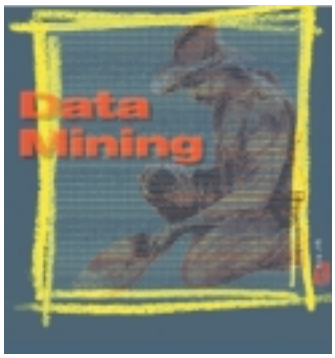
Cover Feature

Constraint-Based, Multidimensional Data Mining



Integrating both constraint-based and multidimensional mining into one framework provides an interactive, exploratory environment for effective and efficient data analysis and mining.

Jiawei Han
Simon Fraser University

Laks V.S. Lakshmanan
Concordia University and Indian Institute of Technology, Bombay

Raymond T. Ng
University of British Colombia

Although there have been many data-mining methodologies and systems developed in recent years, we contend that by and large, present mining models lack human involvement, particularly in the form of guidance and user control. We believe that data mining is most effective when the computer does what it does best—like searching large databases or counting—and users do what they do best, like specifying the current mining session's focus. This division of labor is best achieved through *constraint-based mining*, in which the user provides restraints that guide a search.[1] Mining can also be improved by employing a multidimensional, hierarchical view of the data. Current data warehouse systems have provided a fertile ground for systematic development of this *multidimensional mining*.[2] Together, constraint-based and multidimensional techniques can provide a more ad hoc, query-driven process that effectively exploits the semantics of data than those supported by current stand-alone data-mining systems.

AD HOC AND QUERY DRIVEN

An ad hoc and query-driven data-mining system can be more effective because it better fits queries to the user's intentions. It can make the process of inferring knowledge more efficient by letting a query optimizer deliver high-performance, interactive mining systems that encourage exploratory mining and analysis.

Such a data-mining system incorporates two capabilities, which also distinguish it from a statistical-analysis program or a machine-learning system.[3] First, it should offer an ad hoc mining query language, which is a high-level declarative language comparable to the Structured Query Language (SQL) for relational database management systems. Such a declarative mining language lets users express

- the part of the database to be mined (called the minable view[1]),
- the type of pattern/rule to be mined, and
- the properties that the patterns should satisfy.

These patterns should include not only numerical constraints on statistical properties (like support, confidence, and correlation), but also those based on attribute domains, classes, and aggregates,[1] such as "I.type = 'snacks' and avg(I.price) < 100."

Second, a data-mining system should support efficient processing and optimization of mining queries by providing a sophisticated mining-query optimizer. Such an optimizer exploits the various constraints stated in the user-specified mining query and their properties to generate access plans that guarantee a level of performance commensurate with the constraints in the query.

CONSTRAINTS: ESSENTIALS FOR AD HOC DATA MINING

We divide constraints into five categories:

- *Knowledge type constraints* specify the type of knowledge to be mined, such as concept description, association, classification, prediction, clustering, or anomaly. This constraint, unlike other constraints, is usually specified at the beginning of a query because different types of knowledge can require different constraints at later stages.
- *Data constraints* specify the set of data relevant to the mining task. We often specify such constraints in a form similar to that of an SQL query and process them in query processing.
- *Dimension/level constraints* confine the dimension(s) or level(s) of data to be examined in a database or a data warehouse. Such constraints follow the model of a multidimensional database and demonstrate the spirit of multidimensional mining. Thus, multidimensional mining can be smoothly incorporated within the framework of constraint-based mining.

0018-9162/99/$10.00 © 1999 IEEE

- *Rule constraints* specify concrete constraints on the rules to be mined.
- *Interestingness constraints* specify what ranges of a measure associated with discovered patterns are useful or interesting from a statistical point of view.

The following example illustrates these five constraints at work. Suppose there is a sales multidimensional database with four interrelated relations

- sales (customer_name, item_name, transaction_id),
- lives (customer_name, district, city),
- item (item_name, category, price), and
- transaction (transaction_id, day, month, year),

where lives, item, and transaction are three dimension tables. These tables are linked to the sales table via three keys: customer_name, item_name, and transaction_id.

"Find the sales of what cheap items (with the sum of the prices less than $100) that may promote the sales of what expensive items (with the minimum price of $500) in the same category for Vancouver customers in 1998" is an association mining query. It is expressed in a data mining query language (DMQL1) as shown in Figure 1a.

This mining query may allow the generation of association rules like those shown in Figure 1b.

The rules mean that if a customer in Vancouver bought Census_CD and MS Office 97, there is a 68 percent probability that he will also buy MS SQL Server. The rule further indicates that 1.5 percent of all the customers fulfilled all the criteria.

In this query, the knowledge type constraint is association. The data constraint is lives(C, _ ,"Vancouver"). The dimensions are related to all three dimensions: *lives*, *item*, and *transaction* because the query involves all of them.

The levels are more confined. For lives, we only consider customer_name since city = "Vancouver" is used only in the selection; for item, we consider the levels item_name and category since they are used in the query; and for transaction, we consider only transaction_id since day and month are not referenced and year is used only in the selection. Rule constraints include most portions of the where and having clauses, such as S.year = 1998, T.year = 1998, I.category = J.category, sum(I.price) $100, and min(J.price) ≥ 500. Finally, there are two interestingness constraints (thresholds), min_support = 0.01 and min_confidence = 0.5.

Knowledge type constraints and data constraints can be applied before data mining. That is, they are not intertwined with the mining process itself. After applying these constraints, a mining process may first mine all of the possible rules before applying the remaining three categories of constraints and then filter out the rules that do not satisfy such constraints. However, this may yield an inefficient and sometimes prohibitively expensive mining process. Therefore, it is critical to analyze these constraints for properties of interest. We want to push constraints deeper inside the mining process—eliminate irrelevant item sets earlier—and minimize the number of item sets to be examined.

```
mine associations as
  lives(C, _, "Vancouver") and
    sales+(C, ?{I}, {S}) ⇒ sales+(C, ?{J}, {T})
  from sales
where S.year = 1998 and T.year = 1998 and
  I.category = J.category
group by C, I.category
having sum(I.price) < 100 and
  min(J.price) ≥ 500
with min_support = 0.01 and
  min_confidence = 0.5
```

(a)

```
lives(C, _, "Vancouver") and
  sales(C, "Census_CD", ) and
    sales(C, "MS/Office97",_) ⇒ sales(C,
      "MS/SQLServer",_) [0.015; 0.68]
```

(b)

Figure 1. An association mining query expressed in data mining query language.

HANDLING ASSOCIATION RULES

Association rules are a popular type of rule. A rule constraint comes in various forms. It can be in the form of a predicate in a rule, specifying set/subset relationship, constant initiation of variables, or aggregate functions.[1] In the previous example, the constraints S.year = 1998, T.year = 1998, I.category = J.category, sum(I.price) < 100, and min(J.price) > 500 are rule constraints.

An essential question in constraint-based mining is "What kind of rule constraints can be pushed into the mining process while still ensuring complete answers to a mining query?"

In classical association rule mining, the standard Apriori algorithm[4] exploits an interesting property for finding frequent item sets: Whenever the support of a set *S* of items violates the frequency constraint (that is, its support falls below a specified threshold), then all *S*'s supersets must also violate the frequency constraint. This is called the *antimonotonicity property*.[1] Other than the frequency constraint, there are many rule constraints that also satisfy the antimonotonicity property.

Antimonotone constraints

A rule constraint such as sum(I:price) $ 100 is antimonotone because any item set that already has a sum

It is interesting to examine whether a rule constraint can be pushed deeply into the hierarchy to facilitate mining not only at the current level of abstraction but also at deeper levels.

over 100 cannot become part of the group. Adding more items to the item set will simply make it more expensive, and it will never satisfy the constraint.

Similarly, min(J:price) ≥ 500 is antimonotone in the sense that any item set that violates this constraint should be tossed away since adding additional items will never make it satisfy the constraint.

The constraint S.year = 1998 is also antimonotone since any item with S.year π 1998 will not satisfy the constraint.

Such constraints can be pushed deeply into the mining process because if they are not satisfiable at an early level of the mining process, they have no hope of becoming satisfiable later (as more items are added to the item set).

In contrast, a constraint like avg(I:price) $100, is not antimonotone because the addition of more items could let the item set satisfy the constraint. Such a constraint cannot be pushed inside the mining process while guaranteeing the completeness of the answers to a query.

Succinct constraints

Pruning induced by antimonotone constraints occurs once each iteration of an Apriori-style algorithm. Another property of constraints, succinctness,[1] can also provide an effective pruning method.

The constraint min(J:price) $500 is succinct because we can explicitly and precisely generate all the sets of items satisfying the constraint, without recourse to a generate-everything-and-test approach. Specifically, such a set must contain at least one item whose price is less than $500. Because there is a precise "formula" to generate all the sets satisfying a succinct constraint, there is no need to iteratively check the constraint in the mining process.

In contrast, the constraint avg(I:price) $100 is not succinct because intuitively the average is inherently dependent on all items in a set and this cannot be reduced to a simple selection on individual items—a subset—of the entire item set.

Dimension/level constraints

As mentioned before, data warehouses and multidimensional databases are semantically organized into multiple dimensions and levels. Mining knowledge at a particular combination of dimensions and levels makes it easier to apply to the corresponding abstraction space. Moreover, applying dimension/level constraints in data mining could greatly reduce the search space.

Different users could be interested in associations among the items at different abstraction levels. For example, some may be interested in sales among categories of items, such as soft drinks and chips, whereas some others may be interested in more details, such as sales between Coke and Sunset potato chips. Some mining approaches first find what categories of items are likely to be sold together and then drill down along the item dimension to find particular items in these categories that are sold together. Efficient algorithms have been developed to facilitate level-shared mining.[5,6] The property "if a high-level item is infrequent, none of its descendants can be frequent" indicates that some antimonotone properties exist across levels and can be explored for level-shared mining.

Finally, we examine how to use dimension/level constraints and rule constraints together in mining association rules. As discussed earlier

- dimension constraints help carve out the task-relevant dimension space,
- level constraints determine what combinations of the abstraction space to work on, and
- rule constraints confine the forms of the rules to be generated.

These constraints can be enforced together. It is interesting to examine whether a rule constraint can be pushed deeply into the hierarchy to facilitate mining not only at the current level of abstraction but also at deeper levels.

Returning to our original example, let's suppose we would like to find association rules at multiple levels of abstraction with the same set of constraints. If the frequency constraint maintains the same threshold at different levels, antimonotonicity at a high level of abstraction will still be valid since deeper level items can be viewed like a high-level item with respect to determining antimonotonicity and succinctness. For example, the constraint sum(I:price) $100 is antimonotone at multiple levels of abstraction, as we've explained before. That is, the final rules obtained may associate items at the category level or item_name level. In either case, we can push the same set of constraints that satisfy antimonotonicity and succinctness deeply into the association mining process.

IMPLEMENTATION

Using these ideas, we are developing a data-mining system, which the "Online Analytical Mining Architecture" sidebar describes.

By classifying rule constraints into antimonotone and/or succinct, we have developed an algorithm called CAP (for Constrained Apriori) that delivers a level of performance commensurate with the selection abilities of the constraints. Earlier, we reported experimental results that show CAP to be significantly faster than the straightforward algorithm, which does not push constraints deeper into the mining process.[1]

Online Analytical Mining Architecture

Our Online Analytical Mining (OLAM) architecture, shown in Figure A,[3] facilitates constraint-based, multidimensional mining of large databases and data warehouses. The architecture consists of four layers; the lowest layer is the *data repository layer*, which consists of the supporting databases and data warehouses. On top of it is the *multidimensional database (MDDB) layer*, which provides a multidimensional view of data for online analytical processing and mining. The essential layer for data mining is the *OLAP/OLAM layer*, which consists of two engines, one for online analytical processing and one for mining. Finally, on top of the OLAP/OLAM layer lies the *user interface layer*, which provides easy-to-use interfaces. These interfaces let users construct data warehouses, create multidimensional databases, select the desired sets of data, perform constraint-based interactive OLAP and mining, and visualize and explore the results.

The OLAM architecture provides a modularized and systematic design for a data mining system and provides several benefits. First, it takes advantage of widely available, comprehensive information-processing infrastructure. These systems have been systematically constructed around relational database management systems and data warehouses, which can store huge amounts of data. Data cleaning, integration, and consolidation have been largely performed in the construction of data warehouses. An efficient OLAM architecture should use existing infrastructure in this way rather than constructing everything from scratch.

Another benefit of the OLAM architecture is that it provides an OLAP-based exploratory data analysis environment. The integration of database and data warehouse at one end and online analytical processing and mining at the other facilitates two things. First, it becomes possible to mine different subsets of data and at different levels of abstraction by drilling, pivoting, filtering, slicing, and dicing a multidimensional database and the intermediate data-mining results.

Secondly, it facilitates online, interactive selection of data-mining functions and interestingness thresholds. Performing these functions interactively and viewing the results with data/knowledge visualization tools will greatly enhance the power and flexibility of exploratory data mining.

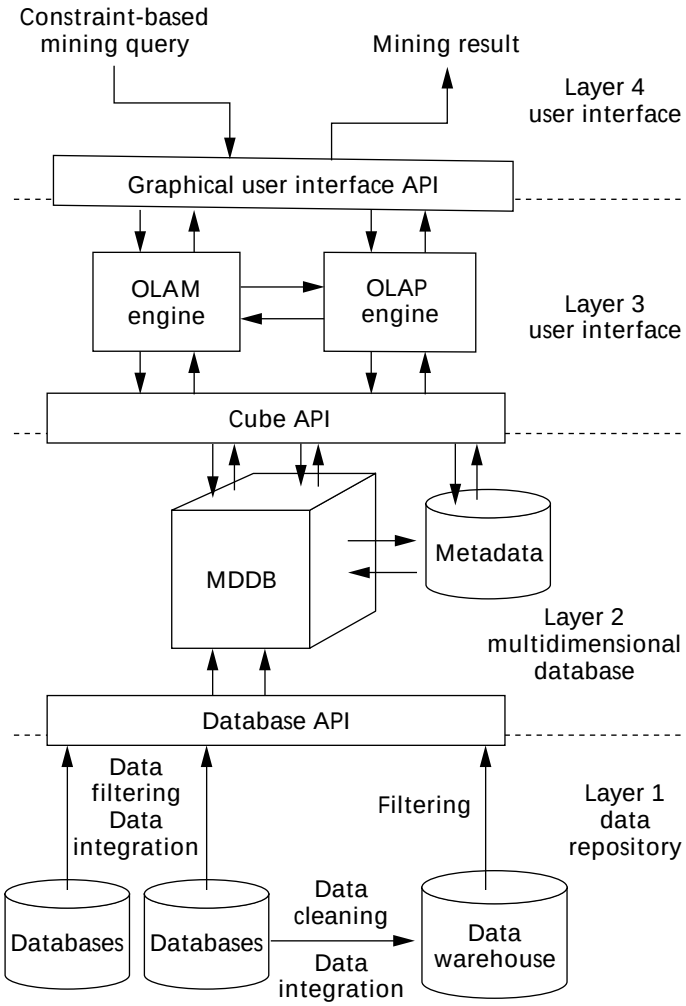


***Figure A. Online analytical mining (OLAM) architecture. Reprinted wih permission from* Communications of the ACM.**

CAP has been embedded in a version of the OLAM engine shown in the sidebar. In that engine, association rule mining, and the mining of other related forms of rules, can be carried out with many user interaction points.[8] This design philosophy allows the user to guide the mining and to authorize expensive computation, which usually yields higher quality results but is too costly to apply in an unfocused, unguided fashion. Multidimensional, constraint-based association mining has also been incorporated in the DBMiner system,[7] in which Associator is one of several major data-mining modules in the system.

Many unsolved problems remain in constraint-based, multidimensional association mining. These include how to integrate constraint-based mining methods based on different measures of interest; how to systematically design and implement a constraint-based data-mining query language; and how to perform incremental mining by relaxing

The notion of a high-level declarative query language, query processing, and query optimization have played a pivotal role in the evolution and maturation of database technology.

certain constraints. More research is needed to develop an integrated, constraint-based association-mining environment. We would also like to know how to apply constraint-based, multidimensional mining to other types of knowledge, such as characterization, classification, clustering, and anomaly analysis. These are unexplored but very promising areas for future research.

The notion of a high-level declarative query language, query processing, and query optimization have played a pivotal role in the evolution and maturation of database technology, leading to the development of flexible and high-performance database systems.[9] It is our vision that the next generation of data mining systems will successfully integrate traditional DBMS capabilities with mining capabilities.[10] Such systems will offer a single, seamless framework that combines traditional database management—which is ad hoc, query driven, multidimensional, and exploratory—with an efficient data analysis and mining tool.

In this context, we see a central role for constraint-based multidimensional mining. Constraints can serve as a unifying paradigm for the various traditional constraints as well as those associated with dimension hierarchies in multidimensional databases. Constraints not only enrich the semantics of mined rules, enabling users to have a say in what they want, but they also offer opportunities to develop powerful analysis/mining query optimizations. ❖

References

1. R. Ng et al., "Exploratory Mining and Pruning Optimizations of Constrained Associations Rules," *Proc. 1998 ACM SIGMOD Int'l Conf. Management of Data*, ACM Press, New York, 1998, pp. 13-24.
2. S. Chaudhuri and U. Dayal, "An Overview of Data Warehousing and OLAP Technology," *ACM SIGMOD Record*, **Month?** 1997, pp. 65-74.
3. T. Imielinski and H. Mannila, "A Database Perspective on Knowledge Discovery," *Comm. ACM*, Nov. 1996, pp. 58-64.
4. R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," *Proc. 1994 Int'l Conf. Very Large Data Bases*, Morgan Kaufmann, San Francisco, Calif., 1994, pp. 487-499.
5. J. Han and Y. Fu, "Discovery of Multiple-Level Association Rules from Large Databases," *Proc. 1995 Int'l Conf. Very Large Data Bases*, Morgan Kaufmann, San Francisco, Calif., 1995, pp. 420-431.
6. R. Srikant and R. Agrawal, "Mining Generalized Association Rules," *Proc. 1995 Int'l Conf. Very Large Data Bases*, Morgan Kaufmann, San Francisco, Calif., 1995, pp. 407-419.
7. J. Han, "Towards On-Line Analytical Mining in Large Databases," *ACM SIGMOD Record*, Mar. 1998, pp.97-107.
8. R. Ng et al., "Exploratory Mining Via Constrained Frequent Set Queries (Demo)" *Proc. 1999 ACM SIGMOD Conf. on Management of Data*, ACM Press, New York, 1999, pp. 556-558.
9. A. Silberschatz, H. F. Korth, and S. Sudarshan, *Database System Concepts, 3rd ed.*, McGraw-Hill, Hightstown, N.J., 1997.
10. S. Sarawagi, S. Thomas, and R. Agrawal, "Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications " *Proc. 1998 ACM SIGMOD Int'l Conf. Management of Data*, ACM Press, New York, 1998, pp. 343-354.

Jiawei Han is a professor of computing science at Simon Fraser University, B.C., Canada. His research interests include data mining and data warehousing, and spatial and Web data mining. Han has a PhD in computer science from the University of Wisconsin at Madison. He is a member of the IEEE Computer Society, the ACM and ACM SIGMOD and ACM SIGKDD.

Laks V.S. Lakshmanan is an associate professor of computer science at Concordia University, Montreal, and the India Institute of Technology, Bombay. His research interests include data mining and data warehousing, semistructured databases, Web technology, network directories, advanced data models and applications, management of uncertain data, and deductive and object-oriented databases. Lakshmanan has a PhD in computer science from the India Institute of Science, Bargalore. He is a member of the IEEE Computer Society, the ACM, and ACM SIGMOD and ACM SIGKDD.

Raymond T. Ng is an associate professor of computer science at the University of British Columbia, Canada. His research interests include data mining and multimedia databases. Ng has a PhD in computer science from the University of Maryland at College Park.

Contact Jiawei Han at the School of Computing Science, Simon Fraser Univ., 8888 University Dr., Burnaby, B.C., Canada V5A 1S6; han@cs.sfu.ca.