Decision Support Systems 46 (2008) 429–439

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

Measuring interestingness of discovered skewed patterns in data cubes

Navin Kumar [a], Aryya Gangopadhyay [b,*], Sanjay Bapna [c], George Karabatis [b], Zhiyuan Chen [b]

[a] Advisory Board Company, United States
[b] Department of Information Systems, University of Maryland Baltimore County (UMBC), United States
[c] Department of Information Systems, Morgan State University, United States

ARTICLE INFO

Article history:
Received 18 August 2006
Received in revised form 15 August 2008
Accepted 25 August 2008
Available online 13 September 2008

Keywords:
OLAP
Data cube navigation
Data warehousing
Navigation rules
Skewness

ABSTRACT

This paper describes a methodology of OLAP cube navigation to identify interesting surprises by using a skewness based approach. Three different measures of interestingness of navigation rules are proposed. The navigation rules are examined for their interestingness in terms of their expectedness of skewness from neighborhood rules. A novel Axis Shift Theory (AST) to determine interesting navigation paths is presented along with an attribute influence approach for generalization of rules, which measures the interestingness of dimensional attributes and their relative influence on navigation paths. Detailed examples and extensive experiments demonstrate the effectiveness of interestingness of navigation rules.

Published by Elsevier B.V.

1. Introduction

With an ever-increasing volume of data collected and archived by organizations, it has become critical to efficiently and effectively navigate through large, multidimensional cubes to identify interesting hidden surprises. While On-Line Analytical Processing (OLAP) tools provide various operations such as roll-up, drill-down, and slicing-dicing for viewing datasets from different angles [13], they offer only minimal guidance to the users in the actual knowledge discovery process. Moreover, the viewing possibilities are combinatorially explosive in number, making it a daunting task to manually detect interesting hidden surprises in the voluminous and complex lattices of multidimensional cubes.

As the size of database increases, the number of navigation paths grows which may overwhelm the users. It then becomes difficult for users to manually go through enormous sets of rules to identify interesting ones. This problem could be alleviated if users were presented with a short list of rules, or a list of navigation paths for analytical studies. We approach the issue of cube navigation by using skewness based navigation rules, also called sk-navigation rules, which identify interesting surprises in data cube lattices. In this context, a surprise reveals how anomalous a set of transactions is, when compared with another set of closely related transactions in the fact table. The anomalous transactions could be defined either by few outliers in the datasets influencing the aggregated datasets or by a group of transactions showing substantial difference on facts, such as profit or cost, from the remaining transactions. Because the notion of a surprise is an intuitive one, different users may have different impressions on what constitutes a surprise. Our rule-driven system allows the users to control the knowledge discovery process by letting them set the baseline for surprises by simply adjusting the skewness level of significance.

The contributions of this paper are as follows. We evaluate the interestingness of discovered sk-navigation rules and then assist users to selectively navigate along the paths that lead to interesting surprises in data lattices. Using the measures of interestingness, users can simply prune the large number of generated rules to only the ones that are interesting. Since sk-navigation rules are discovered based on a measure of skewness, we adopt the interestingness of rules in terms of their *expectedness* of skewness from the rules in the neighborhood. We also introduce an Axis Shift Theory (AST) to determine interesting navigation paths based on the global measures of axis shifts of sk-navigation rules. The rules complement each other to yield interesting navigation paths leading to low-level interesting surprises. Lastly, we introduce a method of generalizing sk-navigation rules to identify interesting dimensional attributes to augment cube navigation. Specifically, we measure the interestingness of attributes in terms of their *attribute influence*, a metric based on the unique navigation paths provided by sk-navigation rules.

The rest of the paper is organized as follows. Section 2 presents the related work while Section 3 provides the preliminaries on data cube model and discovery of sk-navigation rules. In Section 4 we present the different measures of interestingness of sk-navigation rules, and in

* Corresponding author.
E-mail address: gangopad@umbc.edu (A. Gangopadhyay).

0167-9236/$ – see front matter. Published by Elsevier B.V.
doi:10.1016/j.dss.2008.08.003

Section 5 we discuss applying our methodology in the context of a real-world application. An experimental evaluation of interestingness is presented in Section 6 followed by a comparison with previous work in Section 7. Section 8 presents the conclusions and future work.

2. Related work

Interestingness measures are used to filter out irrelevant or mundane patterns from a set of mined patterns such as association or classification rules. A large number of interestingness measures have been proposed in the literature on data mining and knowledge discovery (see [8,19] for a comprehensive discussion). These interestingness measures can be categorized into objective, subjective, and semantic measures. Objective measures require no user input and can be further categorized into probability-based measures and form-dependent measures. Thirty eight probability-based objective measures have been discussed in [8], some examples of which are support, coverage, example and counter example rate, and Laplace correction. Form-dependent objective measures [4,5,7] such as neighborhood-based unexpectedness, surprisingness and logical redundancy have been applied for ranking and clustering patterns such as association rules. Subjective interestingness measures require some form of user input in determining the utility of a mined pattern [18,21,27]. Utility-based measures have been used for objective-oriented association mining (for example, [26,33]), with user-specified objectives. In addition, numerous interestingness measures for summaries have been proposed in the literature including diversity (e.g., [12,33]), conciseness and generality [5], peculiarity [22–25], and surprisingness/unexpectedness [9]. Most of these methods with the exception of [9,10,20,30,32] have been applied for identifying patterns that have been mined from a given static dataset as opposed to providing guidance for navigating through multi-dimensional data cubes of graphs, which is the focus of our paper. Hence, in the rest of this section we focus on those methods that are closely related to our work.

Query driven knowledge discovery [6] and discovery-driven exploration of OLAP data cubes [22–25] have been proposed to address identification of surprises. The authors in [22–23] have used precomputed measures indicating exceptions as surprises. Being precomputed, the surprises cannot be defined by users. Furthermore, it is overwhelming for users to examine surprises by looking at data values in a large number of rows and columns. This work was further extended in [24,25] by discovering surprises in unexplored parts of a data cube using maximum entropy principles computed on the aggregate differences. However, these studies deal with aggregated datasets which often hide the characteristic of detailed data: an "extremely high" value and an "extremely low" value could be aggregated to a "moderate" value, hiding both extreme values. Other work in the area of subgroup patterns [16] addresses the general problem of defining and identifying local subgroups. While important, such methods also presume how a surprise is defined and do not provide a generalized navigational methodology. Another recent approach is presented in [5] where the authors use Simpson's paradox as the basis for defining surprises in multidimensional data but do not address the navigational techniques to reach surprises of interest. Recently, in [17], skewness based navigation rules for data cube exploration were proposed. These rules provide navigation support to users to explore data cubes at the transaction level. While the authors address the navigational techniques to reach surprises of interest, they focus only on the skewness metric to reach surprises.

In another closely related work, interestingness has been used in attribute selection, for attribute-oriented generalization [2], where the authors examine different strategies for choosing the next attribute for generalization. Also, the discovery of interesting summaries in generalization space graphs has been studied in [17,22], using the expectations of users, and relative variance as an interestingness measure. However, unlike the method described in [17,22] we do not require users to specify probability distributions. In addition we deal with the lowest level transactions instead of summaries.

An examination of interestingness measures for data mining appears in [19] and a comprehensive categorization of interestingness measures including analysis of their properties is found in [8]. A rule trivial to one user may not be trivial to another; therefore, with proper guidance users can explore the ruleset by only examining the rules which match their expectations. System guidance is also necessary because it is unrealistic to expect every discovered rule to be a surprise [31]. We model cube navigation by measures of interestingness of navigation rules, so that the users can quickly identify rules of interest, aided by the system's pruning of uninteresting rules. Our work differs from previous work in the sense that we measure the interestingness of navigation rules by their skewness based differences. We use skeweness measure to discover useful interesting rules. These rules assist in the cube navigation resulting in the identification of interesting paths. The paths together with the selected rules, ensure that users reach interesting low level surprises in data cubes.

3. Preliminaries

3.1. Overview of the data cube model

Adapting from the terminology given in [29] let us assume that the data cube consists of m dimensions, $d_1, d_2,\ldots, d_m$. A dimension d_i is associated with a concept hierarchy containing one or more levels of aggregation. Level l_{ij} represents the jth level of dimension d_i, such that $1 \le j \le L_i$ where L_i is the number of levels associated with d_i. A level l_{ij} contains a set of attributes. Let v_{ijk} be the kth attribute at level l_{ij}. The facts are numerical measures, usually the objects of analysis. Assume that there are s facts, $f_1, f_2,\ldots, f_s$ in the fact table, and w_{pq} is the qth value for fact f_p, where $1 \le p \le s$, $w_{pq} \in W_p$ where W_p is the domain of possible values of fact f_p. The fact table contains the complete transaction set $T=[\tau_1, \tau_2,\ldots, \tau_n]$ where n is the total number of transactions. A transaction, τ is represented as $\{(x_1, x_2,\ldots, x_i,\ldots, x_m), (f_1, f_2,\ldots, f_p,\ldots, f_s)\}$, where x_i is an attribute (v_{ijk}) from the lowest level ($=L_i$) of d_i, and f_p is an associated fact. In the data warehouse literature, data cubes formed at different levels of dimensional hierarchies store data at different degrees of aggregation, each of which is called a cuboid. The highest level of aggregation is called the apex cuboid, often denoted as "ALL" [11]. We follow the same nomenclature in this paper. However, instead of storing the aggregates, we utilize the lowest level transactions that are used for calculating the aggregates at each cuboid. Aggregation often hides characteristics of the detailed data: an "extremely high" value and an "extremely low" value can be aggregated to a "moderate" value, hiding both extreme values. Using the lowest level of granularity, the problem of hiding a surprise as a side effect of aggregation is avoided [17].

Fig. 1 illustrates a partial lattice for a grocery database with product (P), time (T) and store (S) dimensions. Every node in the lattice corresponds to a set of transactions, also referred to as a

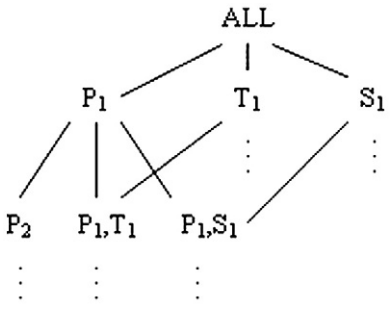

Fig. 1. A partial lattice of cuboids.

dataset. A cuboid consists of a set of nodes. For example, (P_1) represents a one-dimensional cuboid containing level-1 (product category) for the product dimension. Similarly, (P_1,T_1) represents a two-dimensional cuboid constructed by level-1 of product (category) and time (year) dimensions. Given m dimensions, latt(m) is a lattice of cuboids, each being a distinct combination of hierarchical levels of dimensions. An edge shows the possible navigation from one cuboid to another, for instance from (P_1) to (P_1,T_1). A navigation path describes a traversal through the nodes in the lattice. For example, the path $(P_1)\rightarrow(P_2)\rightarrow(P_2,T_1)\rightarrow(P_2,T_1,S_1)$ suggests that a user first looks at a node at (P_1) Product category, drills down to a node at (P_2) product subcategory, and subsequently views the nodes at (P_2,T_1) product subcategory and year and finally (P_2,T_1,S_1) product subcategory, year, and region. Given a node in a navigation path, subsequent nodes are determined either by drilling down one dimension from a preceding node or by including a new dimension that does not exist in the preceding node. One drill-down operation may only involve navigation by one level in a given dimension or traversing to a different dimension, but not both. This process may continue until there are no more nodes to traverse. For example, starting from a current node containing the rule "Year=1992", the candidate nodes to be examined for traversal include the rules {"Quarter=Q1-1992", "Quarter=Q2-1992", …, "Year=1992 and Product Category=Drinks", …, "Year=1992 and Region=Eastern", …}.

3.2. Discovery of sk-navigation rules

To discover the surprises in data cubes, the property of skewness, a measure of the asymmetry in data distribution, has been applied in a four-step recursive algorithm [15] as follows. (1) Given a current node, generate a set of candidate nodes, (2) Measure the skewness of candidate nodes, (3) Apply the test of significance of skewness on candidate nodes, and (4) Transform nodes with significant skewness into sk-navigation rules. Each candidate node in the data cube corresponds to a subset of transactions. Thus, the skewness is computed for the sample values of the random variable fact attribute in the set of transactions corresponding to that node e.g., for a node "category=drinks" and fact as profit, the skewness of that node is computed over the profit values of all transactions with "category=drinks".

Once a node with significant skewness is identified, it acts as the current node for generating candidate nodes at the next level. The algorithm terminates when either it reaches the lowest level nodes in the lattice, or no more nodes exhibit significant skewness in the current iteration.

An sk-navigation rule skr, as shown below, contains information about the dimensional levels, facts, their corresponding values and the level of significance and skewness.

$$(d_1 : l_{1j} = v_{1jk}, d_2 : l_{2j} = v_{2jk,\ldots}, d_i : l_{ij} = v_{ijk,\ldots}, d_m : l_{mj} = v_{mjk}) \rightarrow f_p = w_{pq}\left[\alpha, \sqrt{b_1}\right].$$

Here α and $\sqrt{b_1}$ are the level of significance and skewness of the random variable fact [3,17]. For instance, if the profit for a node "category=drinks" is positively skewed at $\alpha=0.05$ and $\sqrt{b_1}=2.63$, the corresponding sk-navigation rule would be "product category=drinks→profit=sk-high [0.05, 2.63]." relative to its parent node "product category=All". Sk-high in the consequent means a positive skewness in profit. Similarly, a negatively skewed node is represented by sk-low.

A partial discovery of sk-navigation rules is illustrated in Fig. 2. Starting from the sk-navigation rule 2: "year=1993→profit=sk-low", the next set of rules that can be generated are "year=1993: quarter=1993-Q2→profit=sk-low", and "year=1993: quarter=1993-Q4→profit=sk-high" assuming that both the two new set of rules are significantly skewed at $\alpha=0.05$. Cube navigation is facilitated by comparing the discrete sk-high or sk-low values at a prespecified α value for the parent and the child nodes as described in greater details in Section 4.

3.3. Cube navigation using sk-navigation rules

We use the discovered sk-navigation rules to assist users in the cube navigation process. A rule is also called a node of surprise, because it essentially represents a lattice node containing a significant skewed pattern relative to its parent. A user begins the navigation with a root node, and drills down to children nodes. Similarly, a node is rolled up by moving to its parent. A navigation path is defined by the complete traversal from a root node to a leaf node, and comprises the nodes visited during the traversal.

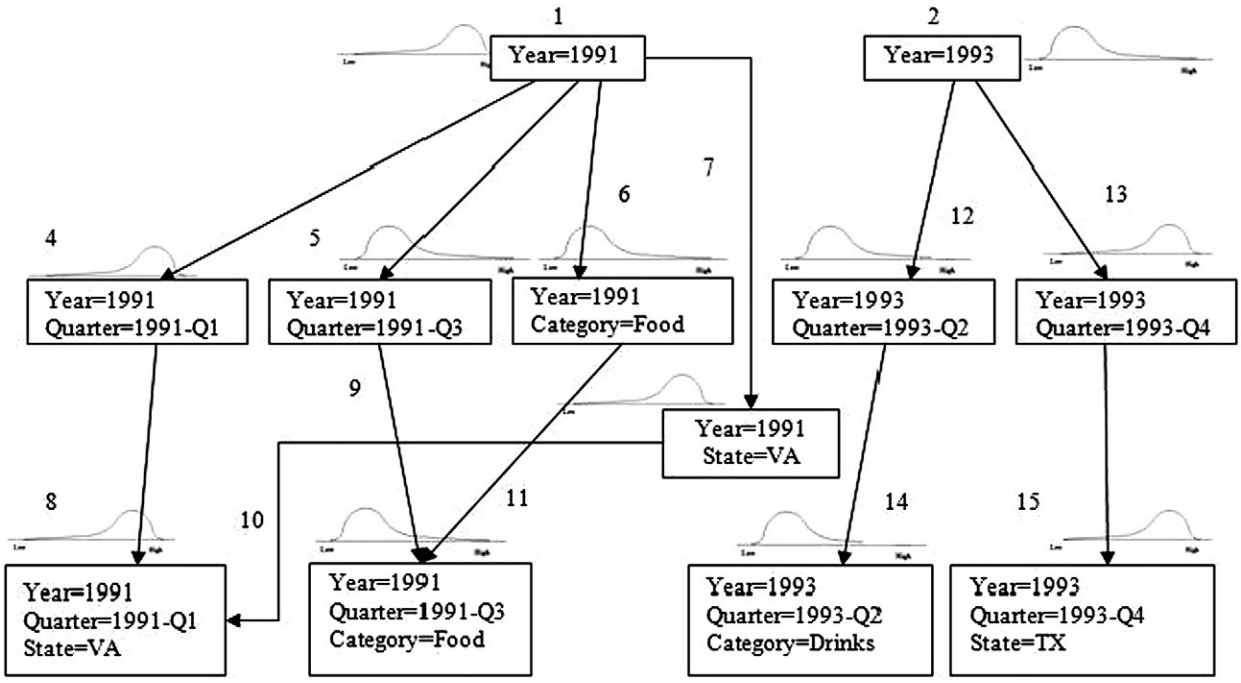

Fig. 2. Navigation paths based on sk-navigation rules at $\alpha=0.05$.

4. Interestingness of sk-navigation rules

In this section, we formally define measures of interestingness. Since our cube exploration approach is based on detecting surprises using skewness, we adopt the interestingness of rules in terms of unexpectedness of skewness and explain how sk-navigation rules can yield interesting navigation paths and dimensional attributes. Specifically, we propose three measures of interestingness as follows.

(1) Expectedness of sk-navigation rules. This measure determines interestingness of rules in terms of their unexpected patterns of skewness from the rules in the neighborhood.
(2) Axis shift in navigation paths. This measure identifies interestingness of navigation paths based on the global measures of axis shifts of sk-navigation rules.
(3) Generalization of sk-navigation rules. This measure quantifies the interestingness of attributes by computing their influence on lattice nodes of surprises.The three measures of interestingness are complementary to each other as each may generate different interesting rules (see examples using a real life data set in Section 5). Put together, these interestingness measures provide a high degree of flexibility to effectively and efficiently navigate combinatorially explosive data cubes. We now present each of the three measures in detail.

4.1. Expectedness of rules

To determine the interestingness of discovered sk-navigation rules, we introduce a neighborhood-based categorization of rules and then examine the rules for their expectedness based on their skewness. The rules are grouped into three categories, 'expected', 'unexpected', and 'not applicable (NA)', according to specific business rules predefined on skewness patterns for the pairs of navigation rules. An example of such a business rule is "profit increases with lower costs". A navigation rule is called expected if it complies with other discovered navigation rules, i.e., it exhibits a consistent pattern with respect to the other discovered rules. On the contrary, an unexpected rule indicates a directional change of skewness on pairs of parent-child rules. The antecedents of two rules are identical if they correspond to the same lattice node, whereas they are different if they correspond to parent-child lattice nodes. Let ante(skr_i) represent the antecedent of a rule. Let skew_diff(f_{p_1}, f_{p_2}) be the difference in the pattern of skewness of two facts f_{p_1} and f_{p_2} which is determined as follows.

$$\text{skew\_diff}(f_{p_1}, f_{p_2}) = \begin{cases} 0, & \text{if } f_{p_1} \text{ and } f_{p_2} \text{ comply with the business rule,} \\ 1, & \text{if } f_{p_1} \text{ and } f_{p_2} \text{ do not comply with the business rule} \end{cases}$$

An example business rule applied on two facts cost and profit is that "the profit increases (decreases) when the cost decreases (increases), assuming that the revenue is constant." Therefore, if two sk-navigation rules show positive skewness on profit and negative skewness on cost, they comply with the business rule, hence, skew_diff(profit, cost)=0. However, if the navigation rules show the same pattern of skewness (either positive or negative) for both cost and profit, they do not satisfy the business rule, thus skew_diff(profit, cost)=1. For example, if profit is positively skewed for one navigation rule and negatively skewed for another navigation rule, skew_diff (profit, profit) equals 1 for this pair of rules. Note that the skewness difference between two navigation rules for an identical fact, skew_diff(f_{p_i}, f_{p_i}), can also be measured.

We now discuss the three possible cases which measure the expectedness of navigation rules based on the skewness difference as illustrated in Fig. 3. Lattice node 1 is a parent node and lattice node 2 is a child node. The link b is a result of drilling-down, the links a, and c come about due to users examining different facts. The navigation rules skr_{i2}, skr_{j1}, and skr_{j2} are in the neighborhood of navigation rule skr_{i1}.

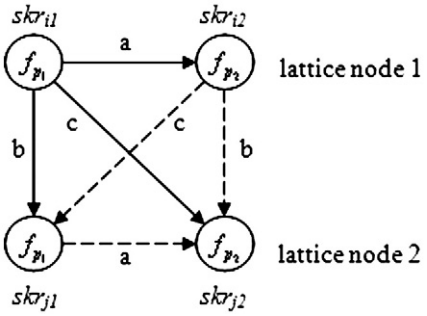

Fig. 3. sk-navigation rules and their neighborhood.

4.1.1. Case (a) same lattice node, different facts

The navigation rules skr_{i1} and skr_{i2} represent the same lattice node (identical antecedent) but contain different facts in the consequent i.e. ante (skr_{i1})=ante (skr_{i2}) and $f_{p_1}(skr_{i1}) \neq f_{p_2}(\text{skr}_{i2})$. The expectedness is measured as follows:

(a) skr_{i1} and skr_{i2} are expected if skew_diff($f_{p_1}(\text{sk}_{i1})$, $f_{p_2}(\text{sk}_{i2})$)=0.
(b) skr_{i1} and skr_{i2} are unexpected if skew_diff($f_{p_1}(\text{sk}_{i1})$, $f_{p_2}(\text{sk}_{i2})$)=1.
(c) skr_{i1} and skr_{i2} are NA if there is no business rules that include f_{p_1} and f_{p_2}.

Example 1. Let the following set of navigation rules represent different facts at the same lattice node.

skr_1: "year=1996→profit=sk-high"
skr_2: "year=1996→cost=sk-high"
skr_3: "year=1996→temperature=sk-low"

Assume that a business rule is defined on profit and cost such that profit and cost are negatively correlated. Then, skr_1 is NA when compared with skr_3 but is unexpected when compared to skr_2.

4.1.2. Case (b) different lattice nodes, same fact

The navigation rules skr_{i1} and skr_{j1} are connected by a parent-child relationship and share the same fact in the consequent, i.e. ante (skr_{i1})⊂ante (skr_{j1}) and $f_{p_1}(\text{skr}_{i1})=f_{p_1}(\text{skr}_{j1})$.

Based on the skewness patterns of navigation rules, we measure the expectedness as follows.

(a) skr_{j1} is expected if skew_diff($f_{p_1}(\text{skr}_{i1})$, $f_{p_1}(\text{skr}_{j1})$)=0.
(b) skr_{j1} is unexpected skew_diff($f_{p_1}(\text{skr}_{i1})$, $f_{p_1}(\text{skr}_{j1})$)=1.
(c) skr_{i1} is NA if l_{navig} (skr_{i1})=1 or skr_{i1} is a root node in its navigation path.

Example 2. The following set of navigation rules represents the same fact at different lattice nodes.

skr_1: "year=1996→profit=sk-high"
skr_2: "year=1996, month=Jan→profit=sk-high"
skr_3: "year=1996, month=May→profit=sk-low"
skr_4: "year=1996, product category=Drinks→profit=sk-high"

Here the parent skr_1 shows a high profit for year 1996. The children navigation rules representing 'Jan 1996' and '1996, drinks' contribute to high profits in 1996, and are identified as expected rules. However, the profit was low for the same year in the month of May (see rule skr_3). This is a surprise, which the user would not have expected by just looking at the parent, thus it is an unexpected rule.

4.1.3. Case (c) different lattice nodes, different facts

The navigation rules skr_{i1} and skr_{j2} represent different lattice nodes such that one rule's antecedent is a subset of another and they

contain different facts in the consequent, i.e. ante (skr_{i1}) ⊂ ante (skr_{j2}) and $f_{p_1}(skr_{i1}) \neq f_{p_2}(skr_{j2})$.

Based on the skewness difference of navigation rules, we measure the expectedness as follows.

(a) skr_{i1} and skr_{j2} are expected if skew_diff($f_{p_1}(skr_{i1}), f_{p_2}(skr_{j2})$)=0.
(b) skr_{i1} and skr_{j2} are unexpected if skew_diff($f_{p_1}(skr_{i1}), f_{p_2}(skr_{j2})$)=1.
(c) skr_{i1} and skr_{j2} are NA if f_{p_1} and f_{p_2} do not define any business rule with each other.

Example 3. Assume the following set of navigation rules.

skr_1: "year=1996→profit=sk-high"
skr_2: "year=1996, month=Jan→cost=sk-high"
skr_3: "year=1996, month=May→cost=sk-low"
skr_4: "year=1996, month=Dec→inventory=sk-low"

In this example the navigation rules represent different lattice nodes. Drilling down on skr_1 reveals skr_2 and skr_3 while following another navigation path reveals skr_4. Based on the business rule, skr_3 is expected when compared with skr_1. However, skr_2 is unexpected compared to skr_1. The navigation rule skr_4 is NA when compared to skr_1.

4.1.3.1. Using the expectedness of navigation rules in path selection. The expectedness of navigation rules reveals useful information for selecting navigation paths. For instance, the paths can be ranked by the number of unexpected navigation rules. Users can simply drill down on paths that contain a large number of unexpected navigation rules, and examine the corresponding datasets that contain many highs and lows in the transaction set. Fig. 2 illustrates an example in path selection using the unexpectedness measure (a path is constructed by a set of sk-navigation rules). Starting with navigation rules 1 (year=1991→profit=sk-high) and 2 (year=1993→profit=sk-low), the first level navigation paths are the following: 1→4, 1→5, 1→6, 1→7, 2→12, and 2→13. From these navigation paths, rules 5, 6, and 13 are unexpected since the skewness difference for these nodes compared to their parent nodes is 1. A preference may be given to navigate the path 1→5→9, 1→6→11 2→13 over the other navigation paths.

4.2. Axis shift in navigation paths

While expectedness is a useful characteristic of individual sk-navigation rules, the interestingness of navigation paths is another valuable property. In practice, a user may seek a short list of paths from a multitude of navigation paths that may lead to potentially interesting surprises. If a measure of interestingness of paths is provided, users can be selective in cube navigation by comparing numerous paths. The following example demonstrates the need for interestingness of paths and reveals that simply applying the expectedness of navigation rules may not be sufficient to differentiate between navigation paths.

Consider two navigation paths np_1 and np_2 with an equal number of expected, unexpected, and NA navigation rules at the level of significance α. Using the expectedness of navigation rules measure, the user cannot discriminate between these paths since they contain the same number of expected and unexpected surprises. However a closer examination may suggest a difference in extremity of surprises of individual navigation rules in the two paths. For instance, np_1 may contain the navigation rules representing surprises with very high skewness. On the other hand, np_2 may contain navigation rules that have been identified as surprises but are just above the significance level α. While np_1 is a more interesting path than np_2, users may initially navigate np_2 due to the lack of discriminatory power related to the level of significance for the expectedness measure.

To address this problem, we propose an Axis Shift Theory (AST) to measure the interestingness of navigation paths. AST uses shift metrics to discriminate between paths that contain multiple interesting navigation rules. A shift is measured by the movement of the mean reference axis when navigating from a parent to a child node. AST evaluates the shifts made by individual navigation rules to determine the overall interestingness of a navigation path. As mentioned earlier, the navigation rules are discovered in parent–child pairs where both parent and child represent their respective datasets and the characteristics of datasets such as mean (μ), standard deviation (σ) and variance (σ^2). Given that $\bar{f}_{p[\text{parent}]}$ and $\bar{f}_{p[\text{child}]}$ are the respective means for parent and child node on a fact f_p, a shift in the mean reference axis is calculated by the difference ($\bar{f}_{p[\text{parent}]} - \bar{f}_{p[\text{child}]}$). We call it a shift because the reference axis essentially shifts from $\bar{f}_{p[\text{parent}]}$ to $\bar{f}_{p[\text{child}]}$. Once moved to nod_{child}, the new mean $\bar{f}_{p[\text{child}]}$ is subsequently used as the reference axis for discovering its children nodes.

Assume that a navigation path np_x contains t sk-navigation rules. Then, $np_x = \{skr_i : 1 \le i \le t\}$. Let f_p be the fact in the consequent for navigation rule skr_i and $\bar{f}_{p[i]}$ be the mean value for f_p. In order to measure the interestingness of navigation paths, we define three shift metrics as follows.

4.2.1. Linear shift (linSh)

We measure the linear shift of a navigation path by taking the cumulative sum of shifts of individual navigation rules as follows.

$$\text{linSh}(np_x) = \sum_{i=1}^{t} \left(\bar{f}_{p[i]} - \bar{f}_{p[i-1]}\right),\ skr_i \in np_x.$$

The linear shift, linSh(np_x), measures the relative shift of the mean reference axis when drilling down from the root node to a leaf node (lowest level navigation rule) in the path. It identifies the paths that contain a significant positive or negative movement of the mean reference axis during navigation. Based on such individual shifts in a path, the linear shift of a path can be positive, negative, or zero. We consider the three cases individually as follows.

$$\text{linSh}(np_x) > 0 \quad \text{(i)}$$

A positive linear shift of a path indicates a significant movement of the root mean reference axis on its right side (positive skewness). It also suggests that the corresponding datasets in the path are likely to be either less negatively or more positively skewed when drilling down to low level sk-navigation rules. An interesting linear shift is observed if a child of a root node is negatively skewed but the linear shift for the path is positive, thereby indicating some unexpected navigation rules with large axis shifts in the navigation path. The path 1→4→8 in Fig. 2 illustrates a positive linear shift. The path 1→6→11may indicate an interesting linear shift.

$$\text{linSh}(np_x) < 0 \quad \text{(ii)}$$

A negative linear shift of a path indicates a significant movement of the root mean reference axis on its left side (negative skewness), which also suggests the likelihood of less positively skewed or more negatively skewed datasets at lower hierarchical levels. An interesting linear shift is observed if a child of a root node is positively skewed but the linear shift for the path is negative, thereby suggesting one or more unexpected navigation rules with large axis shifts in the navigation path. The path 2→12→14 in Fig. 2 indicates a negative linear shift.

$$\text{linSh}(np_x) = 0 \quad \text{(iii)}$$

If a child of a root node is skewed (positively or negatively) but the total linear shift of the path is zero, then the path contains both positively and negatively skewed datasets which balance out each other and result in a zero axis shift. In this case, it is interesting to examine the square shift or absolute shift (described later) to further examine the navigation path for its interestingness. Based on Fig. 2, it is unclear whether the navigation path 1→5→9 has a positive or a negative linear shift.

The paths can be ranked according to their degree of interestingness, by arranging them in a descending order of the magnitude of linear shifts, $|\text{linSh}(\text{np}_x)|$. The paths with higher linear shifts (positive or negative) thus appear at the top for the user's selection list.

4.2.2. Absolute shift (absSh)

The absolute shift measures the total shift of navigation rules in a navigation path irrespective of the direction of individual axis shifts. We define the absolute shift by taking the cumulative sum of scalar axis shifts as follows.

$$\text{absSh}(\text{np}_x) = \sum_{i=1}^{t} |\overline{f}_{p[i]} - \overline{f}_{p[i-1]}|,\ skr_i \in np_x.$$

The absolute shift is a useful metric to identify the navigation paths that contain larger shifts of mean reference axis regardless of individual negative or positive shifts. It preserves the magnitude of total shift by avoiding nullification of positive and negative axis shifts. For example, a path np_x having two axis shifts of $+s$ and $-s$ results in zero linear shift, however it has an absolute shift of $2s$. A nonzero absolute shift implies a definite axis shift for the path. The larger the absolute shift, the greater the axis shifts of navigation rules are expected to be. In conjunction with linear shifts, users can select paths by their absolute shifts. While the linear shift for path np_2 is approximately zero, the absolute shift has a larger value. This suggests that the path np_2 is the result of significant shifts on both the right and the left side of the root mean reference axis, rather than an overall significant yet static shift in one direction.

4.2.3. Root square shift (rsSh)

The root square shift measures the interestingness of a path to distinctively account for the influence of individual high axis shifts. For instance, individual axis shifts, when combined together, may lead to a high absolute shift even if the individual shifts are not high enough. On the other hand, even a single high value shift can dictate the total shift of a path and is interesting to examine. The root square shift identifies these highs and lows in axis shifts to assist in comparing the navigation paths. We define the root square shift as follows.

$$\text{rsSh}(\text{np}_x) = \sqrt{\sum_{i=1}^{t} \left(\overline{f}_{p[i]} - \overline{f}_{[i-1]}\right)^2},\ \text{skr}_i \in \text{np}_x.$$

4.2.4. Using the three shift metrics

The three shift metrics complement each other. The linear shift can be used to examine the paths with a significantly high positive or negative movement of the root mean reference axis. The absolute shift can be used to identify paths with high traversals of the root mean reference axis. The root square shift can be used to reveal paths that contain an unexpectedly high axis shift made by at least one navigation rule. An example which describes the different shifts is provided next.

Example 4. Assume two navigation paths np_1 and np_2. The individual node shifts and the three shift metrics for both paths are shown in Table 1. In this case, the root square shift clearly distinguishes between the interestingness of np_1 and np_2, which would not have been identified using linear and/or absolute shifts.

4.3. Generalization of sk-navigation rules

The axis shift theory determines which navigation paths are interesting as established by the shift metrics. As the size of database increases, the number of sk-navigation rules (and thus the paths) can also grow significantly thus overwhelming the users who typically need to retrieve only a short list of navigation rules or paths for analysis, and also expect proper system guidance with minimal intervention. This can be achieved if the system provides the interestingness information at the earliest possible stages of navigation. It also helps users to have *apriori* knowledge about the types of surprises hidden in subsequent lower level lattice nodes. For instance, if there are four product categories, "Food", "Drinks", "Supplies", and "Gifts" and all of them provide sets of navigation paths to reach the surprises, a question arises as to which of these dimensional attributes are more interesting than others. While "Food" may provide more sk-navigation rules than the others, "Gifts" may lead to more unique surprises. Therefore, a measure of interestingness of attributes can effectively help users navigate only through the paths containing interesting attributes. To determine the interestingness of attributes, we introduce a simple and effective method of generalization of sk-navigation rules, which determines the attributes from dimensional hierarchies that substantially contribute to the discovery of surprises. For example, when a user is given the information that a large number of surprises exists when "Quarter = 1997-Q3", then it is beneficial to navigate along the paths that contain 1997-Q3 as the navigation rules' antecedent.

We examine the navigation rules' antecedents to determine the relative influence or significance of attributes from different dimensions and also detect the unique navigation paths that the attributes belong to. To determine the attribute influence (attrInf) of an attribute v_{ijk} of dimension d_i at level l_{ij}, we perform a two-step generalization of navigation rules on the attribute as follows:

Step 1) Identify the navigation ruleset $\text{skr}(v_{ijk})$, at a skewness significance level of α, in which every navigation rule must either contain the attribute v_{ijk} or an attribute $v_{i'j'k'}$ in the antecedent such that $v_{i'j'k'}$ is a lower level attribute ($i=i', j<j'$) from dimensional hierarchy of dimension d_i, and v_{ijk} is an ancestor of $v_{i'j'k'}$.

Step 2) Determine the attribute influence for v_{ijk} as follows. $\text{attrInf}(v_{ijk}) = \frac{|\text{np}_x(v_{ijk})|}{\sum_{vi,j,k} |\text{np}_x(v_{ijk})|}$.

where $\text{np}_x(v_{ijk})$ is identified by a navigation from the navigation rule "$\text{di}:l_{ij}=v_{ijk}$" to a leaf rule skr_j such that $\text{skr}_j \in \text{skr}(v_{ijk})$ and $|\text{np}_x(v_{ijk})|$ represents the total number of paths belonging to attribute v_{ijk}. The value of $\text{attrInf}(v_{ijk})$ ranges from 0 to 1. If an attribute does not lead to any surprises, its attribute influence equals zero, suggesting that the attribute is not an interesting one. On the contrary, an attribute influence of 1 identifies a highly influential attribute.

To measure attribute influence we choose navigation paths as opposed to the number of leaf nodes (lowest level surprises) to avoid the deficiencies associated by the latter measure. A path essentially considers both the leaf nodes and their accessibility from the attribute. Two or more attributes can lead to an equal number of leaf nodes but exhibit different number of paths to reach them. Also, an attribute can link to multiple paths leading to the same leaf node since a leaf node will connect to at least one navigation path. However, the reverse is not true i.e., two leaf nodes cannot be reached through the same navigation path. If we consider the leaf nodes as the basis of a comparison, two or more dimensional attributes with equal number of leaf nodes will not be distinguishable using attribute influence despite the fact that one attribute guides the users through higher number of paths to reach the surprises. We illustrate such a scenario in Fig. 2: using this navigation ruleset, we determine the unique navigation paths to measure the attribute influence for year 1991 and 1993. For this navigation ruleset, rules 8 and 10 on one hand, and rules 9 and 11 on the other, are identical even though they were generated by different traversals. Table 2 presents the attributes, their attribute influence (given that the total number of paths by all dimensional attributes is 15), and their leaf node counts.

Table 1
Comparison of shift metrics for two navigation paths

Path	Node shifts			linear shift	absolute shift	root square shift
	node_shift1	node_shift2	node_shift3			
np_1	30	35	40	105	105	61.03
np_2	3	2	100	105	105	100.06

Table 2
Attribute influence for 1991 and 1993

Attribute	Paths (using rule_id)	attrInf	# leaf nodes identified
1991	1→4→8 1→5→9 1→6→11 1→7→10	0.267	2 ("quarter=1991-Q1, state=VA", "quarter=1991-Q3, product category=Food")
1993	2→12→14 2→13→15	0.133	2 ("quarter=1993-Q2, product category=Drinks", "quarter=1993-Q4, state=TX")

While both 1991 and 1993 lead to equal number of leaf nodes, 1991 is a more influential attribute than 1993 since it provides the users with a higher number of unique paths to reach the surprises as determined by attrInf(1991)>attrInf(1993).

5. Interestingness measures on a real world dataset

In 2003, motor vehicle crashes ranked third in terms of the number of years of life lost, behind cancer and heart diseases [28] where the number of years of life lost is estimated as the additional number of years an individual is expected to live had he/she not died in the crash. Using a combination of OLAP, GIS, and statistical tools, managers of the road infrastructure are able to determine problem roads and intersections and take corrective actions. Enforcement officers can implement different tactics based on external factors e.g., time and day of week, weather conditions, driving under influence, etc. A prototype system named MSAC was described in [1], which described a crash reduction system which provides OLAP capabilities to all stakeholders through a unified architecture to facilitate collaborative decision-making. MSAC users manually navigate paths of interest to them. Based on our experience with the system implementation and with discussions with various stakeholders, we identified that guided knowledge discovery would be extremely desirable for improving the functionality of the prototype.

Based on manual cube navigation using MSAC, the following prior knowledge was known to the enforcement officers: the vehicle crash costs are the highest on Friday nights after 8:00 pm, and in districts with a large number of highways. In order to test the identification of navigation paths with high degree of interestingness, we obtained the dataset containing 534,941 commercial vehicle crash records from the MSAC researchers that ranged from 1993 to 2001 in the state of Maryland. Each individual crash record details the crash location, day and time, severity of crash, and the crash cost. We examined the interestingness of the skewed patterns for this dataset for the three measures: expectedness of the navigation rules, axis shift in paths, and attribute influence.

On the expectedness measure, 372 expected and 52 unexpected sk-navigation rules were discovered within a total of 2.06 min. Using the axis shift skewed interestingness measure, the most interesting path identified by linear shift, absolute shift, and root square shift all led to the following sk-navigation rule "District=3-Greenbelt, County=Prince George, Day_Of_Week=Tuesday, and Time_Interval=4 pm–8 pm" with high positive shifts in the crash costs at each navigation level. This fact is interesting since it does not conform to the *apriori* knowledge that most crashes take place on Friday after 8:00 pm. Without this information, users would have spent considerable amount of time navigating the paths that have Friday as a node. The second most interesting measures for navigation paths were different at the second navigation level for linear shift, absolute shift, and root square shift; however, in all three of these shifts, "District=3-Greenbelt" was identified as the root navigation node.

Table 3 shows the attribute influence of the top seven attributes. Interestingly "District=7-Frederick" was identified as the attribute with the largest influence on the interesting navigation paths. While, *apriori*, the user might suspect that Frederick district would be interesting, since two Interstate Highways pass through the district, the attribute influence clearly identifies this as an interesting starting node. Thus, there are ample opportunities for enforcement officials to navigate starting from the Frederick node in order to gain more understanding on crashes.

Table 3
Seven best attribute influence for crash dataset

Dimensional attribute	Attribute influence
District='Frederick'	0.390244
District='Annapolis'	0.317073
District='Greenbelt'	0.195122
Day of week='Tuesday'	0.195122
County='Anne Arundel'	0.170732
County='Frederick'	0.170732
Day of week='Saturday'	0.170732
Day of week='Friday'	0.170732
Day of week='Sunday'	0.146341

6. Experimental results

In this section, we present a set of experiments to evaluate the measures of interestingness and their scalability. Specifically, we demonstrate the interestingness from: (a) expectedness of rules, (b) navigation paths using axis shifts, (c) generalization of attributes, along with (d) the execution time, and (e) the space overhead. All experiments were performed on a 1.7 GHz Pentium IV machine with 512 MB RAM running Windows XP. The algorithms were implemented in PL/SQL on an Oracle 10g database.

6.1. Experimental setup

We adapted the Grocery database [14] to produce five test datasets as shown in Table 4. The number of navigation nodes is obtained by adding all lattice nodes in all possible cuboids. Each of the datasets contains three dimensions, Product, Time, and Store, and two facts, Profit and Cost. The dimensional hierarchies are Product {Category, Subcategory, Brand}, Time {Year, Quarter, Month}, and Store {Region, State, City}. The number of attributes at the lowest level varied from 16 to 64 for Product, 20 to 192 for Time, and 14 to 31 for Store dimension. We generated surprises which represent transactions containing 15–20% high or low profit values compared to the rest of the transaction set. The numbers of transactions with surprises varied from 1 to 25 for each of the datasets. The surprises were also generated in a manner such as to conceal their presence at the higher levels of aggregations (for example, the aggregate "Product category=Drinks" would not show the surprises in transactions "Product Brand=Pepsi" and "Product Brand=Ahold 2% Milk").

Due to the nature of navigating the lattice nodes, a surprise at the lowest level affects its higher level lattice nodes, such as a surprise at a node "city=Fairfax" will influence two additional nodes, "state=VA" and "region=Eastern". We measure this phenomenon by defining an influence rate, which is calculated as the number of nodes affected divided by the total number of nodes. Fig. 4 shows the number of

Table 4
Summary of the five experimental datasets

Dataset	# records	# nodes of navigation
DS-1	4480	5893
DS-2	32,240	37,119
DS-3	97,384	107,095
DS-4	20,736	131,759
DS-5	190,464	1,035,893

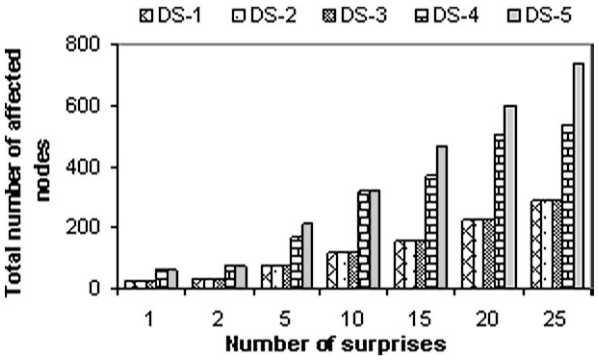

Fig. 4. Surprise vs. total affected nodes.

nodes in the grocery cube lattice that were affected by the lowest level surprises. For instance in dataset DS-5, the existence of only one lowest level surprise (affecting a single transaction) "Product subcategory=Orange Juice, month=1991-Q1_Jan, city=Baltimore" affected 62 nodes from a total 1,035,893 nodes in the lattice giving an influence rate of 0.005%. As expected, increasing the number of surprises to 5 in the same dataset resulted in a higher influence rate of 0.020% by affecting a total of 215 nodes. At the maximum, when 25 lowest level surprises existed in DS-5, the influence rate was 0.071% affecting 738 nodes.

Apart from varying the number of surprises, we also discovered the navigation rules at various levels of significance (α) ranging from 0.005 to 0.1.

6.2. Interestingness from expectedness of navigation rules

Fig. 5 examines the effect of α on the expectedness of discovered navigation rules for the dataset DS-5 when the number of surprises was set to 25. The plot "pos_exp" shows the percentage of expected navigation rules with positive skewness; "neg_unexp" shows the percentage of unexpected navigation rules with negative skewness, and so on. While the total number of expected navigation rules is always greater than the total number of unexpected navigation rules, we observe a higher percentage of unexpected navigation rules discovered with an increase in the value of α. When α was increased to 0.05, 8.17% unexpected navigation rules were discovered. It further increased to 31.62% at α=0.1. This happened as a result of a decrease in critical skewness with higher α's. At lower α's, only the very highly skewed patterns, which were more prominent and mostly expected, were detected. The discovery of 31.62% unexpected navigation rules also suggests that a large number of unexpected skewed patterns were concealed in low level lattice nodes and were not noticeable at higher levels of aggregation at smaller α's. Finally, we notice that there is a relatively larger drop in the percentage of

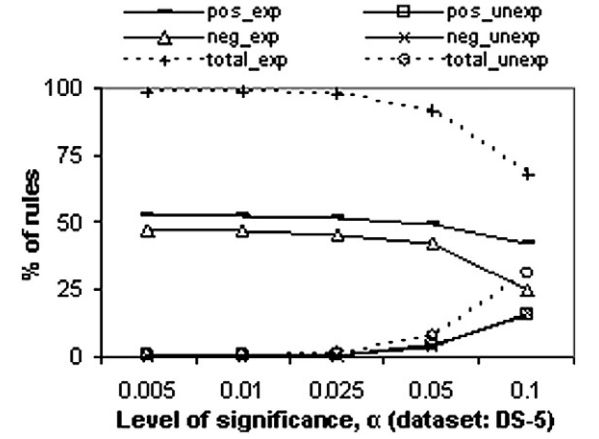

Fig. 5. Interestingness of rules for DS-5 at surprises=25.

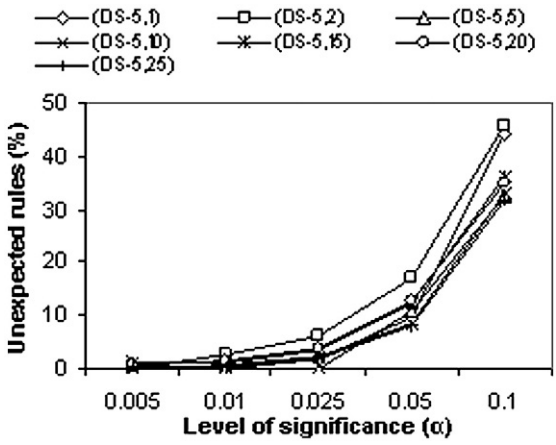

Fig. 6. Discovered unexpectedness rules for DS-5.

navigation rules for neg_exp (from 46.67% to 25.15%) curve compared to pos_exp curve (52.08% to 42.55%). It is attributed to the discovery of large number of unexpected navigation rules at higher α's. It also suggests that there are more positively skewed patterns than negatively skewed patterns in the dataset.

Fig. 6 further illustrates the discovery of unexpected skewed patterns in DS-5 at various values of α and number of surprises. In each of the instances, the percentage of unexpected navigation rules increased with an increase in α. For example, when α increased from 0.025 to 0.05, the number of unexpected navigation rules increased from 5 (1.93% of total) to 41(8.17% of total) for 25 surprises. At the maximum, 45.63% unexpected skewed patterns were detected for 2 surprises and α=0.1.

Fig. 7 compares the percentages of interesting navigation rules for the five datasets when the number of surprises and α were set to 25 and 0.05 respectively. We observe that DS-3 contained a higher percentage of unexpected navigation rules (24.19%) compared to other datasets. Also, this 24.19% was almost equally divided between positively skewed unexpected (12.39%) and negatively skewed unexpected (11.80%), highlighting an important issue with data aggregation: it is difficult to detect skewed patterns by looking at aggregated datasets. However, sk-navigation rules detect these skewed patterns as explained in DS-3.

6.3. Interestingness from navigation paths using Axis Shift

Fig. 8(a) illustrates how the Axis Shift Theory is used in DS-5 to reach surprises through interesting navigation paths when the number of surprises and α were set to 25 and 0.05 respectively. We

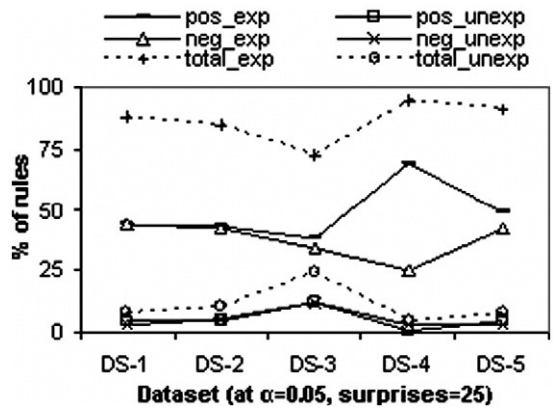

Fig. 7. Interestingness of rules for five datasets with α=0.05.

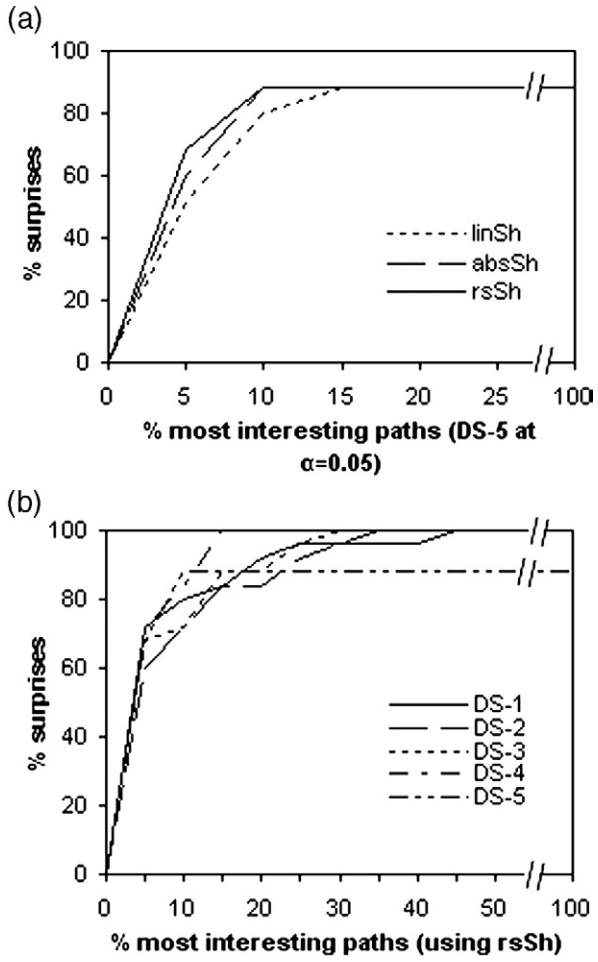

Fig. 8. Interestingness of paths for (a) DS-5, and (b) Five datasets at α=0.05.

first calculated the linear shift (linSh), absolute shift (absSh), and root square shift (rsSh) for discovered navigation paths. Then, the paths were ranked from highest to lowest interestingness by arranging them in descending order of shifts, for each of the three shift metrics. We then examined the percentage of surprises detected by various sizes of top interesting paths.

Fig. 8(a) shows that every shift metric required only a small set of interesting paths to reach all surprises discovered by sk-navigation rules (88% of total implanted surprises). For instance, only the top 15% (most interesting) paths were able to reach all discovered surprises using linSh. Also, both absSh and rsSh reached the surprises by using only 10% of the paths. We observe that rsSh is the steepest of the three shifts, thereby indicating that it performed better than linSh and absSh to reach the discovered surprises. A horizontal straight line after 15% paths for linSh (and 10% for absSh and rsSh) suggests that the same sets of surprises were reached afterwards using the less interesting navigation paths.

Fig. 8(b) further illustrates the use of rsSh to determine interesting paths for five datasets when the number of surprises and α were set to 25 and 0.05 respectively. We observe that only the top 15% (32 out of a total of 218 paths) and the top 10% (23 out of a total of 235 paths) of most interesting paths were able to reach the surprises in DS-4 and DS-5 respectively. This clearly shows that the surprises were reachable by a significantly smaller set of pruned paths. Also, the top 35% and 30% paths were able to reach all the surprises in DS-2 and DS-3. A highest 45% of the paths were needed to reach every surprise in DS-1 but in reality the top 21% paths had detected 96% of the surprises. Only one less-skewed surprise "product subcategory=Juice, quarter=1993-Q2, city=San Diego" was reached by the 64th most interesting path from a total 142 paths, thus requiring 45% paths to reach every surprise in the dataset.

6.4. Interestingness from generalization of sk-navigation rules

Fig. 9 shows the top three most interesting attributes for the five datasets when the number of surprises and α were set at 25 and 0.05 respectively. These attributes were identified based on the frequency of their presence in navigation paths leading to surprises as discussed in the section on generalization of attributes. As shown in the figure, 'MD', 'Supplies', and 'PA' were the top three most interesting attributes in DS-1, DS-2, and DS-3, though they showed different attribute influences for different datasets. For instance, 'MD' in DS-1 identified 53.52% of the possible navigation paths leading to surprises, whereas the same attribute identified 49.33% and 45.91% of the paths in DS-2 and DS-3 respectively. The 'Eastern' region was identified as the most interesting attribute in DS-5 with a 50.21% attribute influence. It is important to note that this interestingness measure identified one attribute in each of the datasets that led to the discovery of at least 45% of the surprises. The attributes ranked by attribute influence provide guidance to users in cube navigation. For example, when navigating through DS-1, users are most likely to reach the surprises if they drill down on 'MD' followed by 'Supplies', and 'PA'.

6.5. Execution time and space overhead

Fig. 10(a) shows the total time to identify the interesting navigation paths in DS-5 as a function of α. We first deduced the navigation paths from sk-navigation rules and then applied the axis shift theory to measure the interestingness of paths in terms of linSh, absSh, and rsSh. The graph suggests an increase in execution time for higher α's. This is expected because at higher α's, a larger number of paths are discovered. However, even for the largest dataset DS-5, the interesting paths were discovered with a very low processing time. For instance, it took only 2083 ms to discover 147 interesting paths with 10 surprises and α set at 0.05. The execution time reached a maximum 2613 ms when 495 interesting paths were identified in DS-5 at a peak α value of 0.1 and 25 surprises. Fig. 10(b) shows the execution time for different datasets at α=0.05. The interesting navigation paths were discovered within a low processing time for each of the five datasets: it took only 1810 and 1882 ms to discover interesting paths in DS-3 and DS-4 respectively for 20 surprises. It took the maximum 2464 ms to discover 235 interesting paths in DS-5 for 25 surprises.

The space overhead to store the interesting navigation rules and navigation paths for the datasets at α=0.05 is very low. It ranges from 0.22% (in DS-5) to 2.05% (in DS-4). This small overhead is due to the fact that the number of navigation rules does not increase in direct proportion to the size of the datasets.

7. Comparison with previous work

We used the motor vehicle crash data described in the previous section for comparing our method with previous work. Even though the literature provides many interestingness measures such as support, confidence, lift, conviction, surprisingness and novelty, none of these measures have been developed for navigating multi-dimensional data

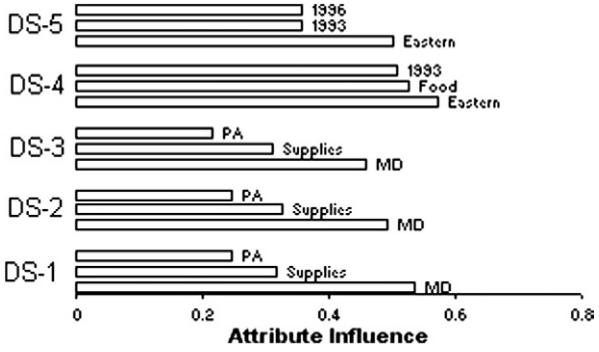

Fig. 9. Attribute influence for datasets at α=0.05.

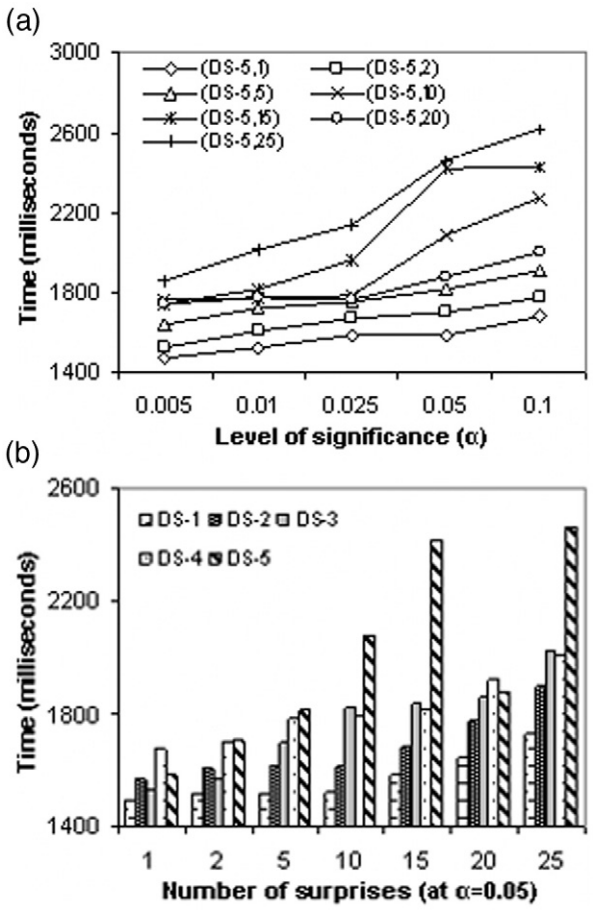

Fig. 10. Execution Time for (a) DS-5 as a function of α, and (b) five datasets at α=0.05.

cubes. To the best of our knowledge, the only method other than what we describe in this paper was discovery driven cube exploration, developed by Sarawagi et al. [22–25]. The crash dataset has nine dimensions and several facts, of which, we will consider only one measure, namely, crash_cost. The dimensions and the number of distinct values for each level of the dimensional hierarchies are shown in the Table 5. The lowest level is shown as "NONE" as there is no possible navigation from there. Some of the dimensions have been flattened out for simplification but this does not impact the generality or the comparison of the two methods. In the method described in [22–25], a user has to view 10,439 screens of all possible exception values (InExp, SelfExp, and PathExp) which is the total number of possible cuboids [11] since the navigation is based on the values of the dimensions at each level. In addition, for each screen, the user will have to visually inspect tables with number of rows ranging from 2 to 9497. Both of the above are impractical for user-driven navigation.

In contrast, our method lists the rules instead of the actual values of the dimensions. So a user can select a rule to drill-down the same dimension or drill across another dimensions which limits the number of cuboids visited. In our proposed method, the maximum number of dimensional values is 88, calculated by counting the total number of possible navigations. Each screen will have a rank-ordered set of 10 rules. It means that in the worst case, the user can view 880 possible screens. Both of the above are worst case scenarios for user-driver cube navigation as summarized in the Fig. 11 below.

Table 5
Description of dimensions

Dimension	Level 1	Level 2	Level 3	Total
Time	9	12	7	756
Location	25	1	None	25
Collision	2	25	None	50
Vehicle	24	None	None	24
Route	9497	None	None	9497
Environment	23	None	None	23
Severity	2	None	None	2
Junction	9	None	None	9
Contribution	53	None	None	53
Total				10,439

8. Conclusions and future work

In this paper, we presented the measures of interestingness of skewness based navigation rules which are used to discover interesting surprises hidden in multidimensional cubes. We investigated interestingness in three different ways examining: unexpectedness of navigation rules; an axis shift in the navigation path, and generalization of navigation rules. First, unexpected navigation rules are discovered by examining their skewness' differences from the navigation rules in the neighborhood. Second, the theory on axis shifts identifies the interesting navigation paths in terms of linear shift, absolute shift, and root square shift, which are tested on a real-world dataset. Finally, the generalization of navigation rules identifies interesting dimensional attributes which lead to large numbers of low level interesting surprises. We also conducted detailed experiments on five different sets of grocery data to evaluate these measures of interestingness.

The measures of interestingness suitably fit into business intelligence arena. Executives and analysts can navigate through OLAP cubes using sk-navigation rules to gain useful insights into multidimensional datasets. In BI dashboards, the interestingness measures can fittingly work as a set of cues to provide visibility into datasets and to help organizations reach stated goals by leveraging information and analytics. For instance, executives can begin cube navigation by first looking at attribute generalization measure to instantly know about interesting attributes. Then, they can select an interesting attribute and navigate through unexpected sk-navigation rules. They can also filter on navigation paths based on axis shifts so as to view the paths that interest them.

The measures of interestingness can suitably be applied to business domains where metrics are real-valued to measure for skewnesss and where the rules can be compared relative to each other (such as rules for their unexpectedness or attributes for their attribute influence). Metrics such as time (production time, shipment time, product assembly time), cost (production cost, operational cost, cost of inventory), revenue, profit, units of product sold, fit our measures of interestingness quite well.

As an example, reducing the "production time" is usually a goal for manufacturing companies. A positively skewed production time (i.e. it is taking longer than average) might suggest to the executives to look for inefficiency factors by product by region. They can drill down on navigation rules to identify the products in regions that have longer than expected production times. This knowledge can then be used to determine the underlying reasons for the production bottlenecks. On the other hand, a negative skewness on production time metric is a good sign.

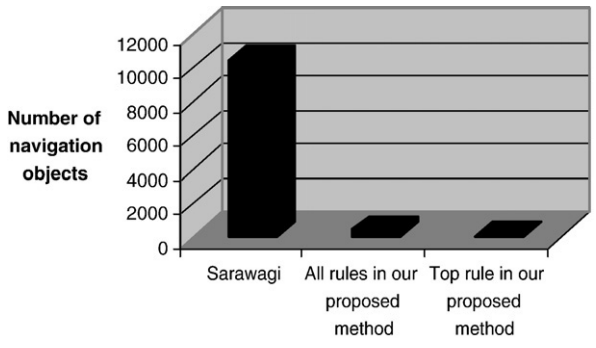

Fig. 11. Comparison with Sarawagi et al. [24,25]

We are currently extending our work on the concept of cube navigation to allow users to view in a tree structure, all possible navigation paths from a navigation rule and then delve directly into a navigation rule of interest. We are further enhancing our work to allow users to investigate the corresponding transaction set for a navigation rule such that the highly skewed transactions are displayed clearly at the top.

References

[1] S. Bapna, A. Gangopadhyay, A web-based GIS for analyzing commercial motor vehicle crashes, Information Resources Management Journal 18 (2005) 1–12.
[2] B. Barber, H.J. Hamilton, A comparison of attribute selection strategies for attribute-oriented generalization, presented at International Symposium on Methodologies for Intelligent Systems (ISMIS'97), Charlotte, NC, 1997.
[3] R.B. D'Agostino, M.A. Stephens, Goodness-of-Fit Techniques, Marcel Dekker, Inc., New York, NY, 1986.
[4] G. Dong, J. Li, Interestingness of discovered association rules in terms of neighborhood-based unexpectedness, presented at Second Pacific-Asia Conference on Research and Development in Knowledge Discovery and Data Mining, 1998.
[5] C.C. Fabris, A.A. Freitas, Discovering surprising instances of Simpson's Paradox in hierarchical multidimensional data, International Journal of Data Warehousing and Mining 2 (2006) 27–49.
[6] M. Garcia, M. Quitales, F. Penalvo, M. Martin, Building knowledge discovery-driven models for decision support in project management, Decision Support Systems 38 (2) (2004) 305–317.
[7] L. Geng, H.J. Hamilton, Finding interesting summaries in Genspace graphs efficiently, presented at Canadian Artificial Intelligence Conference (AI 2004), London, ON, Canada, 2004.
[8] L. Geng, H.J. Hamilton, Interestingness measures for data mining: a survey, ACM Computing Surveys 38 (2006).
[9] H.J. Hamilton, L. Geng, L. Findlater, D.J. Randall, Efficient spatio-temporal data mining with genspace graphs, Journal of Applied Logic 4 (2006) 192–214.
[10] J. Han, Towards on-line analytical mining in large databases, presented at ACM SIGMOD International Conference on Management of Data, 1998.
[11] J. Han, M. Kamber, Data Mining: Concepts and Techniques, 2nd edMorgan Kaufmann, 2006.
[12] R.J. Hilderman, H.J. Hamilton, Knowledge Discovery and Measures of Interest, Kluwer Academic, Boston, MA, 2001.
[13] W.H. Inmon, Building the Data Warehouse, John Wiley & Sons, New York, 1996.
[14] R. Kimball, M. Ross, The Data Warehouse Toolkit, Second edWiley Computer Publishing, 2002.
[15] M. Klemettinen, H. Mannila, H. Toivonen, Interactive exploration of interesting findings in the telecommunication network alarm sequence analyzer Tasa, Information and Software Technology 41 (1999) 557–567.
[16] W. Klosgen, Subgroup discovery, in: W. Klosgen, J.M. Zytkow (Eds.), Handbook of Data Mining and Knowledge Discovery, Oxford University Press, New York, 2002, pp. 354–361.
[17] N. Kumar, A. Gangopadhyay, G. Karabatis, S. Bapna, Z. Chen, Navigation rules for exploring large multidimensional data cubes, International Journal of Data Warehousing & Mining 2 (2006) 27–48.
[18] B. Liu, W. Hsu, Post-analysis of learned rules, presented at Thirteenth National Conference on Artificial Intelligence (AAAI-96), Portland, Oregon, USA, 1996.
[19] K. McGarry, A survey of interestingness measures for knowledge discovery, Journal of Knowledge Engineering Review 20 (2005) 39–61.
[20] B. Padmanavan, A. Tuzhilin, Knowledge refinement based on discovery of unexpected patterns, Decision Support Syetems 33 (3) (July 2002) 309–321.
[21] S. Sahar, Interestingness via what is not interesting, presented at Fifth ACM SIGKDD international conference on Knowledge discovery and data mining, 1999.
[22] S. Sarawagi, Indexing Olap data, presented at IEEE Data Engineering Bulletin, 1997.
[23] S. Sarawagi, Explaining differences in multidimensional aggregates, presented at VLDB Conference, 1999.
[24] S. Sarawagi, User-adaptive exploration of multidimensional data, presented at VLDB, 2000.
[25] S. Sarawagi, R. Agrawal, N. Megiddo, Discovery-driven exploration of Olap data cubes, presented at International Conference on Extending Database Technology, 1998.
[26] Y.D. Shen, Z. Zhang, Q. Yang, Objective-Oriented Utility-based Association Rule Mining, ICDM, 2002, pp. 426–433.
[27] A. Silberschatz, A. Tuzhilin, What makes patterns interesting in knowledge discovery systems, IEEE Transactions on Knowledge and Data Engineering 8 (6) (1996) 970–974.
[28] R. Subramanian, Motor Vehicle Traffic Crashes as a Leading Cause of Death in the United States, National Center for Statistics and Analysis, NHTSA, Washington DC, 2003.
[29] J.S. Vitter, M. Wang, Approximate computation of multidimensional aggregates of sparse data using wavelets, presented at ACM SIGMOD International Conference on Management of Data, Philadelphia, PA, 1999.
[30] K. Wang, Y. Jiang, L.V.S. Lakshmanan, Mining unexpected rules by pushing user dynamics, presented at ACM SIGKDD, 2003.
[31] N. Wu, J. Zhang, Factor analysis based anomaly detection and clustering, Decision Support Systems 42 (1) (2006) 375–389.
[32] H. Zang, B. Padmanavan, A. Tuzhilin, On the discovery of significant statistical quantitative rules, KDD 2004, 2004, pp. 374–383.
[33] N. Zibidi, S. Faiz, M. Limam, On mining summaries by objective measures of interestingness, Machine Learning 62 (3) (2006) 175–198.

Navin Kumar has a PhD in Information Systems from University of Maryland, Baltimore County. His main research interests are in data warehousing solutions and data mining.

Aryya Gangopadhyay is a Professor of Information Systems at the University of Maryland Baltimore County (UMBC). His research interests include privacy preserving data mining, OLAP data cube navigation, and core and applied research on data mining.

Sanjay Bapna is an Associate Professor of Information Science and Systems at Morgan State University. His research has appeared in *Decision Sciences*, and elsewhere. His research interests include data mining, analytics, and security.

George Karabatis is an Associate Professor of Information Systems at the University of Maryland, Baltimore County (UMBC). His current research interests are semantic information integration and mining, and applications for mobile handheld devices.

Zhiyuan Chen is an Assistant Professor at information systems department, UMBC. His research interests include privacy preserving data mining, data navigation and visualization, XML, automatic database tuning, and database compression.