

# **Navigating Multidimensional Data Using Sk-Association Rules**

Navin Kumar and Aryya Gangopadhyay

University of Maryland, Baltimore County (UMBC)

Information Systems Department, 1000 Hilltop Circle, ITE 404, Baltimore, MD 21250

{navin1, gangopad}@umbc.edu

## **Navigating Multidimensional Data Using Sk-Association Rules**

### **Abstract**

*Navigating through multidimensional data cubes is a non-trivial task. Although On-Line Analytical Processing (OLAP) provides the capability to view multidimensional data in different perspectives through roll-up, drill-down, and slicing-dicing, it offers only minimal guidance to end users in the actual knowledge discovery process. It is impractical to navigate through the enormous numbers of cuboids that usually make up data cubes, and consequently users are likely to miss out valuable information in their cube navigation.*

*In this paper, we address this problem by proposing a Discovery of Sk-Association Rules (DISAR) algorithm to drive the knowledge discovery process. First, we develop a new set of rules known as sk-association rules using a powerful test of skewness on the pairs of lattice nodes. Second, we capture the navigation paths in the data cubes by using sk-association rules. Third, we use the sk-association rules to enhance the navigation capabilities in the data cubes. Experimental results demonstrate detailed evaluation of sk-association rules.*

**Key words:** OLAP, data cube lattice, association rules, skewness

## 1 Introduction

With the ever increasing volume of data collected and archived by organizations, it has become critical to be able to efficiently and effectively navigate large, multidimensional databases. Dimensional modeling techniques offer modeling paradigms to capture measures along multiple dimensions, and On-Line Analytical Processing (OLAP) tools provide various operations such as roll-up, drill-down, and slicing-dicing to select target datasets and view them from different angles. However, it is still difficult for end users to detect the hidden patterns in the voluminous and complex lattice of multidimensional databases. Such operations have been compared to finding needles in a haystack.

In this paper, we present *Discovery of Sk-Association Rules (DISAR)*, a novel skewness based algorithm, to detect the hidden surprises in data cubes and then we use these surprises to provide a method for cube navigation. Here, a surprise reveals how anomalous a transaction is when compared with the rest of the transactions. Because the notion of a surprise is context and knowledge dependent, different users may have different thresholds for defining what constitutes a surprise. DISAR provides a notion of surprises based on the level of significance, in which the users can set the thresholds for surprises by simply adjusting the level of significance. Furthermore, our methodology allows users to browse through high-level information, and then can selectively navigate to the deeper recesses of the datasets. This provides a proper guidance while letting the users drive the knowledge discovery process.

DISAR is fundamentally different from the previous work in that it detects surprises by looking at data at the lowest level of granularity at every cuboid in a multidimensional lattice, as opposed to dealing with aggregated data. The logic behind this is that aggregation often hides the characteristics

of the detailed data: an “extremely high” value and an “extremely low” value can be aggregated to a “moderate” value, which hides both of the extreme values. Using the smallest level of granularity, we avoid the problem of hiding a surprise as an inadvertent side effect of aggregation.

To detect the surprises, we apply the statistical property of skewness on the lattice nodes. A dataset is skewed if it is not symmetrically distributed with reference to its central axis (usually the mean). If  $\mu$  and  $\sigma$  are the mean and standard deviation of a population, skewness ( $\sqrt{\beta_1}$ ) is defined by its third standardized moment as follows:

$$\sqrt{\beta_1} = \frac{E(X - \mu)^3}{[E(X - \mu)^2]^{3/2}} = \frac{E(X - \mu)^3}{\sigma^3},$$

where E is the expected value operator. For a symmetric distribution, such as a normal distribution  $N(\mu, \sigma)$ ,  $\sqrt{\beta_1} = 0$ . If the distribution is positively (negatively) skewed,  $\sqrt{\beta_1} > 0$  ( $\sqrt{\beta_1} < 0$ ), the portion of the curve on the right (left) of the central axis contains a longer tail. Also, the single most commonly used distribution in statistical analysis is the normal distribution [13], and is applied in a powerful test of skewness [12] to identify the non-normality in unknown datasets. We use this statistical test of skewness to identify skewed hidden patterns in the lattice, which positively or negatively influence the datasets.

The main contributions of this paper are as follows:

- (1) We use the statistical property of skewness to detect non-normal patterns at the lowest levels in the dimensional hierarchies, which ensures a high recall and precision in the detection of surprises.
- (2) The methodology allows users to adjust the level of significance in determining surprises. A user can navigate through the same dataset multiple times by varying the level of significance. The practical implication of this adjustment is that some of the obvious surprises may already be

known to the user, and therefore, he or she might search for more fine-grained surprises at different iterations.

- (3) Surprises for multiple measures can be discovered and displayed simultaneously. In most real-life settings, multiple measures such as *revenue* and *cost* are viewed simultaneously in order to gain useful insights [16].
- (4) As the number of dimensions, hierarchical levels, and transactional cardinality increase, even a simple presentation of the raw data can be overwhelming to the users. Sk-association rules present the surprises in a more comprehensible form devised in proper navigation paths.

The rest of the paper is organized as follows. Section 2 presents the related work followed by an overview of the data cube model in Section 3. We describe the DISAR algorithm in Section 4. The paper then proceeds with the experimental evaluation of the DISAR in Section 5. We discuss the conclusions and future work in Section 6.

## **2 Related Work**

Cube lattices are commonly used in multidimensional data mining in large databases [10]. Association rule mining, first introduced in [1], is one of the key research topics, and has been developed in different ways, such as the quantitative rules [2], RLSD [5], strong affinity patterns [8], the DGX distribution [9], and the CCMine [15]. These methods focus primarily on finding the strong associations among the items sets. Association rule, for example, establishes interesting relationships among items, but cannot determine an interesting set of transactions containing anomalies if its support or confidence is very low. On the other hand, the outlier detection techniques, such as LOADED [7], are mainly useful in identifying the extreme anomalies in the datasets, but do not provide adequate solutions to describe the overall patterns of the datasets.

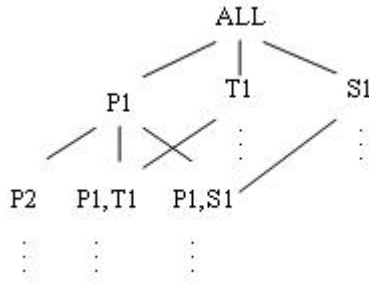
Besides, these methods do not deal with guiding the user through the navigations, which is an important issue in multidimensional databases. While query driven knowledge discovery [11] and discovery-driven exploration of OLAP data cubes [3] have been proposed, they primarily require the end users to work on the datasets to identify the surprises. The authors in [3] defined surprises in a rigid manner implying the users could not view them differently according to their needs. Furthermore, it was overwhelming for users to study the surprises by looking at the datasets presented in a large number of rows and columns. These shortcomings quickly prompt for further research in this area to develop a finer solution for cube exploration. Using DISAR, we aim to extend these existing works by detecting the surprises concealed at the lowest level of granularities in the large multidimensional databases, and then by establishing the concept of cube navigation using the discovered surprises supplemented by dynamic adjustments and novelty of surprises.

### 3 Overview of the Data Cube Model

Let us say that the data cube consists of  $m$  dimensions,  $d_1, d_2, \dots, d_m$ . A dimension  $d_i$  is usually associated with a dimensional hierarchy containing one or more levels of aggregation. Level  $l_{ij} = j^{\text{th}}$  level of dimension  $d_i$ , where  $1 \leq j \leq L_i$ .  $L_i$  is the number of levels associated with dimension  $d_i$ . A level  $l_{ij}$  contains a set of attributes. Let  $v_{ijk}$  be  $k^{\text{th}}$  attribute at  $l_{ij}$ . Let us say there are a total  $s$  facts,  $f_1, f_2, \dots, f_s$  in the fact table.  $w_{pq} = q^{\text{th}}$  value of fact  $f_p$ , where  $1 \leq p \leq s, w_{pq} \in W_p$ . Here,  $W_p$  is a set of discrete values associated with  $f_p$ , such as  $W_{[profit]} = \{\text{low, medium, high}\}$ . The fact table contains the complete transaction set,  $T$ .  $T = [\tau_1, \tau_2, \dots, \tau_n]$  where  $n$  is the number of transactions. A transaction,  $\tau$  is represented as  $\{(x_1, x_2, \dots, x_i, \dots, x_m), (f_1, f_2, \dots, f_p, \dots, f_s)\}$ , where  $x_i$ 's are attributes ( $v_{ijk}$ 's) of  $d_i$ 's, and  $f_p$ 's are associated facts.

Given  $m$  dimensions, a lattice  $latt(m)$  of cuboids can be constructed, where a cuboid is a unique combination of hierarchical levels of dimensions. Figure 1 presents the partial lattice for a grocery

database with *product* (P), *time* (T) and *store* (S) dimensions. (P1) represents a one-dimensional cuboid containing level-1 (*product category*) for *product* dimension. Similarly, (P1,T1) represents a two-dimensional cuboid formed by level-1 of *product* and *time* dimensions. An edge shows the possible navigation from one cuboid to another, for instance from (P1) to



**Figure 1. A partial lattice of cuboids**

(P1,T1). A cuboid consists of a set of nodes, where a node contains the attributes  $v_{ijk}$ 's and the facts  $f_p$ 's. For example, the cuboid (P1) contains three nodes, namely “Food, \$445K”, “Drinks, \$680K”, and “Supplies, \$380K”. Every node corresponds to a set of transactions, also referred as a dataset. A navigation path describes the traversal through the nodes in the lattice. For example, the path (P1)  $\rightarrow$  (P2)  $\rightarrow$  (P2,T1)  $\rightarrow$  (P2,T1,S1) implies that the user first looks at a node at (P1), drills down to a node at (P2), and then subsequently view nodes at (P2,T1) and (P2,T1,S1).

#### 4 The DISAR Algorithm

We propose DISAR, an algorithm to discover hidden surprises using the property of skewness. DISAR utilizes normal distribution as the base mathematical model for detecting skewed nodes existing within a lattice. To discover the surprises, we follow a four-step recursive process as follows.

- (1) Given a current node, generate a set of candidate nodes (explained in Section 4.1),
- (2) Measure the skewness for candidate nodes (explained in Section 4.2),
- (3) Apply the test of significance of skewness on candidate nodes (explained in Section 4.3), and
- (4) Transform the nodes of surprises into sk-association rules (explained in Section 4.4).

Once a node of surprise is identified, it acts as the current node for generating the next level candidates. The algorithm terminates when either the lowest level nodes in the lattice have been reached, or no more nodes exhibit significant skewness in the current iteration. The last step, navigation using the sk-association rules, is discussed in Section 4.5.

#### 4.1 Generation of Candidate Nodes

We start detection of surprises at the root node, which is a level-1 node, such as (P1), (T1), or (S1). Subsequent nodes are determined either by drilling down to a dimension from a preceding node or by including a new dimension that does not exist in the preceding node. Suppose  $nod_{curr}$  is the node where a user is currently looking at a surprise contained by the fact  $f_{p[curr]}$ .

$$nod_{curr} = \{l_{1[curr]}, l_{2[curr]}, \dots, l_{i[curr]}, \dots, l_{m[curr]}\}.$$

Here,  $l_{i[curr]}$  is the hierarchical level for dimension  $d_i$  at node  $nod_{curr}$ . A node  $nod_{cand}$  is then identified as a candidate node for  $nod_{curr}$  under the following conditions.

$$\exists l_{i[cand]} : l_{i[cand]} = l_{i[curr]} + 1 \text{ for exactly one } l_{i[cand]};$$

$$\forall i \quad l_i(nod_{cand}) \geq l_i(nod_{curr}), \text{ where } i = 1, \dots, m.$$

Let us say that  $nod_{curr}$  contains  $m_1$  dimensions ( $1 \leq m_1 \leq m$ ) at  $l_{ij} \neq 0$ . Then, we determine candidate nodes by (i) navigating one level down in the hierarchy for  $d_k$ ,  $d_k \in m_1$ , and (ii) navigating along a new dimension  $d_l$ ,  $d_l \in (m - m_1)$  dimension set.

#### 4.2 Measurement of Skewness for Candidate Nodes

Once the candidate  $nod_{cand}$  is identified, we measure the skewness of the dataset for  $nod_{cand}$  with respect to its central axis defined by the mean value of  $nod_{curr}$ . We examine the patterns at candidate nodes that are significantly influencing the current node. In other words, we identify the datasets at the candidate nodes that are skewed when measured from the current node.

Let us say  $\bar{f}_{p[curr]}$  is the mean value of fact  $f_{p[curr]}$  at  $nod_{curr}$ . Let  $f_{p(j)[cand]}$  be  $f_p$ 's value for  $j^{\text{th}}$  transaction  $\tau_{j[cand]}$ . To estimate the skewness, we first calculate the moments given by the following equation.

$$m_{i[cand]} = \frac{\sum_{j=1}^{n_2} (f_{p(j)[cand]} - \bar{f}_{p[curr]})^i}{n_2}.$$

The skewness of candidate node  $nod_{cand}$  is, then, measured as follows.

$$\begin{aligned} \sqrt{b_1[cand]} &= \frac{m_3[cand]}{m_2[cand]^{3/2}} \\ &= \frac{\sum_{j=1}^{n_2} (f_{p(j)[cand]} - \bar{f}_{p[curr]})^3 / n_2}{\left[ \frac{\sum_{j=1}^{n_2} (f_{p(j)[cand]} - \bar{f}_{p[curr]})^2}{n_2} \right]^{3/2}} \end{aligned}$$

This approach helps us in (i) recognizing the interesting patterns by using the property of skewness, and (ii) drawing the navigation paths by establishing a parent-child relationship. The candidate node  $nod_{cand}$  is labeled as the “child” node for its “parent”  $nod_{curr}$ , which provides a way to navigate from the root node to low-level nodes by using current and candidate nodes.

### 4.3 Test of Significance for Candidate Nodes

Once we have measured the skewness of candidate nodes, we apply the test of significance of skewness [13] to discover the nodes of surprises. We determine the surprises by conducting a one-sided test of skewness as follows.

$$H_0: \text{normality with } \sqrt{\beta_1} = 0$$

$$H_1: \text{non-normality with } \sqrt{\beta_1} > 0, \text{ or } H_1: \text{non-normality with } \sqrt{\beta_1} < 0$$

If  $\sqrt{\beta_1}$  satisfies the critical skewness at level of significance  $\alpha$ , the null hypothesis  $H_0$  becomes false and is rejected. Based on the sample size ( $n_2$ ) of  $nod_{cand}$ , we calculate the critical skewness as follows.

*Critical value for Skewness [ $5 \leq n_2 \leq 35$ ]:*

Skewness  $\sqrt{b_1}_{[cand]}$  for a candidate node is compared with the simulation probability points of  $\sqrt{b_1}$  called Monte Carlo points at  $\alpha$  level.

*Critical value for Skewness [ $n_2 \geq 36$ ]:*

We perform a normal approximation of the null distribution of  $\sqrt{b_1}$  characterized by a Johnson  $S_U$  curve. The obtained Z-value represents approximately a standard normal variable. This Z-value is looked up in the standard normal distribution,  $N(0,1)$ , table at  $\alpha$  level to either reject or accept the null hypothesis  $H_0$ .

*Test of Significance of Skewness:*

If  $\sqrt{b_1}_{[cand]} > \sqrt{b_1}^{(critical)}_{[cand]}$ , we reject the null hypothesis, and therefore, skewness  $\sqrt{b_1}_{[cand]}$  of candidate node  $nod_{cand}$  is found significant at  $\alpha$  level. If  $\sqrt{b_1}_{[cand]} \leq \sqrt{b_1}^{(critical)}_{[cand]}$ , the null hypothesis cannot be rejected which implies that the candidate dataset is normally distributed with no significant skewness in  $\bar{f}_{p[curr]}$ .

#### **4.4 Transformation of Significant Nodes into Sk-Association Rules**

A candidate  $nod_{cand}$ , containing a significant skewed pattern, is transformed into an sk-association rule. The individual dimensions, their hierarchical levels, and the attributes from  $nod_{cand}$  are assigned to the antecedent. Consequent consists of the fact and its skewness. For example, a positive skewness for profit infers that  $nod_{cand}$  is in a high profit region. An sk-association rule a.k.a. *skr* is then represented as follows.

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

Here  $\alpha$  and  $\sqrt{b_1}$  are the respective level of significance and skewness. For example, if  $nod_{cand}$  ('Drinks', \$680K) is positively skewed at  $\alpha = 0.05$  and  $\sqrt{b_1} = 2.63$ , then the corresponding sk-association rule will be “*product category = Drinks*  $\rightarrow$  *profit = sk-high [0.05, 2.63]*.”

The support, denoted by  $v$  for an sk-association rule, is defined as the number of transactions to establish the rule, and is a useful parameter to control the number of rules generated by DISAR. In some instances, using a very small sample size may produce a large number of rules, in which case  $v$  is used to restrict the rule generation by qualifying only those nodes, which satisfy the condition  $n_2 \geq v$ . One point to be noted here is that sk-association rules are neither association rules found in data mining, nor production rules found in expert systems. Association rules will not serve the purpose of discovering surprises as they detect the most commonly found patterns as opposed to uncommon patterns. On the other hand, sk-association rules discover both the common and uncommon patterns hidden in the datasets.

#### **4.5 Navigation using the Sk-Association Rules**

The generation of an sk-association rule implies a *node of surprise* is discovered. A user begins with the root node (level-1 sk-association rules) and continues drilling down to children nodes until no further rules are detected. Thus the rules guide the users to handily reach the surprises of their interest. A navigation path comprises of a set of sk-association rules visited during the traversal. The longest navigation path, recognized by its total number of rules =  $(\sum L_i - 1)$ , describe the most detailed path to reach the lowest levels of nodes of surprises.

To further enhance the navigational capabilities, we provide the users with the capability to control the navigations according to their needs, in three different ways as follows.

- (a) *Level of significance ( $\alpha$ )*: One of the key aspects of DISAR is to be able to change  $\alpha$  dynamically to look at different sets of sk-association rules. The critical skewness increases with

a reduction in  $\alpha$ , and vice-versa. The  $\alpha$ -value is set to 0.05 by default, and is provided as a user-defined parameter. A user, willing to look at only highly extreme surprises, simply needs to decrease the  $\alpha$ -value to 0.01 or 0.005. Similarly, the  $\alpha$ -value can be relaxed to 0.1 or even 0.2 to look at less extreme surprises.

(b) *Dealing with multiple facts*: Given a particular node, the multiple facts can be studied simultaneously by looking at different consequents while keeping the same antecedent. For example, two sk-association rules,  $skr_1$  “ $year=1993 \rightarrow profit=sk-high [0.05, 1.79]$ ” and  $skr_2$  “ $year=1993 \rightarrow cost=sk-low [0.05, -2.06]$ ”, represent the same node but contain two different facts *profit* and *cost*.

(c) *Categorization of sk-association rules*: We categorize sk-association rules into three groups: ‘expected’, ‘unexpected’, and ‘NA’ for enhanced navigation strategies. A rule is *expected* if it shares the same consequent with the parent. The consequent of an *unexpected* sk-association rule differs from that of its parent. An sk-association rule at the root node, having no parents, is assigned ‘NA’. For example,  $skr_1$  “ $Quarter = 1993-Q2 \rightarrow profit = low [0.05, -1.36]$ ” is an unexpected surprise when navigating from  $skr_2$  “ $year = 1993 \rightarrow profit = high [0.05, 1.79]$ ”. This categorization adds purposeful information to the navigation paths. The users can simply drill down on the paths that contain a higher number of unexpected sk-association rules, and examine the corresponding datasets that contain many highs and lows in the transactions.

## 5 Experimental Results

In this section, we present a set of experiments to evaluate the performance of DISAR algorithm by studying the quality of sk-association rules and the scalability. Specifically, we demonstrate: (a) the recall and precision of sk-association rules, (b) execution time, and (c) the space overhead.

### 5.1 Experimental Setup

We adapted the Grocery database [14] to produce five test datasets as shown in Table 1. The primary reason for using synthetic test data in these experiments is to know precisely what the hidden surprises are and then compare them with discovered sk-association rules.

<b>Dataset</b>	<b># records</b>	<b># nodes of navigation</b>
DS-1	4,480	5,893
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

**Table 1. Summary of the five experimental datasets**

Each of the datasets contained three dimensions (*product*, *time*, and *store*), and two measures (profit and quantity). The dimensional hierarchies were *product* (category, subcategory, brand), *time* (year, quarter, month), and *store* (region, state, city). The number of attributes at the lowest level was varied from 16 to 64 for *product*, 20 to 192 for *time*, and 14 to 31 for *store* dimension. A surprise represented a transaction containing 15-20% high or low profit compared to rest of the transaction set. The surprises were introduced in a manner so as to conceal their presence at the higher levels of aggregations. For example, in DS-5, all eight years had their profit values close to \$17.8 million, but at the lowest levels only four years contained the genuine surprises. To further study the effect of varying size of dataset on DISAR, we replicated exactly the same surprises in DS-1, DS-2, and DS-3. We tested the DISAR algorithm by varying the number of surprises in the datasets. All experiments were performed on a 1.7GHz Pentium IV machine with 512MB RAM running Windows XP. The algorithms were implemented in Oracle PL/SQL. The values for  $\alpha$  and  $\nu$  were set to 0.05 and 5 throughout the experiments.

## 5.2 Quality of Sk-Association Rules

	$\alpha = 0.05, v=5$								
	<b>Recall</b>					<b>Precision</b>			
# of surprises	DS-1	DS-2	DS-3	DS-4	DS-5	DS-1, DS-2	DS-3	DS-4	DS-5
1	0.960	0.960	0.960	0.984	0.984	1.000	0.96	1.000	0.726
2	0.939	0.939	0.939	0.961	0.987	1.000	0.738	0.986	0.962
5	0.919	0.919	0.919	0.968	0.977	1.000	1.000	1.000	0.995
10	0.897	0.897	0.897	0.969	0.984	1.000	1.000	1.000	0.960
15	0.880	0.884	0.890	0.960	0.960	1.000	1.000	0.997	0.993
20	0.899	0.899	0.899	0.927	0.956	1.000	1.000	1.000	0.986
25	0.862	0.866	0.866	0.950	0.997	1.000	0.984	1.000	0.989

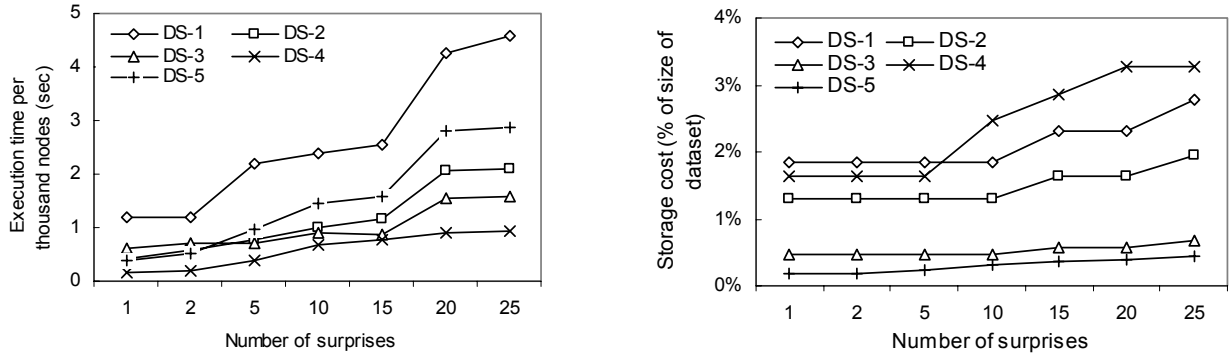
**Table 2. The recall and precision of discovered sk-association rules**

Table 2 shows the recall and precision for varying number of surprises. The purpose of this experiment is to examine how many true surprises (recall) are discovered from the total hidden surprises. The precision reflects the proportion of true surprises in the set of discovered rules. We observe that the DISAR detected the surprises for all five datasets with very high recall and precision values. DS-1, DS-2, and DS-3 had nearly the same recall because they contained the same set of surprises. The recall increased with larger datasets because they generated a higher number of candidate nodes qualifying as true surprises. However, it also decreased the precision because some false surprises were also detected. For DS-1 and DS-2, no false surprises were detected. For DS-5, a total 327 sk-association rules were discovered at number of surprises = 10 in which 314 rules were true surprises, resulting in a precision of 96%.

### 5.3 Execution Time

Figure 2(a) shows an expected increase in execution time with a higher number of surprises but within a reasonable processing time. For example, in dataset DS-3, it took only 1.587 seconds per thousand nodes (total 2.83 minutes) to discover a total 255 nodes of surprises when the number of surprises was set to 25. These 255 nodes were selected from a total of 107,095 nodes of navigation (Table 1). For the same dataset, it took only 0.868 seconds when the numbers of surprises were set to

15. A major benefit of the DISAR algorithm is that it discovers the surprises by starting from level-1. This way a lattice node is pruned at a higher level itself if it did not contain any surprises, hence resulting in an efficient discovery process.



**Figure 2. Scalability in terms of (a) Execution time, (b) Storage cost**

### 5.4 Space Overhead

The sk-association rules are stored in the database using a relational schema as suggested in [4]. Figure 2(b) shows the storage cost needed for sk-association rules, as a percentage of the size of dataset. Although the storage cost expectedly increases with an increase in the number of surprises, it ranges only from 0.18% to 3.28%. It reached a maximum 3.28% when relatively higher numbers of nodes of surprises (513 nodes) were discovered for dataset DS-4 at number of surprises = 25. Therefore, although we have discovered a highly meaningful sk-association rule set, the space overhead to store this rule set is very small. By already establishing a parent-child relationship among the rules, DISAR avoids any post-processing of sk-association rules and does not require any extra storage for the navigation paths.

## 6 Conclusions and Future Work

In this paper, we presented the DISAR algorithm to discover the hidden surprises in the data cube lattices. DISAR produced good results on different grocery datasets with high recall and precision

values. We also established the concept of navigation to traverse through the nodes of surprises. The users could also adjust the level of significance to only view the surprises of their interest.

We are extending our work on the concept of navigation to deal with the scenarios where possibly a large number of rules are generated. Specifically, we are working on an axis shift theory, which ranks the navigation paths based on the distance metrics of sk-association rules, and then provides a comparison matrix to study the usefulness of various navigation paths. We are also working on a rule-driven system to provide the navigational capabilities to end users by using sk-association rules.

## 7 References

- [1] R. Agrawal, T. Imielinski, and A. Swami, "Mining associations between sets of items in large databases", In *Proc. of the ACM SIGMOD*, 1993, pp. 207–216.
- [2] Y. Aumann and Y. Lindell, "A statistical theory for quantitative association rules", In *ACM SIGKDD*, 1999, pp. 261-270.
- [3] S. Sarawagi, R. Agrawal, and N. Megiddo, "Discovery-driven exploration of OLAP data cubes", In *International Conference on Extending Database Technology*, 1998, pp. 168-182.
- [4] N.Kumar, A. Gangopadhyay and G. Karabatis, "Supporting mobile decision making with association rules and multi-layered caching", *Decision Support Systems*, 2005.
- [5] J. Zhang, E. Bloedorn, L. Rosen and D. Venese, "Learning rules from highly unbalanced data sets", In *ICDM*, 2004, pp. 571-574.
- [6] G. Cormode, and S. Muthukrishnan, "Summarizing and mining skewed data streams", In *SDM*, 2005.
- [7] A. Ghoting, M. Otey, S. Parthasarathy, "LOADED: Link-based outlier and anomaly detection in evolving data sets", In *ICDM*, 2004, pp. 387-390.
- [8] Hui Xiong, Pang-Ning Tan, and Vipin Kumar, "Mining Strong Affinity Association Patterns in Data Sets with Skewed Support Distribution", In *ICDM*, 2003
- [9] Z. Bi, C. Faloutsos, and F. Korn, "The dgx distribution for mining massive, skewed data", In *ACM SIGKDD*, 2001.
- [10] A. Casali, R. Cicchetti, and L. Lakhal. "Cube Lattices: a Framework for Multidimensional Data Mining". In *SDM*, 2003, pp. 304-308.
- [11] J. Boulicaut, P. Marcel, and C. Rigotti, "Query driven knowledge discovery in multidimensional data", In *ACM DOLAP*, 1999, pp. 87-93.
- [12] R. D'Agostino, A. Belanger, and R. D'Agostino Jr., "A suggestion for using powerful and informative tests of normality", *The American Statistician*, 44, 1990, pp. 316-322.
- [13] R. D'Agostino, and M. Stephens, *Goodness-of-fit techniques*, New York:M.Dekker, 1986.
- [14] R. Kimball, and M. Ross, *The data warehouse toolkit*, second ed (2002).
- [15] W. Y. Kim, Y. K. Lee, and J. Han, "CCMine: Efficient mining of confidence-closed correlated patterns", In *PAKDD*, 2004.
- [16] Connelly R. and Mosimann R., *The Multidimensional Organization*, Cognos Incorporated, 1999.