Visualization of Data Mining Rules using OLAP

Basilis Boutsinas¹, Eirini Gkiza²
MIS & Business Intelligence Lab, Dept. of Business Administration, University of Patras, 26500 Rio, Patras, Greece
¹vutsinas@upatras.gr; ²eirkgkiza@upatras.gr

Abstract: Data Mining is an emerging knowledge discovery process of extracting previously unknown, actionable information from very large scientific and commercial databases. Usually, a data mining process extracts rules by processing high dimensional categorical and/or numerical data. However, in the data mining context the user often has to analyze hundreds of extracted rules in order to grasp valuable knowledge. Thus, the analysis of such rules by means of visual tools has evolved rapidly in recent years. Visual data mining attempts to take advantage of humans’ ability to perceive pattern and structure in visual form. Researchers have developed many tools to visualize data mining rules. However, few of these tools can handle effectively more than a few dozens of data mining rules. In this paper, we propose a new visualization technique of data mining rules based on OLAP. More specifically, the proposed technique utilizes the standard two-dimensional cross-tabulation table of most OLAP models in order to visualize even a great number of data mining rules. The advantage of the proposed technique is that the user can choose to "drill down" on specific subsets of such rules. We also present experimental results that demonstrate how the proposed technique is useful and helpful for analyzing and understanding extracted data mining rules.

Keywords: Visualization; Data Mining; OLAP

I. INTRODUCTION

Data Mining is an emerging knowledge discovery process of extracting previously unknown, actionable information from very large scientific and commercial databases. It is imposed by the explosive growth of such databases and it is applied to various different domains, from health and finance to education [35] and music [37]. Usually, a data mining process extracts rules by processing high dimensional categorical and/or numerical data. Classification, clustering and association are the most well known data mining tasks.

Classification is one of the most popular data mining tasks. Classification aims at extracting knowledge which can be used to classify data into predefined classes, described by a set of attributes. The extracted knowledge can be represented using various schemas. Decision trees, "if-then" rules and neural networks are the most popular such schemas. Note that there are transformation techniques proposed in the literature which can transform both decision trees and neural networks into "if-then" rules [10,33,62]. Thus, throughout this paper, we shall refer to the output of a classification algorithm as classification rules.

A lot of algorithms have been proposed in the literature for extracting classification rules from large relational databases, such as symbolic learning algorithms including decision trees algorithms (e.g. ID3 [61]), rule based algorithms (e.g. CN2 [20]) and logic-based algorithms (e.g. [57]), statistic-based algorithms (e.g. CART [14]), connectionist learning algorithms (e.g. back-propagation networks [64]), instance-based algorithms (e.g. PEBLS [22]) and hybrid algorithms (e.g. [10]).

Association rules can be used to represent frequent patterns in data, in the form of dependencies among concepts-attributes. In this paper, we consider the special case, that is known as the market basket problem, where concepts-attributes represent products and the initial database is a set of customer purchases (transactions). This particular problem is well-studied in data mining. We consider association rules of the form “90% of the customers that purchase product x also purchase product y” (Boolean association rules), e.g [1,13,15]. Formally, an association rule is a rule of the form \(X \Rightarrow Y\), where \(X\) and \(Y\) are sets of attributes. A clustering algorithm attempts to seek a \(k\)-partition of \(X\) into \(k\) subgroups (medoids of the clusters) as clustering rules.

Classification methods have been widely studied in various scientific fields including Machine Learning, Neural Networks and Statistics. Classification algorithms can be classified [2] as either hierarchical or iterative (partitional, density search, factor analytic or clustering and graph theoretic). Complete-link, average link and single-link algorithms [29,47] are some popular hierarchical clustering algorithms. K-means [45], along with its variants (e.g. [42,72]), and hill-climbing [3] are some popular partitional clustering algorithms.

Formally, given a set of input patterns, \(X=\{x_1,x_2,...,x_n\}\): Partitional clustering attempts to seek a \(k\)-partition of \(X\), \(C=\{C_1,C_2,...,C_k\}\), usually \(k<<n\), such that: 1) \(C \neq \emptyset\), \(1 \leq k \leq n\).
2) \[ \bigcup_{i=1}^{k} C_i \subseteq X, \text{ 3) } C_i \cap C_j = \emptyset, \text{ 4) } l \leq j, \text{ 5) } i \neq j. \]
Hierarchical clustering attempts to construct a tree-like nested structure partition of \( X \), \( H = \{H_1, H_2, ..., H_l\} \), usually \( 1 < n \), such that: \( C_i \subseteq H_p \), \( C_j \subseteq H_p \), and \( p > q \) imply \( C_i \subseteq C_j \) or \( C_j \subseteq C_i = \emptyset \), \( \forall 1 \leq p, q \leq l, i \neq j. \)

In the context of data mining the user often has to analyze hundreds of extracted rules in order to grasp valuable knowledge. Thus, the analysis of such rules by means of visual tools has evolved rapidly in recent years. Visual data mining attempts to take advantage of humans’ ability to perceive pattern and structure in visual form. The use of visualization techniques to facilitate the user in exploring the extracted knowledge is of great importance in the context of all the data mining tasks mentioned above [31]. In the context of classification rules, the need for visualization has been driven by the large amount of rules which constitute a knowledge management problem. In the context of association rules, the need for visualization has been driven by the huge amount of extracted rules which can be more than the original transactions (usually for low support and confidence values). In the context of clustering rules, the need for visualization has not been driven by the number of extracted rules, which usually are only a few. The need for visualization of a clustering solution will depend on the application. For instance, in target marketing or anomaly detection applications the user wish to know how well defined the clusters are, how different they are from each other, which clusters are similar to each other and what their size is.

Researchers have developed many tools to visualize data mining rules. However, few of these tools can handle effectively more than a few dozens of data mining rules [11]. In this paper, we propose a new visualization technique of data mining rules based on OLAP.

OLAP (On Line Analytical Processing) [21,59] must provide a multidimensional conceptual view of the data to be analysed. OLAP meets both the high-level information delivery requirements of executives and the needs of advanced ad hoc analysis of knowledge workers. In general, OLAP tools are designed to support DSS procedures providing the users with various OLAP functions, such as the popular drilling down or slicing and dicing.

More specifically, the proposed technique utilizes the standard two-dimensional cross-tabulation table of most OLAP models in order to visualize even a great number of data mining rules. The advantage of the proposed technique is that it is not necessary to present to the user the whole set of extracted data mining rules. The proposed technique only presents groups of data mining rules to the user (represented by the levels of hierarchies in the OLAP model) and the user can choose to focus (“drill down”) on specific groups. Also, the proposed technique is not based on specific elaborated forms of charts, graphs or trees.

We also present experimental results that demonstrate how the proposed technique is useful and helpful for analysing and understanding extracted data mining rules.

In the rest of the paper we first describe the related work (Section II) and we then present the proposed visualization technique for classification (Section IV), association (Section V) and clustering (Section VI) rules, along with experimental results. Finally, we present some comparison of results (Section VII) and we conclude (Section VIII).

II. RELATED WORK

In this section we summarize the different categories of presentation and visualization of data mining outcomes of classification, association and clustering techniques.

A. Classification Rules

In the visualization of the classification outcome three main techniques have been presented by the research community: decision trees, treemaps and tree ring, and parallel coordinates.

The most frequently used technique is the Tree View which facilitates the hierarchical structure of decision trees with parent-child relationships. A classification rule is represented by the path from the root of the tree to a leaf node where it is assigned a class label. What differentiates the various tree visualization techniques is the representation of nodes by the use of color to depict the classification accuracy, the display of statistics metrics in nodes, or the use of an additional compact representation technique of the entire tree unless all the nodes are visible on the display. The MineSet tool with the Decision Tree Visualizer and the Option Tree Visualizer [16], IBM's Intelligent Miner [43], DBMiner's 3D Tree View, Tree Grid View and Rules View [26], SPSS's Answer Tree [48,69] and SAS's Tree View [65] are some of the most popular commercial programs which implement the tree visualization technique.

Alternative methods of the Tree View are the Treemap [24,46,66] and the Tree Ring techniques [6,65]. Virtually, they depict the topology of a tree and the nodes size in a brief view [6]. Treemap uses a rectangle in the same size of the display area and an appropriate space-filling approach in which each node is a smaller rectangle whose area is proportional to the node size. The root node of the tree is equal to the initial rectangular area, which is then partitioned recursively in such a manner that the child nodes are included in the parent node. The tree ring technique uses the same methods but in a circular way. InterRing of Xmdv Tool [76] is based on a similar radial space filling approach. Finally, RINGS [68] is a technique for visualizing large hierarchies.

Ankerst et al. [4,5,30] has developed a technique for representing the hierarchical tree by using the perception based classification. Each level of the tree is represented by a horizontal bar which includes all nodes of a particular level and where each attribute is placed in a different segment. The class label is depicted with the use of a perception based color scale. In this way the leaf nodes have a more flat color while the nodes in the upper bars have an unclear color.
Kohavi et al. and Becker [8,50,51] have introduced the Decision Table Classifier which uses a spreadsheet representation (matrix). It facilitates the same hierarchical construction as the decision trees but in each level it uses two attributes to separate the data. The values of the attributes are represented in the axes of the matrix and for each combination of values there is a distribution bar. The bar's height represents the number of records that satisfy this condition where the color of the bar represents the class. Kohavi et al. [7] have developed a visualization technique of the naive Bayes Classifier that uses the bar diagrams as well as pie chart diagrams.

Alternative techniques are the 2D [28] and 3D [52] projection techniques that use class preserving methods. Parallel Coordinates are also used in the visualization of classification rules. In RuleViz model [38,39] every attribute is represented by a vertical line and all the lines are placed in a parallel manner. Then, every rule is depicted as a polygonal stripe that crosses the parallel axes and the width of color is signifying the rule's accuracy. The WinViz tool [53-55] uses rectangles to represent the attribute values rather than lines and the class labels are represented by colored rectangles. The width of a rectangle indicates the number of records having the particular value in an attribute. The user can define the rules being visualized at a time with the use of a color polygonal line. Finally, in [9] a circular visualization of sets of classification rules is presented. The visualization is based on graphs where all attribute-value pairs of a rule are represented with edges inside a circle.

B. Association Rules

To facilitate the interpretation of association mining results, various techniques have been presented (see [18] for a recent survey). According to the bibliography three main classes of visualization techniques exist in association mining: tables which are similar to textual presentations, matrices, and graphs.

The use of a rule table is a relatively simple visualization technique of association outcome. This method has been used by the most popular commercial programs, such as SAS Intelligent Miner [65], IBM DB2 Intelligent Miner [43], DBMiner [25,26]. Their main advantages are that they are simpler in interpretation by the non-expert and they give the opportunity to represent sequentially the results by a criteria column (support, confidence, items etc) that the user selects.

Researches have devised several techniques for the visualization of association rules build on matrix techniques. The matrix visualization [16,26,27,43] is based on the design of a two-dimensional matrix where the antecedent and consequent items of a rule are placed on the x and y axes and any relationship is indicated by a bar on the intersecting cell. The height of the bar usually represents the frequency of occurrence, while the color of the bar represents the confidence measure. This method is appropriate in rules of a relatively small number of antecedent and consequent values (one-to-one relations), otherwise it may lead to occlusion. To confront this problem Wong et al. [75] developed the “rule to item matrix” by displaying the support and confidence metrics at the end of the matrix as a wall. A variation of the matrix technique is the CrystalClear [58] which consists of two techniques: enhanced grid view and tree view. What differentiates CrystalClear from the previous techniques is that the rules are represented by colored cells on the matrix according to their support and confidence values and rules with the same values are represented in the same cell. In [36] a matrix technique is proposed which is based on visualizing clusters of rules rather than the whole set. Hofmann et al. [40,41] introduced the Interactive Mosaic Plots and their alternatives Double Decker Plots, as the graphical analogue of a contingency table that yields the association rule as well as the other potential rules in that table, whether they meet the threshold or not. So, each cell of the table is visualized by a tile in the plot, in such a way that the tile's size is directly proportional to the number of cases in the cell.

The use of graphs in the visualization of mining results has been proposed by many researchers [26,43,49,63,74]. All these visualizations represent items as nodes and the associations between them as directed links that connect the nodes (see [34] for describing such techniques in the R package). The key feature that differentiates all these techniques is the representation of support and confidence values and the way the nodes are placed in the plane to avoid the problems of occlusion and cluttering. Roddick et al. [19] introduced a novel technique for visualizing hierarchical association rules by the use of a radial model. Particularly, in this technique the radial model is used for the interpretation of hierarchy between items while the associations are visualized using the direct or trial techniques for connecting the lines.

Other techniques that do not belong to any of the above classifications are Bar Chart Graphs where each rule's condition both on antecedent and consequent clause is visualized by a bar in a chart. A 3D alternative presentation of the Bar Chart is the Perspective Wall. Parallel Coordinates [44] are also used in visualization of mining outcomes where each attribute in an association rule is represented by a vertical line. All the three above techniques are implemented in the VizMiner tool [52]. Brzuzese et al. [17] integrated the graph-based and the parallel coordinate techniques in discovering and representing association rules.

C. Clustering Rules

Many visualization techniques that have been presented in the previous sections are used also in the visualization of clustering mining outcomes. For example, Parallel Coordinates in WinViz [54,55,56] and in Xmdv Tool [32,73] are facilitated in the visualization of clustering results. The Treemaps method is also used in hierarchical clustering and has many applications in biology (microarray analysis) [67].

III. THE ADOPTED OLAP MODEL

In the literature hierarchy, there appear some OLAP models [71], however in our analysis there exist several advantages for solving this specific problem using the OLAP-model that we have proposed in an earlier contribution [12]. Its building block is the data hypercube. A data hypercube is an n-dimensional array: \( DH(d_1,\ldots,d_n) \) of
lists \( l_1 = (e_1, \ldots, e_n) \). Every element of a list is an ordered pair \( e_i=(\text{label}_i, \text{value}_i) \) denoting that each element is defined by two attributes: its label, that is distinct, and its value. All the lists of a data hypercube have the same number of elements, each with the same label in all the lists. To every dimension \( d_i \) is also assigned a label \( \text{label}_i \) denoting that each dimension has a distinct label. A dimension consists of dimension values, which belong to a certain domain. To every dimension value is also assigned a label. We define the function \( \text{label}(i) \) that returns the label of an element or dimension value. The type of the label of an element or dimension value is always alphanumeric. The type of the value of an element could be either numeric or alphanumeric. We also define the function \( \text{dom}(d_i) \) that returns the domain of dimension values of a dimension \( d_i \). In Fig. 1 the data hypercube is \( DHs[d_1, d_2, d_3] \) of lists \( l_1 = (e_i) \), where \( \text{label}(d_1) = \text{MONTH} \), \( \text{label}(d_2) = \text{NAME} \), \( \text{label}(d_3) = \text{ZIPCODE} \), \( \text{label}(e_i) = \text{Sales} \).

![An example dependency tree](image)

Intuitively, elements correspond to measures and represent the actual data that are organized around concepts of the real world that need to be analyzed. Concepts are represented by dimensions in the proposed OLAP-model. Examples of such concepts are sales, purchases, pricing, customer base, etc. The label of an element denotes the concept that this element represents. The value of an element is a measurement of such a concept. The identification of a certain element depends on the identification of the list it belongs to. In order to identify a list one must first define the dimensions of the n-dimensional array that represents the hypercube. These dimensions represent the different views through which the data can be accessed.

The used OLAP-model is defined using the notion of hierarchical data hypercube. A hierarchical data hypercube is based on a data hypercube, the specialized data hypercube \( DHs[d_1, \ldots, d_n] \), where \( \forall v_i \in \text{dom}(d_s) \ \exists e_g \in \text{lg}_s \) and \( v_i = \text{value}(e_g) \) and \( \text{lg}_s \) is a list that belongs to another data hypercube, the generalized data hypercube: \( DHg[d_1, \ldots, d_m] \). Intuitively, a hierarchical data hypercube is a data hypercube, the specialized data hypercube (corresponds to a fact table), whose at least one dimension is identified through the elements of another data hypercube, the generalized data hypercube (corresponds to a dimension). That is, the values of at least one dimension of the specialized data hypercube and hence the identification of its lists and elements, depend on the values of all dimensions of the generalized data hypercube through the identification of its lists and elements. Elements of the lists of generalized data hypercubes are values of dimensions of their specialized data hypercube. In Fig. 1, values of \( d_i \) and \( d_j \) dimensions of the specialized data hypercube \( DHs[d_1, d_2, d_3] \) are identified through the elements of \( DHg1 \) - \( DHg4 \) generalized data hypercubes.

There are three kinds of relations between dimension values of generalized and specialized data hypercubes. The first is a one-to-one relation, where every list of a generalized data hypercube is related to only one dimension value of its specialized data hypercube. The reverse is also true. The second is a one-to-many relation, where every list of a specialized data hypercube is related to many dimension values of its specialized data hypercube while every dimension value of the latter is related to only one list of the corresponding generalized data hypercube. The third is a many-to-many relation, where every list of a generalized data hypercube can be related to many (possibly one) dimension values of its specialized data hypercube and every dimension value of the latter is related to many lists of the corresponding generalized data hypercube. Therefore, the dependencies between the different data hypercubes form a tree rooted at the base data hypercube. Thus, we define the dependency tree of the OLAP-model. The structure of a dependency tree introduces levels where the different data hypercubes belong to (see Fig. 1 for an example, where all relations are one-to-many).

We define a hierarchy as a path of the dependency tree starting from the base data hypercube and ending in a leaf. Since a hierarchy starts from the base data hypercube, it is related with one of its dimensions. Actually, every generalized data hypercube that belongs to a hierarchy can be used in the identification of lists of the base data hypercube, through defining the values of this related dimension. In general, there might be more than one hierarchies related to the same dimension of the base data hypercube.

According to the above, one could access more than one list of elements in the base data hypercube. This situation corresponds to the identification of more than one value for some dimension(s). It is, actually, a rule in the case where the accessed elements are identified indirectly through the definition of elements of generalized data hypercube(s). This is because a list of a generalized data hypercube, usually, consists of many elements that are dimension values of some dimension of its specialized hypercube. For instance, in Fig. 1, \( DHg1[1st \ Quarter] \) refers to the list \{Jan, Feb, Mar\} \( \subset \text{dom}(d_3) \). Since the different lists of elements are identified indirectly, they must also be viewed indirectly. This is achieved by aggregating the different lists of elements of the base data hypercube.

Aggregating is reduced to applying a certain aggregating operator over these lists. Examples of aggregating operators, applying to elements with numerical values, are “Sum”, “Average”, etc. Examples of aggregating operators, applying to elements with alphanumeric values are “Concatenate”, “Substring”, etc. Usually, different aggregating operators are applied to different sets of corresponding elements. Moreover, different aggregating operators can be applied to the same set of corresponding elements viewed by different levels of the dependency tree.
We formally define these aggregating operators by attaching to each node of the dependency tree a list of operators \( o=\{op1,...,op_g\} \), where every aggregating operator \( op_i \) is applied to the \( i \)th element of the different lists of the base data hypercube.

IV. VISUALIZING CLASSIFICATION RULES

We define a classification rule \( R^i \in R \) as an ordered pair \( R^i=(A',M') \). The first element, \( A'=(A_i, AV_i) \), \( A_i \in ATT, AV_i \in dom(A_i) \), is a variable length ordered set of attribute-value pairs, where \( ATT \) is a set of \( d \) attributes. The \( d \)th attribute of \( ATT \) represents the class attribute. By definition, \( 2 \leq |A_i| \leq d \) and the \( (A_i, AV_i) \) attribute-value pair is mandatory. The second element \( M'=(M_j, MV_j) \) \( M_j \in MSR, MV_j \in dom(M_j) \), is a fixed length ordered set of measure-value pairs, where \( MSR \) is a set of \( m \) measures.

During a preprocessing phase, we first order elements of \( ATT=\{A_1,...,A_d\} \) based on the increasing order defined by the entropy measure. Then, for each rule \( R^i \), we add in \( A' \) a void attribute-value pair \( (A_i, NULL) \) for those attributes \( A_i \in ATT \) such that \( -3(A_i, AV_i) \in A' \).

Then we define a hierarchical data hypercube as follows:

\[
\text{input} \quad \text{a set of } d \text{ attributes } ATT \text{ as defined above} \\
\text{input} \quad \text{a set of } m \text{ measures } MSR \text{ as defined above} \\
\text{input} \quad \text{a set of } R \text{ rules } R^i=(A',M') \text{ as defined above} \\
\text{1. define} \quad \text{the base data hypercube } DH_b[d_1,d_2] \text{ of lists} \\
\text{l=(e_1,...,e_m) as: set dimensions } d_1=A_1 \text{ and } d_2=A_d \\
\text{2. assign} \quad \text{lists as:} \\
\text{for each } R^i=(A',M') \in R \text{ set } DH_b[d_1,d_2]=M' \\
\text{3. for each } A_p \in ATT, 2 \leq p \leq d-1 \\
\text{[} \\
\text{define} \quad \text{a generalized data hypercube } DH_{gp}[gd_p] \\
\text{of lists where } \text{dom}(gd_p) = \{AV_p| (A_p, AV_p) \in A' \} \text{ for some } R^i \in R \} \\
\text{assign} \quad \text{lists as:} \\
DH_{gp}[gd_p] = \{AV_p, (A_p, AV_p) \in A', p=gd_p \text{ for some } R^i \in R \} \text{ } \\
\text{]}
\]

From an implementation viewpoint, a hypercube with two dimensions is defined; the base data hypercube \( DH_b[d_1,d_2] \). In most of the cases, the base data hypercube stores only the classification accuracy of the extracted classification rules (thus \( m=1 \)). The second dimension \( (d_2) \) represents the class attribute. The first dimension \( (d_1) \) represents the extracted set of classification rules or the decision tree. More specifically, the hierarchy of the first dimension represents the classification rules or, interchangeably, the paths of the decision tree from the root to the leaves. Usually, only some of the attributes participate in such classification rules or paths. Thus, during the preprocessing phase, they are transformed in order to include an attribute-value pair for every attribute included in \( ATT \). This is achieved by adding void attribute-value pairs, with NULL as an attribute value. Thus, each level of the hierarchy, defined as a generalized data hypercube \( DH_{gp}[gd_p] \), represents a specific attribute. The higher levels of the hierarchy represent the more important attributes (lower entropy) while the lower levels represent the less important ones. Finally, the “count” operation is defined to be the aggregation operation between all the levels of the hierarchy.

From a presentation viewpoint, the two dimensions of the base data hypercube \( DH_b[d_1,d_2] \) are assigned to row and column dimensions. Thus, the end user can access the different rules through the hierarchy of \( d_1 \). If attribute values, represented only by the higher levels, are defined by the user then the number of classification rules (due to “count” operation) including these attribute values is displayed. In order to access the classification accuracy of a rule the user must define attribute values from all the levels, including possible void attribute-value pairs. This is achieved by a classical “drill down” OLAP function. However, in the implementation level, this feature implies a many-to-many relationship between the levels of the hierarchy. Note that the proposed OLAP model [12] supports such relationships, as well as all the features needed to implement the hypercubes described above. There are, also, commercial systems which support these features.

A. A Test Implementation Of A Hierarchical Data Hypercube For Visualizing Classification Rules

The proposed technique can be easily implemented using any OLAP tool. As a test implementation, we implemented a hierarchical data hypercube for visualizing classification rules extracted from the “Nursery” database supplied by the UCI Machine Learning Repository [70], using pivot chart reports of Excel (Excel files are freely available for download at: http://150.140.136.40/lab/images/stories/uploads/files/DMOLAP.rar). The UCI Machine Learning Repository is a collection of databases, domain theories, and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms. The “Nursery” database, having 12960 objects described by 8 attributes, was derived from a hierarchical decision model originally developed to rank applications for nursery schools. The final decision depended on three subproblems: occupation of parents and child’s nursery, family structure and financial standing, and social and health picture of the family. We used C4.5 [60] classification algorithm without a prune check, in order to extract 839 classification rules (\( |R|=839 \)).

Since Excel cannot support many-to-many relationship between the levels of the hierarchy, we used different NULL values (null3-null839) to represent void attribute-value pairs (step 2). The hierarchy of the row dimension (step 3) represents the “health”, “has Nurs”, “parents”, “social”, “housing”, “finance”, “form” and “children” attributes from the more important (lower entropy) to less.
important ones (higher entropy). The column dimension represents the class. Thus, \( ATT = \{ \text{health}, \text{has nurs}, \text{parents}, \text{social}, \text{housing}, \text{finance}, \text{form}, \text{children}, \text{class} \} \) and \( MSR = \{ \text{False Positives} \} \).

If the user selects only the higher levels then only the number of classification rules is displayed for each attribute value (due to “count” aggregation operator). This case is shown in Fig. [2], where the number of rules for “health=not_recom”, “health=priority” and “health=recommended” is shown with respect to different class values. The user could be interested in the great number of rules (328) for “health=recommended” and “class=priority”. Alternatively, the user could be interested in a specific attribute value, e.g. “health=priority”. In any case, the user can “drill down” by choosing to show those attributes that interested her/him the most. This is achieved by right-clicking on a specific attribute and selecting “Group and Show Detail - Show Detail” from the pop-up menu. This case is shown in Fig. [3] where the user, also, chose a different chart type. Finally, the user can select to show only specific values for each attribute from the scroll-down menu next to each attribute label. Of course, the user can define chart properties at will. The above support user to produce simpler and more friendly charts, as the one shown in Fig. [5] which also presents rules \( R^{29} \) and \( R^{30} \). Finally, note that the Excel user can manually define the aggregating operator (menu “Field Settings”) for all levels as well as all the features needed to implement the hypercubes described above. There are, also, commercial systems which support these features.

V. VISUALIZING ASSOCIATION RULES

We define a set of literals \( I = \{ i_1, \ldots, i_l \} \), called items, that represent products and a set of transactions \( D = \{ (T_1, TID_1), \ldots, (T_n, TID_n) \} \), called itemsets, where \( T \subseteq I \) and \( TID_i \) is a unique identifier associated with each \( T \).

We define the number of association rules \( R \in R \) as an ordered pair \( R = (A', M') \). The first element is an order set \( A' = (L_T, R_T) \), where \( L_T, R_T \subseteq I \). As in the case of classification rules, the second element \( M' = (M_j, MV_{ji}) | M_j \in MSR, MV_{ji} \in \text{dom}(M_j) \) is a fixed length ordered set of measure-value pairs, where \( MSR \) is a set of \( m \) measures. Then we define a hierarchical data hypercube as follows:

- **input** a set \( I \) of \( l \) literals as defined above
- **input** a set \( MSR \) of \( m \) measures as defined above
- **input** a set \( R \) of rules \( R = (A', M') \) as defined above
- **input** a grouping threshold \( h \) and grouping parameters \( u \) and \( w \)
1. define the base data hypercube $DH_b[d_1,d_2]$ of lists $l=(e_1,...,e_n)$ as: set 
$\text{dom}(d_i) = \{\text{concatenate}(LT_i) \mid A_i = (LT_i,RT_i), \forall R'=(A',M') \in R\}$,
$\text{dom}(d_2) = \{\text{concatenate}(RT_i) \mid A_i = (LT_i,RT_i), \forall R'=(A',M') \in R\}$,
for $i = 1$ to $h$
2. assign lists as: for each $\forall R'=(A',M') \in R$
$\text{DH}_b[\text{concatenate}(LT_i),\text{concatenate}(RT_i)]=M'$
For $i = 1$ to $h$
3. define a generalized data hypercube $DH_{glh}[gldh]$ as:
$\text{dom}(gldh) = \{\text{concatenate}(PLT_i) \mid PLT_i \in PLT\}$
where $PLT = \{PLT_1,...,PLT_u\}$ is a partition of $\text{dom}(sdl)$, where $sdl$ is a dimension of the specialized data hypercube of $DH_{glh}[gldh]$
assign lists $DH_{glh}[PLT_i]=\{i_i \mid i_i \in PLT_i \subset \text{dom}(sdl)\}$
4. define a generalized data hypercube $DH_{grh}[grdh]$ as:
$\text{dom}(grdh) = \{\text{concatenate}(PRT_i) \mid PRT_i \in PRT\}$
where $PRT = \{PRT_1,...,PRT_w\}$ is a partition of $\text{dom}(sdr)$, where $sdr$ is a dimension of the specialized data hypercube of $DH_{grh}[grdh]$
assign lists $DH_{grh}[PRT_i]=\{i_i \mid i_i \in PRT_i \subset \text{dom}(sdr)\}$

From an implementation viewpoint, a hypercube with two dimensions is defined; the base data hypercube $DH_b[d_1,d_2]$. In most of the cases, the base data hypercube stores only the support and the confidence of the extracted association rules (thus $m=2$). The values of the first dimension ($d_1$) represent the literals (items) $l \in LT$, of a rule $R'=(A',M') \in R$, which are the literals participating in the left part of the extracted association rules. Similarly, the values of the second dimension ($d_2$) represent the literals (items) $l \in RT$, of a rule $R'=(A',M') \in R$, which are the literals participating in the right part of the extracted association rules. However, the number of attribute values of $d_1$ or of $d_2$ is usually large. Therefore, a hierarchy is defined for each of the two dimensions. Each level of the hierarchy represents a partition of the values of its immediate lower level. The “count” operation is defined to be the aggregation operation between all the levels of the hierarchies.

From a presentation viewpoint, the two dimensions of the base data hypercube $DH_b[d_1,d_2]$ are assigned to row and column dimensions. Thus, the end user can access the different rules through the two hierarchies. If literals, represented only by the higher levels of both hierarchies, are defined by the user then the number (due to “count” operation) of association rules including these literals is displayed. In order to access the support and the confidence measure of a rule the user must define literals from the lowest levels in both hierarchies. This is achieved by a classical “drill down” OLAP function. In the implementation level, this feature implies the common one-to-many relationship between the levels of the hierarchies.

A test implementation of a hierarchical data hypercube for visualizing association rules

As a test implementation, we implemented a hierarchical data hypercube for visualizing association rules extracted from a synthetic database, using pivot chart reports of Excel. The synthetic database is built using 100 different items ($|I|=100$). We chose to visualize a set of 102 ($|R|=102$) association rules extracted using the Apriori algorithm [11]. Thus, $d_1$ ($d_2$) consists of 102 concatenations, each for one left (right) part of a rule (step 1). For each pair of these left and right parts, a list is stored in the base data hypercube if they form a rule (step 2).

Items appearing in the left part of association rules are first grouped for the “Left Group 4” level and this grouping is repeated for three more levels (step 3). A similar process is applied to the set of items appeared in the right part of the extracted association rules (step 4).

Fig. 6 Visualization using higher levels of both row and column dimensions

Fig. 7 Visualization using all levels of both row and column dimensions

If the user selects only the higher levels then the number of association rules for a certain group of items in the left and/or the right part is displayed, due to the “count” aggregation operator. This case is shown in Fig. 6, where the user chose to “Hide Details” for the last three levels in both the dimensions representing the left and right part items. Thus, “A1, A2” and “E1, E2” values of the fourth level of the two dimensions are shown respectively. The user could be interested in the great number of rules (50) concerning items in the “A2” and “E2” groups or the user could be interested in items of a specific group. In any case, the user can “drill down” by choosing to show those groups
that interested her/him the most. This is achieved by right-clicking on a specific group and selecting “Group and Show Detail” from the pop-up menu. Finally, the user can select to focus on specific rules, by showing the corresponding values for all groups (levels) and by showing their support or confidence. This case is shown in Fig. 7, where rules $(182\Rightarrow 185, 183\Rightarrow 186, 184\Rightarrow 187, 185\Rightarrow 188, 185, 186\Rightarrow 189$ and $187\Rightarrow 190$ are shown along with their support $(\text{Support})$. Note that in a real life dataset, the group labels can be defined properly in order so as to provide information for the items they include.

As in the case of classification rules, the user can select to show only specific values for each group from the scroll-down menu next to each group label, or define properly the chart properties, or define the aggregating operator in order to produce simpler and friendlier charts.

VI. VISUALIZING CLUSTERING RULES

Similarly to a classification rule, we define a clustering rule $R' \in R$ as an ordered pair $R' = (A', M')$. The first element, $A' = \{(A_i, AV_{i}) | A_i \in \text{ATT}, AV_{i} \in \text{dom}(A_i)\}$, is fixed (contrary to a classification rule) length ordered set of attribute-value pairs, where $\text{ATT}$ a set of $d$ attributes. The second element $M' = \{(M_j, MV_{j}) | M_j \in \text{MSR}, MV_{j} \in \text{dom}(M_j)\}$ is a fixed length ordered set of measure-value pairs, where $\text{MSR}$ is a set of $m$ measures. Then we define a hierarchical data hypercube as follows:

**input** a set of $d$ attributes $\text{ATT}$ as defined above  
**input** a set of $m$ measures $\text{MSR}$ as defined above  
**input** a set $R$ of rules $R' = (A', M')$ as defined above  

1. **define** the base data hypercube $DH_0[d_1,d_2]$ of lists $l=(e_1,...,e_m)$ as: get dimensions $d_1=A_1$  
2. **assign** lists as:  
   for each $R' = (A', M') \in R$ set $DH_0[d_1] = M'$  
3. for each $A_p \in \text{ATT}$, $2 \leq p \leq d-1$  
   
   {  
   define a generalized data hypercube $DH_{g_p}[gd_p]$ of lists where $\text{dom}(gd_p) = \{AV_{p_i} | (A_p, AV_{p_i}) \in A'$  
   for some $R' \in R$}  
   assign lists as:  
   $DH_{g_p}[gd_p] = \{AV_{(p-1)i} | (A_{(p-1)i}, AV_{(p-1)i})$  
   $(A_p, AV_{p_i}) \in A', p = gd_p$ for some $R' \in R \}$  

From an implementation viewpoint, a hypercube with one dimension is defined; the base data hypercube $DH_0[d_1]$. In most of the cases, the base data hypercube stores only the number of objects included in different extracted clusters (thus $m=1$). The hierarchy of the $d_1$ dimension represents the means or medoids of the extracted clustering rules. Since $A_1$ is a fixed length ordered set of attribute-value pairs, each level of the hierarchy, defined as a generalized data hypercube $DH_{g_p}[gd_p]$, represents a specific attribute. The order in which attributes are assigned to levels is not critical. The “count” operation is defined to be the aggregation operation between all the levels of the hierarchy.

From a presentation viewpoint, the $d_1$ dimension of the base data hypercube is assigned either to row or to column dimension. Thus, the end user can access the different rules through the hierarchy of $d_1$. If only some attribute values are defined by the user then the number of clustering rules including these attribute values is displayed. In order to access the number (due to “count” operation) of objects within a cluster the user must define attribute values from all the levels. This is achieved by a classical “drill down” OLAP function. In the implementation level, as in the case of classification rules, this feature implies a many-to-many relationship between the levels of the hierarchy.
If the user selects only the higher levels then the number of clustering rules is displayed. This case is shown in the upper part of Fig. [8], where the user chose to “Hide Details” for “doors”, “persons”, “lug_boot” and “safety” attributes. If the user selects to show details in all levels, as shown in the lower part of Fig. [9], then the rules are displayed along with the number of instances they refer to. Of course, as in the cases of classification and association rules, the user can select to show any level between, to show only specific values for each level, to define properly the chart properties, or to define the aggregating operator.

VII. ON COMPARING THE PROPOSED VISUALIZATION TECHNIQUE

The advantage of the proposed technique is that it can handle effectively a great number of data mining rules, in contrast to related techniques. This is achieved because using the proposed technique there is no need for presenting the whole set of extracted data mining rules. Instead the user can access any interesting subset of them (access from lower levels). Of course, the user can access all of them through aggregation in higher levels. Actually, higher levels guide the user to select the subset of the rules that interests her/him.

Another advantage of the proposed technique is that it can be applied to all data mining techniques. Note that classification and association rules are usually visualized by different techniques. Clearly, efficient visualization techniques (e.g. perception based classification or graph-based techniques for association) are dedicated to either classification or association rules.

A comparison of the proposed technique to related visualization techniques could be performed by visual inspection. However, we explicitly present some concluding remarks of such comparison, in the following.

The most common problem in tree visualization techniques is the display of large trees. The problem exists even in some techniques which try to reduce the visualized information, as Option Tree Visualizer with decision nodes, or in some others which try to efficiently manage the visualization area, as Treemap and Tree Ring techniques. On the other hand, Decision Tables try to tackle the problem by allowing the presentation of only two attributes in each visualization, in contrast to the proposed technique where all attributes are included in the visualization. Also, there are techniques, as perception based classification, capable of visualizing large trees. However, they are based on complex visualizations while they require knowledge of the domain at hand.

The most similar to the proposed technique is tree visualization technique capable of showing/hiding the subtree rooted in each node and capable of focusing on part of the tree. However, in such cases, the user is not guided by the visualization process itself in order to focus on interesting parts of the visualized information, as in the proposed technique.

As far as association rules are concerned, Rule tables, in the presence of too many rules, are equivalent with textual representations not using the potentials of graphical techniques. Their advantage, i.e. to represent sequentially the results by a criteria column, could be also supported in the proposed technique by sorting dimension values. The latter is supported by most OLAP tools.

Matrix techniques cannot handle a great number of rules since they need an $O(n^2)$ space complexity. Also, they may lead to occlusion and misunderstanding of the outcome, in the case of rules with many antecedent and consequent values. The above problems are also true for both Interactive Mosaic Plots or Double Decker Plots and Bar Charts. Extensions of matrix techniques, as "rule to item matrix" and CrystalClear, tackle the occlusion problem but they still cannot handle a great number of rules or rules with a great range of support/confidence, respectively. An important advantage of the enhanced grid view in CrystalClear is that it supports visualization of “differences” in the set of rules due to incremental changes in the data source. The proposed technique can support such a feature by establishing a time dimension for representing evolving sets of rules where new ones are created, old ones removed, and others changed.

Graph-based techniques are more efficient in handling a great number of rules however they cannot tackle the occlusion problem. Moreover, they cannot represent many “metavalues” (e.g. measures of interestingness, as support and confidence). On the contrary, the number of “metavalues” is not limited in the proposed technique, since one can define any number of measure-value pairs in $M$ (see Sections IV, V, VI).

Some variations of graph-based techniques, as Circular Association Rules, support aggregated representations of rules as well as focusing on subsets of rules, as also the proposed technique does. more than 3 levels of headings should be used. All headings must be in 10pt font. Every word in a heading.

VIII. CONCLUSIONS

We presented a new visualization technique of data mining rules based on the standard two-dimensional cross-tabulation table of OLAP. The advantage of the proposed technique is that it is not based on the effective visualization of the whole set of extracted data mining rules but on the fact the user can choose to focus (“drill down”) on specific groups of rules. In all the presented test implementations, the “count” operation is defined to be the aggregation operation between all the levels of all the hierarchies. Thus, the user can access the number of extracted data mining rules related to a higher level of a hierarchy. This information guides the user to "drill down" to lower levels and, finally, to focus on specific rules. Of course, OLAP tools support a lot of aggregating operators which a user can select in real time. Moreover, the user can select different aggregating operators for different hierarchies. For instance during the visualization of classification rules, selecting the “Average” operator the user could get the information of the average classification accuracy of the extracted data mining rules related to a higher level.
Finally, note that the proposed methodology is not based on specific elaborated forms of charts, graphs or trees. It is based on using an OLAP model in order to take advantage of any visualization techniques the adopted OLAP tool supports. Although the visualization technique is developed based on a specific OLAP model, it can be implemented in every OLAP tool, possibly at a cost of limiting user friendliness. Thus, we presented some test implementations of the proposed visualization technique by simply using Excel pivot chart reports.

REFERENCES

[16] C. Brunk, J. Kelly and R. Kohavi, "MineSet: An Integrated System for Data Mining", in Proc. of the 3rd Int. Conf. on Knowledge Discovery and Data Mining (KDD '97), 1997, p. 135-138.
[30] M. Ester, "Cooperative Classification: A Visualization-Based Approach of Combining the Strengths of the User and the Computer", CiteSeerX.


Basilis Boutsinas received his diploma in “Computer Engineering & Informatics” in 1991 from the University of Patras, Greece. He also conducted studies in “Electronics Engineering” at the Technical Education Institute of Piraeus, Greece, and Pedagogics at Pedagogical Academy of Lamia, Greece. He received his Ph.D. on Knowledge Representation from the University of Patras in 1997. He is a professor in the Department of Business Administration at the University of Patras since 2001. His primary research interests include data mining, business intelligence, knowledge representation techniques, nonmonotonic reasoning and parallel AI.

Eirini Gkiza received her diploma in “Business Administration” in 2005 from the University of Patras, Greece. She received her MSc on Mathematics of Computers and Decision Making from the University of Patras in 2007. She is now an executive manager specialized in market research and consulting services. Her primary research interests include data mining and business intelligence.