of heterogeneities occurs when facts in different data marts are in different names, values (inconsistent measures), formats or even on a different scale. Therefore, the paper examines and classifies the heterogeneities which can occur during the integration of independently developed data marts and four methods for heterogeneity detection are proposed and discussed. The methods are as follows: method for metadata extraction, method for detecting schema-instance heterogeneities, method for detecting heterogeneities among dimensions and method for detecting heterogeneities among facts. The paper ends with conclusions about the advantages of the proposed methods for heterogeneity detection during the integration of data marts.

**Key words:** multi-dimensional model, data mart, method.

**JEL Classification:** C8, L86.

1. **Introduction**

One problem that needs to be resolved in many practical cases is the integration of data marts that have been developed and operated independently. In many cases decision making requires data that resides in multiple and independently developed data sources. For example, if we want to compare several KPIs (Key performance indicator) from different data marts or to define new KPI, which is defined by the KPIs residing in several different data marts, we face two possible choices – building a new data mart or the integration of the existing ones. Building a new data mart is a costly and time consuming task, so it would be better to look for mechanisms for integration of the existing data marts. Another case that requires marts integration is in the event of mergers and acquisitions of different companies. In this case, for example, one company acquires another company and the data marts in the acquired company should be integrated into the data warehouse of the acquiring company. Here we have the same possible choices as in the previous case.

The multidimensional model (MDM) on which the concept of data mart is based is built out of three basic constructive elements: the facts which are analyzed, the dimensions (coordinates of the fact) and the measures.
which allow for the quantitative evaluation of the facts (Vassiliadis, P., Sellis, T. K., 1999). It is quite often the case that large organizations need integrating independently developed data marts. These data marts should be based on common dimensions and facts, but in many cases different company develop their own data marts and their integration becomes a difficult task. The difficulties stem from the heterogeneities of the MDM elements and from a semantic point of view they can be classified as dimension and fact heterogeneities. Dimension heterogeneities occur when the dimension schema structures, dimension members or the naming of semantically-related dimensions have semantic discrepancies. Fact heterogeneities occur when the measures in different data marts are given different names, values (inconsistent measures), formats or even different scale. There has been research on the problem of resolving the heterogeneities occurring in data mart integration (Tseng, F. S. C., Chen, C-W., 2005), (Torlone, R., 2008) and (Berger, S. and Schref, M., 2008). The main part of data mart integration is resolving the possible heterogeneities. To the best of our knowledge there are no proposals for methods or techniques which will allow the data mart integration process to automatically identify the possible heterogeneities between the integrated MDM models.

2. Data mart construct and heterogeneities

Intuitively, a DM defines one or more measure variables within fact tables, categorized by some dimensions that are organized in hierarchies of levels. Facts and dimensions consist of both their schema and the corresponding instances. For the dimension instances, the commonly accepted term dimension members (or members for short) (Vassiliadis, P., Sellis, T. K., 1999) is used throughout the paper.

In order to achieve the goal of the current paper, the constructing elements of a data mart need to be analyzed and defined. A DM = {F1, ..., Fn, D1, ..., Dm}, consists of non-empty set of dimensions and fact tables (Berger, S., Schref, M., 2008), (Golfarelli, M., Maio, D., Rizzi, S., 1998).

Now let {τ1, ..., τm} be a finite set of data types (e.g. integers) with their domain defined by function dom(τ).

A dimension D ∈ {D1, ..., Dm} of a DM is composed of:

- The dimension schema SD = {LD, S(LD), HD} containing finite, non-empty set of Levels LD = {l1, ..., lm, lAll}, with level schema S(LD) = {S(l1), ..., S(lm)} and the roll-up hierarchy HD ⊆ LD x LD;
- The level Schema Sl ∈ S(LD) of some level li is an attribute schema (K, N1, ..., Nk) with name l, key K (the dimensional attribute) and optional non-dimensional attributes N1,..., Nk. Every Nk attribute ∈ Sl has a domain dom (Nk) = dom (τk);
- The dimension instance d(SD) over schema SD with name d containing a set of members Vd with each v ∈ Vd being a tuple over a level schema Sl and a family of "roll-up" relationships between the member subsets, defined by the dimension hierarchy.

A fact table F ∈ {F1, ..., Fn} of DM is composed of:

- The fact scheme Sf = {Af, Mf} containing a finite, non-empty set of dimensional attributes Af = {A1, ..., An} and a set of measure attributes Mf = {M1, ..., Mn}. Each Ai ∈ Af is linked with a level li ∈ LD of the dimensions D, each Mj ∈ Mf with a τj. The domain of the attributes in Af is defined as dom(Ai) = members(li) and dom(Mj) = dom(τj);
- The fact instance f(Sf), a set of tuples over ([dom(A1) × ... × dom(An)], [dom(M1) × ... × dom(Mm)]). A tuple f ∈ f(Sf) is called a "cell" or "fact". Moreover, we call the values [f(A1), ..., f(An)] the "coordinates" of a cell, modelling the multi-dimensional context for the measures [f(M1), ..., f(Mm)].
### Methods for Heterogeneity Detection During Multi-Dimensional Data Mart Integration

#### Table 1. Heterogeneities between multidimensional data marts

<table>
<thead>
<tr>
<th>Heterogeneity</th>
<th>Level of heterogeneity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimension naming heterogeneities</td>
<td>Dimension Schema</td>
<td>Different name labels for ontologically same schema elements</td>
</tr>
<tr>
<td>Non-corresponding Dimension schemas</td>
<td>Dimension Schema</td>
<td>Two dimension schemas do not have any common equivalent aggregation level</td>
</tr>
<tr>
<td>Inner-level corresponding</td>
<td>Dimension Schema</td>
<td>Two dimension schemas have at least one, but not all equivalent level(s) in common and the base level is different</td>
</tr>
<tr>
<td>Base-level corresponding</td>
<td>Dimension Schema</td>
<td>Two dimension schemas have at least one, but not all equivalent level(s) in common and the base level is equivalent</td>
</tr>
<tr>
<td>Flat corresponding dimension schemas</td>
<td>Dimension Schema</td>
<td>Two dimension schemas have equivalent sets of aggregation levels with identical base levels, but the hierarchies do not match</td>
</tr>
<tr>
<td>Inner level domain conflicts</td>
<td>Dimension Schema</td>
<td>Two attributes in level schemas of autonomous dimensions assign different sets of allowed values (i.e., different domains).</td>
</tr>
<tr>
<td>Base level domain</td>
<td>Dimension Schema</td>
<td>Two attributes in level schemas of autonomous dimensions assign different sets of allowed values (i.e., different domains).</td>
</tr>
<tr>
<td>Non-dimensional attributes domain</td>
<td>Dimension Schema</td>
<td>Two attributes in level schemas of autonomous dimensions assign different sets of allowed values (i.e., different domains).</td>
</tr>
<tr>
<td>Heterogeneous roll-up functions</td>
<td>Dimension Instance</td>
<td>The roll-up functions of two equivalent members define ambiguous parent members</td>
</tr>
<tr>
<td>Non-dimensional value heterogeneity</td>
<td>Dimension Instance</td>
<td>Some N-attribute contains ambiguous values among two equivalent members that belong to levels with equivalent schemas</td>
</tr>
<tr>
<td>Fact naming heterogeneities</td>
<td>Fact Schema</td>
<td>Different name labels for ontologically same attributes</td>
</tr>
<tr>
<td>Non-corresponding fact schemas</td>
<td>Fact Schema</td>
<td>Two fact schemas do not have any equivalent dimensional attribute in common</td>
</tr>
<tr>
<td>Partially corresponding fact schemas</td>
<td>Fact Schema</td>
<td>Two fact schemas have at least one, but not all equivalent dimensional attribute(s) in common</td>
</tr>
<tr>
<td>Domain heterogeneities</td>
<td>Fact Schema</td>
<td>Two attributes in autonomous fact schemas assign different sets of allowed values</td>
</tr>
<tr>
<td>Overlapping Fact cells</td>
<td>Fact instance (measure)</td>
<td>Two fact tables define identical dimensional attributes and overlapping measure attributes</td>
</tr>
<tr>
<td>Different measures</td>
<td>Fact instance (measure)</td>
<td>Two fact tables have all equivalent dimension attributes, but do not have any measures in common</td>
</tr>
</tbody>
</table>
Based on the formal definition of data mart elements, an analysis of the possible heterogeneities is required. Heterogeneities among independent and autonomous data marts may occur either at the schema or at the instance level and involve both dimensions and facts. Generally, heterogeneities among Data Marts covering the same application domain (e.g., sales figures of grocery stores) result from the use of either (1) different modelling patterns and methodologies, or (2) ambiguous domain vocabularies, or (3) a combination of both these factors. In recent years the heterogeneities that are specific to data warehouses and the multidimensional conceptual model are analyzed in (Torlone, R., 2008), (Berger, S., Schref, M., 2008) and (Golfarelli, M., Maio, D., Rizzi, S., 1998). A classification of the possible heterogeneities between multidimensional data marts is proposed along two dimensions: modelling scope (schema level, instance level) and model entity (dimension, fact). Thus five basic levels of heterogeneities are obtained (fig. 1).

A summary of the possible heterogeneities on the different levels are presented in the table 1.

3. Methods for heterogeneities detection

DWs and data marts are basically the implementation of multidimensional data models along with some tools for manipulating the multidimensional data. Data marts implement/represent the multidimensional data instance either as relational systems (ROLAP), proprietary multidimensional systems (MOLAP), or a hybrid of both (HOLAP).

In ROLAP (relational online analytical processing) systems, the multidimensional data model is an E/R model and it is represented as a relational database (RDB) instance.
Although ROLAP uses a relational database to represent the MD instance, the database must be carefully designed for analytical processing. Star and Snowflake are the two best-known relational schemas that are specifically designed for ROLAP MD databases. ROLAP maps operations on multidimensional data model to standard relational operations. An advantage of these systems is that they can be easily integrated into other relational information systems (Kimball, R., Ross, M., 2002). Based on (Kimball, R., Ross, M., 2002), ROLAPs are the most common implementation of multidimensional data mart, four methods for heterogeneity detection are proposed and developed (fig. 2).

3.1. Method for metadata extraction

In order to achieve the idea of detecting heterogeneities a set of data describing the MDM data model is needed. This first method has as its goal to extract metadata about the data marts which are integrated. The method provides the opportunities for extracting the following metadata (fig. 3):

- Metadata coming from the data mart structure including automatically...
identifying dimensions and fact tables, names of dimensions and facts, names of measures, dimensional and non-dimensional attributes, dimensional and non-dimensional domains, automatic identification of facts measures;

- User’s metadata including defining dimensional and non-dimensional attributes within the dimension schema, defining dimension hierarchy.

The mode of operation and executed tasks of the method for metadata extraction can be described in the following steps (fig. 4):

- Determining dimension schema
  - Determining dimensional attributes and their properties and characteristics;
  - Determining non-dimensional attributes and their properties and characteristics;

Fig. 3. Metadata elements
- Determining facts schema
  - Determining dimensional attributes within the fact schema;
  - Determining measures within the fact schema;

**Determining dimension schema**

**Determining dimensional attributes and their properties and characteristics**

**Determining non-dimensional attributes and their properties and characteristics**

**Determining measures within the fact schema**

**Determining dimensional attributes within the fact schema**

---

**Methods for metadata extraction**

**Input:** relational schema $R = \{ E_1, ..., E_n \}$

**Output:** multi-dimensional data model $MDM = \{ E_1 = (A_f, \text{dom}(A_f), \text{dom}(F_1)) \}$

$$f(G_1, ..., G_n) = (A_{E_n} \text{dom}(A_n), \text{dom}(F_n) f(G)) D_n = \langle d(SD1), \text{dom}(l.K1), \text{dom}(l.N1), \text{LD1}, \text{LD1}, ..., D_n = \langle d(SD1), \text{dom}(l.K1), \text{dom}(l.N1), \text{LDn}, S(LDn), HDn \rangle$$

1: for all $E \in R$ do
2: if $(E_i, \text{getPKCount}() = 1)$ then
3: $\text{Add}(E_i)$ as D to MDM
4: else if $(T_i, \text{getFKCount}() > 0)$ then
5: $\text{Add}(E_i)$ as F to MDM

---

*Fig. 4. Tasks within the method for metadata extraction*

The algorithmic description of the method includes all the necessary steps for retrieval and storage of the data mart's elements and their characteristics in the form of metadata (fig. 5).
The result of the implementation of the method for metadata extraction represent the multi-dimensional data model elements extracted from relational schema. The implementation of this method is very important for the correct functioning and implementation of the following methods, considering that the analysis of the extracted metadata will determine the possible heterogeneities between the integrated data marts.
3.2. Method for detecting schema-instance heterogeneities

The second method for detecting heterogeneities is the method for detecting schema-instance heterogeneities. This method provides possibilities for detecting heterogeneities on schema-instance level using the extracted metadata from the method for metadata extraction.

The purpose of this method is to provide indication of differences between the ways of modeling of the integrated data marts. Sometimes, part of the fact context is modelled using elements at the schema once and elements at the instance level across distinct autonomous data marts once. The schema-instance heterogeneities are very difficult to detect automatically; thus a user interaction is needed to verify the existence of this heterogeneities. The mode of operation and executed tasks of the method for detecting schema-instance heterogeneities can be described in the following steps (fig. 6):

- Determining the count of the dimensions within the both of the integrated data marts;
- Ascertaining if the number of dimensions is equal.

The algorithmic description of the method includes all the necessary steps for solving the defined tasks of the method (fig. 7).

---

**Fig. 6. Tasks within the method for detecting schema-instance heterogeneities**

**Method for detecting schema-instance heterogeneities**

**Input:** two multi-dimensional data models – MDM = \{F_1, ..., F_n, D_1, ..., D_m\} и MDM' = \{F_1', ..., F_n', D_1', ..., D_m'\}

**Output:** true or false, respectively for the presence or absence of this kind of heterogeneities

1: for all \(D \in MDM\) do
2: \(\text{countMDM}++\)
3: end for
4: for all \(D \in MDM\) do
5: \(\text{countMDM}'++\)
6: end for
7: if (\(\text{countMDM} \neq \text{countMDM}'\)) then
8: return true
9: else
10: return false
11: end if

**Fig. 7. Algorithmic description of the method for detecting schema-instance heterogeneities**

If the number of dimensions is different in the integrated data marts, this denotes a different number of dimensional attributes in the fact tables schemas.
3.3. Method for detecting heterogeneities among dimensions

The third method for detecting heterogeneities is the method for detecting heterogeneities among dimensions. This method provides opportunities for detecting heterogeneities on dimensions level using the extracted metadata from the method for metadata extraction, namely (fig. 8):
Methods for Heterogeneity Detection During Multi-Dimensional Data Mart Integration

- Detecting naming heterogeneities among (1) dimensions, (2) dimensional attributes, (3) non-dimensional attributes in level schema;
- Detecting dimension schema heterogeneities – (1) non-corresponding dimension schemas, (2) inner-level corresponding dimension schemas, (3) base-level corresponding dimension schemas, (4) flat corresponding dimension schemas, (5) base-level domain heterogeneities, (6) non-dimensional attributes domains;
- Detecting dimension member heterogeneities – (1) heterogeneous roll-up functions and (2) non-dimensional value heterogeneities.

The algorithmic description of the method includes all the necessary steps for detection of heterogeneities which exist on dimension level (fig. 9).

**Method for detecting heterogeneities among dimensions**

**Input:** dimensions D and D’

**Output:** List of all existing heterogeneities among the dimensions D and D’ L

1: for all l ∈ LD do
2:   for all l’ ∈ LD’ do
3:     ds=1-LevenshteinDistance (name (l), name (l’))/count_charact (name (l), name (l’))
4:     if (dataType(l) = dataType(l’) & ds > 0.6) then
5:       l ≈ l’
6:       count++
7:     else
8:       noncorres++
9:   end if
10: end for
11: end for
12: if(count=0 & (1 – LevenshteinDistance (name (D), name (D’))/ count_charact (name (D), name (D’)) < 0.5) then
13:   Add NonCorrespDimSchemasConf (D,D’) to L
14: go to end
15: else
16:   if (name (D) ≠ name (D’)) then
17:     Add DimNameConf (name (D), name (D’)) to L
18:   end if
19:   for all l ∈ LD do
20:     for all l’ ∈ LD’ do
21:       if (l ≈ l’ & (name (l) ≠ name (l’))) then
22:         Add LevelNameConf (name (l), name (l’)) to L
23:       end if
24:       if (l ≈ l’ & dom (l, k) ≠ dom (l’, k’)) then
25:         Add DimDomainConf (dom (l, k), dom (l’, k’)) to L
26:       end if
27:     for all l,N ∈ SI do
28:       for all l’,N’ ∈ S’I do
29:         if (l ≈ l’ & dom (l, Nk) ≠ dom (l’, N'm)) then
30:           Add NonDimDomainConf (dom (l, Nk), dom (l’, N’m)) to L
31:         end if
32:         if (name (l, Nk)≠name (l’, N’m)) then
33: \text{Add NonDimNameConf (name (l, N_k), name (l'_j, N'_{m_j})) to } L
34: \text{end if}
35: \text{end for}
36: \text{end for}
37: \text{end for}
38: \text{end if}
39: \text{if (l'_j \neq l'_j' \& count < noncorres) then}
40: \text{Add InnerSchemaCorresConf (SD, SD') to } L
41: \text{else if (l'_j = l'_j' \& count < noncorres) then}
42: \text{Add BaselevelCorresConf (SD, SD') to } L
43: \text{if (HD \neq HD') then}
44: \text{Add DriftHierarchConf (HD, HD') to } L
45: \text{end if}
46: \text{for all } v \in Vd \text{ do}
47: \text{for all } v' \in Vd' \text{ do}
48: \text{if (v(l,K) = v'(l'.K) \& r'(l'.K) \neq r'(l'.K)') then}
49: \text{Add HeterRollUpFuncConf (v, v') to } L
50: \text{end if}
51: \text{for all } v(l,N) \in Vd \text{ do}
52: \text{for all } v'(l'.N') \in Vd' \text{ do}
53: \text{if (v(l,N) = v'(l'.N') \& v(l,N) \neq v'(l'.N')) then}
54: \text{Add NonDimValueConf (v(l,N), v(l'_j, N'_j)) to } L
55: \text{end if}
56: \text{end for}
57: \text{end for}
58: \text{end for}
59: \text{end for}
60: \text{end if}
61: \text{return } L

\text{Fig. 9. Algorithmic description of the method for detecting heterogeneities among dimensions}

The result of the implementation of the software method for detecting heterogeneities among dimensions represents a list with all identified heterogeneities among two dimensions.

3.4. Method for detecting heterogeneities among facts

The last method for detecting heterogeneities is the method for detecting heterogeneities among facts. This method provides opportunities for detecting heterogeneities among facts using the extracted metadata from the method for metadata extraction, namely (fig. 10):

- Detecting naming heterogeneities among (1) fact tables, (2) dimensional attributes, (3) measures;
- Detecting fact tables schema heterogeneities – (1) non-corresponding fact schemas, (2) partially corresponding fact schemas, (3) domain heterogeneities;
- Detecting fact tables instances heterogeneities – (1) overlapping fact cells, different measures.

The algorithmic description of the method includes all the necessary steps for detection of heterogeneities which exist on the fact table's level (fig. 11).
Methods for Heterogeneity Detection During Multi-Dimensional Data Mart Integration

**Method for detecting heterogeneities among fact tables**

**Input:** fact tables \( F = \{ S_f = \{ A_f, M_f \}, f(S_f) \} \) and \( F' = \{ S'_f = \{ A'_f, M'_f \}, f'(S'_f) \} \)

**Output:** List of all existing heterogeneities among the fact tables \( D \) and \( D' \)

\[
\begin{align*}
1: & \quad \text{for all } A[l] \in A_f \text{ do} \\
2: & \quad \quad \text{for all } A[l'] \in A'_f \text{ do} \\
3: & \quad \quad \quad \text{if } (A[l] = A[l']) \text{ then} \\
4: & \quad \quad \quad \quad \text{count}++ \\
5: & \quad \quad \quad \text{else} \\
6: & \quad \quad \quad \quad \text{noncorres}++ \\
7: & \quad \quad \text{end if} \\
8: & \quad \text{end for} \\
9: & \text{end for} \\
10: & \text{if } (\text{count}=0) \text{ then} \\
11: & \quad \text{Add NonCorrespondingFactSchemasConf } (F,F') \text{ to } L \\
12: & \text{go to end} \\
13: & \text{else} \\
14: & \quad \text{if } (\text{name } (F) \neq \text{name } (F'))\text{ then} \\
15: & \quad \quad \text{Add FactNameConf } (\text{name } (F), \text{name } (F')) \text{ to } L
\end{align*}
\]
Fig. 11 Algorithmic description of the method for detecting heterogeneities among fact tables
The result of the implementation of the software method for detecting heterogeneities among fact tables represents a list with all identified heterogeneities among two fact tables.

Conclusions

In conclusion we can say that the use of the proposed methods for detecting data mart heterogeneities may shorten the process of data mart integration by reducing the risks of unidentified heterogeneities or user's mistakes. The redundancy of unidentified heterogeneities, in conjunction with common mistakes made by users, are the leading causes of data mart integration project failure (inefficient uses of time and data management in the context of running and maintaining data marts for business purposes). To achieve optimal functionality and reduce the aforementioned risks, and above all to streamline and shorten the process of data mart integration, the implementation of the proposed methods for detecting data mart heterogeneities is recommended.

References


