# A Fuzzy Approach for Mining Generalized Frequent Temporal Patterns

Francisco Guil, Alfonso Bosch
Dept. LyC
Universidad de Almería
{fguil, abosch}@ual.es

Antonio Bailón
Dept. CCIA
Universidad de Granada
bailon@decsai.ugr.es

Roque Marín
Dept. IIC
Universidad de Murcia
roque@dif.um.es

## Abstract

*The incorporation of temporal semantics into the traditional data mining techniques has caused the creation of a new area called Temporal Data Mining. This incorporation is especially necessary if we want to extract useful knowledge from dynamic domains, which are time-varying in nature. Related to this topic, we proposed in [11] an algorithm named $TSET$ for mining temporal patterns (sequences) from datasets. In this paper, we present an extension of the algorithm based on the incorporation of fuzzy sets techniques into the mining process in order to extract general, flexible, and temporal patterns which reflects the uncertainty and imprecision presented in real domains.*

## 1. Introduction

Data mining is an essential step in the process of knowledge discovery in databases that consists of applying data analysis and discovery algorithms that produce a particular enumeration of structures over the data [9]. There are two types of structures: models and patterns. So, we can talk about local and global methods in data mining [17]. In the case of local methods, the simplest case of pattern discovery is finding *association rules* [2]. The initial motivation for association rules was to aid in the analysis of large transactional databases. The discovery of association rules can potentially aid decision making within organizations. Another approach is integrating the data mining process into the development of Knowledge Based Systems [19].

Since the problem of mining association rules was introduced by *Agrawal* in [2], a large amount of work has been done in several directions, including improvement of the *Apriori* algorithm, mining generalized, multi-level, or quantitative association rules, mining weighted association rules, fuzzy association rules mining, constraint-based rule mining, efficient long patterns mining, maintenance of the discovered association rules, etc. We want to point out the work in which a new type of association rules was introduced, the *inter-transaction association rules* [16, 15]. Temporal data mining can be viewed as an extension of this work.

Temporal data mining can be defined as the activity of looking for interesting correlations or patterns in large sets of temporal data accumulated for other purposes [7]. It has the capability of mining activity, inferring associations of contextual and temporal proximity, some of which may also indicate a cause-effect association. This important kind of knowledge can be overlooked when the temporal component is ignored or treated as a simple numeric attribute [22].

Data mining is an interdisciplinary area which has received contributions from a lot or disciplines, mainly from databases, machine learning and statistic. In [26] we found a review of three books, each one written from a different perspective. Although each perspective make strong emphasis on different aspects of data mining (efficiency, effectiveness, and validity), only when we simultaneously take these three aspects into account we may get successful data mining results. However, in the case of temporal data mining techniques, the most influential area is artificial intelligence because its work in temporal reasoning have guided the development of many of this techniques. In non-temporal data mining techniques, there are usually two different tasks, the description of the characteristics of the database (or analysis of the data) and the prediction of the evolution of the population. However, in temporal data mining this distinction is less appropriate, because the evolution of the population is already incorporated in the temporal properties of the data being analyzed.

We can found in the literature a large quantity of temporal data mining techniques. We want to highlight some of the most representative ones. So, we can talk about sequential pattern mining [3], episodes in event sequences [18], temporal association rules mining [4, 12, 13], discovering calendar-based temporal association rules [14], patterns with multiple granularities mining [7], and cyclic association rules mining [20]. However, there is an important form of temporal associations which are useful but could not be discovered with this techniques. These are the inter-

transaction associations presented in [15, 16]. The introduction of this type of associations was motivated by the observation that many real-world associations happen under certain context, such as time, place, etc. In the case of temporal context, inter-transactional associations represents associations amongst items along the dimension of time. Due to the number of potential association becomes extremely large, the mining of inter-transaction association poses more challenges on efficient processing than classical approaches. In order to make the mining of inter-transaction associations practical and computationally tractable, several methods have been proposed in [24, 23, 10].

Working in the same direction, in [11] we presented an algorithm named $TSET$ based on the inter-transactional framework for mining frequent sequences (also called frequent temporal patterns or frequent temporal associations) from several kind of datasets. The improvement of the proposed solution was the use of a unique structure to store all frequent sequences. The data structure used is the well-known set-enumeration tree, commonly used in the data mining area [1, 5, 6, 8], in which the temporal semantic is incorporated. Although the algorithm can extract very interesting temporal patterns, the strictness of the process leads to a method that is not very useful in a lot of application domains where uncertainty and imprecision are presented. The aim of this paper is to extend the algorithm introducing fuzzy sets techniques into the mining process in order to make it more expressive and flexible. This enhance leads into a general framework capable of extract general, practical and flexible temporal patterns.

The remainder of the paper is organized as follows. Section 2 gives a formal description of the problem of mining frequent generalized temporal patterns (sequences) from datasets. Section 3 introduces the algorithm named $TSET^f$. Section 4 presents an example to illustrate the fuzzy approach proposed. Conclusions are finally drawn in Section 5.

## 2. Problem Description

In this section we describe our notation, some basic definition, and we introduce the goals of our algorithm.

**Definition 1** *A* dataset D *is an ordered sequence of records* $D[0]$, $D[1]$,..., *where each* $D[i]$ *can have col attributes, c[0],...,c[col-1]. The 0-attribute will be the dimensional attribute, the temporal data associated with the record, expressed in temporal units. The rest of attributes can be quantitative or categorical. We assume that the domain of each attribute is a finite subset of non-negative integers, and we also assume that the structure of time is discrete and linear. Due to every event registered has its absolute date identified, we represent the time for events with an absolute dating system [21]. In order to simplify the calculations, we transform the original dataset subtracting the date of each record from the date of the first record, the time origin.*

With this generic definition of dataset, and with minimal modifications, the algorithm that we propose works with different types of sources, that is, relational databases, transactional databases and data streams.

**Definition 2** *An* event $e$ *is a 3-tuple* $(c[i], v, t)$*, where* $0 < i < col$*,* $v \in dom\{c[i]\}$*, and* $t \in dom\{c[0]\}$*, that is,* $t \in \mathbb{N}$*. Events are "things that happen", and they usually represent the dynamic aspect of the world [21]. In our case, an event is related to the fact that a value* $v$ *is assigned to a certain attribute* $c[i]$ *with the occurrence time* $t$*. We will use the notation* $e.c$*,* $e.v$*, and* $e.t$ *to set and get the attribute, value, and time variables related to the event* $e$*. The set of all distinct pairs (c[i],v) can be also called* event types*.*

**Definition 3** *Given two events* $e_1$ *and* $e_2$*, we define the* $\leq$ *relation as follows:*

1. $e_1 = e_2$ *iff* $(e_1.t = e_2.t) \wedge (e_1.c = e_1.c) \wedge (e_1.v = e_1.v)$
2. $e_1 < e_2$ *iff* $(e_1.t < e_2.t) \vee ((e_1.t = e_2.t) \wedge (e_1.c < e_2.c))$

We assume that a lexicographic ordering exists among the pairs (attribute, value) in the dataset.

**Definition 4** *A* sequence (or event sequence) *is an ordered set of events* $S = \{e_0, e_1, ..., e_k\}$*, where* $e_i < e_{i+1}$*, for all i = 0,...,k-1. Obviously,* $|S| = k + 1$*. Note that different events with the same temporal unit can belong to the same sequence. Also, the same events with different temporal unit associated can belong to the same sequence. But nevertheless, in any sequence there will exist two or more pairs (attribute, value) associated to the same temporal unit. In other words, an attribute can not take two different values in the same instant.*

**Definition 5** *Let* $U_{tmin}$ *be the minimal dimensional value associated to the sequence* $S$*. In other words,* $U_{tmin} = min\{e_i.t\}$*, for* $e_i \in S$*. If* $U_{tmin} = 0$*, we say that* $S$ *is a* normalized sequence*. Note that any non-normalized sequence can be transformed into a normalized one through a normalization function.*

***Example 1*** *Let* $S_1 = \{(0,0,0), (1,0,0), (3,0,2)\}$*, and* $S_2 = \{(0,0,3), (1,0,3), (3,0,5)\}$ *be two sequences.* $S_1$ *is a normalized sequenced, since it has the minimal value equal to 0 for the temporal dimension. But* $S_2$ *is not a normalized sequence, because its minimal value is not equal to zero. However, we can normalize* $S_2$ *by subtracting its minimal value (*$U_{tmin} = 3$*) from the temporal values as follows:* $S'_2 = \{(0,0,3-3), (1,0,3-$

$3), (3, 0, 5-3)\}$, *resulting in the normalized sequence* $S'_2 = \{(0, 0, 0), (1, 0, 0), (3, 0, 2)\}$.

*Let* $U_{tmax}$ *be the maximal dimensional value associated to the sequence* $S$. *This value indicates the maximum distance amongst the events belonging to the normalized sequence* $S$. *In other words,* $U_{tmax} = e_{k+1}.t$, *where* $|S| = k + 1$. *From both, confidence and complexity points of view [15], this value will be always less or equal than a user-defined parameter called* $maxspan$, *denoted by* $w$.

Here we introduce the definitions that extend the previous problem definition. The extension is based in the relaxation of the counting method in which the concept *approximate temporal distance* amongst events is taking into account. Instead of extract rigid patterns like *"it is likely that the event 1 occurs one temporal unit after the occurrence of the event 2"*, we will obtain flexible patterns like *"it is likely that the event 1 occurs approximately one temporal unit after the occurrence of the event 2"*. The term *approximately* implies that the event 2, for example, could happened 0, 1 or 2 temporal units after. This relaxation let us extract useful patterns from datasets where uncertainty and imprecision are present.

Our approach consists in the use of a fuzzy set [25] in the counting method. This user-defined fuzzy set represent the meaning of the linguistic term *"approximately equal to"*, defined by a 0-centered fuzzy number with membership function $\mu_z$.

Moreover, if the user wants to obtain sequences defined over a temporal unit that is different from the temporal unit specified in the dataset, the algorithm allows to set a parameter g that represents the granularity of the temporal dimension. For example a user can obtain sequences expressed in weeks from a dataset expressed in days setting the parameter $g = 7$.

Having said that, next we present the formulae for computing the support of a sequence.

**Definition 6** *Given an event* $e_i$, *we define the occurrence degree of its type event associated* $(e_i.c, e_i.v)$ *in the instant "x" as:*

$$Occ_x\left(e_i.c, e_i.v\right) = \max\left\{\mu_z\left(gx - e.t\right)\right\} \quad \forall e \in D \quad / \quad e.c = e_i.c \wedge e.v = e_i.v \tag{1}$$

**Definition 7** *The* support *(frequency) of a sequence* $S = (e_1, e_2, \ldots, e_n)$ *over the dataset D is defined as:*

$$Sop(S) = \frac{1}{|D|} \sum_x \left(\min\left\{Occ_{x+g\times e_i.t}(e_i.c, e_i.v)\right\}_{i=1\ldots n}\right) \tag{2}$$

*where the Occ function applied to the first event of the sequence* $e_1$ *always uses a singleton as its* $\mu_z$ *membership function while the other events use the* $\mu_z$ *specified by the user.*

**Definition 8** *A* general sequence *is a sequence whose support value has been calculated with the previous formula.*

**Definition 9** *A* frequent sequence *is a normalized sequence whose support is greater or equal than a user-specified threshold called minimum support* $minsup$.

Given a dataset and $minsup$, the goal of general temporal pattern (or general sequence) mining is to determine in the dataset all the frequent general sequences whose support are greater than or equal to $minsup$.

## 3. The $TSET^F$ Algorithm

In this section we describe the algorithm that we propose for the mining of all frequent sequences from a dataset. $TSET^f$ follows the same basic principles as most apriori-based algorithms [2]. Frequent sequence mining is an iterative process, and the focus is on a *level-wise* pattern generation. At the beginning, all frequent 1-sequences (frequent events) are found, these are used to generate frequent 2-sequences, then 3-sequences are found using frequent 2-sequences, and so one. In other words, (k+1)-sequences are generated only after all k-sequences have been generated. On each cycle, the *downward closure* property is used to prune the search space. This property, also called anti-monotonicity property, indicates that if a sequence is infrequent, then all super-sequence must also be infrequent. Figure 1 outlines a generalized frequent sequences mining algorithm.

algorithm $TSET^f$(dataset $D$, minsup $m$, maxspan $w$)
  begin
    tree.init($D$, $m$, $w$ );
    tree.getSequences($D$, $m$, $w$);
    Output(tree);
  end;

**Figure 1.** $TSET^f$ **Algorithm**

The first step consists in the creation and initialization of the data structure. The "init" method that outlines Figure 2 returns the frequent 1-sequences (frequent events) found in the dataset. It uses a Binary Search Tree auxiliar structure to compute effectively the support of the events presented in the dataset. The inorder traversal of this structure produces the 1-sequences lexicographically ordered. The "getSequences" method (see Figure 3) obtain iteratively the frequent k-sequences using a queue structure. After generating the candidate set, which consists in copying all the events to the right of a given event in the newNode, and

```
procedure init(tree, dataset D, minsup m, maxspan w)
  begin
    BST auxiliarTree = ∅;
    foreach record i in D
      foreach non-dimensional attribute j in record
        auxiliarTree.insert(D[i].c[j], D[i].c[j].v, D[i].c[0]);
    tree.insert(auxiliarTree.traverseInOrder(m));
  end;
```

**Figure 2. The code for the method init**

```
procedure getSequences(tree, dataset D, minsup m, maxspan w)
  begin
    Queue Q = ∅;
    Q.push(tree.root);
    while (Q ≠ ∅)
      begin
        act = Q.pop();
        foreach node n in act
          newNode = n.getCandidates();
          newNode.evaluateSupport(D, w);
          newNode.pruningInFrequent(m);
          if(newNode ≠ ∅)
            begin
              n.child = newNode;
              Q.push(newNode);
            end;
        end;
  end;
```

**Figure 3. The code for the method getSequences**

pruning the infrequent sequences, the new node is pushing into the queue for the next kth-*cycle.*

We want to highlight that the unique difference between $TSET$ and $TSET^f$ is the evaluateSupport() method which use the formulae shown in equation drawn in Definition 7.

## 4. Example

We will use a synthetic dataset to illustrate the fuzzy approach proposed in this work. Suppose that, after some preprocessing, we have obtained the dataset D shown in Table 1, where T represents the temporal attribute expressed in *days* and $C_1$ and $C_2$ two descriptive attributes.

To test the $TSET^f$ algorithm we will try to find meaningful sequences in the datasets working with days and weeks and using strict and approximate distance computations. We assume that the user set the linguistic term "approximately equal to 0" to the fuzzy number shown in Figure 4, and *maxspan* at 2 temporal units.

**Table 1. An example dataset**

| T | $C_1$ | $C_2$ | T | $C_1$ | $C_2$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 16 | 1 | 1 |
| 1 | 1 | 2 | 17 | 0 | 1 |
| 2 | 2 | 1 | 18 | 1 | 2 |
| 3 | 0 | 1 | 19 | 0 | 1 |
| 4 | 1 | 0 | 20 | 2 | 1 |
| 5 | 1 | 1 | 21 | 1 | 2 |
| 6 | 1 | 0 | 22 | 1 | 1 |
| 7 | 1 | 1 | 23 | 1 | 2 |
| 8 | 1 | 2 | 24 | 2 | 1 |
| 9 | 2 | 1 | 25 | 1 | 1 |
| 10 | 1 | 2 | 26 | 0 | 0 |
| 11 | 0 | 1 | 27 | 1 | 1 |
| 12 | 2 | 1 | 28 | 2 | 1 |
| 13 | 1 | 1 | 29 | 1 | 0 |
| 14 | 0 | 0 | 30 | 0 | 1 |

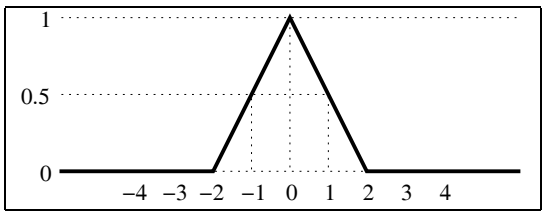


**Figure 4. The user-defined fuzzy number associated with the linguistic term "approximately equal to 0"**

- $g = 1$ (days)

  Using $g = 1$ we try to find frequent sequences expressed in the same temporal units of the D dataset.

  When a rigid distance computing is used, the method can not distinguish some imprecise temporal associations. Setting $minsup = 0.3$, the algorithm extracted the following sequences:

  $S_1 = ((0, 1, 0)), sup = 0.516129$

  $S_2 = ((0, 1, 0), (1, 1, 1)), sup = 0.354839$

  $S_3 = ((1, 1, 0)), sup = 0.580645$

  $S_4 = ((1, 1, 0), (0, 1, 1)), sup = 0.322581$

  $S_5 = ((1, 1, 0), (1, 1, 2)), sup = 0.322581$

  If we use a flexible distance computing, the algorithm extracts frequent sequences that the classical model

does not discover. For $minsup = 0.4$, the discovered sequences are:

$S_1 = ((0,1,0)), sup = 0.516129$

$S_2 = ((1,1,0)), sup = 0.580645$

$S_3 = ((1,1,0),(0,1,1)), sup = 0.435484$

$S_4 = ((0,1,0),(1,1,1)), sup = 0.435484$

$S_5 = ((1,1,0),(0,1,2)), sup = 0.419355$

$S_6 = ((1,1,0),(1,1,1)), sup = 0.403226$

$S_7 = ((1,1,0),(1,1,2)), sup = 0.435484$

$S_8 = ((1,1,0),(0,1,1),(1,1,2), sup = 0.403226$

We can observe that with a greater $minsup$ value, we obtain more sequences with a greater support. This is an very interesting result because with lower computational effort (the search space is lower due to the tree prune) we obtain more interesting results.

- $g = 7$ (weeks)

  Using $g = 7$ we try to find frequent sequences expressed in weeks.

  When a rigid distance computing is used, we need to set a lower $minsup$ value to discover some frequent sequences. With $minsup = 0.3$ the algorithm extracts the following sequences:

  $S_1 = ((0,1,0)), sup = 0.516129$

  $S_2 = ((1,1,0)), sup = 0.580645$

  With $minsup = 0.2$:

  $S_1 = ((0,0,0)), sup = 0.258065$

  $S_2 = ((0,1,0)), sup = 0.516129$

  $S_3 = ((1,1,0)), sup = 0.580645$

  $S_4 = ((0,1,0),(1,1,0)), sup = 0.225806$

  $S_5 = ((0,1,0),(1,1,1)), sup = 0.258065$

  $S_6 = ((1,1,0),(1,1,1)), sup = 0.225806$

  And setting $minsup = 0.15$:

  $S_1 = ((0,0,0)), sup = 0.258065$

  $S_2 = ((0,2,0)), sup = 0.193548$

  $S_3 = ((0,1,0)), sup = 0.516129$

  $S_4 = ((1,0,0)), sup = 0.193548$

  $S_5 = ((1,1,0)), sup = 0.580645$

  $S_6 = ((1,2,0)), sup = 0.193548$

  $S_7 = ((0,0,0),(1,1,0)), sup = 0.16129$

  $S_8 = ((0,1,0),(0,1,1)), sup = 0.16129$

  $S_9 = ((0,1,0),(1,1,0)), sup = 0.225806$

  $S_{10} = ((0,1,0),(1,1,1)), sup = 0.258065$

  $S_{11} = ((0,1,0),(1,1,2)), sup = 0.193548$

  $S_{12} = ((0,1,0),(1,2,0)), sup = 0.193548$

  $S_{13} = ((0,2,0),(1,1,0)), sup = 0.193548$

  $S_{14} = ((1,1,0),(0,1,1)), sup = 0.193548$

  $S_{15} = ((1,1,0),(0,1,2)), sup = 0.16129$

  $S_{16} = ((1,1,0),(1,1,1)), sup = 0.225806$

  $S_{17} = ((1,1,0),(1,1,2)), sup = 0.193548$

  Using the fuzzy approach, the support of the sequences associated to approximated temporal associations is increased, leading to a clear differentiation between meaningful and less important sequences. With a higher $minsup$ value, we obtain more significative sequences. With $minsup = 0.3$:

  $S_1 = ((0,1,0)), sup = 0.516129$

  $S_2 = ((1,1,0)), sup = 0.580645$

  $S_3 = ((0,1,0),(1,1,0)), sup = 0.370968$

  $S_4 = ((0,1,0),(1,1,1)), sup = 0.33871$

  $S_5 = ((1,1,0),(0,1,1)), sup = 0.306452$

  $S_6 = ((1,1,0),(1,1,1)), sup = 0.33871$

## 5. Conclusions

In this paper we have presented an extension of an algorithm which extract temporal patterns from datasets. The extension is based on the relaxation of the counting method using fuzzy techniques. Using a flexible distance computing, the algorithm extracts frequent sequences that the classical model does not discover.

If we set the parameter $\mu_z$ to a singleton 0-centered and $g = 1$, the behavior of the $TSET^f$ algorithm is exactly the same of $TSET$, so we can say that the former is a generalization of the last one.

## References


[1] C. C. Aggarwal. Towards long pattern generation in dense databases. *SIGKDD Explorations*, 3(1):20–26, 2001.

[2] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proc. of the ACM SIGMOD Int. Conf. on Management of Data, Washington, D.C., May 26-28, 1993*, pages 207–216. ACM Press, 1993.

[3] R. Agrawal and R. Srikant. Mining sequential patterns. In P. S. Yu and A. L. P. Chen, editors, *Proc. of the 11th Int. Conf. on Data Engineering, March 6-10, 1995, Taipei, Taiwan*, pages 3–14. IEEE Computer Society, 1995.

[4] J. M. Ale and G. H. Rossi. An approach to discovering temporal association rules. In *Proc. of the 2000 ACM Symposium on Applied Computing, Villa Olmo, Via Cantoni 1, 22100 Como, Italy, March 19-21, 2000*, pages 294–300. ACM, 2000.

[5] R. J. Bayardo. Efficiently mining long patterns from databases. In L. M. Haas and A. Tiwary, editors, *Proc. of the ACM SIGMOD Int. Conf. on Management of Data (SIGMOD 1998), June 2-4, 1998, Seattle, Washington, USA*, pages 85–93. ACM Press, 1998.

[6] F. Berzal, J. C. Cubero, N. Marín, and J. M. Serrano. TBAR: An efficient method for association rule mining in relational databases. *Data & Knowledge Engineering*, 37:47–64, 2001.

[7] C. Bettini, X. S. Wang, and S. Jajodia. Testing complex temporal relationships involving multiple granularities and its application to data mining. In *Proc. of the 15th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, 1996, Montreal, Canada*, pages 68–78. ACM Press, 1996.

[8] F. Coenen, G. Goulbourne, and P. Leng. Tree structures for mining association rules. *Data Mining and Knowledge Discovery*, 8:25–51, 2004.

[9] U. Fayyad, G. Piatetky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AIMagazine*, 17(3):37–54, 1996.

[10] L. Feng, J. X. Yu, H. Lu, and J. Han. A template model for multidimensional inter-transactional association rules. *The VLDB Journal*, 11:153–175, 2002.

[11] F. Guil, A. Bosch, and R. Marín. TSET: An algorithm for mining frequent temporal patterns. In *Proc. of the First Int. Workshop on Knowledge Discovery in Data Streams, to be held in conjunction with the 15th ECML + 8th PKDD Conference (Accepted). Pisa, Italy*, 2004.

[12] C. H. Lee, C. R. Lin, and M. S. Chen. On mining general temporal association rules in a publication database. In N. Cercone, T. Y. Lin, and X. Wu, editors, *Proc. of the 2001 IEEE Int. Conf. on Data Mining, 29 November - 2 December 2001, San Jose, California, USA*, pages 337–344. IEEE Computer Society, 2001.

[13] J. W. Lee, Y. J. Lee, H. K. Kim, B. H. Hwang, and K. H. Ryu. Discovering temporal relation rules mining from interval data. In *Proc. of the 1st EurAsian Conf. on Information and Communication Technology (Eurasia-ICT 2002), Shiraz, Iran, October 29-31, 2002*, volume 2510 of *Lecture Notes in Computer Science*, pages 57–66. Springer, 2002.

[14] Y. Li, P. Ning, X. S. Wang, and S. Jajodia. Discovering calendar-based temporal association rules. *Data & Knowledge Engineering*, 44:193–218, 2003.

[15] H. Lu, L. Feng, and J. Han. Beyond intra-transaction association analysis: Mining multi-dimensional inter-transaction association rules. *ACM Transactions on Information Systems (TOIS)*, 18(4):423–454, 2000.

[16] H. Lu, J. Han, and L. Feng. Stock movement and n-dimensional inter-transaction association rules. In *Proc. of the Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'98), Seattle, Washington, June 1998*, pages 12:1–12:7, 1998.

[17] H. Mannila. Local and global methods in data mining: Basic techniques and open problems. In P. Widmayer, F. Triguero, R. Morales, M. Hennessey, S. Eidenbenz, and R. Conejo, editors, *In Proc. of the 29th Int. Colloquium on Automata, Languages and Programming (ICALP 2002), Malaga, Spain, July 8-13, 2002*, volume 2380 of *Lecture Notes in Computer Science*, pages 57–68. Springer, 2002.

[18] H. Mannila, H. Toivonen, , and A. I. Verkamo. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1(3):259–289, 1997.

[19] C. Ordonez, C. A. Santana, and L. de Braal. Discovering interesting association rules in medical data. In D. Gunopulos and R. Rastogi, editors, *Proc. of the ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, Dallas, Texas, USA, May 14, 2000*, pages 78–85, 2000.

[20] B. Özden, S. Ramaswamy, and A. Silberschatz. Cyclic association rules. In *Proc. of the 14th Int. Conf. on Data Engineering, February 23-27, 1998, Orlando, Florida, USA*, pages 412–421. IEEE Computer Society, 1998.

[21] A. K. Pani. Temporal representation and reasoning in artificial intelligence: A review. *Mathematical and Computer Modelling*, 34:55–80, 2001.

[22] J. F. Roddick and M. Spiliopoulou. A survey of temporal knowledge discovery paradigms and methods. *IEEE Transactions on Knowledge and Data Engineering*, 14(4):750–767, 2002.

[23] A. K. H. Tung, H. Lu, J. Han, and L. Feng. Breaking the barrier of transactions: Mining inter-transaction association rules. In *Proc. of the 5th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, August 15-18, 1999, San Diego, CA, USA*, pages 297–301. ACM Press, 1999.

[24] A. K. H. Tung, H. Lu, J. Han, and L. Feng. Efficient mining of intertransaction association rules. *IEEE Transactions on Knowledge and Data Engineering*, 15(1):43–56, 2003.

[25] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[26] Z. H. Zhou. Three perspectives of data mining (book review). *Artificial Intelligence*, 143:139–146, 2003.