Predicting Occupant Locations Using Association Rule Mining

Conor Ryan and Kenneth N. Brown

Abstract Heating, ventilation, air conditioning (HVAC) systems are significant consumers of energy, however building management systems do not typically operate them in accordance with occupant movements. Due to the delayed response of HVAC systems, prediction of occupant locations is necessary to maximize energy efficiency. In this paper we present two approaches to occupant location prediction based on association rule mining which allow prediction based on historical occupant movements and any available real time information, or based on recent occupant movements. We show how association rule mining can be adapted for occupant prediction and evaluate both approaches against existing approaches on two sets of real occupants.

1 Introduction

Office buildings are significant consumers of energy: buildings typically account for up to 40% of the energy use in industrialised countries [1], and of that, over 70% is consumed in the operation of the building through HVAC and lighting. A large portion of this is consumed under static control regimes, in which heating, cooling and lighting are applied according to fixed schedules, specified when the buildings were designed, regardless of how the buildings are actually used. To improve energy efficiency, the building management system should operate the HVAC systems in response to the actual behaviour patterns of the occupants. However, heating and cooling systems have a delayed response, so to satisfy the needs of the occupants, the management system must predict the occupant behaviour. The prediction system should be accurate at both bulk and individual levels: the total number of occupants of a building or a zone determine the total load on the HVAC system, while knowing the presence and identity of an occupant of an individual office allows us to avoid waste through unnecessary heating or cooling without discomforting the individual.

We believe that in most office buildings, the behaviour of occupants tends to be regular. An occupant's behaviour may relate to the time of day, the day of the week or the time of year. Their behaviour on a given day may also depend on their

Conor Ryan · Kenneth N. Brown

Cork Constraint Computation Centre, Department of Computer Science, University College Cork, Ireland, e-mail: cryan@4c.ucc.ie · k.brown@cs.ucc.ie

location earlier on that day or on their most recent sequence of movements. We require a system which is able to recognize these time and feature based patterns across different levels of granularity from observed data. Further, many office users now use electronic calendars to manage their schedules, and information in these calendars may support or override the regular behaviour. The reliability of the calendar data will depend on the individual maintaining it, so the prediction system needs to be able to learn occupant-specific patterns from the calendars.

We propose the use of association rule mining for learning individual occupant behaviour patterns. We wish to find patterns of any kind which can be used to predict occupant movements, for which association rule mining is ideal as it is designed to find any useful patterns in a dataset. We use the Apriori algorithm [2], and show how the algorithm can be extended to represent time series, incorporating calendar entries. We then propose a number of transformations of the learning mechanism, pruning itemsets and rules to focus in on useful rules, and extending the generation of itemsets in areas where useful patterns will be found. Finally we describe a further modification of this approach which incorporates time-independent sequences. We evaluate the performance on two sets of actual occupant data, and show up to 76% and 86% accuracy on each set respectively.

The remainder of this paper is organized as follows: Section 2 provides an overview of association rules and the existing work on location prediction. Sections 3 and 4 detail the modifications we make to Apriori to make timeslot-specific and timeslot-independent predictions respectively. In Section 5 we outline the datasets we use for evaluation and the other approaches we evaluate against and present our results. We conclude the paper in Section 6.

2 Related Work

Existing methods for predicting occupant locations include bayesian networks [3], neural networks [4], state predictors [5], hidden markov models [6], context predictors [7], eigenbehaviours [8].

The Bayesian network approach presented in [3] predicts the occupant's next location based on the sequence of their previous locations and the current time of day and day of the week. Based on the current room and the day/time, it also predicts the duration of the occupant's stay in the current room. This results in separate predictions for the occupant's next location and for the time they will move.

The neural network approach uses a binary codification of the location sequences as input to a neural network. In [4] both local and global predictors are considered. A local predictor is a network which is trained on and predicts a particular occupant, and thus deals only with codified location sequences. The global predictor takes all occupants' location sequences, along with associated occupant codes, as training data, and can make predictions for any occupant.

The state predictor approach in [5] uses a two-level context predictor with two-state predictors. This method selects a two-state predictor based on the occupant's

sequence of previous locations. Each state within the selected predictor is a prediction; the current state is used as the prediction, and the state may then change depending on whether the prediction was accurate. Being a two-state predictor, each possible location has two corresponding states, so a maximum of two incorrect predictions for any given sequence is necessary to change future predictions, resulting in fast retraining if an occupant changes their behaviour. The second level of this predictor can alternatively store the frequencies of the possible next locations for each sequence. This makes it equivalent to a markov model approach.

These approaches all predict the occupant's next location, and with the exception of the Bayesian network, only use the occupant's recent locations. Our application requires longer term predictions and we believe there may be more general associations between the occupants' locations at different times which allow for such predictions. Association rule mining is intended to discover general patterns in data and so we propose to investigate whether association rule mining can be used to predict occupant locations.

Association rule mining was introduced in [2] as an unsupervised approach to finding patterns in large datasets. The original application was discovering patterns in datasets of transactions, where each transaction was a market basket, i.e. a set of purchased items. In that application items were literals, simple strings which are either present or absent in a transaction; however the algorithm can be applied without modification to sets of attribute/value pairs. We chose Apriori as it is the most basic association rule mining algorithm and thus simplest to modify.

Let U be a universe of items. A dataset D is a set of instances $\{I_1 \dots I_n\}$, where each instance is a set of items from U. An itemset X is a subset of U. The frequency of X, $freq(X)$, is the number of instances I in D for which $X \subseteq I$, while the support is $supp(x) = freq(X)/|D|$. An association rule is an implication of the form $X \Rightarrow Y$ where X and Y are itemsets such that $X \cap Y = \emptyset$. This rule states that each instance which contains X tends to contain Y. The support of the rule is $supp(X \cup Y)$. The confidence of the rule is how often it is correct as a fraction of how often it applies $conf(X \Rightarrow Y) = supp(X \cup Y)/supp(X)$.

The purpose of an association rule mining algorithm is to find the set of rules which are above user-specific thresholds of confidence and support. The first step is to find all itemsets which are 'frequent' according to the support threshold. Association rules are then generated from these itemsets, and any rules which fall below the user-specified minimum confidence are discarded. Confidence is used to measure the reliability of a rule in terms of how often it is correct according to the training data. Finding the frequent itemsets is the more difficult step, as the desired itemsets must be found among the $2^{|U|} - 1$ itemsets which can be generated.

Apriori uses breadth first search to find all frequent itemsets. First all itemsets of size 1 are enumerated. Itemsets whose support falls below the support threshold (infrequent itemsets) are removed, as any superset of an infrequent itemset will also be infrequent. Candidate itemsets of size 2 are then generated by combining all frequent itemsets of size 1, and infrequent itemsets of size 2 are removed. This process continues, finding frequent itemsets of size n by generating candidates

from the itemsets of size n-1 and removing infrequent itemsets, until an n where no frequent itemsets exist is reached.

Once the frequent itemsets have been found, for each frequent itemset X all rules of the form $Y \Longrightarrow X - Y$ where $Y \subset X$ and $Y \neq \emptyset$ are generated, and those which do not obey the confidence threshold are discarded.

3 Adapting Association Rule Mining For Occupant Prediction

The first task in applying association rule mining is to determine the format of the dataset. We define an instance to be a single day for a single occupant, recording for each time slot the location of the occupant. It also includes a set of scheduled locations, specifying where the occupant's calendar stated they would be. Finally, each instance records which occupant and day of the week it applies to. Thus the set of attributes in our dataset is $A = \{d, o, l_i \ldots l_j, s_i \ldots s_j\}$, where d is the day, o is the occupant, l_n is the occupant's location at time slot n, and s_n is the location the occupant was scheduled to be in at time n. Our objective then is to find rules which predict the value of an attribute in $\{l_i \ldots l_j\}$ based on the other attributes. In order to be able to compare confidences meaningfully, we restrict our attention to rules which predict single attributes.

Although this format is all that is needed to run Apriori, it is unlikely to produce usable results. The items in our dataset have semantics which are critical for the eventual application, but Apriori by default treats them all as equivalent. The location attributes $\{l_i \ldots l_j\}$ represent an ordered list of time/location pairs which it is our objective to predict. However, Apriori has no concept of the importance of or ordering over these items, so it will produce rules which run counter to the order, i.e. rules which use later locations to predict earlier locations, and which make useless predictions, e.g. predicting timetable entries.

A further important attribute distinction is that $\{l_i \ldots l_j\}$ and $\{s_i \ldots s_j\}$ are actual location data, whereas d and o are data labeling the location data, i.e. meta-data. Due to this their values are in a sense fixed. For example, in an instance which describes occupant A's movements on a Monday, d and o are fixed at Monday and A respectively, whereas all the other attributes can, in principle, take any value in their domain. This affects the meaning of the support metric as the maximum support for any itemset which includes d or o will be less than 1. Since support is used to determine which itemsets are considered frequent, patterns which occur frequently for certain days and/or agents will be rated as less frequent due to the inclusion of other days and agents in the dataset.

A problem with regard to the content of the data is that the many common patterns tend to be the least interesting, while we require low frequency patterns to be found in order to make predictions in unusual circumstances. Consider for example an occupant who has a 90% chance of being in their office in any timeslot from 9am to 5pm. In this case, any pattern of the form "in at N implies in at M"

where N and M are between 9-5 will have support of at least 80%, thus all such patterns will be found. But there is no real correlation there; all these patterns could be summarized simply as “the occupant is likely to be in”. At the extreme opposite end, we have days when the occupant does not turn up at all, due to illness or other reasons – a very obvious pattern which would be represented by rules such as "out at 9,10,11 implies out at 12”. Such rules could have confidence close to 100% if the occupant tends to be in in the morning, but if absences are rare the itemset behind the rule will have such low support it won’t even be a candidate. Since enumerating every itemset is not feasible, we wish to eliminate the common uninteresting ones and focus on the less common but interesting ones.

3.1 Candidate/Rule Pruning

As mentioned above, standard Apriori has no concept of the relationships between the items in an instance which exist in occupancy data. Due to this it will by default generate some useless rules. The important features are that the location attributes $\{l_i \dots l_j\}$ represent an ordered list and that they are the only attributes we wish to predict. As an itemset which does not contain any of these attributes cannot produce a rule which predicts any of them, we eliminate itemsets which do not contain some subset of $\{l_i \dots l_j\}$ during candidate elimination.

With regard to rule generation, we only wish to predict the future based on the past (i.e. rules which obey the ordering of $\{l_i \dots l_j\}$)), and we only wish to predict a single location at a time in order to allow meaningful comparison of the rules at rule selection time. Thus our rule generation is as follows: for every itemset $\{l_i \dots l_j, x_i \dots x_j\}$, where l is a location item and x is any other type of item, l_j is the consequent and all other items are the antecedent.

3.2 Support Modification

In 2.1 we provided the typical definition of support, the proportion of the instances which contain the itemset/rule. To deal with the reduction in support for itemsets which contain metadata items, we redefine support as $supp(X) = freq(X)/max(freq(X))$. For market basket items, which can in principle occur in every instance, this is the same definition. In the case of our metadata attribute/value pairs however, this definition results in a different value which is normalized such that the maximum value of $supp(X)$ is always 1 for comparison to other support values.

Using this modified support threshold in Apriori allows it to find itemsets when have a lower support due to their metadata attributes. However this greatly increases the area of the itemset lattice which is explored for any given support

threshold. Thus, in order to conserve memory, we mine each possible combination in a separate pass. For every combination of metadata attributes/values *C*, we initialize Apriori with all itemsets of size $|C| + 1$ which are a superset of *C*, instead of standard 1-itemsets. This allows the generation of every itemset which contains that metadata combination in a separate pass.

3.3 Windowing

Some important patterns have such low support that trying to find them by simply lowering the support threshold would result in a combinatorial explosion. Instead we will use the structure of the data to target them specifically. An example of such a pattern is a full day of absence: a very obvious pattern, but one which occurs so infrequently that it won't be learned. As our location attributes form an ordered list we can define subsets of them which are consecutive, temporal windows over the location data. By mining these subsets individually, we can reduce the size of the space of itemsets while still discovering the itemsets which describe consecutive elements of the low support patterns.

We define a window as: $Win(n, m) =< d, o, l_{n\ldots n+m}, s_{i\ldots j} >$ where *i* and *j* denote the first and last timeslots, and *n* and *m* denote the beginning and length of the window respectively. In the windowing phase, we search within every window of the chosen length. This approach ignores patterns which span times which do not fit within a window. We choose to focus on patterns which occur in consecutive time slots as predicting occupant locations based on their most recent movements has been shown to work by the other approaches discussed in section 2.

For distinct patterns windowing is sufficient to find rules which will make the correct predictions should the pattern recur. Taking the example of an occupant who is absent all day, within each window we will learn that consecutive hours of absence imply absence in the next hour. Taken in combination, these rules will state that at any hour of the day, consecutive absence implies continued absence, although we are still not learning sequences in the same sense as the approaches in section 2, as the individual rules are still tied to specific time slots. These rules are added to the rules mined from the complete instances.

3.4 Rule Selection

Once the rules are generated we need a mechanism to choose a rule to make a prediction. When a prediction is required, values for any subset of the possible attributes can be supplied as an itemset *V*. A target for the prediction l_t is also given. We search the generated rules for all rules $X \Rightarrow Y$ where $X \subseteq V$ and $Y = \{l_t\}$. From these we select the rule with the highest confidence as the prediction.

4 Ordered Association Rules

In order to be able to predict occupant locations the timeslot-specific approach above requires that the occupant's behaviour correlates with the time of day. Our evaluation shows that on occupants for whom this does not hold the approach performs poorly. However, existing approaches which find patterns that don't relate to specific times are able to make accurate predictions on such occupants. We now describe a modification to Apriori which allows it to find time-independent sequences of locations using the order of the timeslots. As with the other timeslot-independent approaches, this approach relies on the occupant's most recent locations, and cannot make predictions beyond the next timeslot.

4.1 Ordered Itemsets

In order to represent time-independent sequences we define a new set of attributes $\{q_0 \dots q_j\}$ which represent an ordered list of consecutive locations. These attributes are similar to the timeslot attributes $\{l_i \dots l_j\}$, but rather than being a list of time/location pairs, $\{q_0 \dots q_j\}$ is a list of ordering/location pairs. Thus the first attribute is always '0', as it is the first element in the list. As we deal only with consecutive sequences, for any list of length *j+1*, all elements *0…j* must be present.

Ordered itemsets are itemsets of the form: $\{q_0 \dots q_j\}$. Each ordered itemset is essentially the set of itemsets $\{\{l_k \dots l_{j+k}\} : n \leq k \leq m - j\}$, where *n* and *m* are the minimum and maximum timeslots in the dataset respectively, represented as a single list. An ordered itemset may be instantiated to a time specific itemset by choosing a starting timeslot *k* and adding *k* to every attribute in the list, turning the itemset $\{q_0 \dots q_j\}$ into the itemset $\{l_k \dots l_{j+k}\}$ for the chosen timeslot *k*.

For example, take the ordered itemset {0=>O, 1=>A, 2=>O}, which signifies that an occupant is in their office, leaves for an hour, and then returns. If we set *k* to be 12:00, this itemset becomes {12=>O, 13=>A, 14=>O}, which states that an occupant is in their office at 12:00, leaves at 13:00, and returns at 14:00.

The individual timeslot-specific itemsets which the ordered itemset represents could be found separately by our original approach, however it would require each of them to occur separately with sufficiently high support, and for rules to be generated for each variation it would similarly require each to separately have sufficiently high confidence. Searching for the sequence of movements over all timeslots provides two advantages. First, that a pattern which recurs will be supported even if it recurs at different times, resulting in low support for each individual instance of the pattern, allowing us to find a pattern we otherwise wouldn't. Second, that when we generate an ordered rule from an ordered itemset, it can apply in cases where the pattern had low support, or even in a timeslot where the pattern has never occurred previously.

4.2 Confidence and Support

Confidence and support are defined in the same way for ordered itemsets as for timeslot-specific itemsets. However, *freq(X)* for an ordered itemset *X* counts the number of occurrences of the sequence in each transaction, i.e. $freq(\{q_0 \dots q_j\}) = \sum_{k=n}^{m-j} freq(\{l_k \dots l_{j+k}\})$. This results in values greater than 1 for support, however the anti-monotonicity property still holds, and so using a support threshold to eliminate candidates is still valid for this definition of support, although the threshold no longer represents the fraction of the dataset in which the itemset occurs. We considered alternative definitions for support for ordered itemsets, however they failed to correctly represent the relative frequency of the itemsets of different sizes and/or broke the anti-monotonicity property Apriori relies on.

Confidence for ordered rules is still the fraction of the times that it applies that it is correct, however as with support it is now considered over all occurrences of the time-independent sequence that the rule is based on.

4.3 Candidate Generation

To generate itemsets we use a modified form of Apriori's candidate generation. Since the first attribute of any ordered itemset must be '0' and the attributes must be consecutive, any two itemsets of the same length will have the same attributes. This means that we cannot generate candidates by combining itemsets of the same length, as the only case where a longer itemset would be generated would be when the itemsets have different values for the same attributes, which will result in a support of zero.

Instead, for every possible pair of ordered itemsets of length *j*, we increment the attributes of one of the itemsets by one, shifting the sequence to the right by one timeslot, and then combine them. Thus we combine two itemsets $\{q_0 \dots q_j\}$ and $\{r_0 \dots r_j\}$ if $\forall n: 1 \leq n \leq j-1: q_{n+1} = r_n$, essentially if the latter sequence can provides one item to be appended onto the former sequence. Aside from this modification, candidate generation proceeds as previously described.

4.4 Rule Selection

Rule selection proceeds in the same manner as in the timeslot-dependent approach, except that ordered rules are instantiated to check their applicability. Given the prediction target l_t and the attributes/values to predict on *V*, for each ordered rule $\{q_0 \dots q_{j-1}\} => \{q_j\}$, we set $j = t$ to get the rule $\{l_{t-j} \dots l_{t-1}\} => \{l_t\}$,

before checking whether $\{l_{t-j} \dots l_{t-1}\} \subseteq V$. As before, the applicable rule with the highest confidence is chosen. The ordered and timeslot-specific rules can be combined into a single ruleset and used simultaneously, however simply combining them provides no advantage over selecting and using only the more appropriate ruleset for the dataset.

5 Experimental Evaluation

To test our approach we use two datasets: data recorded by occupants of the 4C lab in UCC, and data from the Augsburg Indoor Location Tracking Benchmarks [9]. We also evaluate three methods which were used in [6] and [10], a HMM, an Elman net and a frequency predictor. The HMM and Elman net were evaluated using the respective tools in MATLAB, while we implemented the frequency predictor ourselves based on the frequency analysis context predictor in [5].

To gather data to test our approach, six occupants of the 4C lab in University College Cork including the authors manually recorded their movements over a period of 5-15 months using google calendar. Each occupant recorded their location by room code if within a campus building, or marked themselves as 'away' if off campus. The data was recorded from 8am to 6pm with half-hour granularity, with any occupancy of significantly shorter duration than 30 minutes filtered out. The occupants also recorded their timetables for the time period, which recorded the locations they were scheduled to be in in the same format as the record of their actual movements. 20 locations were frequented by the occupants including the 'away' location. The test set for this evaluation was the most recent 2 months of data for each occupant, while the training set was all the preceding data each occupant had recorded, which covered between 3 and 13 months.

The Augsburg dataset contains data on 4 occupants for 1-2 weeks in summer and 1-2 months in fall. The format of the dataset is a series of timestamped locations for each occupant. As the data is not broken down into timeslots, we only compare the sequence based approaches on this version of the dataset. In order to be able to apply our timeslot-dependent approach to the data, we converted it to the same timeslot format as our gathered data. An occupant's location in each timeslot is the location in which they spent the majority of that timeslot. Following this conversion there are 7 locations frequented by the occupants including 'away'. We compare all approaches on this version of the dataset.

5.1 Experiments

We generate time-dependent rules from each training set using a minimum support and confidence of 0.2 and 0.5 respectively. During windowing we use a window size of 3 slots and a minimum support of 0.05. We generate time-independent

rules using no support or confidence threshold in order to maximize the coverage of the resulting ruleset. Instead of the support threshold we limit the length of generated rules to 2 items as shorter sequences have proven to be more reliable predictors of occupants' next locations.

Following are the configurations used for the comparison approaches. In the frequency predictor we use a maximum order of 2, again due to shorter sequences being more reliable predictors. For the hidden markov model we use 7-8 hidden states depending on the dataset as this maximized accuracy. The elman net uses the MATLAB default settings for layer delays and hidden layer size as no other combinations of values tested produced higher overall accuracy.

A feature of the frequency predictor is that it continues to train as it predicts; in our evaluation we allow all the sequence-based predictors to retrain with the days they have already predicted included in their training set, in order to maximize their accuracy. Timeslot Apriori predicts with only the initial training run, as the time taken to train makes retraining after every predicted day unfeasible.

We test all approaches on their accuracy in predicting the occupant's exact location in every time slot. As the sequence-based approaches use only recent occupant movements they can only predict for the next timeslot. Timeslot Apriori is tested on its ability to predict Next-Slot and Next-Day, and with or without timetable data available. The former determines whether $l_i \dots l_{n-1}$ are available, where n is the time slot being predicted, 'Next Slot' if this information is available, and 'Next Day' if not. The latter determines whether the values of $s_i \dots s_j$ are available when predicting, and is marked 'no Timetable' if they are not.

5.2 Results

Table 1. Timeslot Apriori accuracy by prediction type on UCC dataset

Next-Slot	Next-Day	Next-Slot (No TT)	Next-Day (No TT)
86%	75%	85%	71%

Table 1 shows the accuracy of Timeslot Apriori making different types of predictions on the UCC dataset. The highest accuracy is achieved on Next-Slot predictions, which confirms that recent occupant movements, on which all the sequence-based approaches rely, are the most reliable predictor of an occupant's next location. For Next-Slot predictions, the timetable only helps marginally. Next-Day predictions are significantly less accurate as they must be made based solely on the occupants' historical data without any knowledge of their movements during the day, however for these predictions the timetable does make a difference to the accuracy.

Figure 1 below shows these results broken down by occupant, and includes Ordered Apriori making Next-Slot predictions. Occupant A's movements are very homogenous, so they are easily predicted by all prediction types. Occupant B has

the most varied movements and the most scheduled events of all the occupants. This makes them harder to predict than the other occupants, however the addition of timetable data makes the largest difference on this occupant, especially on next-day predictions where it allows over 20% higher accuracy. For the other occupants, whose movements follow general patterns with minimal scheduled events, their predictability is primarily contingent on the availability of real-time location data, resulting in one accuracy level for Next-Slot and a slightly lower one for Next-Day. However, Timeslot and Ordered Apriori do vary slightly on Next-Slot predictions, indicating that they do not make exactly the same predictions.

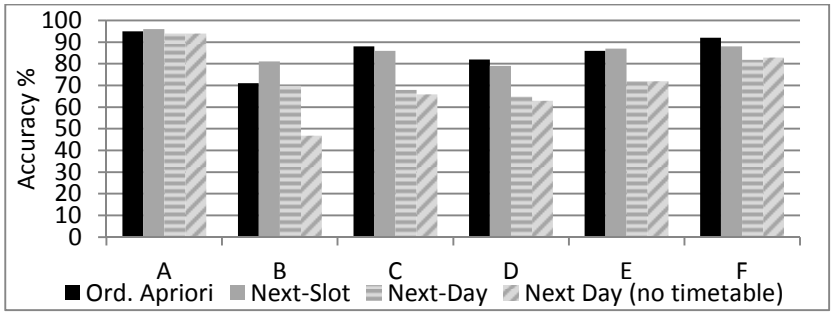

Fig. 1. Accuracy across all occupants on the UCC dataset

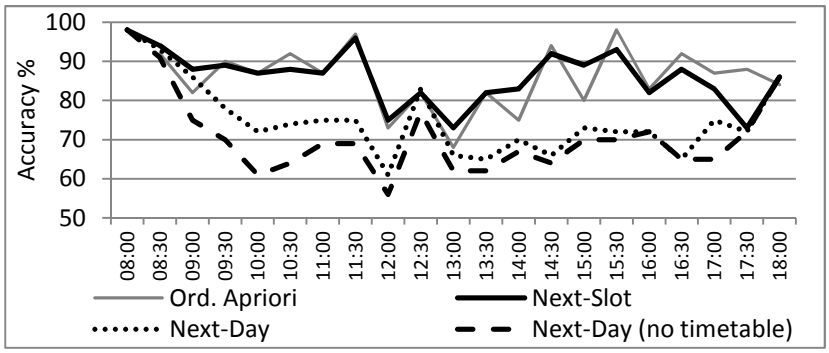

Fig. 2. Accuracy across the day on all occupants in the UCC dataset

Figure 2 breaks the prediction accuracy down by timeslot, showing how the different prediction types fare at different times of day. In general we can see that they match in the morning, evening and around lunch, when the occupants are most predictable. Outside those times, Next-Day predictions drop down as the occupants' activities at those times are more variable, and real-time information is required to maintain the accuracy level. Due to occupant B, Next-Day predictions are improved with the use of timetable data, particularly in the morning.

Table 2. Occupant B and E Confusion Matrices for Timeslot Apriori Next-Slot predictions

		O	M	B	A
Actual Location	O	304	0	1	66
	M	11	19	0	2
	B	21	0	145	7
	A	37	1	0	214

	O	M	B	A
O	148	1	3	65
M	1	6	0	0
B	3	0	3	0
A	29	0	0	581

Table 2 shows the confusion matrices for occupants B and E, classifying the location either as in their own office (O), in a specific group meeting (M), any other room in the building (B) or away (A). The matrices show that the primary source of errors is reversing Office and Away, being uncertain whether the occupant will be in. For the group meeting and events elsewhere in the building, both occupants tend to be either predicted correctly or predicted to be in their office; recognizing that the occupant is in, but not that they will be leaving their office.

Table 3. All approaches accuracy for Next-Slot on UCC dataset

HMM	Elman Net	Freq. Predictor	Timeslot Apriori	Ord. Apriori
77%	86%	87%	86%	86%

Table 3 shows that the sequence predictors generally match Timeslot Apriori for making Next-Slot predictions on the UCC dataset. The HMM performs worse because retraining it on a dataset this size was unfeasible, and so it was only trained on the initial training set, unlike the other sequence-based approaches.

Table 4. All approaches accuracy for Next-Slot on Augsburg dataset

HMM	Elman Net	Freq. Predictor	Timeslot Apriori	Ord. Apriori
74%	76%	77%	39%	76%

Table 4 shows the results of Next-Slot predictions on the Augsburg dataset. The results for all methods are approximately equal, except for Timeslot Apriori, which performs poorly. The occupants in the Augsburg dataset follow predictable patterns in their movements, however they follow these patterns at irregular times. Timeslot Apriori cannot learn patterns independently from the time at which they occur; if an occupant repeats the same sequence of movements in a different timeslot, Timeslot Apriori will attempt to learn separate rules for each timeslot. This is exacerbated by the fact that the Augsburg dataset contains very little training data; Timeslot Apriori could potentially learn every possible instantiation of the occupants' patterns separately, but there aren't enough examples present to do so. The other approaches are successful as they are able to learn the sequences independent of the time at which they occur. Ordered Apriori is similarly time-independent and thus matches the other sequence-based approaches. As Timeslot Apriori performs poorly on this dataset, we do not attempt Next-Day predictions.

Table 5. Comparison of Sequence Learners on Original Augsburg Dataset

Occupant	HMM	Elman Net	Frequency Predictor	Ord. Apriori
A	62%	57%	60%	59%
B	55%	58%	58%	58%
C	47%	46%	47%	47%
D	54%	50%	56%	56%
All	55%	53%	55%	55%

Table 5 shows the accuracy of the sequence learners on the unmodified Augsburg dataset. As this version of the dataset does not have timeslots, predictions are of the occupant's next location only with no concept of time. While the average accuracy across all four occupants is approximately the same for three of the approaches, there are minor variations in accuracy on each individual opponent. These results show that Ordered Apriori is also able to match existing methods on pure sequence data. The results are lower than the corresponding results in [10] as we include predictions when the occupant is leaving their own office and cases where a prediction was not made.

6 Conclusions and Future Work

In this paper we presented two approaches for applying association rule mining to the problem of predicting future occupant locations. We implemented our approaches using modifications of a standard association rule mining algorithm and presented experimental results which show that our modifications can predict actual occupant movements with a high degree of accuracy.

Compared to standard approaches, our timeslot-dependent approach has some advantages and disadvantages. Our aim with this approach is to predict for any time slot using whatever information is available, whether it be the occupant's recent movements on the same day or simply their historical patterns, allowing it to use a wider variety of data to make a wider variety of predictions. This is successful on the UCC dataset, however it performs poorly on the Augsburg set even for Next-Slot predictions due to the time-independent nature of their movements.

Our sequence based approach matches the capabilities of the existing approaches, giving it the same accuracy as those approaches, although the same limitations in terms of the information which is used and what predictions can be made. Using the rulesets generated by both approaches, we are essentially able to predict with whichever approach is more suitable for any given prediction. Thus we can match the accuracy of existing methods for the predictions they can make, while being able to make a wider array of predictions.

Since Timeslot Apriori and Ordered Apriori do not make exactly the same predictions for Next-Slot prediction, it is possible that intelligent selection of the rules

from both sets that we could improve accuracy over what either approach achieves alone. We intend to investigate this possibility as part of our future work.

The approaches evaluated in this paper only consider patterns in the occupant's specific location. There may be patterns which support more general predictions, such as that an occupant will be out of their office without predicting exactly where they will be. There is existing work [11] on extending Apriori to mine association rules with taxonomies. As part of our future work we intend to similarly extend our approach, allowing us to make more generalized predictions.

The eventual goal is to integrate this approach with occupant localization systems such as [12], and predictive control systems such as [13]. Using occupant localization data, a system based on our approach could provide the predictions necessary for more energy efficient building control.

References

1. World Business Council for Sustainable Development, Energy Efficiency in Buildings: Facts and Trends – Full Report, (2008).
2. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules in Large Databases. Proceedings of the 20th International Conference on Very Large Databases, 487-499, (1994).
3. Petzold, J., Pietzowski, A., Bagci, F., Trumler, W. and Ungerer, T.: Prediction of Indoor Movements Using Bayesian Networks. LoCA 2005, LNCS 3479, 211–222, (2005).
4. Vintan, L., Gellert, A., Petzold, J., and Ungerer, T.: Person Movement Prediction Using Neural Networks. Technical Report, Institute of Computer Science, University of Augsburg, (2004).
5. Petzold, J., Bagci, F., Trumler, W., Ungerer, T., and Vintan, L.: Global State Context Prediction Techniques Applied to a Smart Office Building. The Communication Networks and Distributed Systems Modeling and Simulation Conference, San Diego, CA, USA, (2004).
6. Gellert, A., Vintan, L.: Person Movement Prediction Using Hidden Markov Models. Studies in Informatics and Control, Vol. 15, No.1 (2006).
7. Voigtmann, C., Sian Lun Lau, David, K.: A Collaborative Context Prediction Technique. Vehicular Technology Conference, (2011)
8. Eagle, N., Pentland, A. S.: Eigenbehaviors: Identifying structure in routine. Behavioral Ecology and Sociobiology, vol. 63, no. 7, 1057–1066, (2009).
9. Petzold, J.: Augsburg Indoor Location Tracking Benchmarks. Context Database, Institute of Pervasive Computing, University of Linz, Austria. http://www.soft.uni-linz.ac.at/Research/Context Database/index.php, (2005).
10. Petzold, J., Bagci, F., Trumler, W., and Ungerer, T.: Comparison of Different Methods for Next Location Prediction. EuroPar 2006, LNCS 4128, 909–918, (2006).
11. Srikant, R., and Agrawal, R.: Mining Generalized Association Rules. Proceedings of the 21st Internationa1 Conference on Very Large Databases, 407-419, (1995).
12. Najib, W., Klepal, M., Wibowo, S.B.: MapUme: Scalable middleware for location aware computing applications. International Conf. on Indoor Positioning and Indoor Navigation (IPIN), (2011).
13. Mady, A. E.D., Provan, G., Ryan, C., Brown, K. N.: Stochastic Model Predictive Controller for the Integration of Building Use and Temperature Regulation. AAAI (2011).

This work is funded by Science Foundation Ireland under the grants ITOBO 07.SRC.I1170 and Insight 12/RC/2289