Discovery of Association Rules in Temporal Databases[1]

Abdullah Uz Tansel[2] and **Necip Fazil Ayan**
Department of Computer Engineering and Information Science
Bilkent University
06533, Ankara, Turkey
{atansel, fayan}@cs.bilkent.edu.tr

Abstract

Temporal databases naturally contain a wealth of information which can be unearthed by knowledge discovery and data mining techniques. Discovering association rules in market basket data have been widely studied and many algorithms have been developed. In this study, we examine discovery of association rules in temporal databases. We add enumeration operation to the relational algebra to prepare the data for discovery of association rules. To observe the changes in the association rules and their statistics over the time, we apply knowledge discovery techniques on series of datasets obtained over consecutive time intervals, instead of applying on the whole database.

Keywords. Association rules, data mining, temporal databases, enumeration operation.

Introduction

With the recent developments in computer storage technology, many organizations have collected and stored massive amounts of data. The widespread use of bar-codes for most commercial products, the computerization of many business and government transactions, and the advances in data collection tools have paved the way for the storage of huge amounts of data. Given these massive data sets, it is very difficult for humans to process them by traditional methods of data analysis. Even though voluminous information is buried within these datasets, the information is not directly available for the users. Therefore, an urgent need exists for developing techniques and tools that assist human beings to analyze and automatically extract the hidden knowledge buried within large volumes of data. Knowledge discovery in databases includes techniques and tools to address this need.

The improved capabilities of computer hardware and software also opened the way for implementing temporal databases, which store past, present and possibly future data (Tansel et al. 1993). Temporal databases are append-only and current data values become historical data as new data values are added to the database. Therefore, temporal databases naturally are fertile data repositories for data mining and knowledge discovery. Among many possible applications of knowledge discovery in temporal databases, we examine the discovery of association rules.

This explosive growth in data and the advances in database technology contributed to the emergence of a new field, which devices techniques and tools that assist human beings to automatically transform the stored data into useful and understandable information. This new field is known as *knowledge discovery in databases* and it refers to the whole process for extracting useful information from large amounts of data nuggets automatically and efficiently. It includes data pre-processing, selection of suitable techniques, discovery of frequent and interesting patterns, and post-processing of the results. Data mining is an important step of the knowledge discovery in databases. It deals with the discovery of frequent and interesting patterns, exceptional cases, sequential patterns, and anomalous cases. The primary data mining tasks are classification, regression, clustering, summarization, dependency modeling, and change and deviation detection (Fayyad, Piatetsky-Shapiro, and Smyth 1996). Data mining is application dependent and different applications may require different mining techniques.

Association rules are one of the promising subjects of data mining, and have been widely explored to date. An association rule $A \rightarrow B$ is a statement of the form "for s% of the transactions, if the value of an attribute set A is a particular value, then the value of attribute set B tends to have another particular value". In this manner, association rules aim at discovering the patterns of co-occurrences of the attributes in the database. The problem of discovering association rules was first explored in (Agrawal, Imielinski, and Swami 1993) on a supermarket basket data, that is the set of transactions that include items purchased by the customers. In this pioneering work, mining of association rules was decomposed into two subproblems: the first one is to discover all frequent patterns, and the second is to generate the association rules from the sets representing those frequent itemsets. The second subproblem is straightforward, and can be done efficiently in a reasonable time. However, the first subproblem is very tedious and computationally expensive for very large databases, which is the case for many real life applications. Depending on the size of the database, many efficient algorithms have been proposed for finding the frequent patterns in a database (Agrawal, Imileinski, and Swami 1993; Agrawal and Srikant 1994; Brin et al. 1997; Chen, Park, and Yu 1995;

[2] On leave from Baruch College, the City University of NewYork.

Mannila, Toivonen, and Verkamo 1994; Savasere, Omiecinski, and Navathe 1995; Toivonen 1996). However, the time complexity of such a process is still very large. Similarly, the time complexity of mining temporal data is NP-hard (Wijsen and Meersman 1997).

In large databases, the number of discovered rules could be very high. Because of this reason, a post-processing step on the rules is needed to interpret the extracted knowledge. For this purpose, algorithms for effectively pruning the rules have been proposed. Those algorithms are based on the concept of interestingness of patterns. Algorithms solely based on computers, i.e. automatically eliminating redundant and transitive rules, and systems with the human interaction to guide the pruning process have been introduced for reducing the number of rules obtained at the end of the discovery process.

In this paper, we will examine the problem of discovery of association rules in temporal databases. Instead of extracting rules throughout the whole time line, we will extract the rules for consecutive time intervals with different time granularities. By means of this approach, the users of the knowledge discovery system will be able to see the changes and fluctuations in the association rules as well as the time periods over which these rules are valid. Our approach is different than those in (Agrawal and Srikant 1995) and (Betini, Wang, and Jajodia 1996) that find the sequence of patterns in a customer's transactions, and patterns of temporal events, respectively.

The rest of this paper is organized as follows. Section 2 introduces temporal databases, along with the interpretation of time in temporal databases and the enumeration operation. In Section 3, we present our formalism to discover the association rules in temporal databases. Finally, we conclude in Section 4 with the advantages of our approach and the future research.

Temporal Databases

There are different concepts of time: valid time, transaction time, event time, etc. Valid time denotes when attribute values become valid (effective) whereas transaction time denotes when data values are recorded in the database. A temporal database can accommodate any one or a combination of these concepts of time. Time can be added to tuples (Snodgrass 1987), hence utilizing 1NF relations. Time can also be added to attributes that require N1NF relations (Gadia 1988; Tansel 1986; Tansel 1997). Time points, time intervals, temporal elements –sets of disjoint time intervals (Tansel et al. 1993) or sets of time points can be used as timestamps to represent time reference of values. The type of temporal database and the type of timestamp used are irrelevant for discovering association rules. The methodology we propose applies all types of temporal databases.

The Time

A temporal database evolves over time that is represented by a calendar system, that has multiple granularities such as hours, days, etc. We assume that there is an absolute time line represented by real numbers (Wang et al. 1997). Clock ticks of a calendar are represented by natural numbers. Each clock tick corresponds to an interval of real numbers. Thus, clock ticks of a calendar system are mapped to consecutive intervals of real numbers that are disjoint. This definition allows representation of multiple time granularities, which is essential for modeling temporal data as well as mining useful information from it.

Generally, data is batched for a meaningful period of time and data mining techniques are applied to discover association rules. However, in a temporal context, we can batch data by different time granularities, i.e. hour, day, month, quarter, etc. and this may reveal further useful information. Let's assume that the absolute time line is represented by a time unit of seconds. A calendar based on minutes would see every consecutive 60 seconds as one minute. A calendar of hours would consider every 60 minutes –3600 seconds- as one hour and so on.

Supermarket transactions are in general recorded at the second level. These transactions then are accumulated for a one-week period. Each of the original transactions is directly placed in one batch. It is also possible to combine several transactions into one representative transaction. For instance, if the same customer performs two or more transactions a day, these transactions can be combined into one representative transaction. We add a new operation, *enumeration*, to the relational algebra to prepare temporal data for discovering association rules (Tansel 1987).

The Model

Let R (TID, CID, I_1, …, I_n, S, E) be a relation scheme. The attributes of R are transaction identifier (TID), customer identification number (CID), and items (I_1, …, I_n). Moreover, customer specific data including a customer identification number can be placed into a separate relation, C (C_1, …, C_k, S, E). S and E denotes an interval [S, E] indicating the validity period of the data represented by a tuple in R. (Note that R is a generalized temporal relation scheme. For supermarket data, time interval [S, E] is a time point, i.e. S=E, and one time attribute would be sufficient. This is called an event relation in (Tansel 1987)).

Let D be a database that consists of temporal relations such as R. For the sake of simplicity, we use tuple timestamping in explaining our formalism. Our formalism applies equally on attribute timestamping as well.

Enumeration operation extracts portions of a temporal relation and prepares it for data mining process. Let R be a temporal relation containing supermarket transactions. Let $X \subseteq \{TID, C_1, \ldots, C_k\}$ and f_I be aggregate function whose parameters are I_1, …, I_n and T be a one-column relation whose tuples designate the intervals of interest. Enumeration operation is defined as follows:

$$R < X, f_I > T = \{[X].y.t \mid r \in R \wedge t \in T \wedge \\ y = f(\{u[I] \mid u \in S \wedge u[X] = r[X] \wedge \\ [u[S], u[E]] \cap t \neq \emptyset\})\}$$

Enumeration operation partitions the tuples in R with respect to the attributes in *T* and *X*, that contains transaction specific and/or customer specific attributes. The function f_I is applied on each attribute *I* of each partition. Possible functions are given in Table 1. Note that enumeration can be redefined to apply aggregate functions on individual attributes in *I*.

Function	Definition
FIRST	Returns I-component of the first tuple in chronological order
LAST	Returns I-component of the last tuple in chronological order
ANY	Returns I-component of any tuple
ALL	Returns I-components of all the tuples
COUNT	Returns count of tuples
CON	Combines I-components of the tuples of a partition into one tuple

Table 1: Aggregate Functions

Finding Association Rules

In this section, we will briefly introduce the formal description of the problem of discovery of association rules in temporal databases, and give how this problem can be decomposed into two subproblems, as proposed in (Agrawal, Imielinski, and Swami 1993).

Formal Description of the Problem

Agrawal et al. define the problem of discovering association rules in databases in (Agrawal, Imielinski, and Swami 1993; Agrawal and Srikant 1994). We adopt this definition on the temporal relation scheme *R*.

Let $\{I_1,...,I_m\}$ be a set of binary attributes, called items. Let *D* be a database of temporal relations. A tuple in *R* represents a transaction that contains a subset of items *I*. Each transaction *T* is represented as a binary vector, with *T*[*k*]=1 if *T* bought the item I_k, and *T*[*k*]=0 otherwise. Let *X* be a set of items in *I*. We say that a transaction *T* satisfies *X* if for all items I_k in *X*, *T*[*k*]=1.

By an association rule, we mean an implication of the form $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$. We call *X* as the *antecedent* of the rule, and *Y* as the *consequent* of the rule. The rule $X \Rightarrow Y$ holds in *D* with confidence *c* if *c*% of the transactions in *D* that contain *X* also contain *Y*. The rule $X \Rightarrow Y$ has support *s* in *D* if *s*% of the transactions in *D* contain $X \cup Y$. Confidence is a measure of rule's strength while support corresponds to statistical significance.

An itemset *X* is called a *k*-itemset if it contains *k* items from *I*. $X + Y$ is a *l*-extension of *X* if *Y* is a *l*-itemset and $X \cap Y = \emptyset$.

Given a temporal relation *R*, the problem of mining association rules is to generate all association rules that have support and confidence greater than the user-specified *minsup* and *minconf*, respectively.

Decomposition of the Problem

The problem of finding association rules can be decomposed into two parts (Agrawal, Imielinski, and Swami 1993; Agrawal and Srikant 1994):

Step 1: Generate all combinations of items that have fractional transaction support above a certain threshold, called *minsup*. Those items are called *large* itemsets, and all other combinations whose support is below the threshold are called *small* itemsets.

Step 2: Use the large itemsets to generate the association rules. For every large itemset *l*, find all non-empty subsets of *l*. For every such subset *a*, output a rule of the form $a \Rightarrow (l\text{-}a)$ if the ratio of *support* (*l*) to *support* (*a*) is at least *minconf*. If an itemset is found large in the first step, the support of that itemset should be maintained in order to compute the confidence of the rule in the second step efficiently.

The key for optimization of the computation of association rules lies in the techniques used to create candidate itemsets. The smaller the number of candidate itemsets is, the faster the algorithm would be. However, in large retailing applications, the number of transactions might be in the order of millions, and the number of items might be in the order of thousands. If there are *m* items in the database, then there are 2^m possible itemsets, indicating that brute force search techniques require exponential time. Thus, many fast algorithms are proposed to find the frequent patterns in a data set working in time linearly to the number of large itemsets and number of transactions (tuples) in the database. These algorithms use clever data structures to speedup the search process.

Algorithms for discovering large itemsets make multiple passes over the data (Agrawal, Imileinski, and Swami 1993; Agrawal and Srikant 1994; Brin et al. 1997; Chen, Park, and Yu 1995; Mannila, Toivonen, and Verkamo 1994; Savasere, Omiecinski, and Navathe 1995; Toivonen 1996). Associated with each itemset is a counter that is used to keep its support. In the first pass, the supports of the individual items are counted, and then large ones are determined. Each subsequent pass starts with a seed set of itemsets found to be large in the previous pass, and uses this seed set for generating new potentially large itemsets (candidate itemsets). Then, the actual supports for these candidate itemsets are counted throughout the data. At the end of each pass, large itemsets are determined out of the candidate itemsets, and they are included into the seed set for the next pass. This process continues until no new large itemsets are found.

Association Rules in Temporal Databases

An entire temporal database can be used for mining association rules. However, the resulting rules may not be interesting since data accumulated over a longer time span is used. Consumer behavior and preferences change over time. Observation of these changes provides useful information for various decision-making purposes. A temporal database provides us the opportunity to observe these changes, by mining various subsets of a temporal database.

Our approach aims at detection of the changes in the association rules. The changes in association rules occur over periods of time, and include a decrease/increase in the support/confidence of an association rule and addition/removal of itemsets from a particular itemset. In order to observe how a frequent pattern fluctuates within certain periods, it is necessary to extract association rules from datasets accumulated over consecutive time periods. In this process, the vital parameters are the length of the interval through which the set of association rules will be extracted, and the minimum support and confidence for these rules.

We decompose the problem of observing changes in association rules into three subproblems. The first one is the pre-processing step that extracts the portion of the temporal data valid during a given interval. The second step is the discovery of association rules, and the final step is the presentation of rules and their fluctuations to the users. Certainly, the final step requires a good user interface design, which will allow the user to observe the changes in specific association rules in an easy and understandable manner. In the following subsections, we will explain the details of each subproblem.

Pre-processing Step. In the pre-processing step, we first decide on the length of the interval during which data is accumulated, and the number of such intervals. Then, we repeatedly walk over the database to extract a subset of temporal data for each interval. For instance, if we search for the association rules for one-week periods, we should take the portion of the database for weekly periods, beginning from the first week and continuing with each succeeding week until the present time or a cut-off date. This process is performed using the enumeration operation previously described in the paper.

Discovery of Association Rules. Our approach discovers the rules within a certain period, and repeats this process for the whole database for consecutive time periods. Considering the nature of the application, different time intervals varying in length can be tested. For example, for supermarket basket data, this interval is usually a week. However, mining even daily transactions may reveal interesting results.

Let $TI = [s, e]$ be a time period where s denotes its starting time point and e denotes its end. R_{TI} denotes the portion of a temporal relation that is valid during the interval $[s, e]$. Let A_{TI} be the set of large itemsets derived from R_{TI}. We can obtain A_{TI} by any of the techniques proposed in the literature (Agrawal, Imileinski, and Swami 1993; Agrawal and Srikant 1994; Brin et al. 1997; Chen, Park, and Yu 1995; Mannila, Toivonen, and Verkamo 1994; Savasere, Omiecinski, and Navathe 1995; Toivonen 1996). Moreover, we can calculate A_{TI} for any number of periods in sequence. Let $TI_1, \ldots, TI_n$ be consecutive, non-overlapping, fixed length time intervals, and $A_{TI_1}, \ldots, A_{TI_n}$ be the corresponding large itemsets as depicted in Figure 1. Algorithm 1 finds the large itemsets A_{TI_i}, for $i = 1, \ldots, n$. Let L be the length of a time interval, $START$ be the start of TI_1, and END be the end of TI_n. The time unit granularity of the temporal database is g.

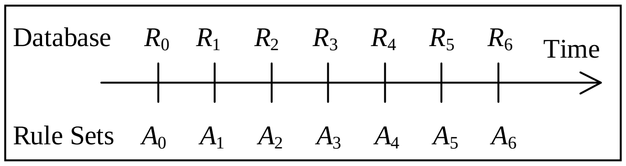

Figure 1: Association Rules in Sequence

Algorithm 1 allows us to walk the database in different time granularities. Changing L to $n \times L$ walks through the database in time intervals of $[s, s+n \times L]$. For example, if L is one day, $7 \times L$ would be a week. This is particularly useful for observing seasonal, cyclical patterns, and changes in such patterns. Moreover, it is possible to walk through the database by sliding the starting points of intervals different than the interval length. Changing lines 2 and 8 in Algorithm 1 by $2 \times L$ would suffice to do this type of walking. We can also compare the rules obtained from any two different database subsets, R_i and R_j, $i \neq j$.

```
1  s = START
2  e = START + L
3  while s ≤ END do
4  begin
5      R_[s,e] = R <X, f_I>[s, e]
6      compute A_[s,e]
7      s = s + L + g
8      e = s + L
9  end
```

Algorithm 1: Discovery of Association Rules in a Temporal Database

Algorithm 1 combines the two steps of pre-processing and discovery of association rules. There is a natural opportunity for optimization in this process. As the enumeration operation is carried out to prepare the data, initial step of discovering association rules, i.e. computing large 1-itemsets and their counts, can be done at the same time in one pass over the temporal data. Then, the second step of discovering association rules, i.e. computing large k-itemsets for $k>1$, is completed. There are two possible approaches. One alternative is to prepare the entire data set by the enumeration operation for the designated interval and to use this data in the second step of association rule discovery. The second alternative is retrieving the data for subintervals of the original interval. Naturally, datasets for the subinterval would be smaller and may fit the memory. In this case, the original enumeration operation is replaced by a sequence of enumeration operations, one for each subinterval. The second step of the discovery process utilizes the results of these smaller datasets. This lends itself to the use of parallel processing techniques. Among the algorithms for discovering association rules, Apriori algorithm proposed by Agrawal and Srikant 1994 seems a good fit for the first alternative whereas Partition algorithm proposed by Savasere, Omiecinski, and Navathe 1995 can be efficiently used in the second alternative.

After extracting the patterns for small periods, the large itemsets for the larger time periods can be obtained by using the patterns extracted before. This is exactly same as the second step in the Partition algorithm (Savasere, Omiecinski, and Navathe 1995). The only important point is that the minimum support for small periods should be held small enough in order to prevent missing any large itemsets. Of course, the minimum support should be adjusted well for the larger time intervals in order to have a reliable and complete set of association rules within those periods.

As the length of the time intervals decreases, the minimum support for the rules should be selected optimally, i.e. small enough not to miss any large itemset, and large enough to have a reliable set of association rules. It is very difficult to set these parameters a priori since data mining applications are generally user and application dependent. In other words, a pattern that is interesting to one user may be of no interest to another user, and the interestingness of patterns varies from application to application. Therefore, instead of examining all the possible values, it may be preferable to adjust the minimum support by a user-assisted system, in which a human being takes active role in the data mining application by supplying the minimum supports for each interval.

Presentation of the Resulting Association Rules. If the temporal sequence contains few large itemsets –hence rules, the user can observe the changes easily, i.e. disappearance or emergence of a large itemset, continuous existence of a large itemset, etc. However, if there are many large itemsets in the temporal sequence, tools are needed at the user interface to browse through the large itemsets and associated rules.

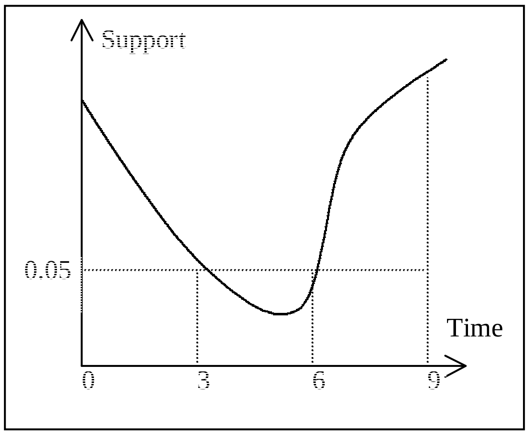

Figure 2: Changes in the Support of an Itemset

The presentation of the association rules includes two parts. The first one presents the support of a large itemset or support/confidence of an association rule through the valid time interval in the database. This is simply a graph where x-axis represents the time line, and y-axis denotes the support of a large itemset or support/confidence of a particular rule. This is especially useful when the user wants to see the fluctuations in a particular rule. Figure 2 is an example to present the user how the support of a large itemset changes over the time. (The minimum support is taken as 0.05.)

The second part of the presentation exhibits the time intervals where the itemset is large. This type of presentation is useful when the user wants to see the changes in co-occurrences of items throughout the time. For each item in the database, a time line is presented along with the additional items through the whole time line. It is sufficient to show only the large itemsets that include the specified item and that is not a subset of any large itemset. This is sufficient since any subset of a large itemset is also a large itemset. For example, in the upper part of Figure 3, *A* is large within time points 0-3, *ABC* is large within time points 6-9, and *A* is small within time points 3-6. The user also has a chance to see the time lines for the *k*-itemsets, where *k*>1. Figure 3 presents some example time lines for different itemsets.

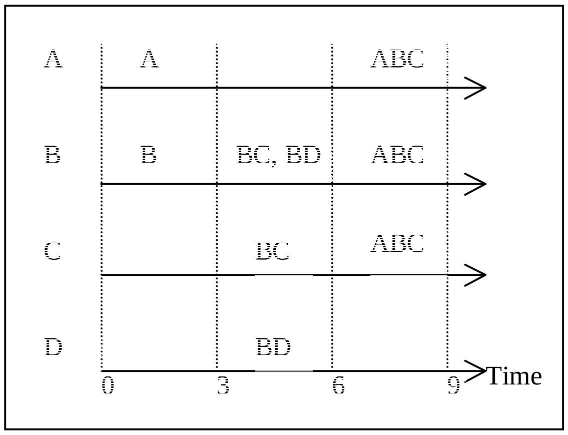

Figure 3: Large Itemsets in Each Interval

By means of these presentation techniques, the user is able to see the changes and fluctuations in the association rules. In this manner, the users are able to see how the frequent patterns changed throughout the time, perhaps providing insight about the behavior of related agents, about which data is collected.

Conclusion

Knowledge discovery in general and discovering association rules in particular involves extraction of useful information from large datasets. Temporal databases are naturally good sources for knowledge discovery. Moreover, temporal databases provide the opportunity to see how the unearthed knowledge changes throughout the time. In this paper, we have developed a formalism to make temporal databases as a good source for knowledge discovery. The enumeration operation prepares the data for knowledge discovery. Retrieval of the temporal data in processing the enumeration operation also is combined with the first step of discovering association rules. In this process, the optimization tools available in a temporal database will significantly affect the performance of the entire operation.

Our approach in discovery of association rule allows the user of a knowledge discovery system to observe the changes and fluctuations in the association rules. It is also possible to observe the seasonal or cyclical changes and fluctuations in the rules. Moreover, the formalism presented in this paper can be used to extract the association rules in a given time interval, beginning the discovery process in small time intervals, and combining the results of them to obtain a complete set of association rules in the specified time interval. This approach will yield important results for the decision-makers in fields where customer preferences are strongly interrelated with the time. For example, the application of our formalism to the market basket data will be beneficial while interpreting the results of the promotions, discounts, etc.

This paper attempts to extract the wealth of information buried in temporal databases. There are many problems for future research such as optimizing the enumeration operation and the discovery of association rules, evaluating the performance of different algorithms for discovering association rules, determining the right interval size, etc.

References

[1] Agrawal, R.; Imielinski, T.; and Swami, A. 1993. Mining Association Rules between Sets of Items in Large Databases. In Proceedings of ACM SIGMOD, 207-216, Washington, D.C.

[2] Agrawal, R., and Srikant, R. 1994. Fast Algorithms for Mining Association Rules. In Proceedings of the 20th International Conference on Very Large Databases (VLDB'94), 487-499, Santiago, Chile.

[3] Agrawal, R., and Srikant, R. 1995. Mining Sequential Patterns. In Proceedings of International Conference on Data Engineering (ICDE'95*)*, Taipei, Taiwan.

[4] Betini, C.; Wang, X. S.; and Jajodia, S. 1996. Testing Complex Temporal Relationships Involving Multiple Granularities and Its Application to Data Mining. In Proceedings of PODS'96, 68-78, Montreal, Canada.

[5] Brin, S.; Motwani, R.; Ullman, J. D.; and Tsur, S. 1997. Dynamic Itemset Counting and Implication Rules for Market Basket Data. In Proceedings of ACM SIGMOD, 255-264, Arizona, U.S.A.

[6] Chen, M. S.; Park, J. S.; and Yu, P. S. 1995. An Effective Hash Based Algorithm for Mining Association Rules. In Proceedings of ACM SIGMOD, 175-186, San Jose, California.

[7] Fayyad, U. M.; Piatetsky-Shapiro, G.; and Smyth, P. 1996. From Data Mining to Knowledge Discovery: An Overview. In *Advances in Knowledge Discovery and Data Mining*, 1-31, eds. Fayyad, U. M.; Piatetsky-Shapiro, G.; Smyth, P.; and Uthurusamy, R. AAAI/MIT Press.

[8] Gadia, S. K. 1988. A Homogenous Relational Model and Query Languages for Temporal Databases. *ACM Transactions on Database Systems* 13(4): 418-448.

[9] Mannila, H.; Toivonen, H.; and Verkamo, A. I. 1994. Efficient Algorithms for Discovering Association Rules. In Proceedings of AAAI'94 Workshop on Knowledge Discovery in Databases (KDD'94), 181-192, Seattle, Washington.

[10] Savasere, A.; Omiecinski, E.; and Navathe, S. 1995. An Efficient Algorithm for Mining Association Rules in Large Databases. In Proceedings of the 21st International Conference on Very Large Databases (VLDB'95), Zurich, Switzerland.

[11] Snodgrass, R. 1987. The Temporal Query Language TQuel. *ACM Transactions on Database Systems* 12(2): 247-298.

[12] Tansel, A. U. 1986. Adding Time Dimension to Relational Model and Extending Relational Algebra. *Information Systems* 11(4): 343-355.

[13] Tansel. A.U. 1987. A Statistical Interface to Historical Relational Databases. In Proceedings of the 3rd International Conference on Data Engineering (ICDE'87), 536-546.

[14] Tansel, A .U.; Clifford, J.; Gadia, S. K.; Jajodia, S.; Segev, A.; and Snodgrass, R. eds. 1993. *Temporal Databases: Theory, Design and Implementation.* Redwood City, California: Benjamin Cummings.

[15] Tansel, A. U. 1997. Temporal Relational Data Model. *IEEE Transactions on Knowledge and Data Engineering* 9(3): 464-479.

[16] Toivonen, H. 1996. Sampling Large Databases for Association Rules. In Proceedings of the 22nd International Conference on Very Large Databases (VLDB'96), Mumbay, India.

[17] Wang, X. S.; Betini, C; Brodsky, A.; and Jajodia. S. 1997. Logical Design for Temporal Databases with Multiple Granularities. *ACM Transactions on Database Systems* 22(2): 115-170.

[18] Wijsen, J., and Meersman, R. 1997. On the Complexity of Mining Temporal Trends. In Proceedings of SIGMOD'97 Workshop on Research Issues on Data Mining and Knowledge Discovery, SanJose, Claifornia.