AAAI-02 Workshop "Automation as Caregiver", pages 24-30

Learning Models of Human Behaviour with Sequential Patterns

Valerie Guralnik and **Karen Zita Haigh**
Automated Reasoning Group
Honeywell Laboratories
3660 Technology Drive
Minneapolis, MN 55418
{vguralni,khaigh}@htc.honeywell.com

Abstract

The Independent Lifestyle Assistant (I.L.S.A.) is an agent-based system to aid elderly people to live longer in their homes. One of the biggest challenges for I.L.S.A. is that every installation will (1) be in a home with a different layout and suite of sensor and actuator capabilities, and (2) supporting a technophobic client with unique capabilities, needs and care-giving support network. I.L.S.A. therefore must be able to adapt to its environment over time. This paper describes our approach to modelling one particular aspect of the I.L.S.A. domain: using sequential pattern learning to learn models of human behaviour. We describe the problem, our enhancements to the basic algorithm, and present experimental results collected from four test sites [1].

Introduction

Historically, 43% of Americans over the age of 65 will enter a nursing home for at least one year. When it becomes apparent that a loved one can no longer safely take care of themselves, a nursing home is often the only option, in spite of the financial and emotional strain placed on the family.

We are developing an automated monitoring and caregiving system called *Independent LifeStyle Assistant* (I.L.S.A.) that will be a better alternative [8]. Researchers and manufacturers are developing a host of home automation devices that will soon be available. I.L.S.A. will be the integration of these individual functions, augmented with advanced reasoning capabilities to create an intelligent, coherent, useful assistant that helps people enjoy a prolonged independent lifestyle.

One of the biggest challenges for I.L.S.A. is due to the fact that every I.L.S.A. installation will (1) be in a home with a different layout and suite of sensor and actuator capabilities, and (2) supporting a technophobic client with unique capabilities, needs and care-giving support network. I.L.S.A. thus must be rapidly deployable, easy to configure and easy to update as the client ages and technology changes.

We have been exploring Machine Learning (ML) techniques to enable I.L.S.A. to adapt to its environment over time. Success in this domain requires that I.L.S.A. capture the complex interactions of its resources, and be responsive to constant changes. Using ML techniques, the fielded system will 1) tune itself to its actual operating environment, greatly reducing the amount of tuning and knowledge acquisition required at setup, 2) respond to changes in the users and the domain, directly reducing maintenance costs, and 3) capture the user's preferences, enhancing system usability. Machine Learning will permeate I.L.S.A. in almost every one of its functions.[2]

This paper describes one particular approach to modelling one particular aspect of the I.L.S.A. domain; here we focus on using sequential pattern learning [1] to learn models of human behaviour. We extended the approach to incorporate reasoning about the time of the activities. We also define and present our domain-specific filter to identify useful patterns.

In the following section we outline the problem domain in more detail. We then follow with a description of the technical approach, including how we incorporate time and the postprocessing filters. We then present results from data collected in four homes.

Technical Approach

One of the machine learning goals we have for I.L.S.A. is to model human behaviour with enough accuracy to recognize their actions and respond to situations. Our data is sensor firings. We need to learn what sensor firings correspond to what activities, in what order, and at what time. The technique of sequential pattern mining [1] provides a basis for recognizing these activities and their order. A *sequential pattern* in our context is a list of sensor firings ordered in time. Sequential pattern mining addresses the problem of finding frequent sequential patterns. We extend the basic approach to

[1] This work was performed under the support of the U.S. Department of Commerce, National Institute of Standards and Technology, Advanced Technology Program, Cooperative Agreement Number 70NANBOH3020

[2] This work is being patented as [7].

incorporate reasoning about the time of the activities.

Examples of discovered sequential patterns are:

- The Bathroom-Motion sensor fires after the Bedroom-Motion sensor fires 75% of the time
- The pattern Kitchen-Motion sensor followed by Living-Room-Motion sensor followed by Bedroom-Motion sensor occurs in 60% of the days

Unfortunately, one cannot simply apply the sequential pattern algorithm to sensor data and hope to discover sequential patterns which will *meaningfully* model a human's behavior. The reason for this is that we cannot accurately explain what each discovered pattern respresents. For example, the Bedroom-Motion sensor may represent several types of activity, such as a person waking up in the morning, or a person entering bedroom in the evening to go to sleep. One way of attaching a meaning to a pattern is to determine the time periods during which the sensors are each firing. The examples of sequential patterns above now become:

- The Bathroom-Motion sensor fires between 7:00 AM and 8:00 AM after the Bedroom-Motion sensor fires between 6:45 AM and 7:45 AM 75% of the time.
- The pattern Kitchen-Motion sensor fires between 6:00 PM and 6:30 PM followed by Living-Room-Motion sensor firing between 6:20 PM and 7:00 PM followed by Bedroom-Motion sensor firing between 9:00 PM and 10:00 PM occurs in 60% of the days.

Identification of appropriate time periods for each sensor makes it easier to attach a meaning to the pattern learned through unsupervised learning. In the above example, the first pattern might represent a person's waking-up routine, while the second pattern might represent the person's after work evening routine.

The pattern learning algorithm generates many thousands of sequences, and hence the final step of our algorithm is to filter the set of discovered patterns, and then cluster them for presentation to an analyst.

Calculating Time Intervals

The first step in identifying intervals of occurrence of each event in the pattern is to identify appropriate time periods for each sensor.

One possible approach is to generate the intevals during the sequential patterns discovery phase. However, the overhead is intractable, since all possible intervals must be considered.

Our solution to this problem is to predetermine the time intervals *before* learning the sequential patterns. This approach has the potential drawback that the quality of the discovered patterns can be affected by the way the expert defines the intervals. However, to minimize this possibility, we do not predefine *fixed-width* time intervals, but instead determine time intervals based on the *distribution* of sensor firings during the day. We use a *probability density function*, which

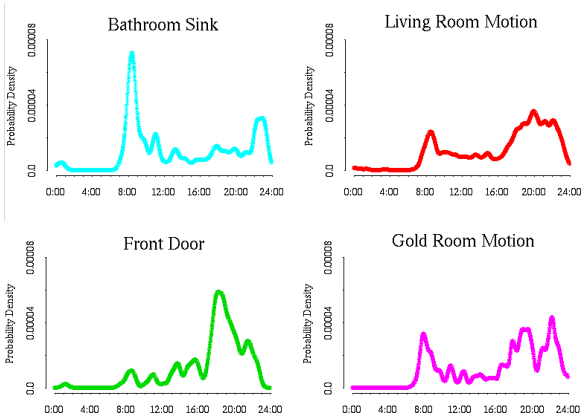

Figure 1: Individual Probability Density Function for four sensors in house 1.

Let s	be the sensor of interest
Let $\mathcal{T}$	be the timestamps of all firings of s
Let $\mathcal{P}$	be the probability density function of $\mathcal{T}$
Let $\mathcal{M}$	be the timestamps corresponding to the local minima of $\mathcal{P}$
Let $\mathcal{M}'$	be the sorted list [0:00, $\mathcal{M}$, 24:00]
$\forall$ pairs$(m_i, m_{i+1}) \in \mathcal{M}'$	
	Let x be $t_j \in \mathcal{T}$ such that $m_i < t_j$ and $\nexists t_k \in \mathcal{T}$ such that $m_i < t_k < t_j$
	Let w be $t_{j+1} \in \mathcal{T}$ such that $t_{j+1} < m_{i+1}$ and $\nexists t_k \in \mathcal{T}$ such that $t_{j+1} < t_k < m_{i+1}$
	interval$_i \leftarrow [x, w]$

Table 1: Calculating Time Intervals

is essentially a smoothed histogram of sensor firings, to determine the distribution. The area under the curve from time t_1 to time t_2 gives us the probability that a sensor will fire between t_1 and t_2.

We define the time intervals as intervals between successive local minimas of the probability density function. The key idea behind this approach is that a person usually performs each daily activity during a certain time period. For example, a person may typically wake-up between 8 AM and 9 AM. A sequence of sensor firings with attached timings will hence represent an activity.

Since some sensors fire more frequently than others, and moreover measure different kinds of activity, we determine time intervals by calculating probability density functions for each sensor separately. Figure 1 shows four different density functions, where one can see, for example, that the bathroom sink is used much more heavily during the morning routine.

The algorithm is presented in table 1. We first calculate the probability density function for each sensor, based on all the data, and use this function to calculate the time intervals. Then, for each sensor firing and its associated timestamp, we attach its interval.

Sequential Pattern Learning

Let us now cast the problem of learning a model of human daily activities into sequential pattern discovery.

First, let us define some terms from sequential discovery. Each distinct sensor and time partition pair is an *item*. A *sequence* is an ordered list of items. A sequence α is denoted as $(\alpha_1 \rightarrow \alpha_2 \rightarrow \ldots \rightarrow \alpha_n)$, where each sequence element α_j is an item. An item can occur multiple times in a sequence.

After the time intervals are determined, the list of daily event sequences is translated into a data set suitable for learning sequential patterns. A sensor event maps to an item, and has a unique identifier. Without loss of generality, we assume that no event sequence has more than one event with the same time-stamp, and that the events are sorted by the time of occurrences.

The input database consists of a number of these event sequences. The support of a pattern α is the fraction of sequences (or days) in the database that contain that pattern. According to this definition, the support of a pattern α is counted only once per sequence even though it may occur multiple times. Given a user-specified threshold called *minimum support*, we say that a pattern is *frequent* if its support is greater than the minimum support.

Once the list of sensor firings is mapped to database of sequences, the frequent sequential patterns can be computed efficiently using a variety of sequential pattern discovery algorithms [1; 10; 13]. For experiments described in this paper we used tree projection sequential patterns algorithm [6]. This algorithm is particularly suitable for data sets with a small number of distinct items and long sequences.

Post Processing

Due to the complicated structure and redundancy of the data, the algorithm generates an enormous number of highly frequent, but not necessarily useful sequential patterns. In data sets with characteristics like I.L.S.A., the number of patterns generated by pattern analysis can be the same magnitude as the number of sensor firings in the data set [2]. A list of several thousand patterns is only marginally more useful than raw data. Our goal is to find *interesting* sequences that will help us create a model of person's activities during the day.

Our first filter is a domain heuristic, based on the assumption that the closeness in time of occurrence of events provides evidence in support of those firings related to one activity. The closer the events occur in time, the more likely that events belong to the same activity. Notice that domain knolwedge is fairly fuzzy, and it is hard to define specific parameter values for 'closeness in time'.

In experiments described in this paper we use a simple heuristic that states that a pattern represents two different activities if it contains events that have non-itersecting time intervals. Let's consider the following example:

- Bedroom Motion [6:00 - 7:00] $\rightarrow$ Bathroom Motion [6:00 - 7:00]
- Kitchen Motion [12:00 - 13:00] $\rightarrow$ Dining Room Motion [12:30 - 13:30] $\rightarrow$ Kitchen Motion [13:00 -14:00]
- Bedroom Motion [6:00 - 7:00] $\rightarrow$ Kitchen Motion [12:00 - 13:00] $\rightarrow$ Dining Room Motion [12:30 - 13:30]

While the first two patterns represent one activity each (waking up and having lunch, respectively), the third respresents a combination of the parts of those two activities. Application of this filter generally limits the learned patterns to ones that represent one activity.

Our second filter eliminates redundant patterns by removing all subpatterns that have the same supporting days. The closed set [15] reduces the set of all frequent patterns by several orders of magnitude, while not losing any information. Let's consider the follwing example:

- Bedroom Motion [6:00 - 7:00] $\rightarrow$ Bathroom Motion [6:00 - 7:00]
- Bedroom Motion [6:00 - 7:00] $\rightarrow$ Kitchen Motion [6:30 - 7:30]
- Bedroom Motion [6:00 - 7:00] $\rightarrow$ Bathroom Motion [6:00 - 7:00] $\rightarrow$ Kitchen Motion [6:30 - 7:30]

Let's assume that the first pattern is supported by days 1, 4, 14, 21 and the last two patterns are supported days on 1, 4, and 21. The second filter will eliminate the second pattern since it is a subpattern of the last one and both are supported by the same set of days. The first pattern is not redundant because the set of its supporting days is different fromt the set of supporting days of the last pattern even though the first pattern is a subpatterns of the last one.

Our third filter exploits the domain knowledge that some sensors will keep firing even though the same event triggered them. For example, if a person is cooking dinner in the kitchen for half an hour, the motion detector will keep firing with regular intervals.

The potential users of our approach have indicated that this kind of pattern is confusing, and they would much rather be presented with its subpattern, in which multiple sequential firings of the same sensor are reduced to one.

We show in our results section that our interestingness filters substantially reduce the number of patterns presented to analysts. Nevertheless, this number is occasionally still large enough to require further aggregation of patterns. To facilitate analysts' understanding of patterns, we apply clustering techniques to the remaining patterns. Each cluster then represents diferent variations of the same activity. In experiments presented in this paper, we used a vector-spaced k-means algorithm [14]. To apply this algorithm we needed to

map our patterns to a vector space: dimensions are the distinct event types of the all the patterns. In n is the dimensionality of that space, then a pattern is represented as an n-dimensional vector of zeros and ones, with ones corresponding to all the events present in that particular pattern.

Experimental Results

Experimental Setup

I.L.S.A. has been installed in the homes of four of the project engineers, with 10-20 sensors per home. These sensors include motion detectors, pressure pads, door latch sensors, toilet flush sensors, and a panic button.

We have been collecting data from this prototype system since July 2001. Table 2 briefly describes each home and shows the total data collected. Since these installations were primarily focussed on configuring the system, we did not use all of the data – the table indicates how much of the data was useful.

One of the reasons we discarded data was due to sensors designed and configured for a slightly different task. Door sensors for example, are designed for security systems and raise an alarm when the door is opened. The sensor, however, is not appropriate for a continuous monitoring environment when doors are often left open for long periods of time: the sensor starts 'shouting' and drowning out the signals of other sensors.

As we grew to understand the characteristics of the sensors, we built simple filters to remove corrupted files, aggregate signals and remove noise. We also replaced problematic sensors with sensors from different manufacturers or different sensing specifications.

We also removed several "exceptional" days, for example, when the data server went down, or the occupants left the home on vacation.

Learning Results

Our experiments to date have focussed on two important questions: (1) whether we can learn patterns that can explain the lifestyle of household, and (2) whether we miss important patterns. To answer these questions we tried to draw some conclusions based on the patterns we found for each household, and then interviewed the occupants about our perceptions. Given that patterns are calculated over many days of data, it is not feasible to compare to a ground truth, say a continuous video tape of activity. Examples of the patterns learned in each test houses are shown in Table 3.[3]

Our observations of house 1 indicate that the occupant lives a very regular lifestyle. She is in bed every night by midnight, and up at around 8am. She almost universally exits the house through the back door. The bathroom sink is used every morning, but rarely in the evenings. We were unable to explain the firings of the motion sensors that occur for an hour during the late-afternoon before the back door opens, but the occupant immediately recognized that the dog starts getting excited about her pending return from work.

House Num	Pattern	Support
1	UpstairsM [06:48,09:58] → LivingRoomM [06:48,09:58] → BathrmSinkPP [07:07,09:58] → LivingRoomM [07:07,09:58]	64.5%
1	BackDoor1 [16:12,19:07] → LivingRoomM [16:53,19:07] → KitchenM [16:53,19:07] → LivingRoomM [16:53,19:24]	62.9%
2	LR/DRM [15:19,16:59] → KitchenM [15:19,17:43]	55.0%
2	BathroomM [21:29,23:56] → LR/DRM [21:53,23:56]	52.5%
3	TV-LRM [21:45,23:25] → KitchenM [21:45,23:25] → TV-LRM [21:45,23:43] → BedroomM [21:45,23:43] → BathroomM [21:45,23:43] → BedroomM [21:45,23:59]	75.3%
3	BedroomM [03:44,07:28] → BathroomM [04:53,07:28] → BedroomM [04:53,07:38] → KitchenM [05:25,07:42]	75.3%
4	BedRmPP [04:23,06:12] → KitchenM [06:12,08:23] → Cupboard [06:45,09:04] → FrontDoor [06:56,08:43]	52.9%

Table 3: Example Patterns Found

Unfortunately not all sensors were working correctly when we started collecting data in house 2. For example, only the last 20 useful days contained data from the bedroom and bathroom. In addition, there were a lot of sensors that didn't fire at all: some sensors are designed to fire only in extreme circumstances, e.g. glass break and keypad panic. As a result, the majority of firings recorded for house 2 came from only three sensors: Living Room Motion, Back Door and Front Door.

The majority of patterns we learned involved only those sensors, typically consisting of repeated firings of Living Room Motion sensors interleaved with other sensors such as the Back Door. As a result, it was very hard interpret what the learned patterns meant. One interesting observation is that the sensors fired during the day on weekdays. It turns out that one person in the household works from home. Another conclusion we drew is that the residents of that house prefer to use the Back Door over the Front Door.

The patterns obtained from data of house number 3

[3] Abbreviations: LR = LivingRoom, DR = Dining Room, PP = Pressure Pad, M = Motion Sensor

House Number	Number of Occupants	Number of Sensors	Number of Days of Collected Data	Number of Days of Useful Data
1	1 adult, 1 80-lb dog	16	80	62
2	2 adults	20	87	40
3	2 adults	10	123	81
4	1 adult	10	63	34

Table 2: Collected Data

House Number	Minimum Support Threshold	Number of Patterns Found	Number of Patterns After Filtering
1	60%	11411	119
2	50%	5375	50
3	75%	2725	182
4	50%	340	54

Table 4: Number of Patterns Found

suggested that its residents wake up very early in the morning and go to bed very late at night – the occupant later confirmed that the alarm clock is set to 5:15 on weekday mornings. Another observation was made that people in that household consistently go to the kitchen late in the evening (between 9 pm and 11 pm). It turned out that the residents get an evening snack to eat in front of TV in the evening. We also accurately identified that the only access to the bathroom is through the bedroom. Table 3 shows that the occupants enter and use the bathroom before going to the kitchen for breakfast.

After looking at the patterns learned in household 4, it became clear that the engineer living in that house lives a hectic life. While there were some morning patterns that can be interpreted as "getting ready to go to work patterns (with a regular breakfast)", there were very few evening patterns. The results were confirmed by the engineer, who told us that he had not expected us to find *any* patterns in his house.

Table 4 summarizes learning results obtained from the four test houses. The first observation we can make is that increasing the minimum support threshold does not guarantee an increase in number of patterns learned. In fact, the number of patterns is determined by two factors:

1. how regular the person's daily activities are, and
2. how may different sensors that fire regularly are installed in the house (sensors like glass-breakage fire extremely rarely).

The second observation we can make is that while the pattern discovery algorithm produces an enormous number of patterns, the post-processing step reduces it to a reasonable number. While it is impossible for a human analyst (e.g. care giver) to analyze 5375 patterns, he will be able to comprehend 50 patterns. It can be argued that 182 patterns (as in case of household 3) is still a large number for analysts to navigate through, and hence we use clustering techniques (as described above) to group patterns similar to each other. The analyst can then look at the discovered patterns in batches. An example of one such cluster containing patterns test house 1 is presented in Table 5. The patterns in that cluster probably represent a morning routine.

Related Work

The problem of mining for sequential patterns was first introduced in Agrawal and Srikant [1]. A generalization of the class of patterns being mined and performance enhancements were presented in [13]. There are many domains where sequence mining has been applied, which include discovering customer buying patterns in retail stores, identifying plan failures [16], finding network alarm patterns [9], and so on. There are several differences between data in those domains and data in ours.

First, in those domains each event has a unique identifier, whereas in our domain each sensor firing can represent different types of events. For example, the Bedroom-Motion sensor can fire due a person walking into the bedroom, exiting the bedroom, or moving within the bedroom. Second, the usual data sets contain thousands of different types of events, whereas each house in our application has no more of 20 sensors installed. Third, the sequences in regular applications contain about 20 events, whereas a regular sequence in our application contains hundreds of sensor firings collected each day.

Performance issues aside, these differences mean that straightforward application of the sequential discovery algorithm to I.L.S.A. data will most likely yield almost all the possible combination of sensors. The discovered patterns will not be useful in modeling human's daily activities. Hence we must exploit whatever additional information we have, notably time.

The search space of patterns satisfying some statistical properties of dominance is still large. Moreover, as pointed out by Silberschatz and Tuzhilin [11], not all patterns that are statistically dominant are of interest. The notion of 'interestingness' for event sequences is addressed by Berger and Tuzhilin [3] who suggest that a pattern is interesting if the ratio of its actual by expected occurrences exceeds a given threshold. Das *et al* [4] perform sequence mining and rank discovered rules according to their 'informativeness'. They use the

Pattern	Support
UpstairsM [06:48,10:40] → BathroomSinkPP [07:07,10:30]	69.3%
BathroomSinkPP [07:07,10:30] → KitchenM [06:54,10:28]	62.9%
LRM [06:19,09:58] → BathroomSinkPP [07:07,10:30]	66.1%
LRM [06:19,09:58] → BathroomSinkPP [07:07,10:30] → KitchenM [06:54,10:28]	61.2%
UpstairsM [06:48,10:40] → LRM [06:19,09:58] → BathroomSinkPP [07:07,10:30] → LRM [06:19,09:58] → BathroomSinkPP [07:07,10:30]	61.2%
UpstairsM [06:48,10:40] → LRM [06:19,09:58] → BathroomSinkPP [07:07,10:30] → LRM [06:19,09:58]	64.5%
LRM [06:19,09:58] → BathroomSinkPP [07:07,10:30] → LRM [06:19,09:58] → BathroomSinkPP [07:07,10:30]	62.9%

Table 5: A cluster of Patterns

J-measure of informativeness proposed by Smyth and Goodman [12]. A common theme among the various criteria for measure of interest is the concept of *novelty* or *unexpectedness* of a rule, where results that were previously known by the data anlaysts are not considered interesting. Our measure of interest on the other hand is the *usefulness* of a pattern for our modelling purposes.

Conclusion and Future Work

We are currently working on several enhancements to the pattern learner. The first, and most important, is to use the learned patterns as a basis for I.L.S.A.'s automatic behaviour recognition system. We also would like to utilize a clustering algorithm that takes event *order* into account, such as described in [5]. We also suspect that sensors that are co-located are indicative of the same behaviour, and would like to explore this hypothesis to bias the learner. Finally, one complaint that we received from engineers is that the time intervals of events in the patterns are too wide. Our suspicion is that weekend vs. weekday patterns are sufficiently different that the learner is unable to create fine-grained time intervals; however, due to the small amount of available data, we have not yet partitioned the data set.

ML has been discussed as a means to enhance the accuracy and coverage of fielded monitoring systems, with no serious effort to evaluate its capabilities, in part because of an assumption that the domains had strong structure and little variance (e.g. piloting fighter jets, airline ticket sales, logistics form generation, etc.). Not only are those assumptions radically untrue in I.L.S.A.'s domain, they've also proved somewhat false in the other domains as well. I.L.S.A., by utilizing learning in each decision-making component of the system, for many types of sensor data, and over an extended lifetime, will be a significant step forward.

The discovered patterns can be used to construct behaviour models of the elderly, which in turn will be able to provide means for on-going adaptation and improvement of I.L.S.A.'s responsiveness relative to the needs of the elderly. The behaviour models will enable I.L.S.A. to automatically configure itself and to adapt to changing conditions, minimizing the time and labor involved in set-up and maintenance. Further, the learned models of behaviour can be employed to assist in selecting the most apporiate response plan to actions of the elderly.

References

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In *Proceedings of the Eleventh International Conference on Data Engineering (ICDE)*, pages 3–14, 1995.

[2] R. J. Bayardo. Efficiently mining long patterns from databases. In *ACM SIGKDD Management of Data Conference*, 1998.

[3] G. Berger and A. Tuzhilin. Discovering unexpected patterns in temporal data using temporal logic. *Temporal databases - Research and Practice, volume 1399 of Lecture Notes in Computer Science*, pages 281 – 309, 1998.

[4] G. Das, K. I. Lin, H. Mannila, G. Renganathan, and P. Smyth. Rule discovery from time series. In *4th International Conference on Knowledge Discovery and Data Mining*, pages 16–22, 1998.

[5] V. Guralnik and G. Karypis. A scalable algorithm for clustering sequential data. In *First IEEE Conference on Data Mining*, 2001.

[6] V. M. Guralnik. *Mining Interesting Patterns from Sequential Data.* PhD thesis, University of Minnesota, Minneapolis, Minnesota, 2001.

[7] K. Z. Haigh, S. A. Harp, and V. Guralnik. Methods for learning the patterns of behaviour of an actor or environment, November 2001. Patent docket number H0003384. Provisional Disclosure filed.

[8] K. Z. Haigh, J. Phelps, and C. W. Geib. An open agent architecture for assisting elder independence. In *The First International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2002.

[9] K. Hatonen, M. Klemettinen, H. Mannila, P. Ronkainen, and H. Toivonen. Knowledge discovery from telecommunications network alarm

databases. In *12th International Conference on Data Engineering*, 1996.

[10] M. V. Joshi, G. Karypis, and V. Kumar. Universal formulation of sequential patterns. Technical report, University of Minnesota, Department of Computer Science, Minneapolis, 1999.

[11] A. Silbershatz and A. Tuzhilin. What makes patterns interesting in knowledge discovery systems. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):970–974, 1996.

[12] P. Smyth and R. Goodman. An information theoretic approach to rule induction from database. *IEEE Transactions on Knowledge and Data Engineering*, 4:301–316, 1992.

[13] R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. In *Proceedings of the Fifth International Conference on Extending Database Technology*, pages 3–17, 1996.

[14] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 2000.

[15] M. J. Zaki. Generating non-redundant association rules. In *ACM SIGKDD Conference*, 2000.

[16] M. J. Zaki, M. J. Lesh, and N. Ogihara. PLANMINE: Sequence mining for plan failures. In *4th International Conference on Knowledge Discovery and Data Mining*, 1998.