Finding Cross-Object Relationships from Large Databases

Ling Feng **Eric Tsang**
Department of Computing, Hong Kong Polytechnic University,
Hong Kong, China, {cslfeng, csetsang}@comp.polyu.edu.hk

abstract

While traditional association rules demonstrate strong potential values such as to improve market strategies for retail industry, they are limited to finding associations among items within the same transaction. Consider a database of supermarket transactions, the traditional association rules can represent such knowledge as "*80% of customers who buy Chinese tea also buy teapot* **at the same time**." However, they fail to represent some more interesting rules like "*If a customer buys Chinese tea, s/he may most likely buy teapot* **within 3 days**", where the association may span across different transactions. To capture this contextual semantics which are also vital to the validation of associations, in this study, we introduce the notion of *cross-object* relationships. Two algorithms for mining cross-object association rules from large databases are developed by extension of Apriori algorithm. We show that traditional associations can be treated as a special case of cross-object relationships from both conceptual and algorithmic point of view.

1 INTRODUCTION

Since the problem of mining association rules was introduced in [1], a large amount of work has been done in various directions, including efficient, Apriori-like mining methods [2, 9, 18, 15, 16, 22, 24, 5], mining generalized, multi-level, or quantitative association rules [19, 20, 6, 12, 10, 8], association rule mining query languages [11, 23], constraint-based rule mining [13, 21, 23, 3, 7], incremental maintenance of discovered association rules [4], cyclic and interesting association rule mining [14, 17], etc. Despite these efforts, there is an important form of association rules which are useful but could not be discovered with existing association rule mining framework.

Consider a database of supermarket transactions, the traditional association rules can represent such knowledge as "*80% of customers who buy Chinese tea also buy teapot* **at the same time**." However, they fail to represent some more interesting rules like "*If a customer buys Chinese tea, s/he may most likely buy teapot* **within 3 days**", where the association may span across different transactions. In order to capture this contextual semantics into the association rules, in this paper, we introduce the notion of *cross-object* association rules. Two algorithms for mining cross-object relationships from large databases are developed by extension of Apriori algorithm.

The remainder of the paper is organized as follows. In section 2, we give a formal definition of cross-object association rules. Two algorithms for mining such association rules from databases are described in section 3. We evaluate the performance of the algorithms on both synthetic and real-life data sets in section 4. Section 5 concludes the paper.

2 CROSS-OBJECT ASSOCIATION RULES

2.1 Preliminaries

In the traditional association rule mining, the database to be mined is organized as a set of records based on *transaction time*. Except being largely used as record partition measurement, such transaction time has received less exploration.

In reality, many real-world associations do happen in certain *context*. Transaction time here is one kind of contextual information. Taking supermarket as an example, we may find rules like "*if a customer buys Chinese tea, s/he will most likely buy teapot 3 days later.*" Here, besides items, the intervals of transactions also play equally important role in the occurrence of associations. We refer to such interval as one kind of context.

In addition to time property, we can integrate other contextual information like location into

0-7803-5731-0/99$10.00 ©1999 IEEE

TID	dimensional attribute	item list
T1	day1	a, b, c
T2	day2	c, d, e
T3	day3	a, b
T4	day4	a, b, c, d
T5	day5	b, c, d, e

Table 1: A single-dimensional database

association rule mining as well. Virtually, we can enhance the traditional transaction model by a set of dimensional attributes, describing the context where the transaction happens. The dimensional attributes can be any kind of attributes as long as they are meaningful to the applications. Time, distance, temperature, latitude, etc., are typical dimensional attributes.

In this paper, we focus on single-dimensional context, where only one dimensional attribute is used. Assume that the domain of the dimensional attribute is ordinal and can be divided into equal-sized intervals. For instance, time can be divided into day, week, month, etc., and distance into meter, mile, etc. We use Δ to describe the relative difference between two transactions represented by the dimensional attribute, such as a day, a kilometer, etc.. When Δ=0, the cross-object association rules become the traditional association rules.

2.2 Definitions

Let $\mathcal{I} = \{i_1, i_2, \ldots, i_s\}$ denote a set of literals, called items. Let $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$ be the domain of the dimensional attribute. Each element of $\mathcal{C}$ can be thought of as an *address* in the context. A transaction T_i is defined to be a set of items I_i ($I_i \subseteq \mathcal{I}$) happening in c_i ($c_i \in \mathcal{C}$), denoted by $T_i.(c_i, I_i)$. For example, the price of stock A rises on January 4, shop B is opened in the suburb, plant C blooms 2500 metres above sea level, etc. A simple one-dimensional database is shown in Table 1.

For each item, we can similarly attach an address, stating where the item occurs. Here, item address can be in two different formats: (1) *absolute address*, denoted by $i.(c)$, indicates that item i appears at the address c; (2) *relative address*, denoted by $i(\Delta_i)$, indicates that i appears Δ_i away from a reference base point. If the base address is c_0, the absolute address of i is $c_0 + \Delta_i$. In the following, we refer to an item with its relative address as an **extended item**. Note that, with the same base address c_0, two extended items $a(0)$ and $a(2)$ are different, although both of them have the same item a. We use $\mathcal{I}'$ to represent the set of all the possible extended items. Similarly, we call a set of extended items as **extended itemset**. For simplicity, we use *extended itemset* and *itemset* interchangeably. With the above notations, we are now in a position to define the cross-object association rules.

Definition 1 *Given a set of transactions* $T = \{T_1.(c_1, I_1), T_2.(c_2, I_2), \ldots, T_k.(c_k, I_n)\}$ *and an extended itemset* $X = \{i_1(\Delta_1), i_2(\Delta_2), \ldots, i_m(\Delta_m)\}$, *$T$ is said to* **contain** *X if and only if there exists a base address c_0 ($c_0 \in \mathcal{C}$), such that for every $i_x(\Delta_x) \in \mathcal{I}'$, there exists $T_j.(c_j, I_j) \in T$, so that (1) $c_0 + \Delta_x = c_j$; and (2) $i_x \in I_j$*

In the example database shown in Table 1, the transaction set $\{T_2, T_3, T_4\}$ contains extended itemset $\{c(0), a(1), a(2)\}$, with respect to the reference base address *day2*.

Definition 2 **A cross-object association rule** *is an implication of the form $X \Rightarrow Y$, where $X \subset \mathcal{I}'$, $Y \subset \mathcal{I}'$, and $X \cap Y = \emptyset$.*

Based on this definition, a rule that predicts the stock price movement, "*if the price of stock A increases one day, and the price of stock C increases the following day, then most probably the price of stock B will increase on the third day*" can be expressed by a cross-object association rule as: $a(0), c(1) \Rightarrow b(2)$.

Similar to traditional association rules, we use *support* and *confidence* as two major cross-object association rule measurements.

Definition 3 *Let T_{xy} be the set of transaction sets containing extended itemsets $X \cup Y$, T_x be the set of transaction sets containing X, and $|D|$ be the total number of transactions in the database. The* **support** *and* **confidence** *of a cross-object association rule are defined as:*
$support(X \Rightarrow Y) = |T_{xy}|/|D|$
$confidence(X \Rightarrow Y) = |T_{xy}|/|T_x|$.

3 MINING CROSS-OBJECT ASSOCIATION RULES

Similar to the traditional association rule mining, we first discover the frequent itemsets which have transaction support above a predetermined minimum support; and then use the frequent itemsets to generate cross-object association rules.

Under the framework of cross-object association rules, a k-itemset is of the form $\{i_1(\Delta_1), i_2(\Delta_2), \ldots, i_k(\Delta_k)\}$, where item i_j, $1 \leq j \leq k$, is attached by a non-negative value Δ_j indicating the relative address with respect to the base address of the set. For example, a 3-itemset $\{a(0), b(1), c(3)\}$ contains three items expressed in relative addresses along the dimension. That is, taking a transaction containing item a as the base transaction, $b(1)$ denotes an item b contained in a transaction with 1 unit distance away from the base transaction, and $c(3)$ represents an item c in a transaction 3 unit distances away from the base transaction. This is quite different from the classical definition of itemset $\{i_1, i_2, \ldots, i_k\}$ in which all items lie within the same transactions. To find the frequent itemsets, two algorithms, *E-Apriori* and *EH-Apriori*, were implemented which are extensions of Apriori-based algorithms [2, 15]. Let L_k represent the set of frequent k-itemsets, and C_k the set of candidate k-itemsets. Both algorithms make multiple passes over the database. Each pass consists of two phases. First, the set of all frequent (k-1)-itemsets L_{k-1}, found in the (k-1)th pass, is used to generate the candidate itemset C_k. The candidate generation procedure ensures that C_k is a superset of the set of all frequent k-itemsets. The algorithms now scan the database. For each list of consecutive transactions, they determine which candidates in C_k are contained and increment their counts. At the end of the pass, C_k is examined to check which of the candidates are actually frequent, yielding L_k. The algorithms terminate when L_k becomes empty.

As previously reported in [15], the processing cost of the first two iterations (i.e., obtaining L_1 and L_2) dominates the total mining cost. The reason is that, for a given minimum support, we usually have a very large L_1, which in turn results in a huge number of itemsets in C_2 to process. In the cross-object association rules, this situation becomes much more serious as a lot of additional 2-itemsets like $\{a(0), a(1)\}$ may be added into C_2, thus leading to a huge amount of $|C_2|$. In order to construct a significantly smaller C_2, *EH-Apriori* adopts a similar technique of hashing as [15] to filter out unnecessary candidate 2-itemsets. When the support of candidate C_1 is counted by scanning the database, *EH-Apriori* accumulates information about candidate 2-itemsets in advance in such a way that all possible 2-itemsets are hashed to a hash table. Each bucket in the hash table consists of a number to represent how many itemsets have been hashed to this bucket thus far. Such resulting hash table can be used to greatly reduce the number of 2-itemsets in C_2.

4 PERFORMANCE STUDY

To assess the performance of the mining algorithms, we conducted a series of experiments on both synthetic data and real-life data.

4.1 Generation of Synthetic Data

The method used by this study to generate synthetic data is similar to the one used in [2] with some modifications noted below. Table 2 summarizes the parameters used and their settings. We first generate a set L of the potentially frequent itemsets, which may span several transactions, e.g., $\{a(0), b(1), c(2)\}$, and then assign a frequent itemset from L to transactions. Items and their relative addresses (intervals) in the first frequent itemset are chosen randomly, where item is picked up from 1 to N, and its interval is picked up from 0 to W. To model the phenomenon that frequent itemsets often have common items and intervals, some fraction of items and their intervals in subsequent itemsets are chosen from the previous itemset generated. We use an exponentially distributed random variable with mean equal to the *correlation level* to decide this fraction for each itemset. The remaining items and their intervals are picked at random. After generating all the items and intervals for a frequent itemset, we revise each of its intervals by subtracting the minimum interval value of this frequent itemset. In this way, the minimum interval of each potentially frequent itemset is always 0.

After generating the set L of potentially frequent itemsets, we then generate transactions in the database. Each transaction is assigned a series of potentially frequent itemsets. However, upon the generation of one transaction, we need consider a list of consecutive ones starting from this transaction, as items in a frequent itemset may span across different transactions. For example, after selecting the frequent itemset $\{a(0), b(1), c(2)\}$ for current transaction T_c, we should assign item a to T_c, item b to its next transaction T_{c+1}, and item c to T_{c+2}. If the frequent itemset picked on hand does not fit in the current or any one of its successive transactions, this itemset is put in these transactions

Parameter	Meaning	Setting
$\|D\|$	number of transactions	5000 - 25000
$\|T\|$	average size of the transactions	4 - 8
$\|MT\|$	maximum size of the transactions	7 - 11
$\|L\|$	number of potentially frequent itemsets	1000
$\|I\|$	average size of the potentially frequent itemsets	1.5, 3
$\|MI\|$	maximum size of the potentially frequent itemsets	4, 7
N	number of items	300 - 500
R	maximum interval of itemsets	0 - 3

Table 2: Parameters and settings

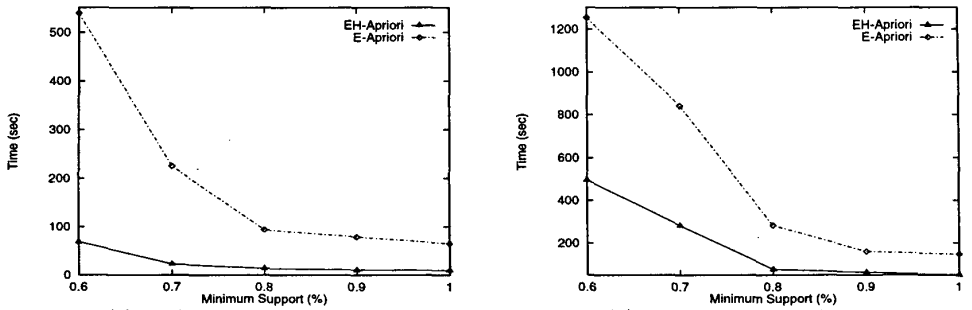
anyway in half the cases, and the itemset enters an *unfit* queue for the next transaction the rest of the cases. Each time, we pick itemsets from this queue first according to the first-in-first-out principle. Only when the queue is empty, do we perform random selection from the set L.

4.2 Experiments on Synthetic Data

Figure 1 shows the basic behavior of the algorithms when the minimum support changes.
When the minimum support increases, the execution times of both *E-Apriori* and *EH-Apriori* decrease because of reduction in the total number of candidate and frequent itemsets. Throughout the experiments, *EH-Apriori* is always far superior over *E-Apriori*. For example, in Figure 1(a), when minimum support is 0.7%, the mining time of *EH-Apriori* is about 23 seconds while that of *E-Apriori* is about 226 seconds. The former only needs to spend 10% time of the latter in getting the same cross-object association rules. This is due to the fact that, although the execution time of the first pass of *EH-Apriori* is slightly longer than that of *E-Apriori* due to the extra overhead required for building $HashTable$, it incurs significantly smalled execution time than *E-Apriori* in later Pass 2, as $|C_2|$ is decreased greatly. By checking the detailed experimental records, we find that the number of candidate 2-itemsets reduces from 305283 to 17403 by means of hash filtering (at T5-I3-W3-N500-D10K, minimum support=0.7%). Less $|C_2|$ results in much less time to test against each transaction of the database. Because the execution time of the first two passes dominates the total execution time for mining cross-object association rules, we employ hash techniques only in the early Pass 2 to achieve performance improvement. From this preliminary experiment, we note that strategies aiming at pruning unnecessary candidates for the cross-object association rules can offer much more benefits than for the traditional association rules.

4.3 Experiments on Real Data

To test the applicability of cross-object association rules, we run the algorithms against a data set collected from Singapore Stock Exchange (SSE), from which we create a LOSER set that contains the counters whose closing prices are less than the previous closing prices. We like to have the complete data set for every trading day of 1996. As the result, we have 250 records corresponding to 250 trading days in 1996. Although there are a few hundred counters in SSE, we only have complete data for 84 counters. Furthermore, as the major trend for SSE in 1996 is down side, the average transaction size of LOSER is pretty large (more than 70).
Figure 2 shows the experimental results on the LOSER data set, from which one example rule found is $UOL(0), SIA(1) \Rightarrow DBS(2)$. It says that, if UOL goes down and SIA goes down the following day, DBS will go down on the third day with confidence more than 99%. (Here, UOL stock represents lands market, SIA stock represents loans and debentures, and DBS represents banking market.) This rule reveals the closely causal relationships among three major stocks in Singapore. As known, the land properties in Singapore play an important role in national economic development. The decaying of land properties inevitably leads to bad performance of loans and debentures and bank. From such discovered rule, we may get to know some characteristics of economic structure in Singapore more or less.



(a) T5-I3-W3-N500-D10K (d) T8-I1.5-W3-N500-D10K

Figure 1: Minimum support versus execution time

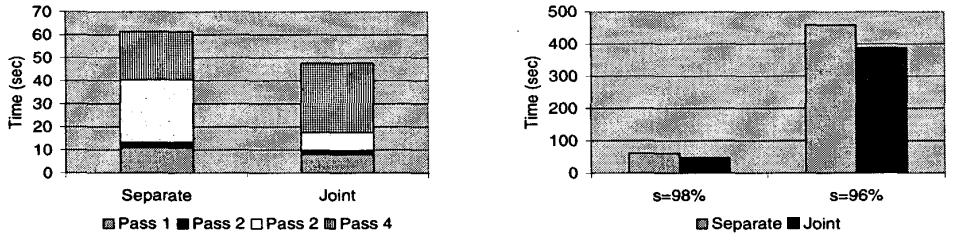


Figure 2: Execution time on LOSER data set, support=98%

Our study using stock movement data is ongoing. The results obtained so far do indicate that, with cross-object association rules, we can discover more comprehensive and interesting knowledge, and *EH-Apriori* algorithm can always achieve better performance than *E-Apriori* algorithm regardless of large or small average transaction size.

5 CONCLUSION

In this paper, we introduce the notion of *cross-object association rules*. These rules can represent not only the associations of items *within* transactions, but also associations of items *among* different transactions. The classical association rules can be viewed as a special case of the cross-object association rules. We implement two algorithms for finding such association rules by extension of Apriori. Our performance studies on both synthetic and real-life data sets show that the strategies to prune unnecessary 2-itemsets from the candidate set C_2 is more beneficial to the overall performance than it is in the case of traditional association rules.

As we only consider single-dimensional mining context in this paper, one future work is to incorporate multiple dimensional attributes into the mining process. Performing distributed and/or parallel cross-object association rules mining is another further work we are interested in.

Acknowledge

This work is supported by Hong Kong Government University Grants Council - CERG PolyU 5074/98E.

References

[1] R. Agrawal, T. Imielinski and A. Swami. Mining association rules between sets of items in large databases. In *Proc. ACM SIGMOD Intl. Conf. Management of Data*, pages 207–216, Washington D.C., USA, May 1993.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. 20th Intl. Conf. Very Large Data Bases*, pages 478–499, Santiago, Chile, September 1994.

[3] E. Baralis and G. Psaila. Designing templates for mining association rules. *Journal of Intelligent Information Systems*, Volume 9, Number 1, pages 7–32, 1997.

[4] D. Cheung, J. Han, V. Ng and C.Y. Wong. Maintenance of discovered association rules in large databases: an incremental updating technique. In *Proc. Intl. Conf. Data Engineering*, pages 106–114, New Orleans, Louisiana, USA, February 1996.

[5] M. Fang, N. Shivakumar, H. Garcia-Molina, R. Motwani and J.D. Ullman. Computing iceberg queries efficiently. In *Proc. 24th Intl. Conf. Very Large Data Bases*, pages 299–310, New York, USA, August 1998.

[6] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *Proc. 21st Intl. Conf. Very Large Data Bases*, pages 420–431, Zurich, Switzerland, September 1995.

[7] J. Han and Y. Fu. Meta-rule-guided mining of association rules in relational databases. In *Proc. 1st Intl. Workshop on Integration of Knowledge Discovery with Deductive and Object-Oriented Databases*, pages 39–46, Singapore, December 1995.

[8] M. Kamber, J. Han and J.Y. Chiang. Metarule-guided mining of multi-dimensional association rules using data cubes. In *Proc. Intl. Conf. Knowledge Discovery and Data Mining*, pages 207–210, California, USA, August 1997.

[9] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen and A.I. Verkamo. Finding interesting rules from large sets of discovered association rules. In *Proc. 3rd Intl. Conf. Information and Knowledge Management*, pages 401–408, Gaithersburg, Maryland, November 1994.

[10] B. Lent, A. Swami and J. Widom. Clustering association rules. In *Proc. Intl. Conf. Data Engineering*, pages 220–231, Birmingham, England, April 1997.

[11] R. Meo, G. Psaila and S. Ceri. A new sql-like operator for mining association rules. In *Proc. 22th Intl. Conf. Very Large Data Bases*, pages 122–133, Mumbai, India, September 1996.

[12] R.J. Miller and Y. Yang. Association rules over interval data. In *Proc. ACM SIGMOD Intl. Conf. Management of Data*, pages 452–461, Tucson, Arizona, USA, June 1997.

[13] R. Ng, L.V.S. Lakshmanan, J. Han and A. Pang. Exploratory mining and pruning optimizations of constrained association rules. In *Proc. ACM SIGMOD Intl. Conf. Management of Data*, pages 13–24, Seattle, Washington, June 1998.

[14] B. Özden, A. Ramaswamy and A. Silberschatz. Cyclic association rules. In *Proc. Intl. Conf. Data Engineering*, 1998.

[15] J.-S. Park, M.-S. Chen and P.S. Yu. An effective hash based algorithm for mining association rules. In *Proc. ACM SIGMOD Intl. Conf. Management of Data*, pages 175–186, San Jose, CA, May 1995.

[16] J.-S. Park, M.-S. Chen and P.S. Yu. Mining association rules with adjustable accuracy. Technical Report IBM Research Report, 1995.

[17] S. Ramaswamy, S. Mahajan and A. Silberschatz. On the discovery of interesting patterns in association rules. In *Proc. 24th Intl. Conf. Very Large Data Bases*, pages 368–379, New York, USA, August 1998.

[18] A. Savasere, E. Omiecinski and S. Navathe. An efficient algorithm for mining association rules in large databases. In *Proc. 21st Intl. Conf. Very Large Data Bases*, pages 432–443, Zurich, Switzerland, September 1995.

[19] R. Srikant and R. Agrawal. Mining generalized association rules. In *Proc. 21st Intl. Conf. Very Large Data Bases*, pages 409–419, Zurich, Switzerland, September 1995.

[20] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Proc. ACM SIGMOD Intl. Conf. Management of Data*, pages 1–12, Montreal, Canada, June 1996.

[21] R. Srikant, Q. Vu and R. Agrawal. Mining association rules with item constraints. In *Proc. 3rd Intl. Conf. Knowledge Discovery and Data Mining*, pages 67–73, Newport Beach, California, August 1997.

[22] H. Toivonen. Sampling large databases for association rules. In *Proc. 22th Intl. Conf. Very Large Data Bases*, pages 134–145, Mumbai, India, September 1996.

[23] D. Tsur, J.D. Ullman, S. Abitboul, C. Clifton, R. Motwani and S. Nestorov. Query flocks: a generalization of association-rule mining. In *Proc. ACM SIGMOD Intl. Conf. Management of Data*, pages 1–12, Seattle, Washington, June 1998.

[24] M.J. Zaki, S. Parthasarathy, M. Ogihara and W. Li. New algorithms for fast discovery of association rules. In *Proc. 3rd Intl. Conf. Knowledge Discovery and Data Mining*, pages 283–286, Newport Beach, CA, USA., August 1997.