OMARS: The Framework of an Online Multi-Dimensional Association Rules Mining System

Wen-Yang Lin
Dept. of Information Management
I-Shou University
Kaohsiung 84008, Taiwan
wylin@isu.edu.tw

Ja-Hwung Su
Institute of Information Engineering
I-Shou University
Kaohsiung 84008, Taiwan
bb0820@ms22.hinet.net

Ming-Cheng Tseng
Institute of Information Engineering
I-Shou University
Kaohsiung 84008, Taiwan
clark.tseng@msa.hinet.net

Abstract

Recently, the integration of data warehouses and data mining has been recognized as the primary platform for facilitating knowledge discovery. Effective data mining from data warehouses, however, needs exploratory data analysis. The users often need to investigate the warehousing data from various perspectives and analyze them at different levels of abstraction. To this end, comprehensive information processing and data analysis have to be systematically constructed surrounding data warehouses, and an on-line mining environment should be provided. In this paper, we propose a system framework to facilitate on-line association rules mining, called OMARS, which is based on the idea of integrating OLAP service and our proposed OLAM cubes and auxiliary cubes. According to the concept of OLAM cubes, we define the OLAM lattice framework that exploit arbitrary hierarchies of dimensions to model all possible OLAM data cubes.

1. Introduction

Over the past few years, data warehouse has been recognized and widely adopted as a platform for storing integrated, detailed, summarized, and historical data. Nowadays, most data warehousing systems are designed to accomplish On-Line Analytical Processing (OLAP), the process of creating and managing multidimensional data for analysis. But with the rapid growth of various requisitions, the OLAP-like tools that provide aggregation charts or tables to visualize data cannot supply further exploration of requisition from different levels of business management. This heralds the development of data mining techniques to discover implicit knowledge from a huge collection of data. One of the predominant techniques in data mining community is association rule mining, which refers to the discovery of associations from a list of database transactions. An example association rule is that "80% of customers who buy product 'A' also buy product 'B'". Such rules reveal customers' buying behavior and can be used in various decision strategies, such as catalog design, cross-sale, customer classification and product layout.

In real world applications, data are featured in multidimensional attributes. The decision makers usually want to analyze the data from various aspects to inspire new insight to achieve competitive advantage. To date, though many researchers have proposed various methods to discover associations from data warehouses [1][2][3][4][6][10][11][12], many problems remain unsolved. Some of them are

1) Nowadays, most data warehouses adopt the star schema to organize the stored data, which allows the users to explore the data from diverse aspects. Current association mining methods, however, fail to exploit all possible patterns hidden in the star schema. This would restrain an expert from discovering new insight.
2) It is well-known that the techniques of data preprocessing and view materialization widely used in query processing and OLAP analysis also could favor on-line re-mining task. For example, the work in [3][4][12] suggested using the OLAP data cube to employ on-line data mining such as association, sequential patterns, classification, etc. However, an inappropriate cube definition cannot fit the on-line mining environment. First, the OLAP-like aggregated cube cannot exploit all of the item relationships such as intra-dimensional or hybrid associations. Mining associations from OLAP cubes still requires a lot of computations. Second, the OLAP cube based paradigm is unfeasible when users want to change the constraints such as support threshold and employ a sequence of re-mining tasks. Each re-mining requires executing the whole procedure to generate frequent itemsets.

To this end, we propose a system framework for on-line association rules mining, called **OMARS** (**O**nline **M**ultidimensional **A**ssociation **R**ules mining **S**ystem), which adopts the idea of pre-computing and materializing cube like OLAP, and integrates the OLAP cubes and our proposed OLAM cubes and auxiliary cubes to realize on-line association mining environment.

The remaining of this paper is organized as follows. In Section 2, we introduce some background material. The previous work related to our research is described in Section 3. Section 4 explains our OMARS framework and details each constituent. Finally, we conclude with suggestions and future work in Section 5.

2. Background

2.1 Data Warehouse and Data Model

The term 'data warehouse' was first coined by W. H. Inmom as a "*subject-oriented, integrated, time-variant and nonvolatile collection of data in support of management's decision-making process*" [7]. Although the data stored in data warehouses have been cleaned, filtered, and integrated, it still takes much time to translate the data into the strategic information because of massive amounts of data. To improve the performance, most data warehouses adopt some preprocessing tools like OLAP service. The term 'OLAP' is the acronym of 'On-Line Analytical Preprocessing', which refers to the process of creating and managing multidimensional data for analysis and visualization. Typical OLAP operations include *roll-up* (increasing the level of abstraction), *drill-down* (decreasing the level of abstraction), *slice and dice* (selection and projection) and *pivot* (re-orienting the multidimensional view of data) [5].

In the world of decisions making, managers tend to view the analyzed data from different business perspectives. As a primary repository for decision support, the data warehouse has to model the data multi-dimensionally. The most popular dimensional modeling of data warehouse is *star schema* [8]. A star schema consists of a central table, called *fact table*, and several *dimension table*s. The fact table consists of numeric measures and some foreign keys. Each dimension table contains a set of attributes that represent the user's business perspectives and are often organized in the form of a hierarchical relationship. Figure 1 shows an example star schema for sales data.

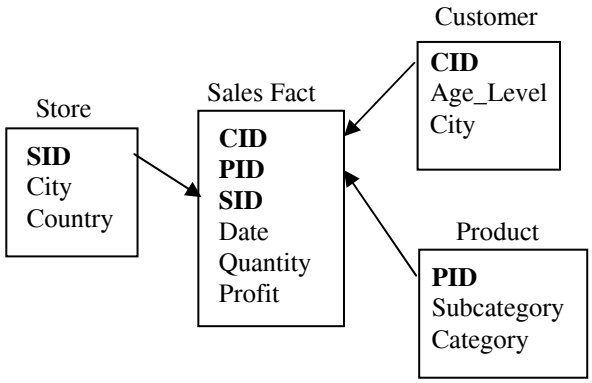

Figure 1. An example star schema for sales data.

2.2 Association Rules Mining

One of the predominant techniques used in data mining is association rule. An association rule, $A \rightarrow B$, denotes the relations between itemsets A and B, which means itemset B tends to appear together with itemset A. For this rule being interesting, the support of A and B, i.e., the number of transactions containing both A and B, denoted as $sup(A \cup B)$, should be more than a user-specified minimum support, called *minsup* ($sup(A \cup B) \geqq minsup$), and the confidence, measured as the conditional probability $P(B|A)$, should also be more than a user-specified minimum confidence, called *minconf* ($P(B|A) \geqq minconf$).

In general, the work for mining association rules can be decomposed into two phases:

1) Frequent itemsets generation: find all itemsets whose supports sufficiently exceed the *minsup*.
2) Rules construction: from the discovered frequent itemsets, generate the association rules. More precisely, for each frequent itemset X, construct the rule $(X-Y) \rightarrow Y$, if $P(Y|X-Y) \geq minconf$, where Y is is a proper subset of X.

2.3 Multi-Dimensional Association Rules

The association mining is originally motivated by discovering frequently purchased patterns from a list of supermarket transactions. Traditionally, only one attribute, e.g., Product, is concerned in the association mining.

In real life, data often involves several tables and a table usually contains multiple attributes. For example, a data warehouse fact table may involves multiple dimension tables with multiple hierarchies. When mining from this kind of multidimensional data sets, users may want to observe the inter relations among the values of dimensions, i.e., a multi-dimensional view of the associations. We call such kind of association rules the *multidimensional association rule*s. In [12], Han, Chee, and Chiang defined three different types of association rules involved multidimensional attributes.

1) *Inter-dimensional association rule*: This kind of rules reveals the associations between a set of dimensions.
2) *Intra-dimensional association rule*: This kind of rules exploits the associations between different items within only one dimension.
3) *Hybrid association rule*: This kind of rules also exploits the associations between a set of dimensions, but allows some items in a rule are from one dimension. It can be regarded as a combination of inter-dimensional and intra-dimensional rules.

According to the above description, intra-dimensional association rules, indeed, correspond to single-dimensional association rules, which can be defined as follows:

Definition 1. Consider a transaction table composed of k dimensions $\{d_1, d_2, \ldots, d_k\}$. An intra-dimensional association rule can be expressed as

$$(d_i, \text{“}x_1\text{”}), (d_i, \text{“}x_2\text{”}), \ldots, (d_i, \text{“}x_m\text{”}) \rightarrow (d_i, \text{“}y_1\text{”}), (d_i, \text{“}y_2\text{”}), \ldots, (d_i, \text{“}y_n\text{”}),$$

where x_j and y_j represent the values of *dimension* d_i, and $1 \leq i \leq k$, $1 \leq m \leq j$, $1 \leq n \leq j$.

For example, a intra-dimensional association rule

(Product, “diet coke”) → (Product, “pretzels”)

says that most of customers who buy product “diet coke” also buy “pretzels” at the same transaction time. We thus say this kind of rules reveals the intra-relation among

products in the same dimension "Product".

Following the concept in [12], we can define the multi-dimensional association rules, inter-dimensional or hybrid, as follows:

Definition 2. Consider a transaction table composed of k dimensions. Let x_{im} and y_{jn} be the values of dimensions X_i and Y_j, respectively. The form of a multi-dimensional association rule is:

$$(X_1, \text{“}x_{1m}\text{”}), (X_2, \text{“}x_{2m}\text{”}), \ldots, (X_i, \text{“}x_{im}\text{”}) \rightarrow (Y_1, \text{“}y_{1n}\text{”}), (Y_2, \text{“}y_{2n}\text{”}), \ldots, (Y_j, \text{“}y_{jn}\text{”})$$

For example, an inter-dimensional rule

$$(\text{City, “Taipei”}), (\text{Gender, “Female”}) \rightarrow (\text{Product, “pretzels”})$$

says that female customers living in "Taipei" tend to buy "pretzels". And a hybrid association rule

$$(\text{City, “Taipei”}), (\text{Gender, “Female”}), (\text{Product, “juice”}) \rightarrow (\text{Product, “pretzels”})$$

says that female customers living in "Taipei" that buy "juice" tend to buy "pretzels".

3. Related Work

Without any pre-computation, the authors in [2] proposed an algorithm, called Carma, to realize on-line computation of frequent itemsets using only two database scans. In the course of the first database scan, the algorithm continuously constructs a lattice of potential frequent itemsets, displays the result, and allows users to adjust the threshold at any time. At the second scan, it determines the precise support of each set of lattice and then removes all infrequent itemsets. We observed that

1) The performance of Carma is better than Apriori or DIC when the support threshold is much small. However, it cannot defeat Apriori or DIC when the support threshold is larger than a critical value.
2) The performance of Carma is worse than on-line mining with preprocessing technique, because the whole mining procedure is carried out repeatedly. If the data is so large that it cannot be handled in real-time, the performance may be much worse.
3) It ignores the fact that in real world most data have diverse characteristics. The users may want to explore the associations from multi-dimensional viewpoints.

Aggarwal and Yu [1] considered the on-line rules generation and provided a lattice-based approach, called *adjacency lattice*, to pre-fine and pre-store the primary itemsets. They analyzed the on-line queries and proposed several adjacency lattice based algorithms. Users working with their on-line mining framework are free to launch different queries at different thresholds. Their approach, however, may suffer the following problems:

1) If the adjacency lattice is complex and large, the preprocessing time for constructing the lattice will become unacceptable.
2) It is difficult to trade off the amount of pre-stored data against the query time.

Recently, a promising direction for realizing on-line data mining has been proposed. The basic idea is to combine OLAP and data mining, called *OLAP Mining* [3][4][6][12]. However, we observe the following problems about this approach.

1) In most cases, inter association rules and hybrid association rules appear simultaneously in multi-dimensional OLAP mining. For example, if we group 'Customer' into transactions and choose 'Product' and 'Store' as mining attributes, the rules may be:

   Store("Mexico") → Product("Berry")

   or

   Store("Mexico"), Product("tents") → Product("Berry").

   Thus, it is hard to differentiate hybrid association rule from inter association rule in multi-dimensional association mining.
2) Because of lacking support for on-line re-mining, one has to invoke the whole mining procedure, and so cannot discover the frequent itemsets in real-time when the data cube is very large.

In [11] the authors proposed the concept of using materialized itemsets for fast mining of association rules. The approach divides the database into a set of non-overlapping partitions according to some attributes, e.g., education type of customers, store location, product category, and generates the frequent itemsets in each partition according to the local threshold. Then, the positive borders corresponding to the frequent itemsets in each partition are computed. Finally, all positive borders are combined to re-mine the new frequent itemsets with supports greater than the global threshold.

The FARM framework presented in [10] addressed the problem of mining associations from multi-attribute databases. Their framework particularly concentrates on exploring the mining spaces without specifying what attributes are used to group records into transactions and what attributes are used to define items. The method for counting support, however, is different from the existential aggregating functions. They also proposed three different types of downward closures and used them to implement an efficient Apriori-like mining algorithm. The FARM framework, however, ignores the following issues:

1) Lacking interaction: Users cannot perform the queries with different thresholds.
2) Costly for virtuous refinement: Like OLAP mining, FARM needs to discover the frequent itemsets by repeating the whole mining process whenever the threshold is increased or decreased.

4. The OMARS Framework

In this section, we describe the proposed framework for integrating data warehouse, OLAP processing and our OLAM cubes and auxiliary cubes to build an on-line multidimensional association mining environment.

Through this system, the users can interactively change their viewpoints, refine the constraints according to the observed result, and perform further exploration.

4.1 Panorama

Figure 2 depicts the OMARS framework, which is deployed to meet the following characteristics of on-line multidimensional association mining.

1) Users can perform OLAP-like explorations. That is, users can interactively change the dimensions that comprise the associations.
2) Users can refine the constraints such as minimum support and minimum confidences, and see the response in no time.

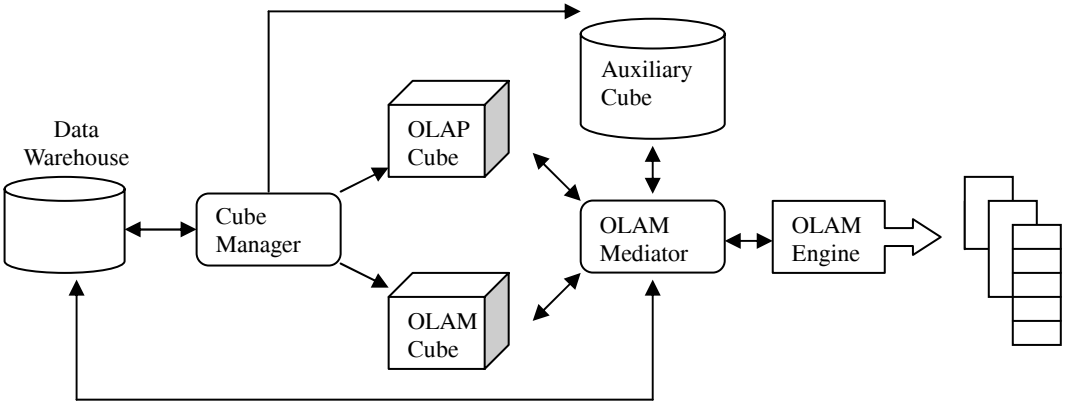

Figure 2. The OMARS framework.

To this end, we propose the concept of OLAM data cube, which stores the frequent itemsets with respect to various groupings of dimension attributes. We also make use of the OLAP service supported by most data warehousing systems, and integrate the OLAP cube and our OLAM cube to meet simultaneously these two different analyses. Our intention is to develop an add-on service instead of a stand-alone system to commercial data warehousing products. The task of on-line association mining is employed through the following two phases:

1) Off-line preprocessing phase: This phase concerns the construction of the data cubes. The data in data warehouses are preprocessed and stored in different repositories, including OLAP Cube, OLAM Cube and Auxiliary Cube. The Cube Manager component is in charge of the cube construction and maintenance, either regularly or irregularly.
2) On-line mining phase: Once the itemsets with supports above a presetting threshold, *prims*, in the preprocessing phase are materialized, the OLAM Engine forwards the user query to the OLAM Mediator, which then searches for the most appropriate cube to answer the association query. However, if the threshold of the query is lower than the presetting threshold specified in the preprocessing phase, the OLAM Engine will make use of the auxiliary cubes and re-scan the database to answer the query.

The critical design issues for these two phases are:

1) Off-line preprocessing phase: how to define the presetting minimum support, and how to construct the OLAM cube and the auxiliary cubes that store the frequent itemsets with supports greater than *prims* and the infrequent itemsets with supports less than *prims*, respectively.
2) On-line mining phase: how to re-use the existential information stored in OLAM, OLAP and auxiliary cubes to facilitate the on-line association mining from various perspectives.

The first phase involves Data Warehouse, Cube Manager, OLAP Cube, OLAM Cube and Auxiliary Cube, while the second phase involves OLAM Engine, OLAM Mediator, OLAP Cube, OLAM Cube and Auxiliary Cube. In the following sections, we will elaborate each constituent of the framework.

4.2 Cube Manager

As stated previously, Cube Manager is responsible for cube generation and maintenance. More precisely, the work of Cube Manager consists of three different parts.

1) Cube selection: This part concerns the problem of how to select the most appropriate set of data cubes for materialization in order to minimize the query cost and/or maintenance cost under the constraint of limited storage.
2) Cube computation: This part deals with the work for efficiently generating the set of materialized cubes selected by the cube selection module.
3) Cube maintenance: This deals with the task of how to maintain the materialized cubes when there are updates to the data warehouse.

4.3 OLAM Mediator and OLAM Engine

The primary task of OLAM Engine is to generate the association rules. It accepts various queries from users and invokes the most appropriate algorithm according to the dimensional attributes and the *minsup* of the query. After receiving a query, the OLAM Engine first analyzes the query and forwards the necessary information to OLAM Mediator, and then waits for the most relevant cube from OLAM Mediator to generate the qualified association rules.

On the other hand, when OLAM Mediator receives the messages about the user query from OLAM Engine, it will look for the most matching cube. First, OLAM Mediator judges whether the input *minsup* is greater than *prims* or not. If the input *minsup* is smaller than *prims*, it has to coordinate and communicate with Auxiliary Cube and data warehouse, and return the matching jointed table to OLAM Engine. Otherwise, it will check that whether the required cube exists in OLAM Cube or not. If not, it will search for OLAP Cube and return the matching cube to OLAM Engine. In the worst case, when in OLAM Cube and OLAP Cube no cube matches the query, it will re-scan the data warehouse and notify OLAM Engine to execute the whole mining procedure afresh.

Figure 3 depicts the cooperation between OLAM Mediator and OLAM Engine.

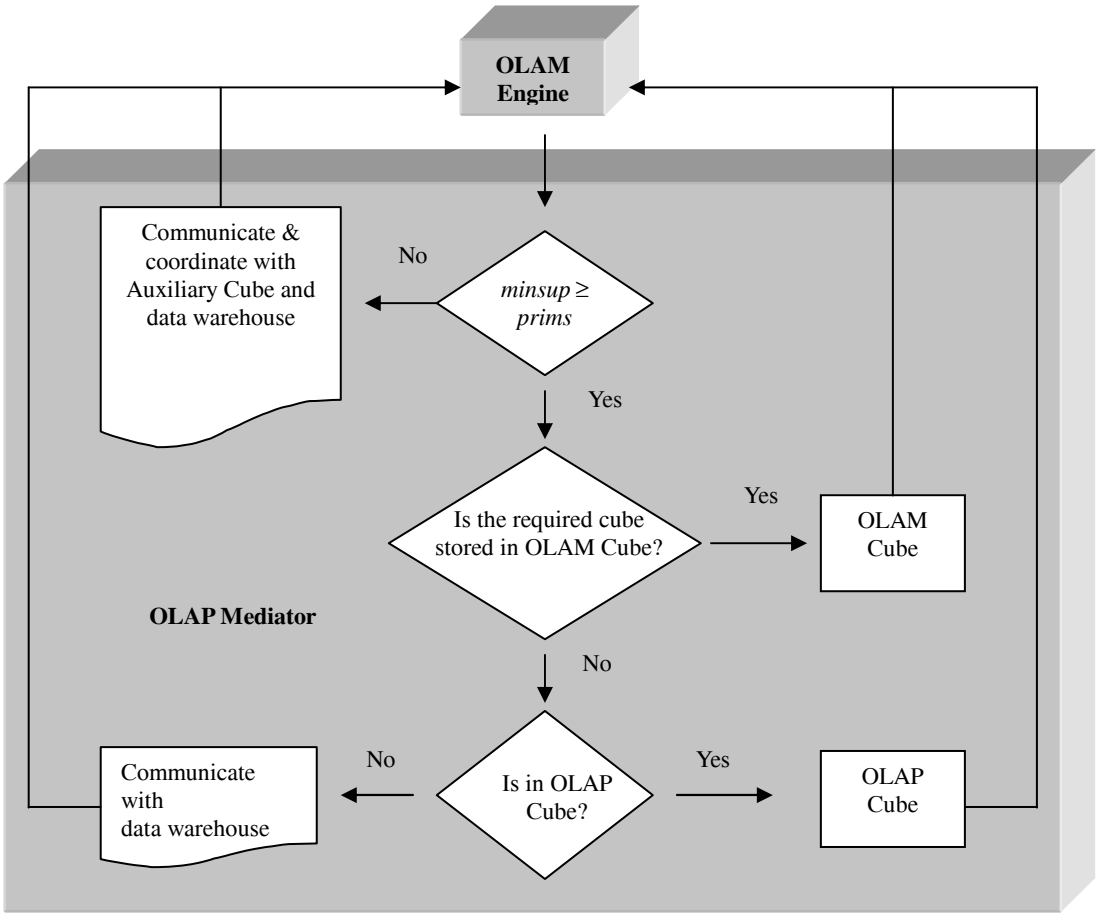

Figure 3. Flow diagram of OLAM Mediator and OLAM Engine.

4.4 OLAM Cube and OLAM Lattice

The concept of OLAM data cube is used to store the frequent itemsets with supports greater than the pre-setting support threshold *prims*, in the off-line preprocessing phase. The intuition is to accelerate the process of association mining by utilizing these materialized frequent itemsets. To clarify the structure of OLAM cubes, we first specify the form of multidimensional association query, which is defined as a four-tuple mining meta-pattern.

Definition 3. Consider a star schema S containing a fact table and m dimension tables $\{D_1, D_2, ..., D_m\}$. Let T be a jointed table from S composed of $a_1, a_2, ..., a_r$ attributes, such that $\forall a_i, a_j \in A(D_k)$ there is no hierarchical relation between a_i and a_j, and $1 \le i, j \le r$, $1 \le k \le m$. A *meta-pattern* of multidimensional associations from T is defined as follows:

$$MP: < t_G, t_M, ms, mc >,$$

where *ms* denotes the minimum support, *mc* the minimum confidence, t_G the group of transaction attributes, t_M the

group of mining attributes, t_G, $t_M \subseteq \{a_1, a_2, \ldots, a_r\}$ and $t_G \cap t_M = \varnothing$.

Note that this meta-form specification of multi-dimensional association queries can present three different multidimensional association rules defined in [12], intra-dimensional, inter-dimensional, and hybrid associations.

For example, consider a jointed table T involving three dimensions from the star schema in Figure 1. The content of T is shown in Table 1. If the mining attribute set t_M consists of only one attribute, then the discovered rules present the intra-association. For example, if t_G = {Cid, Date}, t_M = {Category}, we may have

(Category, "B"), (Category, "D") → (Category, "E")

Otherwise, if $|t_M| \geq 2$, then the resulting associations will be inter-association or hybrid association, e.g., t_G = {Cid, Date}, t_M = {Category, Age_level, C.City}, we have

(Age_level, "21-30"), (C.City, "Taipei") → (Category, "B"),

which is an inter-association, and

(Age_level, "31-40"), (C.City, "New York"), (Category, "C") → (Category, "D"),

a hybrid association.

Table 1. A jointed table *T* from star schema.

Tid	Cid	Date	Category	Age_level	City
1	C1	2001/01/12	B, C, D, E	21-30	Taipei
2	C1	2001/01/23	A, B, C	21-30	Taipei
3	C2	2001/02/01	B, C, D	31-40	New York
4	C4	2001/03/16	A, D	below 20	Toronto
5	C3	2001/03/16	A, B, D	41-50	Seattle
6	C2	2001/08/09	C, D, E	31-40	New York
7	C4	2001/08/09	D	below 20	Toronto
8	C3	2001/09/25	B, C, E	41-50	Seattle
9	C1	2001/09/26	B, D, E	21-30	Taipei
10	C2	2001/10/12	B, C, D, E	31-40	New York

* A: Magazines, B: Electrical, C: Hardware, D: Drinks, E: Paper Products

We now proceed to clarify the OLAM cube structure.

Definition 4. For a meta-pattern *MP* with transaction attributes t_G and mining attributes t_M, and a presetting *minsup*, *prims*, the correspondent OLAM cube, *MCube*(t_G, t_M), is the set of the frequent itemsets with supports larger than *prims*.

Example 1. A hybrid OLAM cube: Let $|t_M| = 3$, t_G = {Cid, Date}, t_M = {Category, Age_level, City}, and *prims* = 3. The resulting OLAM cube is shown in Table 2.

Note that in an OLAM cube, an attribute value is viewed as an item and a tuple containing *k*-items represents a frequent *k*-itemset. For example, the last tuple represents a frequent 4-itemset, where the "Category" dimension of the tuple contains several values, such as "C" and "D". An association rule derived from the last tuple will be

(C.City, "New York"), (Age_level, "31-40") → (Category, "C"), (Category , "D")

This rule reveals that the customers whose age is between 31 and 40 and who live in New York tend to buy product category "C" and "D".

Table 2. An example hybrid OLAM cube expressed in table.

Age_level	City	Category	Support
-	-	A	3
-	-	B	7
-	-	C	6
-	-	D	8
-	-	E	5
21-30	-	-	3
31-40	-	-	3
-	Taipei	-	3
-	New York	-	3
21-30	-	B	3
31-40	-	C	3
31-40	-	D	3
-	Taipei	B	3
-	New York	C	3
-	New York	D	3
21-30	Taipei	-	3
31-40	New York	-	3
21-30	Taipei	B	3
31-40	New York	C	3
31-40	New York	D	3
31-40	New York	C, D	3

Example 2. An intra-dimensional OLAM cube: Let $|t_M| = 1$, t_G = {Cid, Date}, t_M = {Category}, and *prims* = 4. The resulting OLAM cube is shown in Table 3.

Table 3. An example intra-dimensional OLAM cube expressed in table.

category	Support
B	7
C	6
D	8
E	5
B, C	5
B, D	5
B, E	4
C, D	4
C, E	4

In this example, the attributes "Cid" and "Date" are grouped into transaction attributes and the "Category" is viewed as the mining attribute. There are 10 transactions in this table. Again, a tuple containing *k*-items represents a frequent *k*-itemset. From the last tuple that represents a

frequent 2-itemset, we can derive the following association rule:

(Category, "C") → (Category, "E").

This rule says that those customers who buy category "C" tend to buy category "E".

To allow the users mining associations from various perspectives, we have to exploit all possible OLAM cubes. Indeed, according to the definition of OLAM cube, all of the possible OLAM cubes generated from a given star schema can form an OLAM lattice. To provide a hierarchical navigation and multidimensional exploration, we model the OLAM lattice as a three-layer structure. The 1st layer lattice exploits all of the dimensional combinations. The 2nd layer then further exploits the inter-attribute combinations for the selected dimensions in the 1st layer. Finally, the 3rd layer exploits all of the possible OLAM cubes corresponding to the meta-pattern that can be derived from the chosen subcubes in the 2nd layer. Note that the first two layers only serve the purpose for hierarchical navigation and dimensional exploration. The real OLAM cubes are stored in the 3rd layer.

For example, consider the star schema shown in Figure 1. We obtain eight possible dimensional combinations and construct the 1st layer lattice as shown in Figure 4. Each node in the 1st layer lattice represents a possible dimensional combination.

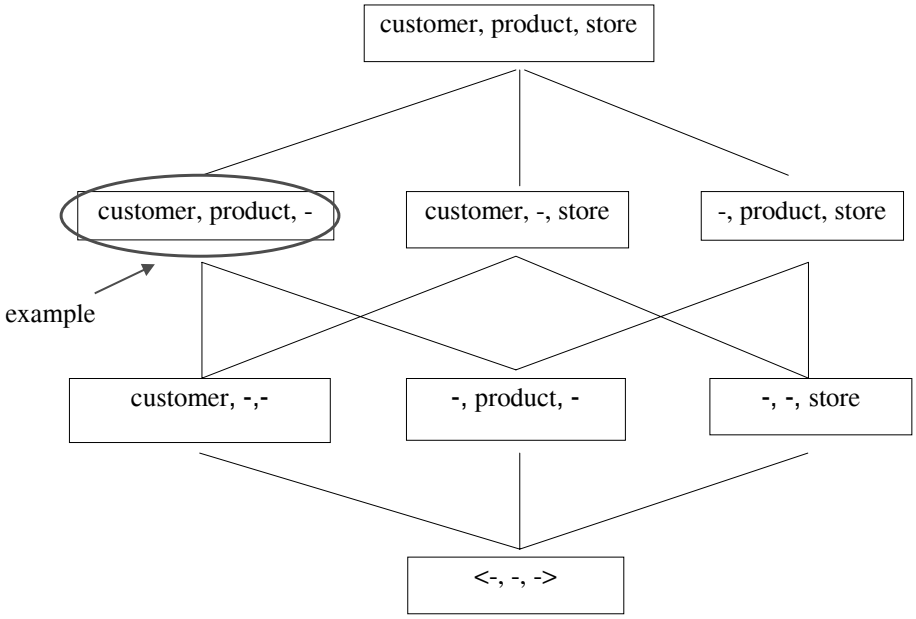

Figure 4. The 1st layer OLAM lattice for the example star schema in Figure 1.

Any node in the 1st layer lattice then can be extended to form a 2nd layer lattice. For example, consider the node composed of "customer" and "product" dimensions. Figure 5 shows the 2nd layer lattice derived from this node. Each node of the 2nd layer lattice is constructed by attaching any attribute chosen from the selected dimensions and the attributes are added one by one as traversing down the lattice. The traversing stops when all attributes of the selected dimensions appear in this bottom node.

Similarly, any node of the 2nd layer lattice can be further extended to form a 3rd layer lattice. Consider the <(City, Age_level), (Category)> node in Figure 5. Figure 6 shows the 3rd lattice by extending <(City, Age_level), (Category)>. There are eight different nodes in this lattice, where each node represents an OLAM cube. As Figure 6 shows, the nodes can be divided into two different categories:

1) Intra-association cubes: These refer to at the 3rd level the nodes that contain only one mining attribute.
2) Inter- or hybrid association cubes: These include at the 1st level the nodes that contain no transaction attribute and at the 2nd level those that contain one transaction attribute.

Note that not all of the OLAM cubes derived in the lattice have to be materialized and stored. Indeed, the concept hierarchies defined over the attributes in the star schema provide the possibility to prune some redundant cubes.

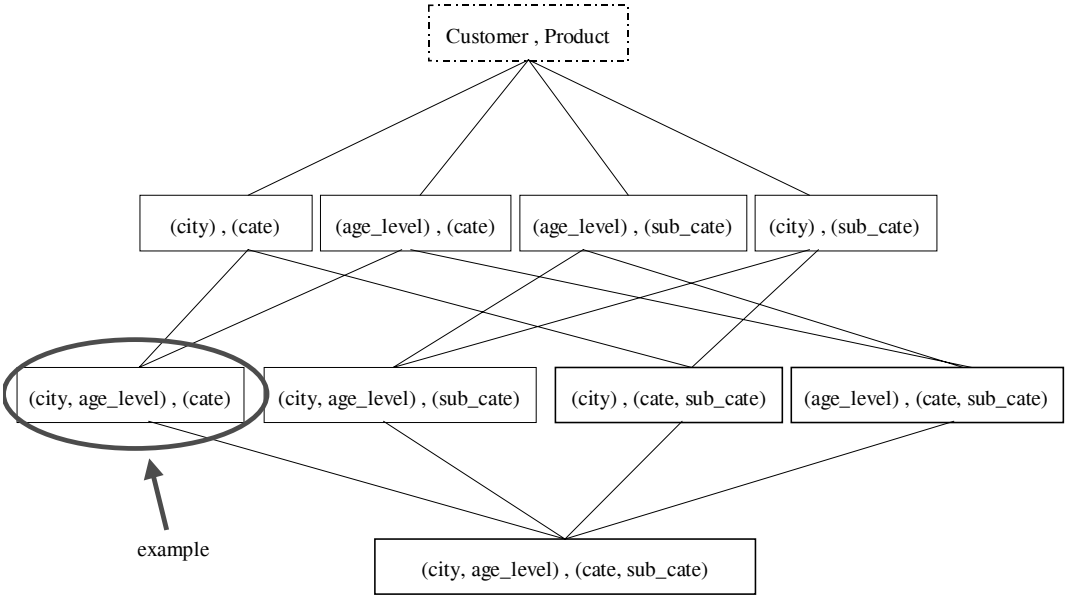

Figure 5. The 2nd layer lattice derived from <customer, product, -> in the 1st layer.

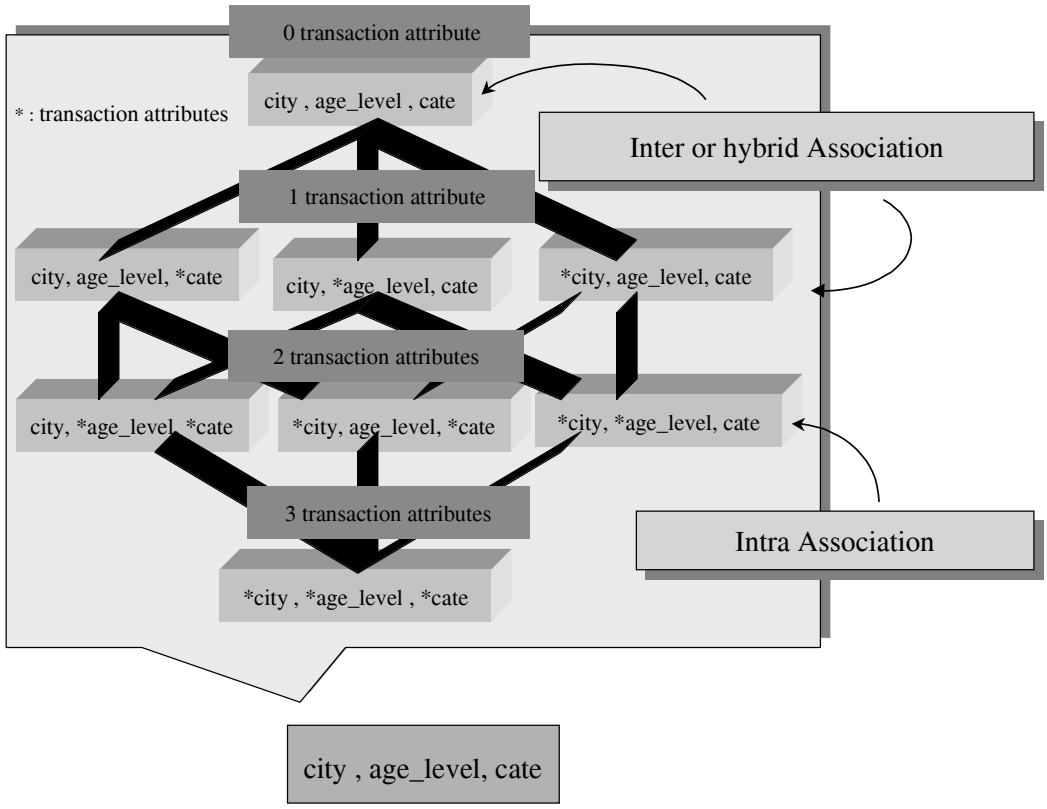

Figure 6. The 3rd layer lattice derived from the subcube <city, age_level> in the 2nd layer.

Consider an OLAM cube, $MCube(t_G, t_M)$. We observed that there are two different types of redundancy.

Observation 1. Schema redundancy: Let $a_i, a_j \in t_G$. If a_i, a_j are in the same dimension and a_j is an ancestor of a_i, then $MCube(t_G, t_M)$ is a redundancy of cube $MCube(t_G - \{a_j\}, t_M)$.

Example 3. Consider the jointed table in Table 4. Let $|t_M| = 1$ and t_M = {Subcategory}. The resulting table by grouping "City" and "Country" as transaction attributes is shown in Table 5. Note that this table is the same as that by grouping "City" as the transaction attribute, as shown in Table 6. Thus, the resulting cube $MCube(\{City, Country\},$

{Subcategory}) is the same as *MCube*({City}, {Subcategory}).

Table 4. A jointed table from star schema.

Cid	City	Country	Category	Subcategory
C1	Tokyo	Japan	Personal Computer	Notebook
C1	Tokyo	Japan	Printer	Ink_Jet
C2	NY	USA	Personal Computer	Notebook
C2	NY	USA	Printer	Ink_Jet
C3	HK	China	Personal Computer	Desktop PC
C4	Toronto	Canada	Storage Hardware	Hard Disk
C5	Toronto	Canada	Storage Hardware	Hard Disk

Table 5. The resulting table by grouping {City, Country} as transaction attributes for Table 4.

City	Country	Subcategory
Tokyo	Japan	Notebook, Ink_Jet
New York	USA	Notebook, Ink_Jet
Hong Kong	China	Desktop PC
Toronto	Canada	Hard Disk

Table 6. The resulting table by grouping {City} as transaction attribute for Table 4.

City	Subcategory
Tokyo	Notebook, Ink_Jet
New York	Notebook, Ink_Jet
Hong Kong	Desktop PC
Toronto	Hard Disk

Table 7. The resulting OLAM cube *MCube*({Cid}, {Category, Subcategory}).

Category	Subcategory	Support
PC	-	3
Printer	-	2
Storage Hardware	-	2
-	Notebook	2
-	Hard Disk	2
-	Ink_Jet	2
Printer	Notebook	2
Printer	Ink_Jet	2
PC	Ink_Jet	2
PC	Notebook	2
Storage Hardware	Hard Disk	2
-	Notebook, Ink_Jet	2
PC, Printer	-	2
PC, Printer	Ink_Jet	2
PC, Printer	Notebook	2
PC	Notebook, Ink_Jet	2
Printer	Notebook, Ink_Jet	2
PC, Printer	Notebook, Ink_Jet	2

Observation 2. Values Redundancy: Let $a_i, a_j \in t_M$. If a_i, a_j are in the same dimension and a_j is an ancestor of a_i, then $MCube(t_G, t_M)$ is a cube with values redundancy.

Example 4. Consider the jointed table in Table 4. Let $|t_M| = 2$, t_G = {Cid}, t_M = {Category, Subcategory} and *prims* = 2. The resulting OLAM cube is shown in Table 7. We observe that the tuples with gray area in this table are redundant patterns. Therefore, it satisfies the values redundancy. Note that if it holds the values redundancy, we must prune the redundant patterns during the generation of frequent itemsets.

In addition, we observe that any OLAM cube is useless if it satisfies the following property.

Observation 3. Useless Property: Let $a_i \in t_G$ and $a_j \in t_M$. If a_i, a_j are in the same dimension and a_j is an ancestor of a_i, then $MCube(t_G, t_M)$ is a useless cube.

Example 5. Let $|t_M| = 1$ and t_M = {Category}. The resulting table by grouping {Subcategory} as transactions is shown in Table 8. The cardinality of every transaction in this table is 1. Therefore, we cannot find any association rule from this table.

Table 8. The resulting table by grouping {Subcategory} as transaction attribute for Table 4.

Subcategory	Category
Notebook	Personal Computer
Ink_Jet	Printer
Desktop PC	Personal Computer
Hard Disk	Storage Hardware

4.5 Auxiliary Cube

Though the OLAM cube is useful to generate associations for *minsup* larger than the pre-setting threshold, it becomes useless when the *minsup* is less than the threshold. In that case, we have to perform the mining task afresh, which probably will not satisfy the on-line demand.

To alleviate this problem, we propose the concept of auxiliary cube, which is used to store infrequent α-itemsets generated in the off-line preprocessing phase, where α denotes the *cutting-level*. For example, consider Table 1 and let $\alpha = 3$ and *prims* = 4. Table 9 shows the auxiliary cube with respect to the intra-dimensional association, where "Cid" and "Date" are grouped as transaction attributes while "Category" is regarded as mining attribute. All tuples containing three items present infrequent 3-itemsets.

Though auxiliary cube can speed up on-line mining when *minsup* ≤ *prims*, it consumes a lot of cube scanning time to locate the qualified itemsets when the auxiliary cube is very large. To prevent this problem, we partition the auxiliary cube into several sub-cubes to reduce the I/O cost. The structure of the partitioned auxiliary cube is shown in Figure 7. Assume *prims* = 0.5%. We partition the auxiliary cube into three sub-cubes, with respect to three different ranges of thresholds, "0.5%-0.2%", "0.2%-0.1%" and "0.1%-0.0%". Next, we judge which sub-cube can answer the query. Because most of the infrequent itemsets appear

in "0.1%-0.0%" sub-cube, the I/O cost of scanning the other sub-cubes would be much low. For example, if the input threshold is between 0.5% and 0.2%, it just needs to scan the "0.5%-0.2%" sub-cube instead of scanning the whole auxiliary cube.

Table 9. An example of auxiliary cube for intra-dimensional association.

Itemset	Support
A, B, C	1
A, B, D	1
B, C, D	3
B, C, E	3
B, D, E	3
C, D, E	3

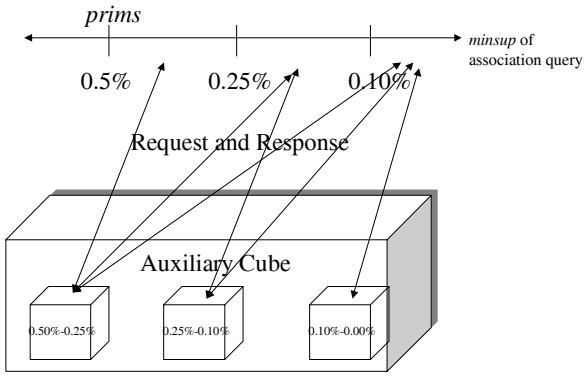

Figure 7. An example partition of auxiliary cube.

5. Conclusions

In this paper, we have proposed a system framework for on-line association rules mining, called OMARS. The goal of this framework is to provide an on-line mining environment to facilitate multidimensional exploration of association rules from data warehouses. To reach this goal, we adopted the concept of preprocessing, and proposed the concept of OLAM cubes and auxiliary cubes to store the frequent itemsets over a presetting *minsup* and the infrequent itemsets. We also proposed a three-layer OLAM lattice to organize all of the possible OLAM cubes in a systematic way. Through this three-layer lattice, users can carry out OLAP-like multidimensional exploration of association rules.

This paper present a preliminary result of our project on building a real on-line multidimensional association mining system. There remains much work, theoretical or implemental, to accomplish to realize the proposed ORMAS on-line mining system.

Acknowledgement

This work was supported in part by the National Science Council of ROC with grant No. NSC90-2213-E-214-040.

References

[1] C.C. Aggarwal and P.S. Yu, "Online generation of association rules," *IEEE Transactions on Knowledge and Data Engineering*, pp. 402-411, 1998.

[2] C. Hidber, "Online association rule mining," in *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, pp. 145-156, 1999.

[3] J. Han, "OLAP Mining: An integration of OLAP with data mining," in *Proceedings of the 7th IFIP 2.6 Working Conference on Database Semantics* (*DS-7*), pp. 1-9, 1997

[4] J. Han, S. Chee, and J. Chiang, "Issues for on-line analytical mining of data warehouse," in *Proceedings of the 1998 ACM SIGMOD, Workshop on Research Issues on Data Mining and Knowledge Discovery*, 1998.

[5] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2000.

[6] J. Han, "Towards on-line analytical mining in large databases," *SIGMOD Record*, Vol. 27, No. 1, 1998.

[7] W.H. Inmon and C. Kelley (1993) *Rdb/VMS: Developing the Data Warehouse*, QED Publishing Group, Boston, Massachussetts.

[8] R. Kimball, *The Data Warehouse Toolkit Practical For Building Dimensional Data Warehouses*, JOHN WILEY & SONS, 1996.

[9] G. Psaila and P.L. Lanzi, "Hierarchy-based Mining of Association Rules in Data Warehouse," in *Proceedings of ACM'00*, pp. 307-312, 2000.

[10] C. Perng, H. Wang, S. Ma and J.L. Hellerstein, " FARM: A framework for exploring mining spaces with multiple attributes," in *Proceedings of IEEE International Conference on Data Mining*, pp. 449-456, 2001.

[11] M. Wojciechowski and M. Zakrzewicz, "Itemset Materializing for Fast Mining of Association Rules," in Proceedings of Second East European Symposium on Advances in Databases and Information Systems (ADBIS'98), Poznan, Poland, 1998, Lecture Notes in Computer Science 1475, Springer 1998.

[12] H. Zhu, *On-Line Analytical Mining of Association Rules*, Master Thesis, Simon Fraser University, December 1998.