Scientific Journal of Riga Technical University
Computer Science. Applied Computer Systems
2011
Volume 46

On Implicitly Discovered OLAP Schema-Specific Preferences in Reporting Tool

Natalija Kozmina, *Faculty of Computing, University of Latvia,*
Darja Solodovnikova, *Faculty of Computing, University of Latvia,*

***Abstract* – We propose content-based methods for construction of recommendations for reports in the OLAP reporting tool. Recommendations are generated based on preference information in user profile, which is updated implicitly by collecting and analyzing user activity in the reporting tool. Taking advantage of data about user preferences for data warehouse schema elements, existing reports that potentially may be interesting to the user are distinguished and recommended. The approach used for recommending reports is composed of two methods – *cold-start* and *hot-start*.**

***Keywords* – OLAP schema, recommendations, reports, user preferences.**

I. Introduction and Motivation

OLAP applications are built to perform analytical tasks within large amount of multidimensional data. During working sessions with OLAP applications the working patterns can vary. Due to the large volumes of data the typical OLAP queries, performed via OLAP operations by users, may return too much information that sometimes makes further data exploration burdening or even impossible. In case too many constraints are chosen, the result set can be empty. In other cases, when the user tries to explore previously unknown data, OLAP query result may highly differ from expectations of the user. Therefore, a user is rather limited in expressing his/her intentions or likes and dislikes in order to get more satisfying results.

Our experience in using standard applications for producing and managing data warehouse reports at the University of Latvia as well as participation in scientific projects and development of our own data warehouse reporting tool [1] served as a motivation for further studies in the field of OLAP personalization. Data warehouse reporting tool is accessed by users of different groups with distinctive rights, interests and skills. Sometimes during sessions a user has no notion about what kind of data he/she is able to find in data warehouse reports. Moreover, a user might be unaware of a potentially useful report, because, for instance, it has been created recently and the user hasn't examined it yet. In this paper we focus on acquiring user preferences implicitly – either by analyzing his/her previous activities or by learning the structure of the browsed report – in order to suggest him/her other reports that might be helpful. Thus, we propose a way to orient in a variety of data warehouse reports, saving time and effort.

We summarized the research made in the field of personalization in OLAP in our previous works [2], [3]. In [2] we have provided an evaluation in order to point out (i) personalization options described in existing approaches, and their applicability to OLAP schema elements, acceptable aggregations, OLAP operations, (ii) the type of constraints (hard, soft or other) used in each approach, and (iii) methods for obtaining user preferences and collecting user information. In [3] a new method was proposed, which provided exhaustive description of interaction between user and data warehouse using the concept of Zachman Framework [4], [5]. Moreover, in [3] a model of user preferential profile was described. To develop user preference metamodel, various user preference modeling scenarios have been considered, which later have been divided into two groups:

- preferences for the contents and structure of reports (OLAP preferences),
- visual layout preferences.

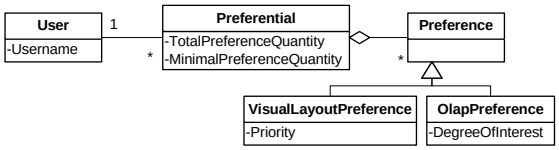

Fig. 1. Preferential profile metamodel (generalized)

Although user preference metamodel contains two distinct classes of preferences – OlapPreference and VisualLayoutPreference (fig. 1) – in this paper we will pay attention only to OLAP preferences. Preferences for visual layout of reports will be covered in a separate paper.

The rest of the paper is organized as follows: section 2 introduces a quick overview of existing OLAP personalization approaches and recommender systems principles, section 3 describes the reporting tool, its logical metadata and OLAP preferences, in section 4 two methods for generation of recommendations for reports are presented, and section 5 concludes the paper.

II. Related Work

A query personalization method that takes user likes and dislikes into consideration exists in traditional databases [6]. Similar ideas seem attractive also for research in the field of data warehousing and topicality of this issue is demonstrated in the recent works of many authors on data warehouse personalization.

There are different aspects of data warehouse personalization. Data warehouse can be personalized at schema level [7], employing the data warehouse multidimensional model, user model and rules for the data warehouse personalization. As a result, a data warehouse user is able to work with a personalized OLAP schema. Users may express their preferences on OLAP queries [8]. In this case,

the problem of performing time-consuming OLAP operations to find the necessary data can be significantly improved. The other method of personalizing OLAP systems is to provide query recommendations to data warehouse users. OLAP recommendation techniques are proposed in [9] and [10]. In [9] former sessions of the same data warehouse user are being investigated. User profiles that contain user preferences are taken into consideration in [10], while generating query recommendations. Another aspect of OLAP personalization is visual representation of data. In [11] authors introduce multiple layouts and visualization techniques that may be used interactively for different analysis tasks.

We also consider methodologies most commonly used in recommender systems [12]. Recommender systems operate with such entities as users and items. A user of the recommender system expresses his/her interest in a certain item by assigning a *rating* (i.e., a numeric equivalent of user's attitude towards the item within a specific numerical scale). In [12] an overview and analysis of algorithms employed in recommender systems is presented. One may distinguish user-based, item-based and hybrid algorithms. A considered user (or item) is referred as an *active* one in order to be distinguished from all other users (or items) of the recommender system.

User-based methods refer to collaborative filtering and user-based k-NN algorithm (introduced by [13]). The similarities between each pair of users are calculated, according to the ratings given to common items that both users have expressed their opinion on. Then the neighborhood is formed around the active user, which consists of users with the closest similarity values to active user's one. Prediction on item's value is made, taking into account ratings of neighborhood users on the same item.

Item-based methods refer to content-based filtering and involve item-based k-NN algorithm (introduced by [14]). The similarities are calculated for each pair of items rated by a common user, which belongs to a subset of users who have assigned a rating to both items in a pair. Active item's predicted value may be computed by means of weighted average of user's ratings on similar items.

Hybrid methods combine principles of user-based and item-based methods.

III. OLAP Reporting Tool

We consider a reporting tool developed at the University of Latvia as an experimental environment for introducing OLAP personalization. The reporting tool is the part of the data warehouse framework [1] developed at the University of Latvia. All operation of the data warehouse framework and the reporting tool as a part of it is based on metadata that are used to describe data warehouse schemata, their storage in relational database and semantics of data stored in a data warehouse, and to accumulate information about reports defined by users on data warehouse schemata.

A. Reporting Metadata

The metadata repository of the data warehouse framework describes a data warehouse at three levels, namely – logical, physical and semantic levels. Besides, the repository also contains reporting metadata (fig. 2), which describes the structure of reports generated by users. Basically, reports are worksheets that contain data items defined by calculations that specify computation formulas from parameters and table columns, which usually correspond to schema elements (measures and attributes). Reports also consist of user-defined conditions and joins between tables.

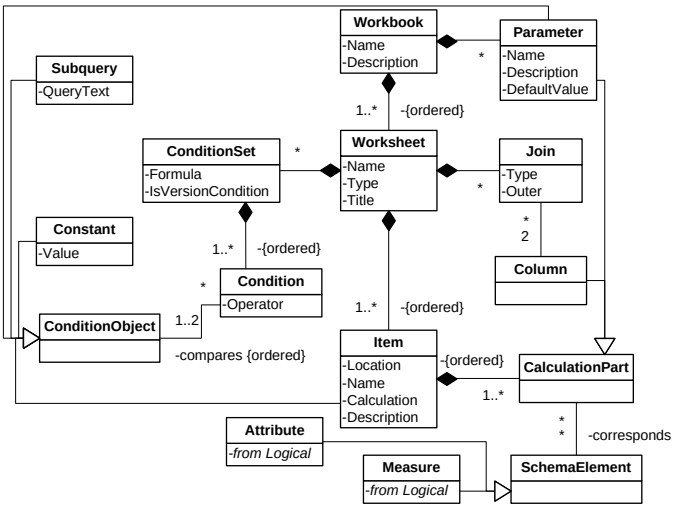

Fig. 2. Reporting metadata

Reports in the tool are defined by developers or users themselves by choosing desired elements of a data warehouse schema and defining conditions, parameters, etc. To define one report, users are allowed to select measures and attributes belonging to one schema. According to the report definition, reporting metadata are created for each report. When a user runs a report, an SQL query is built based on the report definition in reporting metadata [15], and its result is displayed to a user.

B. OLAP Preferences Metadata

Data about user preferences are collected into the OLAP preference metamodel [3], which describes OLAP preferences, classified either as OLAP schema-specific or report-specific.

In the user preference metamodel, OlapPreference is an abstract class, which splits into two classes – *Schema-Specific* and *Report-Specific* preferences. *OlapPreference* class has an attribute – user's degree of interest (*DegreeOfInterest*, *DOI* [6]). *DegreeOfInterest* attribute value is a real number from the interval [0; 1].

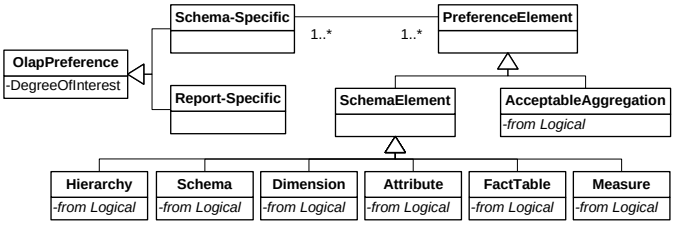

Fig. 3. OLAP preference metadata

The part of the OLAP preference metamodel, which describes Schema-Specific preferences, is depicted in fig. 3. *PreferenceElement* class describes the preference element, which may be either OLAP schema element (e.g. dimension, fact table, attribute, etc.) or acceptable aggregation. Report-Specific preferences refer to preferences for particular reports and data restrictions in reports. Also, there are two ways of collecting user preferences: explicitly (i.e., manually entered by user) or implicitly (i.e., analyzing user's activity by means of web-logs, visited links, etc.). In this paper we consider only implicitly defined OLAP schema-specific user preferences. Report-Specific preferences will be covered in the separate paper.

C. Logical Level Metadata

Schema-Specific preferences are set for the data warehouse schema elements, which are defined in the logical level metadata at the data warehouse repository. The model of the logical level metadata is shown in Fig. 4.

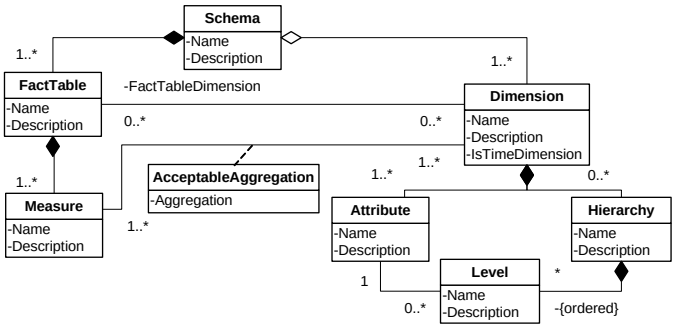

Fig. 4. Logical level metadata

Metadata at the logical level describe the multidimensional data warehouse schema. The logical level metadata are based on the OLAP package of Common Warehouse Metamodel (CWM) [16] and contain the main objects from this package such as dimensions with attributes and hierarchies, fact tables (cubes in CWM) with measures. Fact tables and dimensions are connected by *FactTableDimension* associations. Only dimensions and fact tables connected by *FactTableDimension* associations can be included together in one report. OLAP package of CWM was extended by the class *AcceptableAggregation*, which stores information about aggregate functions (SUM, AVG, COUNT, MIN, MAX) acceptable for each measure and dimension. These metadata are essential for correct queries. The detailed description of all metadata levels of data warehouse used in our approach, including the description of the logical level, is found in the paper [17].

According to the logical level metamodel, data warehouse schema elements are included into the hierarchical structure: a data warehouse schema is composed of interconnected fact tables and dimensions, which are composed of measures and attributes respectively. Dimensions also include hierarchies composed of ordered levels defined by attributes. A fact table belongs to exactly one schema, but a dimension can be shared among multiple schemata. In our approach we take advantage of the hierarchical structure of data warehouse schema elements to automatically estimate degree of interest that a user has got for schema elements located at different levels in the logical level metamodel.

Interconnection of Report Items and OLAP Schema Elements

The models of logical level metadata and reporting metadata are interrelated. Report items are defined by computation formulas from calculation parts that correspond to table columns or parameters. If a calculation part corresponds to a certain dimension attribute or measure, then this schema element is connected to the class *CalculationPart* by the association *'corresponds'* in the reporting metadata. Data warehouse schema elements (measures and attributes) that were used to calculate report items are determined, according to the correspondence associations between calculation parts of the item and schema elements in the reporting metadata. Knowing the attributes and measures that correspond to items of a certain report, it is possible to determine appropriate dimensions and fact tables respectively. Hierarchies (from zero to many) are related to a dimension. Data warehouse schema is defined through association with fact table or dimension.

In our approach we consider only items visible as report columns, rows, data items or page items. Other items used in conditions or joins are omitted, because we regard them only as supplementary to visible report items, which are interesting or useful for a user. For instance, conditions are used to formulate restrictions on data, thus, having an impact on contents of reports, but not on structure. The information about item visibility is obtained from the attribute *Location* of the class *Item* in the reporting metadata.

It is also possible to determine aggregate functions applied to measures to calculate report items. These aggregate functions are derived from the attribute *CalculationFormula* of the class *Item* in the reporting metadata.

IV. Methods for Generation of Recommendations in OLAP Reporting Tool

We introduce two methods, applicable for generation of recommendations in the reporting tool – *hot-start* and *cold-start*. The detailed description of the features of each method is stated in this section.

The hot-start method for generation of recommendations is applied for the user who has had a rich activity history with the reporting system. The cold-start method for generation of recommendations is applied, when (i) a user of the reporting tool starts exploring the system for the first time, or (ii) a user has previously logged in the system, but he/she has been rather passive (count of activity records is lower than a pre-defined *threshold* value). In case (i) it is impossible to generate recommendations by analyzing user previous activity, because it is absent. In case (ii) poor user activity history does not reflect user's interests in full measure, which may lead to generation of either one-sided or too general recommendations, thus, affecting the quality of recommendations. A threshold is defined as a positive constant, which represents the number of records in web-log appurtenant to a certain user and is considered to be sufficient to start employing hot-start method for generation of

recommendations. In other words, a threshold is a borderline between the cold-start and the hot-start approaches. There is no universal value for the threshold, because of various factors that might affect it – for instance, the overall number of reports in the reporting tool, the number of users in the system, the number of reports available for user to access, the overall volume of data warehouse, etc. We suggest that one chooses a threshold value, taking into consideration peculiarities of a particular data warehouse and its reports.

A. Hot-Start Method

The hot-start method is composed of two steps. Firstly, user preferences for data warehouse schema elements are discovered from the history of user's interaction with the reporting tool. Secondly, we determine reports that are composed of data warehouse schema elements, which are potentially the most interesting to a user.

In the hot-start method, weights of schema elements are used to propagate the degree of interest from subelements to the elements of higher level. When a new schema is defined in the data warehouse repository, weights of the new schema elements are calculated and weights of the existing schema elements are adjusted. A *weight* of a schema element is computed in the following way:

- The weight of a schema S_i equals to $W(S_i)=2$.
- The weight of a fact table F_i equals to $W(F_i)=\frac{1}{n}$, where n is the number of fact tables belonging to one schema.
- Since a dimension can belong to multiple schemata, the weight of a dimension is calculated separately for each schema, to which a dimension belongs. The weight of a dimension D_i in a schema S_j equals to $W(D_i,S_j)=\frac{1}{k\cdot m_i}$, where $k=\sum_{l=1}^{n}\frac{1}{m_l}$, n is the number of dimensions belonging to the schema S_j, and $m_i \in m_1,\ldots,m_n$ is the number of schemata, to which the dimension D_i is related. The number of schemata, to which a dimension belongs, is taken into account, because we consider that a dimension used in multiple schemata is less specific for the particular schema. For example, time dimension is involved almost in every schema in a data warehouse and it is not specific for any of the schemata.
- The weight of a measure M_i of a fact table F_j equals to $W(M_i)=\frac{1}{n}$, where n is the number of measures belonging to the fact table F_j.
- The weight of an attribute A_j of a dimension D_j equals to $W(A_i,D_j)=\frac{1}{n}$, where n is the number of attributes belonging to the dimension D_j.
- To compute the degree of interest of a hierarchy, we use the weight of each attribute in that hierarchy. The weight of an attribute A_j, which is a level of a hierarchy H_j, equals to $W(A_i,H_j)=\frac{W(A_i,D_k)}{n}$, where n is the number of attributes that make up levels of the hierarchy H_j, and D_k is the dimension, to which the attribute A_i belongs. Basically, the weight of an attribute in a hierarchy is the weight of the attribute in a dimension divided by the number of levels in the hierarchy. Thus, the weight of a schema element is equal to the sum of the weights of its subelements, except for hierarchies, which do not contribute to the weight of a dimension.

Discovering User Preferences

We maintain and update the degree of interest in OLAP user preferences by analyzing user behaviour in the reporting system. When a user runs a report, items of the report are obtained by means of the reporting metadata analysis. After we determine schema elements used in the report as described in section "Interconnection of Report Items and OLAP Schema Elements", user's degree of interest for each schema element employed in the report is updated hierarchically, starting from the lower level elements. An update of the degrees of interest is conducted according to the algorithm 1, which is executed for each attribute or measure used in the report.

Algorithm 1

Input: User OLAP preferences for schema elements with the degrees of interest for each element and the schema element *E* used in a report, which was executed by the user. DOI(*SE*) is the user's degree of interest for the schema element *SE*, according to the user profile. Output: User OLAP preferences with updated degrees of interest.

```
// if element E is a measure
if E instanceOf(Measure) then
    DOI(E)=DOI(E)+1;
    // getting a fact table,
    // to which the measure E belongs
    F=getFactTable(E);
    DOI(F)=DOI(F)+W(E);
    // getting a schema,
    // to which the fact table F belongs
    S=getSchema(F);
    DOI(S)=DOI(S)+W(F)*W(E);
// if element E is an attribute
else if E instanceOf(Attribute) then
    // getting a dimension,
    // to which the attribute E belongs
    D=getDimension(E);
    DOI(E,D)=DOI(E,D)+1;
    // getting a schema,
    // to which the dimension D belongs
    S=getSchema(D);
    DOI(D,S)=DOI(D,S)+W(E,D);
    DOI(S)=DOI(S)+W(D,S)*W(E,D);
    // getting hierarchies, levels of which
    // correspond to the attribute E
    hierarchies=getHierarchies(E);
    foreach H in hierarchies do
        DOI(H)=DOI(H)+W(E,D)/countLevels(H);
    end
end
```

After updating the degrees of interest for schema elements, we update the degrees of interest of all acceptable aggregations used in the report. For each triple of measure, attribute and aggregate function applied to the measure we obtain acceptable aggregation in the logical level metadata and increase its degree of interest by 1.

Recommending Reports

When degrees of interest are updated in the user's OLAP preferences, we compare the user profile with all reports defined in the reporting metadata and determine reports, which are potentially interesting for the user.

To classify reports into potentially interesting/uninteresting, we adopt the content-based filtering approach [12], which is widely used in item-based recommender systems. We compare user's OLAP preferences with schema elements used in each report to estimate the *hierarchical similarity* between a user profile and a report. The hierarchical similarity between a report and a user profile depends on the number of schema elements used in the report and the degree of interest for these elements set in user profile. Data warehouse schema elements that were used in the report are determined similarly as described in the section "Interconnection of Report Items and OLAP Schema Elements".

Our algorithm for comparing reports with user profiles is based on the following assumptions:

- If the user's degree of interest for a measure *M* is 0, but the degree of interest for a fact table *F* containing *M* is positive, then the user might be also interested in *M*.
- If the user's degree of interest for an attribute *A* is 0, but the degree of interest for a dimension *D* containing *A* is positive, then the user might be also interested in *A*.
- If the user's degree of interest for an attribute *A* is 0, but the degree of interest for a hierarchy *H* containing *A* is positive, then the user might be also interested in *A*.
- If the user's degree of interest for a dimension or fact table *E* is 0, but the degree of interest for a schema *S* containing *E* is positive, then the user might be also interested in *E*.

To calculate the hierarchical similarity, we adopt and adjust the formula used to compute the user-item similarity score for items defined by a hierarchical ontology [18]. Thus, the hierarchical similarity between a report and a user profile is computed as follows:

$$sim = \frac{\sum_{i=1}^{n} DOI(E_i)}{\sum_{j=1}^{m} DOI(G_j)}, \qquad (1)$$

where $E_1,\ldots,E_n$ are schema elements used in the report, and $G_1,\ldots,G_m$ are all schema elements in the user profile.

In our approach, we produce report recommendations of two types: *fact-based* and *dimension-based*. In fact-based recommendations we consider and rate higher only those reports that contain measures from the fact tables with user's positive degree of interest. In dimension-based recommendations we consider and rate higher only those reports that contain attributes from the dimensions with user's positive degree of interest. This way, to produce separately fact-based and dimension-based recommendations, we calculate the value of hierarchical similarity by formula 1 for each report for measures, fact tables and schemata (fact-based similarity), and for attributes, hierarchies, dimensions and schemata (dimension-based similarity) separately. Then *Top-N* reports with the highest fact-based similarity and *Top-N* reports with the highest dimension-based similarity are recommended to the user. If during ranking of the *Top-N* reports, according to the dimension-based similarity, some reports have equal dimension-based similarity value, we order such reports by fact-based similarity value. If during generation of fact-based recommendations some reports have equal fact-based similarity value, we order such reports by dimension-based similarity value and then by summarized degree of interest for aggregate functions applied to the measures of the report.

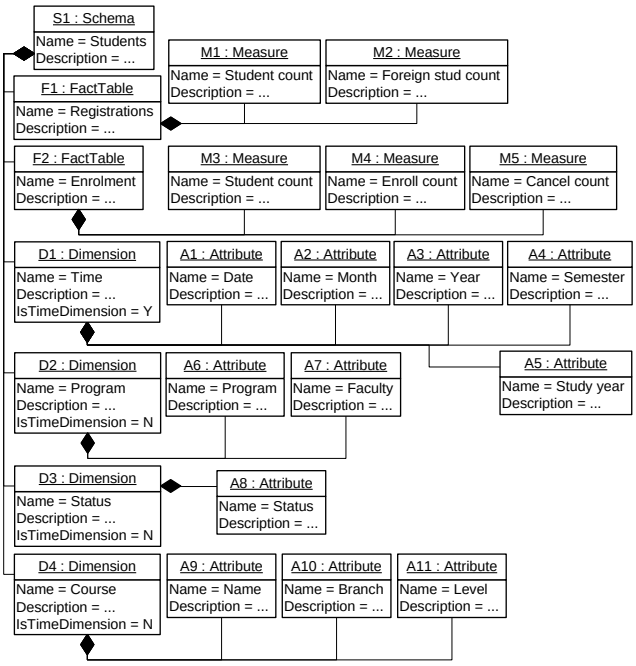

Fig. 5. Students data warehouse schema

To demonstrate our hot-start method for recommending OLAP reports, let us consider an example of a data warehouse schema, which stores data about students.

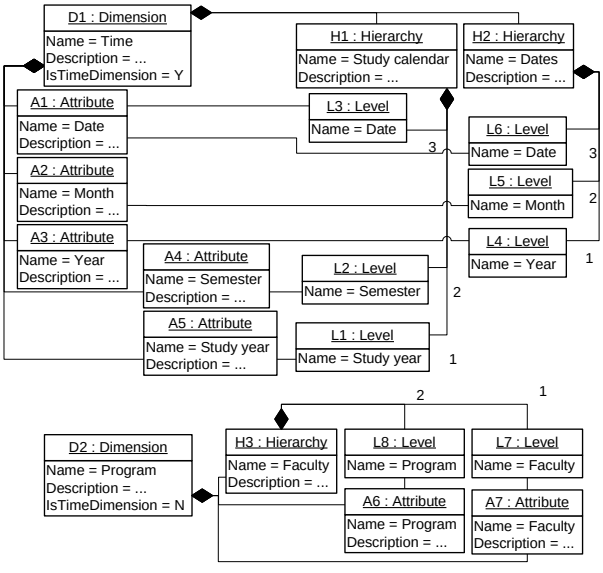

Fig. 6. Hierarchies of dimensions Time and Program

TABLE 1

WEIGHTS AND DOI OF STUDENTS DATA WAREHOUSE SCHEMA AND ITS ELEMENTS

	Schema	Fact tables		Measures					Dimensions				Attributes										
	S_1	F_1	F_2	M_1	M_2	M_3	M_4	M_5	D_1	D_2	D_3	D_4	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}
Weight	2	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{13}$	$\frac{4}{13}$	$\frac{4}{13}$	$\frac{4}{13}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$
DOI	$\frac{4723}{780}$	$\frac{7}{2}$	3	0	7	5	0	4	$\frac{9}{5}$	2	5	$\frac{5}{3}$	0	1	4	4	0	0	4	5	4	1	0

The logical metamodel of the example schema *Students* (Fig. 5) consists of two fact tables: *Registrations* and *Enrolment*, and four dimensions: *Time*, *Program*, *Status* and *Course*. *Registrations* fact table stores information about the number of students, registered for studies at the university per study program (dimension *Program*) and date (dimension *Time*). *Enrolment* fact table contains data about the number of students, enrolled into courses, the number of enrolment actions and the number of enrolment cancellations for each course (dimension *Course*), study program (dimension *Program*), status (dimension *Status*) and date (dimension *Time*).

Dimensions *Time* and *Program* contain hierarchies with corresponding levels, which are shown in Fig. 6.

Let us now compute weights of the schema elements.

Suppose that *Time* dimension is used in 3 other schemata additionally to the *Students* schema, and other dimensions *Program*, *Status* and *Course* belong only to the *Students* schema. Weights of the elements are shown in table 1 and weights of the hierarchy levels are shown in table 2.

Assume that user's degrees of interest for the schema elements computed by the algorithm 1 are such as shown in table 1 row DOI, and the user's degrees of interest for hierarchies with levels composed of attributes are such as shown in table 2 row DOI.

TABLE 2

WEIGHTS OF HIERARCHY LEVELS AND DOI OF THE HIERARCHIES

	Hierarchies			Attributes/Hierarchy Levels							
				Hierarchy H_1			Hierarchy H_2			Hierarchy H_3	
	H_1	H_2	H_3	A_5	A_4	A_1	A_3	A_2	A_1	A_7	A_6
Weight				$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{4}$	$\frac{1}{4}$
DOI	$\frac{4}{15}$	$\frac{1}{3}$	1								

To calculate the hierarchical similarity, let us consider two example reports. The first report is R_1 – *Average foreign student count for each study program per semester*. The second report is R_2 – *Total student count enrolled into courses for each faculty per year*. The hierarchical similarity values for the reports R_1 and R_2 are computed separately for fact-based recommendations $simF_{R1}$ (formula 2) and $simF_{R2}$ (formula 3), and for dimension-based recommendations $simD_{R1}$ (formula 4) and $simD_{R2}$ (formula 5) respectively.

$$simF_{R1} = \frac{1}{DOI(S_1)+DOI(F_1)+DOI(F_2)+DOI(M_1)+\ldots+DOI(H_3)} \cdot \left(DOI(M_2)+DOI(F_1)+DOI(S_1)\right) \approx 0.26 \quad (2)$$

$$simF_{R2} = \frac{1}{DOI(S_1)+DOI(F_1)+DOI(F_2)+DOI(M_1)+\ldots+DOI(H_3)} \cdot \left(DOI(M_3)+DOI(F_2)+DOI(S_1)\right) \approx 0.22 \quad (3)$$

$$simD_{R1} = \frac{1}{DOI(S_1)+DOI(F_1)+DOI(F_2)+DOI(M_1)+\ldots+DOI(H_3)} \cdot \left(\begin{array}{l} DOI(S_1)+DOI(D_2)+DOI(A_6)+DOI(H_3)+ \\ +DOI(D_1)+DOI(A_4)+DOI(H_1) \end{array}\right) \approx 0.24 \quad (4)$$

$$simD_{R2} = \frac{1}{DOI(S_1)+DOI(F_1)+DOI(F_2)+DOI(M_1)+\ldots+DOI(H_3)} \cdot \left(\begin{array}{l} DOI(S_1)+DOI(D_2)+DOI(A_7)+DOI(H_3)+ \\ +DOI(D_1)+DOI(A_3)+DOI(H_2) \end{array}\right) \approx 0.30 \quad (5)$$

According to the fact-based similarity values between OLAP preferences in the user profile and reports R_1 and R_2, the report R_1 is ranked higher than the report R_2, but in compliance with the dimension-based similarity values, these reports are ordered the other way.

B. Cold-Start Method

Instead of parsing user activity as in the hot-start method, we propose the cold-start method, which is suitable for the user who is either a newbie or a passive one. The essence of the cold-start method is composed of two components: firstly, structural analysis of existing reports is performed, and secondly, likeliness between two selected reports is revealed.

In the cold-start method, by *report's structure* we mean data warehouse schema elements and acceptable aggregate functions, which are related to items of a certain report. We discover schema elements used in a report as described in section "Interconnection of Report Items and OLAP Schema Elements" and define a report's structure. Each report is

represented as a *Report Structure Vector* (*RSV*) by formula 6, which is of the following form:

$$RSV = ((e_{11}, e_{12}, \ldots, e_{1k_1}), \ldots, (e_{n1}, e_{n2}, \ldots, e_{nk_n})), \qquad (6)$$

where e_{ik_i} is a vector coordinate, i.e., a binary value that indicates presence (equals 1) or absence (equals 0) of the instance of the report structure element, k_i is the number of elements in *i*-th structure, *i* is the index number of each structure (i = 1, 2, …, n), n is the total number of distinct structure elements in reports. In our case n = 7 as there is a finite set, *S*, of 7 elements, *S* = {attribute, measure, fact table, dimension, schema, acceptable aggregation, hierarchy}.

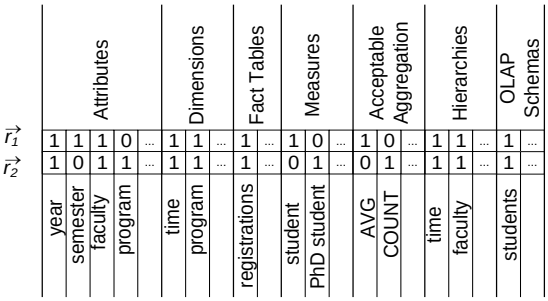

Fig. 7. Two instances of report structure vector (RSV)

Two instances of *RSV* depicted in Fig. 7 provide an example of *RSV* application:

- vector $\vec{r}_1$ describes the structure of the report *R1 – Average student count for each faculty per semester*,
- vector $\vec{r}_2$ describes the structure of the report *R2 – Total PhD student count for each study program per year*.

Both reports belong to the same OLAP schema (*Students*), utilize common hierarchies (*Time*: year → semester, *Faculty*: faculty → program), share a fact table (*Registrations*) and dimensions (*Time*, *Program*). The sets of attributes, measures and acceptable aggregations in the reports *R1* and *R2* are not equal. Note that *RSV* includes all OLAP schemas, their elements (attributes, measures, etc.) and acceptable aggregations. We used *"…"* to substitute other elements of the report structure, because they are not essential for current analysis.

To measure likeliness (also referred to as similarity) is offered to make use of Cosine/Vector similarity. Cosine/Vector similarity was introduced by Salton et al. [19] in the field of information retrieval in order to calculate similarity between a pair of documents by interpreting each document as a vector of term frequency values. Later Breese et al. [20] adopted this formalism in collaborative filtering. In [20] users were treated as documents and items' user rating values as term frequency values. In recommender systems literature Cosine/Vector similarity is extensively used [21], [22], [23], etc. to compute a similarity coefficient for a pair of users (in collaborative filtering) or items (in content-based filtering).

In order to estimate quantitatively the difference between the reports *R1* and *R2*, Cosine/Vector Similarity is applied. Cosine/Vector similarity of the vectors $\vec{r}_1$ and $\vec{r}_2$ is calculated by formula 7:

$$sim = \frac{\vec{r}_1 \cdot \vec{r}_2}{|\vec{r}_1| * |\vec{r}_2|}, \qquad (7)$$

where *"·"* is the dot-product of two vectors (i.e, the sum of pairwise products of vectors' coordinates) and $|\vec{r}_i|$ is the length of each vector (i = 1, 2).

Similarity value, *sim*, of the pair of vectors $\vec{r}_1$ and $\vec{r}_2$ in Fig. 7 is: $sim = \frac{\vec{r}_1 \cdot \vec{r}_2}{|\vec{r}_1| * |\vec{r}_2|} = \frac{8}{\sqrt{11} * \sqrt{11}} \approx 0{,}727$.

Discovering Similarities

The cold-start approach is oriented on providing users with recommendations while working with a certain report. Assume that the report browsed by the user at the moment is called an *active* report. Thus, in order to generate cold-start recommendations, we propose to calculate similarity among the active report and all the rest of the data warehouse reports using report structure vectors (*RSV*). Taking into consideration the fact that (i) a new report might be created or (ii) there might be changes in existing reports' structure, *RSV* and *sim* values have to be recalculated dynamically every time any of the mentioned events takes place.

Recommending Reports

Finally, *Top-N* recommendations, i.e., links to the reports with *N* highest *sim* values sorted in descending order are shown to the user.

V. Conclusions and Future Work

We proposed an approach, which involves two methods – the hot-start method and the cold-start method. The hot-start method defines user OLAP preferences in a reporting tool by means of analyzing user's past activity. In terms of the hot-start method an algorithm for discovering user preferences for data warehouse schema elements and computing degree of interest (*DOI*) in each report was proposed. If the number of records in user's activity log is insufficiently high (i.e., lower than a pre-defined threshold value), the cold-start method is applied. The cold-start method examines the structure of the report being browsed at the moment and calculates the similarity of it with the rest of the reports taking report structure vector (*RSV*) of each considered report as input data. The ultimate goal of our approach is to provide user with recommendations on data warehouse reports.

In our preference metamodel presented in [3], user preferences are classified as OLAP preferences (i.e., involving contents (report-specific) and structure of reports (schema-specific)) and visual preferences (i.e., involving visual representation of reports, for instance, the type of table, graph, font, etc.). However, in this paper the approach is limited to producing recommendations based on automated processing of OLAP schema-specific user preferences.

There are several subjects of our future work; we plan to estimate the quality of recommendations for the group of users

of reporting tools with different rights (e.g., students, professors, staff, etc.), and to extend and complete the approach proposed in this paper by adding methods for (i) explicit definition and processing of OLAP and visual user preferences, (ii) implicit handling of report-specific user preferences to state the most useful data in reports, (iii) collecting and taking advantage of demographical information about users, and (iv) involving collaborative filtering. Evaluation of the methods presented in this paper will follow.

Acknowledgments

This work has been supported by ESF project No. 2009/0216/1DP/1.1.1.2.0/09/APIA/VIAA/044.

REFERENCES

1. **Solodovnikova, D.** Data Warehouse Evolution Framework. *In: Proceedings of the Spring Young Researcher's Colloquium On Database and Information Systems (SYRCoDIS'07)*, Moscow, Russia, 2007, [Online] http://ceur-ws.org/Vol-256/submission_4.pdf [Accessed 20.07.2011]
2. **Kozmina, N., Niedrite, L.** Research Directions of OLAP Personalizaton. *In: Proceedings of the 19th International Conference on Information Systems Development (ISD'10)*, Prague, Czech Republic, 2010. To be published.
3. **Kozmina, N., Niedrite, L.** OLAP Personalization with User-Describing Profiles. *In: Forbrig, P., Günther, H. (eds.) BIR 2010*. LNBIP, Springer, Heidelberg, 2010, vol. 64, pp. 188-202.
4. **Zachman, J. A.** The Zachman Framework: A Primer for Enterprise Engineering and Manufacturing. *In: Zachman International*, 2003.
5. *The Zachman Framework™ for Enterprise Architecture*, [Online] http://zachmaninternational.com/2/production/C4/downloads/Zachman_Framework.pdf [Accessed 20.07.2011]
6. **Koutrika, G., Ioannidis, Y.E.** Personalization of Queries in Database Systems. *In: Proceedings of the 20th International Conference on Data Engineering (ICDE'04)*, Boston, MA, USA, 2004, pp. 597-608.
7. **Garrigós, I., Pardillo, J., Mazón, J.-N., Trujillo, J.** A Conceptual Modeling Approach for OLAP Personalization. *In: Laender, A.H.F. (ed.) ER 2009*. LNCS, Springer, Heidelberg, 2009, vol. 5829, pp. 401-414.
8. **Golfarelli, M., Rizzi, S.** Expressing OLAP Preferences. *In: Winslett, M. (ed.) SSDBM 2009*. LNCS, Springer, Heidelberg, 2009, vol. 5566, pp. 83-91.
9. **Giacometti, A., Marcel, P., Negre, E., Soulet, A.** Query Recommendations for OLAP Discovery Driven Analysis. *In: Proceedings of the 12th ACM International Workshop on Data Warehousing and OLAP (DOLAP'09)*, Hong Kong, 2009, pp. 81-88.
10. **Jerbi, H., Ravat, F., Teste, O., Zurfluh, G.** Preference-Based Recommendations for OLAP Analysis. *In: Proceedings of the 11th International Conference on Data Warehousing and Knowledge Discovery (DaWaK'09)*, Linz, Austria, 2009, pp. 467-478.
11. **Mansmann, S., Scholl, M. H.** Exploring OLAP Aggregates with Hierarchical Visualization Techniques. *In: Proceedings of 22nd Annual ACM Symposium on Applied Computing (SAC'07)*, Multimedia & Visualization Track, Seoul, Korea, 2007, pp. 1067-1073.
12. **Vozalis, E., Margaritis, K.G.** Analysis of Recommender Systems Algorithms. *In: Proceedings of the 6th Hellenic European Conference on Computer Mathematics and its Applications (HERCMA'03)*, Athens, Greece, 2003, pp. 732-745.
13. **Resnick, P., Iacovou, N., Sushak, et al.** Grouplens: An Open Architecture for Collaborative Filtering of Netnews. *In: ACM 1994 Conference on Computer Supported Cooperative Work*, New York, NY, 1994, pp. 175-186.
14. **Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.T.** Item-based Collaborative Filtering Recommendation Algorithms. *In: Proceedings of the 10th International World Wide Web Conference (WWW'10)*, Hong Kong, 2001, pp. 285-295.
15. **Solodovnikova, D.** Building Queries on Multiple Versions of Data Warehouse. *In: Haav, H.-M., Kalja, A. (eds), Databases and Information Systems V - Selected Papers from the 8th International Baltic Conference DBIS 2008*, IOS Press, 2008, pp. 75–86.
16. *Object Management Group: Common Warehouse Metamodel Specification, v1.1*, [Online] http://www.omg.org/cgi-bin/doc?formal/03-03-02 [Accessed 20.07.2011]
17. **Solodovnikova, D.** Metadata to Support Data Warehouse Evolution. *In: Proceedings of the 17th International Conference on Information Systems Development (ISD'08)*, Paphos, Cyprus, 2008, pp. 627-635.
18. **Maidel, V., Shoval, P., Shapira, B., Taieb-Maimon, M.** Ontological Content-based Filtering for Personalised Newspapers: A Method and its Evaluation. *Online Information Review*, 2010, vol. 34 issue 5, pp. 729-756, [Online] http://www.emeraldinsight.com/journals.htm?issn=1468-4527&volume=34&issue=5 [Accessed 20.07.2011]
19. **Salton, G., McGill, M.** *Introduction to Modern Information Retrieval*. McGraw-Hill Inc., New York, NY, USA, 1983.
20. **Breese, J.S., Heckerman, D., Kadie, C.** Empirical Analysis of Predictive Algorithms for Collaborative Filtering. *In: Proceeding of the 14th Conference on Uncertainty in Artificial Intelligence (UAI'98)*, Madison, WI, USA, 1998, pp. 43-52.
21. **Vozalis, M., Margaritis, K.G.** Enhancing Collaborative Filtering with Demographic Data: The Case of Item-based Filtering. *In: Proceedings of the 4th International Conference on Intelligent Systems Design and Applications (ISDA'04)*, Budapest, Hungary, 2004, pp. 361-366.
22. **Rashid, A.M., Karypis, G., Riedl, J.** Influence in Ratings-Based Recommender Systems: An Algorithm-Independent Approach. *In: Proceedings of the 5th SIAM International Conference on Data Mining*, Newport Beach, CA, USA, 2005, pp. 556-560.
23. **Adomavicius, G., Manouselis, N., Kwon, Y.-O.** Multi-Criteria Recommender Systems. *In: Ricci, F., et al. (eds) Recommender Systems Handbook*, Springer, Springer Science+Business Media LLC, 2011, Part 5, pp. 769-803.

Natalija Kozmina obtained her master's degree in computer science from the University of Latvia in Riga in 2010. Her master's thesis was dedicated to personalization in data warehouses. Currently she is a Ph. D student in computer science.

She works at the University of Latvia since 2007. She took part in e-University project and worked at the Information Technology Department as a Developer. At the end of 2009 she joined the Computer Science Department of the University of Latvia and since that time participates in the ESF project 'Computer Science Applications and its Connections with Quantum Physics' (activity 'Research in Data Warehousing') as a Scientific Assistant.

Her current research interests cover the field of data warehousing and OLAP, mainly OLAP Personalization, user-oriented data warehousing and OLAP, as well as data warehouse business requirement modeling and recommender systems.
e-mail: natalija.kozmina@lu.lv

Darja Solodovnikova received her Ph.D. in computer science from the University of Latvia in Riga in 2011. The doctoral thesis covered the topics of evolution in data warehouses.

In 2007 she joined the Computer Science Department of the University of Latvia as a Teaching Assistant. Since 2011 she is a Docent at the Computer Science Department of the University of Latvia. Since 2004 she also works as a Database Analyst at the Information Technology Department of the University of Latvia. In 2010 she joined the ESF project 'Computer Science Applications and its Connections with Quantum Physics' (activity 'Research in Data Warehousing') as a Researcher.

Her current research interests are in the field of data warehousing and OLAP, especially evolution of data warehouses, multiversion data warehouses, user-oriented data warehousing and OLAP, OLAP personalization.
e-mail: darja.solodovnikova@lu.lv