Discovering generalized episodes using minimal occurrences

Heikki Mannila and Hannu Toivonen
University of Helsinki
Department of Computer Science
P.O. Box 26, FIN-00014 Helsinki, Finland
Heikki.Mannila@cs.Helsinki.FI, Hannu.Toivonen@cs.Helsinki.FI

Abstract

Sequences of events are an important special form of data that arises in several contexts, including telecommunications, user interface studies, and epidemiology. We present a general and flexible framework of specifying classes of *generalized episodes*. These are recurrent combinations of events satisfying certain conditions. The framework can be instantiated to a wide variety of applications by selecting suitable primitive conditions. We present algorithms for discovering frequently occurring episodes and episode rules. The algorithms are based on the use of minimal occurrences of episodes; this makes it possible to evaluate confidences of a wide variety of rules using only a single analysis pass. We present empirical results on the behavior of the algorithms on events stemming from a WWW log.

Introduction

Sequences of events are a common form of data that can contain important knowledge to be discovered. Examples of such data are telecommunications network alarms, user interface actions, crimes committed by a person, occurrences of recurrent illnesses, etc. Recently, interest in knowledge discovery from sequences of events has increased: see, e.g., (Dousson, Gaborit, & Ghallab 1993; Laird 1993; Wang *et al.* 1994; Morris, Khatib, & Ligozat 1995; Bettini, Wang, & Jajodia 1996).

In a previous paper (Mannila, Toivonen, & Verkamo 1995) we showed how sequences of events can be analyzed by locating frequently occurring *episodes* from them. An episode is a combination of events with a partially specified order; it occurs in a sequence, if there are occurrences of the events in an order consistent with the given order, within a given time bound.

In a telecommunication application, the rules found using the methods of (Mannila, Toivonen, & Verkamo 1995) have proven to be useful and they have been integrated in alarm handling software (Hätönen *et al.* 1996). However, application studies both in the telecommunications domain and elsewhere have shown that there is a need for extensions to the methods.

In this paper we study what types of episodes can be efficiently discovered from long sequences of events. We present a simple and powerful framework for building episodes out of predefined predicates, and describe how such episodes can be discovered efficiently. The framework allows one to express arbitrary unary conditions on the individual events, and to pose binary conditions on the pairs of events, giving one the possibility to exactly target the types of event combinations that are interesting in the applications.

The algorithms described in the paper are based on *minimal occurrences* of episodes. In addition to being simple and efficient, this formulation has the advantage that the confidences and frequencies of rules with different time bounds can be obtained quickly, i.e., there is no need to rerun the analysis if one only wants to modify the time bounds. In case of complicated episodes, the time needed for recognizing the occurrence of an episode can be significant; the use of stored minimal occurrences of episodes eliminates unnecessary repetition of the recognition effort.

Episodes: patterns in event sequences

We use a fairly standard way of modeling events in time. Given the set $R = \{A_1, \ldots, A_m\}$ of *event attributes* with domains $D_{A_1}, \ldots, D_{A_m}$, an *event* e over R is a $(m+1)$-tuple $(a_1, \ldots, a_m, t)$, where $a_i \in D_{A_i}$ and t is a real number, the *time* of e. We refer to the time of e by $e.T$, and to an attribute $A \in R$ of e by $e.A$. An *event sequence* $\mathcal{S}$ is a collection of events over R, i.e., a relation over $R \cup \{T\}$, where the domain of attribute T is the set of real numbers.

An *episode* P on variables $\{x_1, \ldots, x_k\}$, denoted $P(x_1, \ldots, x_k)$, is a conjunction

$$\bigwedge_{i=1}^{k} \varphi_i(y_i, z_i),$$

where $y_i, z_i \in \{x_1, \ldots, x_k\}$ are event variables, and each conjunct $\varphi_i(x, y)$ has one of the forms $\alpha(x.A)$, $\beta(x.A, y.B)$, or $x.T \leq y.T$. Here A and B are event attributes, α is a predefined unary predicate on D_A, and β is a predefined binary predicate on $D_A \times D_B$. We assume that the available unary predicates include testing for equality with a constant, and that the binary predicates include equality. The *size* $|P|$ of an episode P is the number of conjuncts not involving time in P.

Example 1 In the telecommunications domain, event attributes are, e.g., *type*, *module*, and *severity*, indicating the type of alarm, the module that sent the alarm, and the severity of the alarm, respectively. An episode might look as follows:

$$x.type = 2356 \ \wedge \ y.type = 7401,$$

or

$$\begin{aligned} &x.type = 2356 \ \wedge \ y.type = 7401 \ \wedge \\ &x.T \leq y.T \ \wedge \ x.module = y.module. \end{aligned}$$

The first episode indicates that two alarms of type 2356 and 7401 have to occur, whereas the second says that alarm 2356 has to precede 7401 and that the alarms have to come from the same module. Specification of, e.g., this condition was not possible in the framework of (Mannila, Toivonen, & Verkamo 1995).

The episode

$$\begin{aligned} &x.type = 2356 \ \wedge \ y.type = 7401 \ \wedge \\ &neighbor(x.module, y.module) \end{aligned}$$

captures the situation when alarms of type 2356 and 7401 are sent from adjacent modules. □

Example 2 In analyzing a WWW log, the events have attributes *page* (the accessed page), *host* (the accessing host), and time. An example episode is

$$x.page = p_1 \ \wedge \ y.page = p_2 \ \wedge \ x.host = y.host,$$

indicating accesses to p_1 and p_2 from the same host. □

An episode $P(x_1, \ldots, x_k)$ *occurs* in a sequence of events $\mathcal{S} = (e_1, \ldots, e_n)$, *at interval* $[t, t']$, if there are disjoint events $e_{j_1}, \ldots, e_{j_k}$ such that $P(e_{j_1}, \ldots, e_{j_k})$ is true,[1] and $t \leq min_i\{e_{j_i}.T\}$ and $t' \geq max_i\{e_{j_i}.T\}$.

Note that one could write an SQL query that tests whether an episode of the above form occurs in a sequence, when the sequence is represented as a relation over $R \cup \{T\}$. The evaluation of such a query would, however, be quite inefficient, as the sequence form of the data could not easily be used. The recent work on sequence data in databases (see (Seshadri, Livny, & Ramakrishnan 1996)) provides interesting openings towards the use of database techniques in the processing of queries on sequences.

An occurrence of an episode P at $[t, t']$ is *minimal* if P does not occur at any proper subinterval $[u, u'] \subset [t, t']$. The *set of (intervals of) minimal occurrences* of an episode P is denoted by $mo(P)$:

$$mo(P) = \{[t, t'] \mid [t, t'] \text{ is a minimal occurrence of } P\}.$$

An *episode rule* is an expression $P\,[V] \Rightarrow Q\,[W]$, where P and Q are episodes, and V and W are real numbers. The informal interpretation of the rule is that if episode P has a minimal occurrence at interval $[t, t']$ with $t' - t \leq V$, then episode Q occurs at interval $[t, t'']$ for some t'' such that $t'' - t \leq W$.[2]

Example 3 An example rule in the WWW log is

$$\begin{aligned} &x.page = p_1 \ \wedge \ y.page = p_2 \ \wedge \\ &x.host = y.host\,[60] \\ \Rightarrow \ &x.page = p_1 \ \wedge \ y.page = p_2 \ \wedge \\ &z.page = p_3 \ \wedge \ x.host = y.host \ \wedge \\ &y.host = z.host\,[120] \end{aligned}$$

expressing that if some host accesses pages p_1 and p_2 within a minute, page p_3 is likely to be accessed from the same host within two minutes. □

An episode $P(x_1, \ldots, x_k)$ is *serial*, if P includes conjuncts enforcing a total order between the x_i's. The episode is *parallel*, if there are no conditions on the relative order of the events.

The *frequency* $freq(P)$ of an episode P in a given event sequence $\mathcal{S}$ is defined as the number of minimal occurrences of P in the sequence $\mathcal{S}$, $freq(P) = |mo(P)|$. Given a *frequency threshold* min_fr, an episode P is *frequent* if $freq(P) \geq min\_fr$.

The *episode rule discovery task* can now be stated as follows. Given an event sequence $\mathcal{S}$, a class $\mathcal{E}$ of episodes, and time bounds V and W, find all frequent episode rules of the form $P\,[V] \Rightarrow Q\,[W]$, where $P, Q \in \mathcal{E}$.

To make episode rule discovery feasible, the class of episodes has to be restricted somehow. We consider the restricted but interesting task of discovering episodes from the following classes $\mathcal{E}_S(\Gamma, \Delta)$ and $\mathcal{E}_P(\Gamma, \Delta)$. The class $\mathcal{E}_S(\Gamma, \Delta)$ consists of serial episodes with unary predicates from Γ and binary predicates from Δ. The class $\mathcal{E}_P(\Gamma, \Delta)$ is defined to consist of parallel episodes, with predicates from Γ and Δ.

[1] For reasons of brevity we omit the standard formal definition of satisfaction of a formula, and just use the notation $P(e_{j_1}, \ldots, e_{j_k})$ to indicate this.

[2] There is a number of variations for the relationship between the intervals; e.g., rules that point backwards in time can be defined in a similar way.

Differences to the previous model The episodes described in (Mannila, Toivonen, & Verkamo 1995) were based on conditions on the event types. That is, of our examples only the first episode of Example 1 would be representable within that framework. In the applications it is necessary to be able to state both more relaxed and more restricted episode specifications by having conditions related to, e.g., the network topology. That is, binary predicates are necessary. Also, there was only one fixed time bound associated with a rule. In the applications it is preferable to have two bounds, one for the left-hand side and one for the whole rule, such as "if A and B occur within 15 seconds, then C follows within 30 seconds".

One notable difference in the algorithms is that here we are able to give methods that can be used to compute the confidences of rules of the above form for various values of the time bounds; the previous methods had to make one pass through the data for each time bound. There are also some smaller technical differences, e.g., the exact definition of the confidence of a rule is somewhat changed (and is now more natural, it seems to us).

The work most closely related to ours is perhaps (Srikant & Agrawal 1996). They search in multiple sequences for patterns that are similar to the serial episodes of (Mannila, Toivonen, & Verkamo 1995) with some extra restrictions and an event taxonomy. Our methods can be extended with a taxonomy by a direct application of the similar extensions to association rules. Also, our methods can be applied on analyzing several sequencies; there is actually a variety of choices for the definition of frequency of an episode in a set of sequencies.

There are also some interesting similarities between the discovery of frequent episodes and the work done on inductive logic programming (see, e.g., (Muggleton 1992)); a noticeable difference is caused by the sequentiality of the underlying data model, and the emphasis on time-limited occurrences. Similarly, the problem of looking for one occurrence of an episode can be viewed as a constraint satisfaction problem.

Finding minimal occurrences of frequent episodes

In this section we describe the algorithms used to locate the minimum occurrences of frequent episodes from the classes $\mathcal{E}_P$ and $\mathcal{E}_S$. The algorithms are based on the idea of first locating minimal occurrences of small episodes, and using this information to generate candidates for possibly frequent larger episodes.

An episode P is *simple*, if it includes no binary predicates. Recognition of simple episodes turns out to be considerably easier than for arbitrary episodes, as is shown by the following two theorems.

Theorem 4 Finding whether a simple serial or parallel episode P has an occurrence in an event sequence $\mathcal{S}$ can be done in time $|P||\mathcal{S}|$. □

Theorem 5 Finding whether a serial or parallel episode P has an occurrence in an event sequence $\mathcal{S}$ is an NP-complete problem. □

One should not be discouraged by the NP-completeness result, however; the situations used in the reduction are highly contrived. It seems that in practice episodes similar to the ones in our examples can be recognized and discovered fast.

We move to the discovery algorithm for simple episodes. Given an episode $P = \bigwedge_{i=1}^{k} \varphi_i(y_i, z_i)$, a *subepisode* P_I of P determined by a set $I \subseteq \{1, \ldots, k\}$ is simply $P_I = \bigwedge_{i \in I} \varphi_i(y_i, z_i)$. The basic properties of simple episodes are given in the following lemmas.

Lemma 6 (i) If an episode P is frequent in an event sequence $\mathcal{S}$, then all subepisodes P_I are frequent. (ii) If $[t, t'] \in mo(P)$, then P_I occurs in $[t, t']$ and hence there is an interval $[t_I, t'_I] \in mo(P_I)$ such that $t \le t_I \le t'_I \le t'$. □

Lemma 7 Let P be a simple serial episode of size k, and let $[t, t'] \in mo(P)$. Then there are subepisodes P_1 and P_2 of P of size $k-1$ such that for some $t_1 \in [t, t'[$ and $t_2 \in]t, t']$ we have $[t, t_1] \in mo(P_1)$ and $[t_2, t'] \in mo(P_2)$. □

Note that Lemma 7 does not hold for general episodes: a minimal occurrence of a general episode does not necessarily start with a minimal occurrence of a subepisode.

Lemma 8 Let P be a simple parallel episode of size k, and let $[t, t'] \in mo(P)$. Then there are subepisodes P_1 and P_2 of P of size $k-1$ such that for some $t_1, t_2, t'_1, t'_2 \in [t, t']$ we have $[t_1, t'_1] \in mo(P_1)$ and $[t_2, t'_2] \in mo(P_2)$, and furthermore $t = \min\{t_1, t_2\}$ and $t' = \max\{t'_1, t'_2\}$. □

We use the same algorithm skeleton as in the discovery of association rules (Agrawal & Srikant 1994; Mannila, Toivonen, & Verkamo 1994). Namely, having found the set L_k of frequent simple episodes of size k, we form the set C_{k+1} of *candidate episodes* of size $k+1$, i.e., episodes whose all subepisodes are frequent, and then find out which candidate episodes $P \in C_{k+1}$ are really frequent by forming the set $mo(P)$.

Algorithm 9, below, discovers all frequent simple serial episodes. The discovery of simple parallel episodes is similar; the details of steps 9 and 10 are a bit different.

Algorithm 9 Discovery of frequent simple serial episodes.
Input: An event sequence $\mathcal{S}$, unary predicates Γ and binary predicates Δ, and a frequency threshold *min_fr*.
Output: All frequent simple episodes from $\mathcal{E}_S(\Gamma, \Delta)$ (and their minimal occurrences).
Method:

1. $C_1 :=$ the set of episodes of size 1 in $\mathcal{E}_S(\Gamma, \Delta)$;
2. **for** all $P \in C_1$ compute $mo(P)$;
3. $L_1 := \{P \in C_1 \mid P$ is frequent$\}$;
4. $i := 1$;
5. **while** $L_i \neq \emptyset$ **do**
6. $C_{i+1} := \{P \mid P \in \mathcal{E}_S(\Gamma, \Delta)$, all subepisodes of P are in $L_i\}$;
7. $i := i + 1$;
8.   **for** all $P \in C_i$ **do**
9.   select subepisodes P_1 and P_2;
10.   compute $mo(P)$ from $mo(P_1)$ and $mo(P_2)$;
11.   **od**;
12. $L_i := \{P \in C_i \mid P$ is frequent$\}$;
13. **od**;
14. **for** all i and all $P \in L_i$, output P and $mo(P)$;

Finding the minimal occurrences For simple serial episodes, the two subepisodes are selected on line 9 so that P_1 contains all events except the last one and P_2 in turn contains all except the first one. P_1 and P_2 also contain all the predicates that apply on the events in the episode. The minimal occurrences of P are then computed on line 10 by

$$mo(P) = \{[t, u'] \mid \text{there are } [t, t'] \in mo(P_1) \text{ and } [u, u'] \in mo(P_2) \text{ such that } t < u, t' < u', \text{ and } [t, u'] \text{ is minimal}\}.$$

Note the correspondence to Lemma 7.

For simple parallel episodes, the subepisodes P_1 and P_2 contain all events except one; the omitted events must be different. Again, P_1 and P_2 contain all the applicable predicates. See Lemma 8 for the idea of how to compute the minimal occurrences of P. To optimize the efficiency, $|mo(P_1)|+|mo(P_2)|$ should be minimized.

The minimal occurrences of a candidate episode P can be found in a linear pass over the minimal occurrences of the selected subepisodes P_1 and P_2. The time required for one candidate is thus $O(|mo(P_1)| + |mo(P_2)| + |mo(P)|)$, which is $O(n)$, where n is the length of the event sequence.

While minimal occurrences of episodes can be located quite efficiently from minimal occurrences, the size of the data structures may be even larger than the original database, especially in the first couple of iterations. A solution is to use in the beginning other pattern recognition methods.

Finding general episodes The recognition problem for general episodes is considerably harder than that for simple episodes. The difficulty is caused by the failure of Lemma 7. Consider as an example the event sequence

$$(page_1, host_1, 1), (page_1, host_2, 2), (page_2, host_2, 3), (page_2, host_1, 4), (page_3, host_1, 5),$$

and the episode P

$$x.host = y.host \ \wedge \ y.host = z.host.$$

The minimal occurrences of the two (actually equivalent) subepisodes $x.host = y.host$ and $y.host = z.host$ of P are at $[2, 3]$, whereas $mo(P) = \{[1, 5]\}$.

While the minimal occurrences of a general episode cannot be built as easily from the minimal occurrences of subepisodes as for simple episodes, the occurrence of subepisodes is still a necessary condition for the occurrence of the whole episode (Lemma 6). We use this property to implement a simple exhaustive search method for finding minimal occurrences of general episodes; the use of minimal occurrences of subepisodes guarantees that the exhaustive search has to be applied only to small slices of the long event sequence.

Finding minimal occurrences of general episodes In line 9 of Algorithm 9 choose incomparable subepisodes P_1 and P_2 of P such that $|mo(P_1)| + |mo(P_2)|$ is as small as possible. Then, for each pair of intervals $[t_1, t_1'] \in mo(P_1)$ and $[t_2, t_2'] \in mo(P_2)$ such that $\max(t_1', t_2') - \min(t_1, t_2) \leq W$ search for a minimal occurrence of P in the time interval $[\max(t_1', t_2') - W, \min(t_1, t_2) + W]$. Here W is the upper bound on the length of minimal occurrences. The search can be done using basically exhaustive search, as the sizes of episodes and the number of events in such small intervals are small. We omit the details.

A problem similar to the computation of frequencies occurs in the area of active databases. There triggers can be specified as composite events, somewhat similar to episodes. In (Gehani, Jagadish, & Shmueli 1992) it is shown how finite automata can be constructed from composite events to recognize when a trigger should be fired. This method is not practical for episodes since the deterministic automata could be very large.

Finding confidences of rules

In this section we show how the information about minimal occurrences of frequent episodes can be used to

min_fr	Number of frequent episodes and informative rules							
	Time bounds U (s)							
	$15, 30$		$30, 60$		$60, 120$		$15, 30, 60, 120$	
50	1131	617	2278	1982	5899	7659	5899	14205
100	418	217	739	642	1676	2191	1676	3969
250	111	57	160	134	289	375	289	611
500	46	21	59	49	80	87	80	138

Table 1: Experimental results: number of episodes and rules

obtain confidences for various types of episode rules without looking at the data again.

Recall that an episode rule is an expression $P\,[V] \Rightarrow Q\,[W]$, where P and Q are episodes, and V and W are real numbers. To find such rules, first note that for the rule to be interesting, also the episode $P \wedge Q$ has to be frequent. So rules of the above form can be enumerated by looking at all frequent episodes $M = \wedge_{i \in I} \alpha_i$, and then looking at all partitions of I as $I = J \cup K$, and forming the left and right-hand sides as subepisodes: $P = M_J$ and $Q = M_K$. The evaluation of the confidence of the rule $P\,[V] \Rightarrow Q\,[W]$ can be done in one pass through the structures $mo(P)$ and $mo(Q)$, as follows.

For each $[t, t'] \in mo(P)$ with $t' - t \leq V$, locate the minimal occurrence $[s, s']$ of Q such that $t \leq s$ and $[s, s']$ is the first interval in $mo(Q)$ with this property. Then check whether $s' - t \leq W$.

The time complexity of the confidence computation is $O(|mo(P)| + |mo(Q)|)$. If one wants to find the confidences of rules of the form $P\,[V] \Rightarrow Q\,[W]$ for all $V, W \in U$ for some set of times U, then by using a table of size $|U|^2$ one can in fact evaluate all these in time $O(|mo(P)| + |mo(Q)| + |U|^2)$. For reasons of brevity we omit the details.

Experimental results

We have experimented with the methods using as test data a part of the WWW server log from the Department of Computer Science at the University of Helsinki. The log contains requests to see WWW pages at the department's server; such requests can be made by WWW browsers at any host in the Internet.

An event in the log can be seen as consisting of the attributes *page, host,* and *time*. The number of events in our data set is 116308, and it covers three weeks in February and March, 1996. In total, 7634 different pages are referred to from 11635 hosts. Requests for images have been excluded from consideration. For simplicity, we only considered the *page* and *time* attributes; we used relatively short time bounds to reduce the probability of unrelated requests contributing

min_fr	Execution times (s)			
	Time bounds U (s)			
	$15, 30$	$30, 60$	$60, 120$	$15, 30, 60, 120$
50	158	210	274	268
100	80	87	103	104
250	56	56	59	58
500	50	51	51	52

Table 2: Experimental results: execution times

to the frequencies.

We experimented with frequency thresholds *min_fr* between 50 and 500, and with time bounds between 15 s and 2 min. (In three cases we used two time bounds, i.e., $U = \{V, W\}$, and in one case we searched simultaneously for all combinations of four time bounds in U.) Episode rules discovered with these parameters should reveal the paths through which people navigate when they know where they want to go.

Table 1 shows the number of frequent episodes and the number of informative rules with confidence at least 0.2. (A rule with time bounds V, W is considered informative if its confidence is higher than with time bounds V', W' where $V' < V$ and $W' > W$.)

The number of frequent episodes is in the range from 40 to 6000, and it seems to grow rather fast when the frequency threshold becomes lower. Our data is relatively dense, and therefore the effect of the time bounds on the number of frequent episodes is roughly linear. The largest frequent episodes consist of 7 events. Note that the method is robust in the sense that a change in one parameter extends or shrinks the collection of frequent episodes but does not replace any.

Table 2 shows the execution times for the experiments on a PC (90 MHz Pentium, 32 MB memory, Linux operating system). The data resided in a 3.0 MB flat text file.

The experiments show that the method is efficient. The execution times are between 50 s and 5 min. Note, in particular, that searching for episodes with several different time bounds (the right-most columns in the tables) is as fast as searching for episodes with only the largest time bound. Minimal occurrences are thus a very suitable representation for queries with different time bounds.

Following are some examples of the episode rules found (we use the titles of the pages here, or their English translations, which should be self-explanatory).

- "Department Home Page", "Spring term 96" [15 s] $\Rightarrow$ "Classes in spring 96" [30 s] (confidence 0.83). In other words, in 83 % of the cases where the departmental home page and the spring term page had been accessed within 15 seconds, the classes page was requested within 30 seconds (that is, within 30 seconds from the request for the departmental home page).
- "Research at the department" $\Rightarrow$ "Staff of the department" [2 min] (confidence 0.29). (There is no time bound for the left-hand side since there is only one event.)
- "Department Home Page", "Department Home Page in Finnish", "Classes in spring 96", "Basic courses" [15 s] $\Rightarrow$ "Introduction to Document Preparation (IDP)", "IDP Course Description", "IDP Exercises" [2 min] (confidence 0.42).

Experiments with the data set of (Mannila, Toivonen, & Verkamo 1995) show that — with comparable parameters — the present method is as fast or faster than the one presented in (Mannila, Toivonen, & Verkamo 1995). The new method has, however, two important advantages: the rule formalism is more useful, and rules with several different time bounds can be found with the same effort.

Concluding remarks

We have presented a framework for generalized episodes, and algorithms for discovering episode rules from sequences of events. The present framework supplies sufficient power for representing desired connections between events.

The work presented here is in many ways preliminary. Perhaps the most important extensions are facilities for rule querying and compilation, i.e., methods by which the user could specify the episode class in high-level language and the definition would automatically be compiled into a specialization of the algorithm that would take advantage of the restrictions on the episode class. Other open problems include a theoretical analysis of what subclasses of episodes are recognizable from episodes in polynomial time, and the combination of episode techniques with intensity models.

References

Agrawal, R., and Srikant, R. 1994. Fast algorithms for mining association rules in large databases. In *Proceedings of the Twentieth International Conference on Very Large Data Bases (VLDB'94)*, 487 – 499.

Bettini, C.; Wang, X. S.; and Jajodia, S. 1996. Testing complex temporal relationships involving multiple granularities and its application to data mining. In *Proceedings of the Fifteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'96)*. To appear.

Dousson, C.; Gaborit, P.; and Ghallab, M. 1993. Situation recognition: Representation and algorithms. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*, 166 – 172.

Gehani, N.; Jagadish, H.; and Shmueli, O. 1992. Composite event specification in active databases. In *Proceedings of the Eightteenth International Conference on Very Large Data Bases (VLDB'92)*, 327 – 338.

Hätönen, K.; Klemettinen, M.; Mannila, H.; Ronkainen, P.; and Toivonen, H. 1996. Knowledge discovery from telecommunication network alarm databases. In *12th International Conference on Data Engineering (ICDE'96)*, 115 – 122.

Laird, P. 1993. Identifying and using patterns in sequential data. In Jantke, K.; Kobayashi, S.; Tomita, E.; and Yokomori, T., eds., *Algorithmic Learning Theory, 4th International Workshop*, 1 – 18. Berlin: Springer-Verlag.

Mannila, H.; Toivonen, H.; and Verkamo, A. I. 1994. Efficient algorithms for discovering association rules. In Fayyad, U. M., and Uthurusamy, R., eds., *Knowledge Discovery in Databases, Papers from the 1994 AAAI Workshop (KDD'94)*, 181 – 192.

Mannila, H.; Toivonen, H.; and Verkamo, A. I. 1995. Discovering frequent episodes in sequences. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD'95)*, 210 – 215.

Morris, R. A.; Khatib, L.; and Ligozat, G. 1995. Generating scenarios from specifications of repeating events. In *Second International Workshop on Temporal Representation and Reasoning (TIME-95)*.

Muggleton, S. 1992. *Inductive Logic Programming*. London: Academic Press.

Seshadri, P.; Livny, M.; and Ramakrishnan, R. 1996. SEQ: Design & implementation of sequence database system. In *Proceedings of the 22nd International Conference on Very Large Data Bases (VLDB'96)*. To appear.

Srikant, R., and Agrawal, R. 1996. Mining sequential patterns: Generalizations and performance improvements. In *International Conference on Extending Database Technology (EDBT'1996)*.

Wang, J. T.-L.; Chirn, G.-W.; Marr, T. G.; Shapiro, B.; Shasha, D.; and Zhang, K. 1994. Combinatorial pattern discovery for scientific data: Some preliminary results. In *Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD'94)*, 115 – 125.