Knowledge Discovery in Entity Based Smart Environment Resident Data Using Temporal Relation Based Data Mining

Vikramaditya Jakkula, Aaron Crandall, and Diane Cook
EECS, Washington State University, Pullman, WA-99164.
{vjakkula, acrandall, cook}@eecs.wsu.edu

Abstract

Time is an important aspect of all real world phenomena. In this paper, we present a temporal relations-based framework for discovering interesting patterns in smart environment datasets, and test this framework in the context of the CASAS smart environments project. Our use of temporal relations in the context of smart environment tasks is described and our methodology for mining such relations from raw sensor data is introduced. We demonstrate how the results are enhanced by identifying the number of individuals in an environment, and apply the resulting technologies to look for interesting patterns which play a vital role to predict activities and identify anomalies in a physical smart environment.

1. Introduction

Smart home research has been around for some time and significant progress has been achieved in the areas of data analysis and mining of data collected in smart environments. Despite this, there is still much room for further work in the areas of temporal data mining. Since smart home datasets are time-stamp datasets, we see that temporal mining would make the analysis and use of these datasets more effective and efficient than simpler data mining strategies.

While making sense of sensor data can be challenging for smart environment applications, the problem is made even more complex when the environment houses more than one resident. To aid the capabilities of our temporal data mining, and to reveal the complexities of multi-inhabitant spaces, an entity discovery tool was developed. This tool analyzes the raw sensor data to attribute given events with entities in the space. By attributing events to different entities the temporal miner will be in a position to discover events series that occur, or do not occur, based on occupancy of the space. This increased granularity to the data will lead to a better understanding and prediction of events in smart environments. We will define a formal representation for temporal relationships between events in a smart home, and use this formalism as a basis for reasoning over these relationships. Temporal reasoning gives us the groundwork for performing such activities as anomaly detection and prediction in the context of the smart environment. Representing and reasoning about activities, primarily about actions and events, is an interesting problem within the smart home domain. The focus of this paper is the mining for unique and interesting patterns within the data that can eventually be used for tasks such as event prediction and anomaly detection.

Activities in a smart home include a resident's physical activities as well as instrumental activities. Physical activities may include walking, sitting on a couch, turning on a lamp, and using the coffeemaker, for instance. We see that many of these activities are not instantaneous, but have distinct start and end times. We also see that there are well-defined relationships between time intervals for different activities. These temporal relations can be represented using Allen's temporal relations [1] and can be used for knowledge and pattern discovery in day-to-day activities. These discoveries can be used for developing systems which act as reminder assistants, for detecting anomalies, and for aiding smart homes in automatically taking preventive measures to keep residents safe.

A question may arise as to why Allen's temporal relations should be used for generating temporal intervals. The temporal relations defined by Allen form the basic representation of temporal intervals. When these are used with constraints they become a powerful method of expressing expected temporal orderings between events in a smart environment. There are projects which employ sequential information to predict activities [2], and other methods for identifying suspicious states in a smart environment have been researched [3]. We extend these methods to incorporate valuable information about the interval of time each event spans. While other methods treat each event as a separate entity (including, for instance,

turning on a lamp and later turning off the same lamp), our interval-based analysis considers these two events as members of one interval. Each interval is expressed in terms of start time and end time values. Let us consider a typical activity in a smart environment given below.

Consider a simple scenario which involves a television, fan and a lamp being used in a smart home. We see that the resident turns on the television and after some period of time turns on the fan. As time progresses, feeling cold, the fan is turned off and the individual continues watching the television. Later on, the television is turned off and the individual turns on the lamp to illuminate the room. We see that this scenario involved three activities each defined by interaction with a single device, namely a television, a fan and a lamp. Now we apply Allen's logic to establish the temporal relations among the activities which occurred. The scenario is illustrated in figure 1. These activities can be represented as television "contains" fan and "meets" lamp. We can also represent these relationships as television "meets" lamp and fan "before" lamp.

The thirteen possible temporal relations are illustrated in Table 1. Consider two general events X and Y; we use this to represent the relations in the table. In Table 1, the interval constraints compare the start time (Start) and end time (End) of the activities, X and Y [4].

In this paper we use the temporal relations and identify interesting patterns of activity on the entity-less and entity-tagged data collected in a smart environment. Later we can use this mined knowledge with a decision maker in a real world environment.

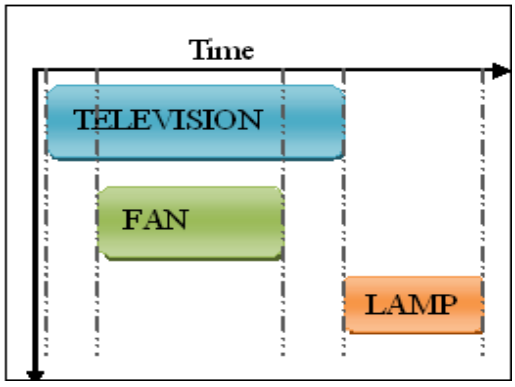
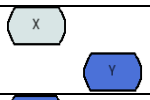
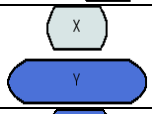
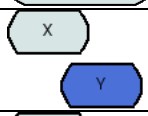
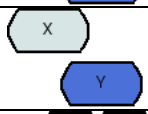
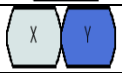
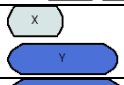
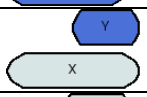


Figure 1. Illustration of temporal intervals

Table 1. Temporal relations representation.

Temporal Relations	Pictorial Representation	Interval constraints
X Before Y		Start(X)<Start(Y); End(X)<Start(Y)
X After Y		Start(X)>Start(Y); End(Y)<Start(X)
X During Y		Start(X)>Start(Y); End(X)<End(Y)
X Contains Y		Start(X)<Start(Y); End(X)>End(Y)
X Overlaps Y		Start(X)<Start(Y); Start(Y)<End(X); End(X)<End(Y)
X Overlapped-By Y		Start(Y)<Start(X); Start(X)<End(Y); End(Y)<End(X)
X Meets Y		Start(Y) = End(X)
X Met-by Y		Start(X)= End (Y)
X Starts Y		Start(X)=Start(Y); End(X)≠End(Y)
X started-by Y		Start(Y)=Start(X); End(X)≠End(Y)
X Finishes Y		Start(X)≠start(Y); End(X) = End (Y)
X Finished-by Y		Start(X)≠Start(Y); End(X)=End(Y)
X Equals Y		Start(X)=Start(Y); End(X)=End(Y)

2. Environmental sensing

We define a smart environment as one with the ability to adapt the environment to the residents in order to improve their experience in the environment. Our CASAS project treats a smart environment as an intelligent agent, as well as working towards agents that represent individual users. The current agent perceives the state of the environment using sensors, and can act upon the environment using power line controllers, illustrated in figure 2. The intelligent agent will select actions that optimize its goals. For a smart environment, these goals could include energy efficiency, enhancing resident productivity, improving their comfort, and/or ensuring their safety. For the study we report here, we focus on the CASAS goal of ensuring the health and safety of the smart environment resident. In order to achieve these goals,

the house should be able to predict, reason, and adapt to its resident. In our smart workplace environment, shown in figure 3, the sensor network data is the primary source of data collection and collects entity-less data, or data describing the events of a single resident in the environment [4].

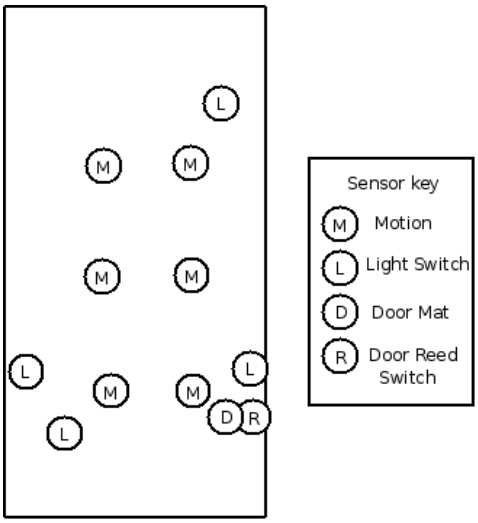


Figure 2. CASAS alpha room sensor layout (source of entity-tagged datasets).

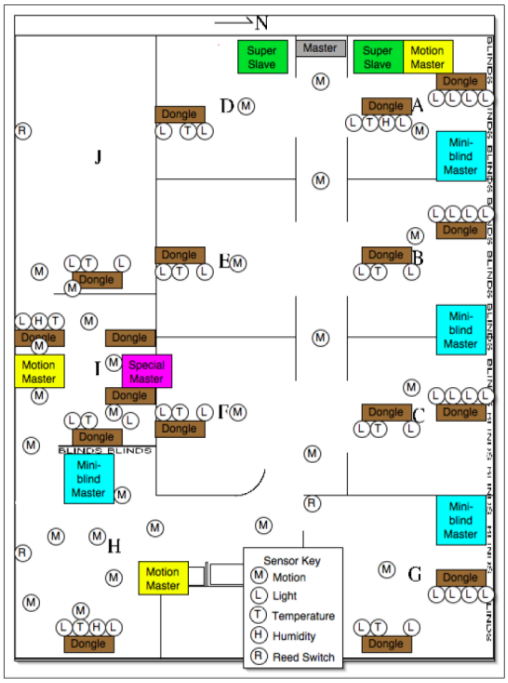


Figure 3. Smart workplace sensor layout (source of entity-less datasets) [3].

The data collection system consists of an array of motion sensors, which collect information using X10 devices and the sensor network. Our dataset is collected for a resident working in the smart workplace (see Figure 3) and consists of two months of data. The lab consists of a presentation area, kitchen, student desks, and faculty room. There are over 100 sensors deployed in the lab that include light, temperature, humidity, and reed switches. This data was collected from a single resident and is being used as the entity-less data source for the experimentation in this paper.

For the data collection process in the multi-resident CASAS environment, we used a series of repeated scripted events in our Alpha Room (see figure 2). The Alpha Room is the CASAS initial data gathering environment, designed to be used for the latest algorithms before they are used in spaces where people are living and working. The room itself is laid out to have two working desks and a white board for collaboration in between. Currently, there are six motion detectors on the ceiling, four lights controlled by Insteon™ power line control devices, a pressure-sensitive mat at the entrance door and a door open/closed switch. A data collection computer receives the power line events via a USB bridge, and the rest of the sensors feed through a microprocessor system to a serial port. The room itself is not large, but big enough to make use of the Entity Discovery tool with two people.

3. Experimentation Process and Results

3.1. Entity Discovery and Tagging

In the CASAS environment, we are enriching the data with information about entities moving within the space. This comes in the form of an entity (in this case, resident) identification number that is attached to each event, matching events to entities. As an entity traverses the space they trigger events. Our Entity Discovery tool uses only information about the space and the raw sensor data to make these attributions. An illustration of a single sensor neighborhood in CASAS environment is given in figure 5. An illustration of entity identification is shown in figure 6. For the Entity Discovery tool, the concept of neighborhoods of sensors was used to determine location and quantity of entities within the space, as well as attributing events to individuals. To get a neighborhood, the sensors available are mapped to a Cartesian grid then assembled into a graph connecting all close sensors within a certain distance. This physical distance around a given sensor (see figure 5) is the neighborhood size, which turns out to be a vital value for the operation of this system and needs to be tuned properly for things to function. The result is that for

every sensor you get a list of neighbors, hence its neighborhood.

Table 2. Entity Discovery Algorithm Pseudo-code.

Entity Discovery Algorithm Pseudo-code:
1. Build model of sensors based on physical layout
2. Generate neighborhoods within sensor graph
3. **While** [Event Received]
4. Parse event
5. Query collection of live entities for neighborhood membership of event
6. **If** zero entities returned **then**
7. Create new entity at event location, attribute event to entity
8. **Else If** one entity returned **then**
9. Move entity to location of event and attribute event to entity
10. **Else If** two or more entities returned **then**
11. Finish all entities and create a new entity at event location, attribute event to new entity
12. **End if**
13. **Loop**

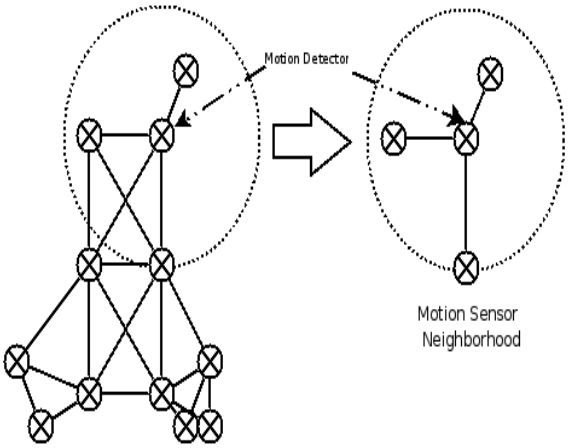


Figure 5. An illustration of a single sensor neighborhood in CASAS environment.

There are three possible scenarios encountered using this tool:

Case A - No entities are returned: In this case, there is no way to attribute the event to a known entity properly so a new entity is created and placed at that location, and that event is attributed to that new entity. For an example of the attribution, see Table 3, below.

Case B - One entity is returned: Here we can attribute the event to a single known entity. The result is that the event is set to that entity's ID and the entity is moved to the location of the event.

Case C - Two or more entities are returned: Because we are unable to determine, with this early strategy, which of the entities caused the event, the tool must play it safe. All of the entities are "finished" and no longer considered in the space, as we can no longer say anything specifically about their impact on the environment. A new entity is created and placed at the location of the event for future consideration.

Table 3. Example event data before and after being processed by the Entity Discovery tool

Example raw event data before being tagged with an entity attribution:			
Timestamp		Sensor ID	Event Message
2007-05-18	14:44:04	001.009	ON
2007-05-18	14:44:08	07.75.3d	OFF
Example event data after being attributed to an entity, in this case two separate entities are noted:			
Timestamp		Sensor ID	Message Entity
2007-05-18	14:44:04	001.009	ON 0
2007-05-18	14:44:08	07.75.3d	OFF 1

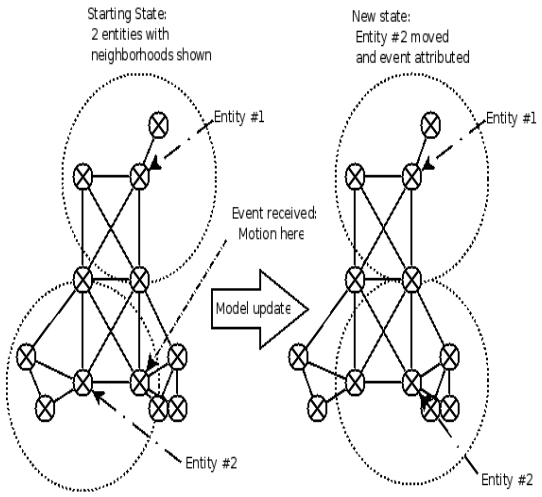


Figure 6. Internal world model update upon event receipt.

The Entity Discovery tool has shown promise, even with this basic of algorithms, at providing information useful to the Knowledge Discovery tool. The next step of this experimentation involves looking for interesting patterns.

3.2. Mining for Interesting Patterns

A common technique used is the discovery of patterns which are frequent and happen often. But using temporal relations to mine, the entire concept of sequence is now represented by temporal relations. Thus, in a smart home scenario these attributes of

interestingness are as follows: frequency can be the number of times that particular pattern is found, length is the number of temporal relations involved in that pattern and the periodicity can be the measure of time span it takes before it repeats itself. The second step after identifying the entities involve forming temporal relations based datasets for mining.

Table 4. Temporal Relation Discovery Algorithm Pseudo-code[16]

Temporal Relation Discovery Algorithm Pseudo-code:
1. **While** [Event && Event + 1 found]
2. Find paired "ON" or "OFF" event in data to determine temporal range.
3. Read next event and find temporal range
4. Take both events and look up kind of relation from possible relation types (see Table 1)
5. Write out relation type and related data
6. increment Event Pointer
7. **Loop**

As we use temporal related datasets, we mine using the longest common subsequence technique with maximal consecutive, where we find a long sequence which is a subsequence, or a common subsequence of all sequences in a set of sequences. In our implementation, we look for maximal consecutive sequences within a two-day period. The reason for performing a maximal consecutive look up is that a three commonly-shared subsequence is of significant interest compared to one sequential subsequence. For instance, we have two strings X and Y, where X = "**AA** BB PP **BB** ZZ **CC** KK **DD** UU VV **EE** RR" and Y = "**AA** **BB** **CC** **DD** EE FF GG". Using this technique, AA BB CC DD EE is of more interest compared to a simple AA BB CC DD subsequence which is sequential in order.

Here we identify common patterns for two days at a time and identify all interesting patterns over the 66 days of entity-less data and 33-days of entity-tagged data. Later we analyze the attributes of interestingness over the longest common subsequences identified in the experimental datasets. The longest common subsequence problem is NP-Hard for a general case of arbitrarily long input sequences and the problem is solvable in polynomial time using dynamic programming [5]. The reason we are interested in longest length and not the shortest one is because the shortest ones just identify the simple temporal relations which can be identified using other frequent mining techniques but the interesting part is the longest length because they are unique for most of the days based on the inhabitant [6]. Using the longest common subsequence tool we look for patterns in both the entity-less and entity-tagged datasets and we patterns found are noted. The next step of the experimentation would be analysis of the results. Table 6 shows us the number of patterns found and the total number of days used, while Figure 7 visualizes these results.

Table 5. Number of patterns found in entity-less vs. entity-tagged data sets

	No of Patterns Found	Total No of Days
Entity-Less	18	66
Entity 0	30	33
Entity 1	21	33
Entity 2	0	33
Entity 3	0	33

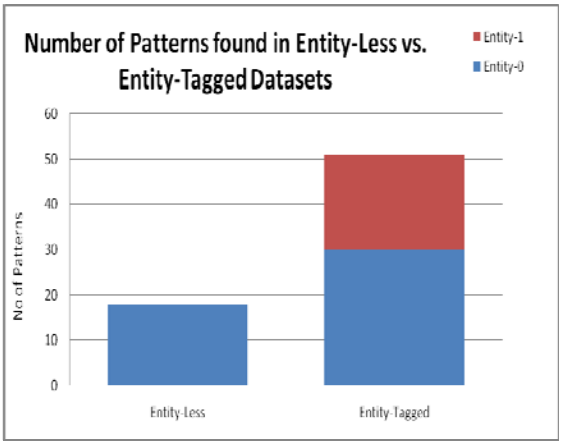


Figure 7. Number of patterns found in entity-less vs. entity-tagged data sets

Table 7 summarizes the frequency or the number of times a particular pattern is repeatedly found. We note that the entity-less dataset has no repeated patterns whereas when we discover different entities and have patterns associated to them we see that there are some patterns which repeat over time. Figure 8 visualizes these results.

Table 6. Pattern frequency found

	Frequency (Number of times a particular pattern repeated)
Entity-Less	0
Entity 0	3
Entity 1	3

Entity 0 + Entity 1	4

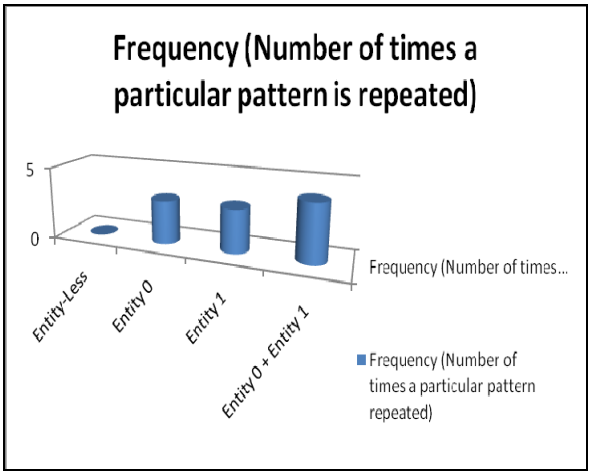


Figure 8. Pattern frequency discovered

Table 8 and figure 8 illustrate the number of cycles found among the patterns identified in the entity-less and entity-tagged datasets. We see that we have three cycles of repetitions in the entity-tagged datasets compared to none in entity-less datasets.

Table 7. Periodicity discovered within the data.

	Periodicity(Number of cycles in Data)
Entity-Less	0
Entity-Tagged	3

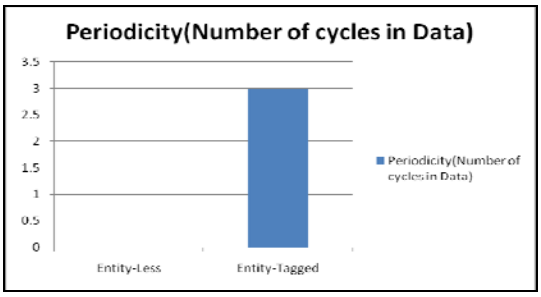


Figure 9. Periodicity discovered within the data.

4. Conclusion

What makes this work novel is the unification of identifying entities and pairing them with temporal reasoning to look for interesting patterns solving the problem of knowledge discovery for a multi-inhabitant smart environment. We think this as an intuitive and relatively simple framework made of complex temporal relations for reasoning and mining. The leveraging of entity discovery provides a new dimension to the data available to the prediction process. By having information about which entity in the space is causing events, even in a rough manner, we are able to mine out more detailed information having to do with the event patterns in the space. Often, people will act and prefer different things when there are multiple individuals operating in a space. By adding this extra interpretation of what is happening in the space, the data mining and prediction systems can take these kinds of higher-order interactions into account.

5. Acknowledgement

This work is supported by NSF grant IIS-0121297.

6. References

[1] James F. Allen, and George Ferguson, "Actions and Events in Interval Temporal Logic", *Technical Report 521*, July 1994.

[2] K. Gopalratnam and D. J. Cook, "Active LeZi: An Incremental Parsing Algorithm for Sequential Prediction", *International Journal of Artificial Intelligence Tools*, 14(1-2):917-930, 2004.

[3] G. Michael Youngblood, Lawrence B. Holder, and Diane J. Cook. "Managing Adaptive Versatile Environments", *Proceedings of the IEEE International Conference on Pervasive Computing and Communications*, 2005.

[4] Vikramaditya Jakkula, and Diane J. Cook, "Learning temporal relations in smart home data", *Proceedings of the second International Conference on Technology and Aging*, Canada, June 2007.

[5] Wikipedia, et al. Longest Common Subsequence Problem. *Wikipedia*. Wikipedia Foundation, Inc. Retrieved May 25, 2007. http://en.wikipedia.org/wiki/Longest_common_subsequence_problem

[6] Wikibooks, et al. Algorithm Implementation Strings Longest Common **S**ubsequence. *Wikibooks*. Wikimedia Foundation, Inc. Retrieved May 25, 2007. http://en.wikibooks.org/wiki/Algorithm_implementation/Strings/Longest_common_subsequence.