Occupant Location Prediction in Smart Buildings Using Association Rule Mining

Conor Ryan
Department of Computer Science
National University of Ireland, Cork

April 2016

Thesis submitted to the National University of Ireland in fulfilment of the requirements for the Degree of Doctor of Philosophy in Computer Science

Research Supervisor: Dr. Kenneth N. Brown
Head of Department: Prof. Barry O' Sullivan

Declaration

All the work presented in this thesis is original. This work was carried out under the supervision of Dr. Kenneth N. Brown between October 2008 and April 2016 in the Department of Computer Science, University College Cork, Ireland. This dissertation has not been submitted in whole or in part for any other degree, diploma or qualification at any other institution.

____

Conor Ryan

April 2016

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. We present an approach to occupant location prediction based on association rule mining, allowing prediction based on historical occupant locations.

Association rule mining is a machine learning technique designed to find any correlations which exist in a given dataset. Occupant location datasets have a number of properties which differentiate them from the market basket datasets that association rule mining was originally designed for. This thesis adapts the approach to suit such datasets, focusing the rule mining process on patterns which are useful for location prediction. This approach, named OccApriori, allows for the prediction of occupants' next locations as well as their locations further in the future, and can take into account any available data, for example the day of the week, the recent movements of the occupant, and timetable data. By integrating an existing extension of association rule mining into the approach, it is able to make predictions based on general classes of locations as well as specific locations.

Acknowledgments

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

Thanks to the anonymous volunteers who contributed to the occupant location dataset by recording their locations and timetables.

Thanks to my family and friends for their support before and during this work.

Special thanks to my supervisor Dr. Ken Brown for his invaluable guidance throughout my research.

List of Tables

List of Figures

Contents

1. Introduction

1.1. Motivation

Buildings are significant consumers of energy in industrialised countries. Recent figures show that commercial and residential buildings are responsible for 41% of total energy consumption in the US (EIA 2015) and 39% in the EU (Eurostat 2014). In non-domestic buildings, Heating, Ventilation and Air Conditioning (HVAC) systems, which control the temperature and air quality in buildings, are estimated to be responsible for up to 57% of energy consumption (Pérez-Lombard et al. 2008). Traditionally buildings are controlled under the assumption of fixed occupancy patterns, with HVAC systems operating on a fixed schedule, regardless of the actual use of the building. If HVAC schedules could be set based on the actual occupancy patterns, energy savings of up to 40% would be possible (Nguyen et al. 2013).

Heating and cooling systems cannot respond instantly, so to take occupant movements into account when controlling the building, the occupants' movements must be predicted in advance. The HVAC system may include micro level actuators which respond quickly, for example electric heaters, or building-wide heating/cooling systems which respond slower, for example overnight heating or cooling of the building's thermal mass. Thus depending on the system, it may be necessary to make predictions into the near future, such as occupants' next locations, or further into the future, such as occupants' movements throughout the day tomorrow. Such a prediction system should be able to predict occupancy at the bulk and individual levels. Bulk predictions are necessary as knowing the total number of occupants that will be in the building tells us what the total load on the HVAC system will be. Data on future movements of individuals on the other hand allows us to avoid wasting energy through unnecessary heating or cooling without discomforting the individual.

Prediction of occupants' future locations necessitates data on their current locations, which can be recorded to create a database of historical occupant locations. An increasing amount of data on occupant movement in buildings is becoming available, from both dedicated tracking systems and using existing infrastructure. Examples which can potentially provide room-level granularity include the use of wireless access points (Melfi et al. 2011), radio frequency Identification (RFID) tags (S. Li et al. 2011), passive infrared (PIR) sensors (Yun et al. 2014), cameras (Corvee et al. 2012) and mobile phone masts (Otsason et al. 2007). Swipe card systems implicitly track occupants as they record entrances and exits for security reasons. Online calendars and room booking systems store information on where occupants are supposed to be in the future. Any system which can record occupants' locations throughout the day is sufficient to provide the data needed to make predictions, though more predictions are possible if real time updates on occupants' current locations and data on where occupants are going to be are also available.

Occupant behaviour can depend on many factors including: Time of day, day of week, recent movements, earlier movements, time spent at current location, etc. Their behaviour can also relate to unpredictable factors, such as being summoned for an ad-hoc meeting or non-work-related events. An occupant prediction system should be able to determine which factors in the dataset correlate with occupants' behaviour, individually or as a whole, and leverage that data when making predictions. Extra data, for example schedule data from an online calendar system, may also be provided, and should be incorporated into the predictions when available.

Association rule mining, introduced in (Agrawal et al. 1993), is a data mining technique designed to find all frequent patterns in a dataset in an unsupervised fashion. In the context of a dataset containing historical occupant movements, this approach should find any and all patterns in the occupants' movements. This is a suitable basis for the approach presented in this thesis as the aim is not any one type of prediction – the approach may need to predict any occupant's location at any time, with differing data available on which to base the prediction. Finding all the patterns in the dataset accommodates this. This also allows for predictions into both the near and distant future, assuming that there are patterns to the occupants' movements which allow for such a range of predictions, because the difference between these types of predictions is simply the presence or absence of real-time tracking data. Finally, association rules are an established approach to data mining, with a number of extensions to find new types of patterns, which can easily be applied to occupant location prediction once the base approach is in place.

1.2. Research Goals and Contributions

The goal of this research is to extend association rule mining such that it can predict patterns of occupancy in buildings. The extended algorithm should match or exceed the accuracy of existing approaches, while also making a wider variety of predictions. The hypothesis of this thesis is as follows:

> *Association rule mining can be applied to data on the past movements of a building's occupants, generating rules which predict their future locations. Such an algorithm can be modified to take account of the specific attributes and metadata that occupancy data includes in order to reach higher accuracy. Furthermore the algorithm can be extended to learn more general concepts and sequences in the data to extend the range of patterns which are found and can be used for prediction.*

The contributions of this research are:

- *Mining patterns specific to occupant location prediction*

  Association rule mining, being an unsupervised process, can be applied to appropriately formatted occupancy data as-is, and will generate a set of rules with some degree of predictive power. However, some types of useful patterns will not be found, while some types of useless patterns will. This thesis investigates these failures and by modifying the algorithm to find more useful patterns, while excluding unnecessary ones, brings the predictive power up to par with existing approaches, while keeping the flexibility of the approach.

- *Learning pure location sequences with association rules*

  While predictions are usually occupant locations at specific times of day, for some types of occupant their recent locations, ignoring time of day, are the most reliable predictor. This thesis investigates the difficulty in learning sequences of locations independent of the time of day with association rules, and modifies the association rule mining algorithm to learn these patterns to produce sets of rules containing both types of prediction.

- *Learning general location patterns*

  One of the strengths of association rule mining is that there is existing work which extends the approach in various ways. An example is generalised association rules, which are designed to find general patterns which apply to multiple specific items. This thesis applies these generalised rules to occupant prediction to increase prediction accuracy by making types of prediction which could not be made before.

- *Frequency based occupant model*

  Building on established work in occupant modelling, this thesis develops a synthetic dataset generator which is, like the prediction method, based on occupants' historical location frequency throughout the day. A detailed evaluation is performed on a model of a set of real occupants to show that the model can capture realistic occupancy patterns.

1.3. Thesis Outline

The rest of this thesis is structured as follows. Chapter 2 provides background and discusses related work in the areas of association rule mining, occupant location prediction and occupant modelling.

Chapter 3 details the datasets used in evaluating OccApriori. The evaluation uses two collected datasets, one of which was collected externally and has been used for similar evaluations in the past, the other collected by several occupants in UCC for the purposes of this research. The chapter then develops an occupant model, including the structure of the model itself and an overview of the process of modelling the occupants from the UCC dataset, and close with an evaluation of the similarity of the location data generated by that model and the original dataset.

Chapter 4 discusses the application of association rule mining to occupant prediction in detail, opening with an overview of the issues which arise when the unmodified algorithm is used for the task. It follows with a description of OccApriori in detail, starting with the structure applied to the occupancy data in order to be able to make useful predictions, and moving on to solutions to each of the issues identified with the unmodified algorithm. The chapter then evaluates OccApriori and several existing approaches on the two collected datasets, showing that OccApriori can match their accuracy on the type of predictions they make on one of the datasets. On the other dataset, the extremely variable movements of the occupants prove difficult for OccApriori to predict. This chapter briefly discuss this issue, and then closes with an evaluation of the contribution each component of the algorithm to the final accuracy.

Chapter 5 addresses the issue of predicting more variable occupant movements introduced in the previous chapter. It starts by discussing in detail why OccApriori has difficulty with these occupants, while other approaches are able to predict them with a high degree of accuracy. With the issue identified, it proposes a new type of rule that encapsulates sequences of locations independent of any particular time of day. Such rules can capture the movement patterns of and successfully predict these occupants. The chapter then describes the modifications to the algorithm necessary to mine these new rules, and evaluates their performance, finding that OccApriori can now match existing approaches on the more variable occupants.

Chapter 6 discusses the application of generalised association rules to occupant prediction. It examines two cases in particular; one where the aim is to make a generalised prediction of the occupant's future location, and another where the aim is to use general patterns in the past to make exact future predictions that wouldn't otherwise be possible. It then details the application of generalised association rule mining to these problems using OccApriori, and show how the generalised rules allow for the extra predictions. As neither of the collected datasets contains sufficient examples of the two types of prediction discussed, the model of the UCC occupants from Chapter 3 is modified to include such examples. The chapter closes with an evaluation of the performance of OccApriori with and without the generalisation enabled, showing that OccApriori works as proposed on a practical example.

Chapter 7 closes the thesis with a discussion of the work presented and of potential avenues of future research in this area.

The software implemented during this research and the anonymised version of the collected occupant location dataset can be found at (Ryan 2015). Some of this work has already been published in peer-reviewed publications:

Ryan, Conor, and Kenneth N Brown. 2012. "Occupant Location Prediction Using Association Rule Mining." In *Proceedings of Workshop on AI Problems and Approaches for Intelligent Environments*, 27–32. (Ryan et al. 2012)

This paper contains parts of Chapters 3 and 4.

Ryan, Conor, and Kenneth N Brown. 2013. "Predicting Occupant Locations Using Association Rule Mining." In *Research and Development in Intelligent Systems XXX*, 63–77. (Ryan et al. 2013)

This paper contains parts of Chapters 4 and 5.

2. Related Work

This chapter provides background and discusses existing work in the areas of occupant location prediction, association rule mining, and occupant modelling. Section 2.1 discusses the state of the art in occupant location prediction with a focus on approaches which, as is the aim of this thesis, can predict occupants further into the future than their next location or use additional context such as the time of day or day of week when making predictions. It also describes in detail several next location prediction approaches which are compared against in the evaluations in later chapters. Section 2.2 provides an overview of what association rule mining is and what it is used for. This is followed by a description of the association rule mining algorithm which the approach in this thesis is built on, as well as descriptions of some other well-known algorithms. Section 2.3 follows with an overview of several extensions of association rule mining which find extra types of rules that may be useful for occupant location prediction. Section 2.4 provides an overview of work on occupant modelling, focusing on models which capture the level of detail in occupant movements which this thesis aims to learn and predict.

2.1. Occupant Location Prediction

This section discusses various existing approaches to occupant location prediction. The priorities of these approaches vary, and so they make different types of prediction using different data. The approaches discussed are broken down into four general classes, and presented in order of relevance to the approach developed in this thesis. Section 2.1.1 discusses approaches which focus on predicting occupant location in the future, i.e. their location significantly later on the current day or on future days, for example where an occupant will be tomorrow at 3pm. By contrast the approaches in section 2.1.2 focus more on prediction in the nearer future, such as an occupant's next location after their current location, but use more context when making their predictions, for example prediction the next location based on the occupant's duration of stay in the current location. In section 2.1.3 older approaches are covered which focused purely on predicting the occupants' next locations with limited context, and finally section 2.1.4 briefly discusses work which extends beyond prediction of occupants of office-type buildings.

2.1.1. Future Location Prediction

Relatively few approaches focus on predicting occupants' locations in the future. One such approach is (Furey et al. 2012), which uses a discrete Bayesian filter to predict occupant movements as part of the HABITS occupant tracking system. The focus of the approach is occupant location tracking, the intention being to fill in gaps in imperfect existing tracking systems. For example, if tracking an occupant based on the wireless access points (APs) they are in proximity to, they may be temporarily not be in range of the number of APs required to give accurate location data. At this point HABITS predicts their movements to fill in the blind spot. The system can factor in real-time tracking data, the topology of the locations in question and, as the name suggests, the habitual patterns of the occupants when making a prediction.

The system is also capable of making location predictions further into the future. The habitual element of the system is based on the concept of "preferred paths" – repeated journeys occupants make from one node to another (not necessarily adjacent) node. These preferred paths inform the system when it is trying to predict occupant movements in real time by determining if they are following a preferred path, but the system can also determine from the historical data the likelihood that occupants will follow these paths in the future.

The authors break the day down into three time periods, morning, lunch and evening. From the historical data, the system can learn the frequency, and therefore probability, with which occupants follow each preferred path in each time period on each day of the week. Their attempts to predict occupants on this basis results in an overall accuracy of 70%, varying from 55%-82% per occupant.

In (S. Y. Kim et al. 2014) individual's locations are predicted based on personality factors using a neural network. Based on previous work in (H. Kim et al. 2012), location data recorded by the mobile devices of five volunteers is clustered into significant locations, resulting in a probability profile of presence in each of three locations (School, House and Etc.) across the 24-hour day. For the same five volunteers personality factors are collected using a questionnaire. Using the probability profiles and associated personality factors as input, back propagation is then used to train a neural network to predict the probability of presence in each location based on the personality factors of the target individual.

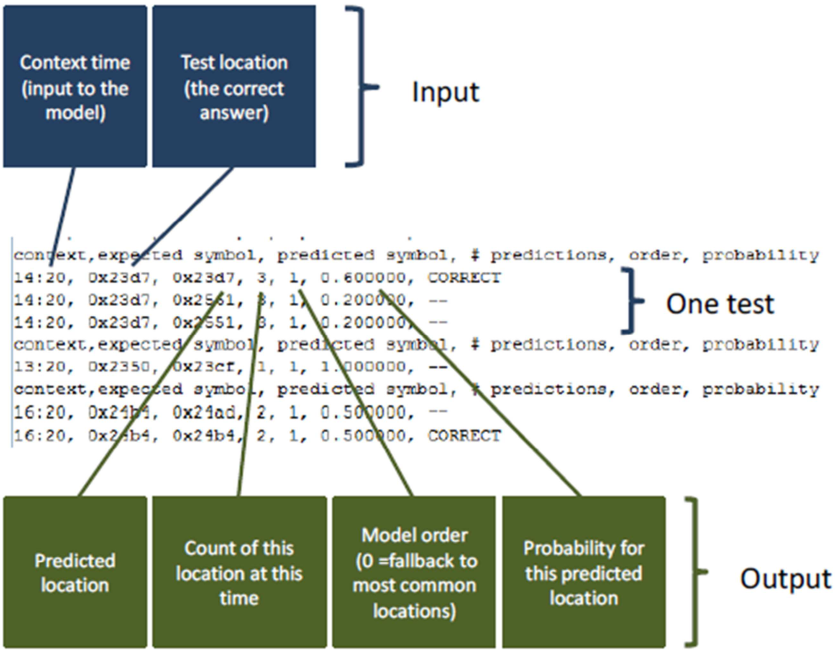
This approach is capable of predicting the likelihood of an individual being at each of the three locations at any given hour, as long as their personality factors are known. Unusually, the identity of the target occupant is not an input when making the prediction; individual occupant patterns are not learned, only the personality factors and the target time are inputs. The approach is evaluated on the same individuals used to train it. While no quantitative measure of accuracy is given, the actual and predicted profiles of presence for each participant are shown, and in general the predictions achieve a good fit to the source data.

Non-linear time series analysis is applied to historical location data in (Scellato et al. 2011) to predict future times of arrival and durations of stay at significant locations. For each individual, their history of visits to their significant locations is broken down into two time series per location, one series comprising the visit start times and the other the visit durations. Given a sequence of recent visit times for a location, future visit times are predicted by finding similar sequences of visit times earlier in the visit time time series, and averaging the times which follow those sequences. The next n visit times can be predicted, but as n increases the prediction accuracy may drop. Durations are predicted by averaging the durations which correspond with the times used for the time prediction.

While this approach cannot make a direct prediction of an individual's location at a given time in the future, such predictions can be extrapolated as follows. For each of the target individual's significant locations, a sequence of next visit times and durations is predicted such that the sequence extends past the target time. The sequences are then checked for a visit which occurs during the target time. If such a visit exists, its location is predicted. If multiple visits are predicted to occur at the target time, one of the locations is randomly selected. If no visit matches the target time, the occupant is predicted to not be at a significant location.

The approach is evaluated on four datasets, each of which have their location data broken down into significant locations. Individuals spend a maximum average of 15% of their time in significant locations across the four datasets. Due to this, accuracy is only considered for predictions of being in a significant location; with 85%+ of the data being outside the significant locations, including predictions of individuals being outside their significant locations in the accuracy could skew the value. The approach is evaluated predicting a range of amounts of time into the future. It achieves over 90% accuracy on the most predictable dataset for predictions five minutes into the future, dropping to just under 80% at an hour. The longest time tested was eight hours, at which point the maximum accuracy achieved was approximately 50%.

In (Burbey et al. 2008) a variable-order Markov model is used in different configurations to predict occupants' next locations and future locations. A Markov model predicts the next element in a sequence based on the most recent n elements, n being the order of the model. The approach uses the Prediction-By-Partial-Match algorithm, which lowers the order of the model if it cannot make a prediction at the current order. The dataset on which the model is trained and tested is formatted as a sequence of the form $time_n, location_n, time_{n+1}, location_{n+1}$ …, where the times are regular ten-minute timestamps. First-order and third-order Markov models are considered in the paper. The first order model can predict an occupant’s location given a timestamp as context, while the third order model can predict a location given the target timestamp, the previous timestamp, and the previous location as context. The model can also fall back to a ‘zeroth-order’ where the most frequent location is predicted with no context, if no other prediction can be made.


Context time (input to the model)
Test location (the correct answer)
Input
One test
Predicted location
Count of this location at this time
Model order (0 =fallback to most common locations)
Probability for this predicted location
Output

```
context,expected symbol, predicted symbol, # predictions, order, probability
14:20, 0x23d7, 0x23d7, 3, 1, 0.600000, CORRECT
14:20, 0x23d7, 0x2561, 3, 1, 0.200000, --
14:20, 0x23d7, 0x2551, 3, 1, 0.200000, --
context,expected symbol, predicted symbol, # predictions, order, probability
13:20, 0x2350, 0x23cf, 1, 1, 1.000000, --
context,expected symbol, predicted symbol, # predictions, order, probability
16:20, 0x24b4, 0x24ad, 2, 1, 0.500000, --
16:20, 0x24b4, 0x24b4, 2, 1, 0.500000, CORRECT
```

Figure 2-1 – Example output from the first-order Markov predictor from (Burbey et al. 2008)

As the model is based on the sequence of (time, location) pairs, the third order model only predicts the occupant’s next location. However the first order model has no context aside from the target timestamp, and thus doesn’t differentiate between predicting the next location and future locations. However, as it has no further context, it can only predict the location that the occupant was most frequently in at the target time. Further context cannot be easily added, as adding context actually involves creating a new higher order model with a whole new set of possible states.

The work is extended in (Burbey 2011) to consider the question of when an occupant will be in a specific location. The Markov model performs poorly (~10% accuracy) for this type of prediction, as when asking for a time rather than a location there are many more possible answers and it is harder to determine the correct one. If an occupant is almost always in their office from 09:00 until 12:00 for example, asking where they are most likely to be at 10:00 is easily answered, but asking when they are most likely to be in their office would result in every timestamp from 09:00 to 12:00 inclusive as a an equally likely answer.

In the same work the “Traversing the Sequence” model, also known as “SEQ” model, is proposed. In this model the occupant location data is a sequence of (arrival time, location) pairs (as opposed to arbitrary timestamps and corresponding locations). Rather than learning the most probable output for each possible input as in the case of the Markov model, the SEQ approach goes through the original dataset when a request for a prediction is made, searching for sequences which match the context given. During the search process each instance is processed by a short algorithm which determines if the instance matches, and what its answer is if it does. As with the Markov model, the SEQ model allows different items of context, with the algorithm varying slightly depending on the context as follows.

The model can be zeroth order with no context, in which case instances which include the target location are searched for, and the times at which it occurs in each instance are recorded. The first order model takes a context time or location and searches for instances which contain the target location preceded by the context time/location. The first time the target location appears after the context is recorded. As locations can repeat, a single instance may match more than once if the context is a location, i.e. there are two first times location B appears after location A if the occupant visits A and B twice, in which case both times would be recorded. The second order model takes both a context location and time. When considering each instance, the algorithm first checks that the target location occurs. It then selects the latest timestamp in the instance which is earlier than the context time. This fuzzy match alleviates the issue of exact timestamps not matching, for example the arrival times 10am and 12am are in an instance, and the context is 11am, in which case 10am would be selected. Finally the algorithm searches for the context location in between the selected timestamp and the first occurrence of the target location after that timestamp. If all these steps are successful the instance is considered a match, and the aforementioned first occurrence of the target location's time is recorded. In all cases, once the times have been recorded from each instance which matches the context as described, the most frequent time is predicted.

The first-order Markov model achieved 87% accuracy in predicting occupant locations on data which recorded each occupant's location in ten minute blocks. The work also considers a version of the dataset which records the occupant's location every minute, on which the accuracy is lower at 78%. This is because it is easier to predict, for example, that an occupant will be out at 07:50 and in at 08:00 than it is to predict that they will be out at 07:58 and in at 07:59, due to the greater number of possible answers and greater variability in location transition time at one-minute granularity. The highest accuracy achieved by the SEQ model was 92% correct to within 20 minutes on the ten minute data with the zeroth order model, with the extra context provided to the higher order models lowering the accuracy. The highest accuracy on the one minute data was again the zeroth order model, getting 77% correct to within 20 minutes.

In (Vu et al. 2011) a Naïve Bayesian classifier is used to predict individuals' future locations, durations of stay and who they will come in contact with. For the location prediction the features used are the type of day (weekday or weekend) and the timeslot (2 hour timeslots starting at 00:00), with the location at that time on that type of day being the output of the classifier. The predicted location is then simply the most probable location based on the training data given an input type and time of day. Duration prediction outputs the mean and standard deviation of the distribution of duration of stay across instances where the individual is at the predicted location and the type and time of day match the inputs. Contact prediction works the same way as location prediction, but with individuals instead of locations.

As with the first-order Markov model above, this approach can predict an individual's location in the future given a timestamp, but in this case the context of the type of day is also included. This approach should be able to support further context as well, although this possibility is not discussed by the authors. However, due to the structure of the instances, the approach is effectively limited to future location prediction; as each timeslot appears is a separate instance, recent locations cannot be referred to. While it could be asked to predict an individual's location in the near future (i.e. next location), it would predict with the same information as if it was predicting their location tomorrow.

The approach is evaluated on location data gathered on fifty participants using Wi-Fi and Bluetooth tracking, with the full results available in (Vu et al. 2010). The evaluation consisted of ten iterations, predicting in each the location of the individual in each of 200 randomly selected test instances. For some instances the individual's location is not known; predictions on these instances are not factored into the accuracy values. This evaluation resulted in an average location prediction accuracy of over 70% for 35% of the participants, and over 45% for 80% of the participants. Duration of stay prediction was considered correct if the actual duration fell within the predicted standard deviation from the predicted mean, and resulted in an average accuracy of over 60% for 80% of participants.

2.1.2. Context-Based Prediction

The above approaches both consider predicting occupants' future locations, but have a limited ability to use extra context elements when making a prediction. Context is important as it defines the ability of an approach to make different predictions in different circumstances. Future location prediction approaches which can't take any extra context elements will always predict that an occupant will be in the same place at the same time of day. In fact the occupant's identity and the time of day are themselves context elements – as demonstrated by the zeroth order Markov model discussed above, without the time of day element only the most frequent location for the occupant is predicted. Without knowing the occupant, only the most frequent location in the dataset will be predicted. Just as it is necessary to know the occupant's identity and the target time of day in order to make a good prediction, as the correct prediction depends on these factors, it is also important to be able to include other elements of context which influence the occupants' locations. Various extra elements of context have been considered in existing approaches which focus on incorporating extra context elements into their predictions.

One such approach is (Petzold et al. 2005) a Bayesian Network approach is used to predict where an occupant will go next, when, and how long they will stay there for. The approach models a time slice (of arbitrary duration), which represents a stay in a single location, as a Bayesian network. The nodes in the network are the current room, current duration (i.e. length of stay), time of day and day of week, with current duration being dependent on the other three nodes. An occupant's sequence of movements then becomes a sequence of length n of identical networks, where n is the number of previous locations which are considered. In each time slice the current room is dependent on the time of day and day of week of the previous time slice, and on the current room in all previous time slices. In this way the model predicts an occupant's next location based on their recent locations, but also the current time of day and the day of the week. The duration of stay in both the current and predicted time slice are predicted solely off the location in the same time slice.

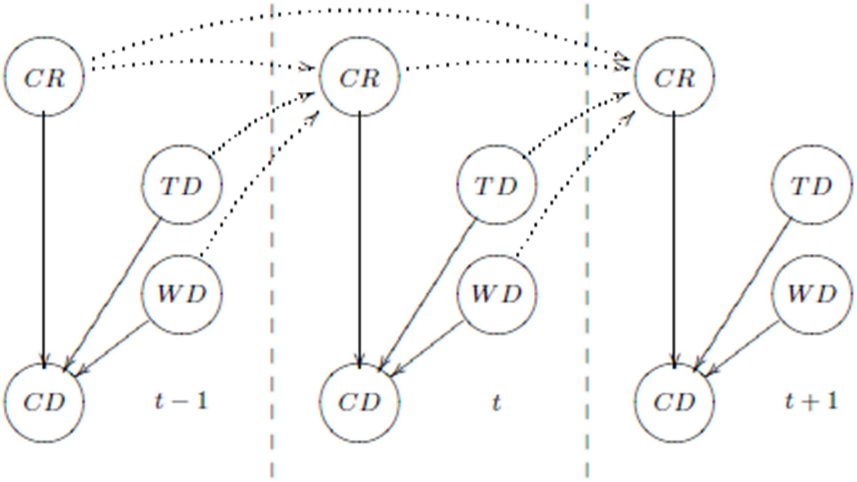


Figure 2-2 – Bayesian network from (Petzold et al. 2005)

As the structure of the network is fixed, the prediction of the occupant's location is always dependent on the time of day and the day of the week, as well as their current location, which means that these elements of context must always be used. To change the context used, the structure of the network would have to be restructured and the model retrained. Similarly the number of time slices used is fixed, resulting in all predictions being made using a fixed number of previous locations. As the model can train while predicting, location prediction accuracy with and without training on a training set was considered. For a sequence length of two, making predictions only when the occupant is not in their office, the approach achieved overall accuracy of 65% without training and 72% with training. Stay duration prediction was tested with networks including different combinations of parent nodes, and with and without training data. Using current room and time of day, making predictions only when the occupant is not in their own office, the approach achieved 70% accuracy without training and 78% with training.

In (Koehler et al. 2014) an ensemble approach is presented which combines the predictions of four classifiers (decision tree, 3-nearest neighbour, support vector machine and gradient boost) to independently predict when an occupant will transition to a new location and what location that will be. The approach uses a variety of elements of context: current location, time of arrival at current location, minutes passed since arriving at current location, current time, day of week, arrival time in the building, number of significant locations visited thus far today, previous two significant locations, duration of stay at previous significant location and time taken to transition to current location. The approach uses ten minute timeslots and significant locations are those which the occupant frequently remains in for at least one timeslot.

In order to determine which features from this list should be used when making a prediction the approach uses sequential floating forward selection (SFFS), which repeatedly add features to the feature set in order to improve accuracy, removing those which can be removed without negatively affecting accuracy. SFFS is applied independently for each classifier in order to find and use the best feature set for each. This process is repeated after each predicted day to react to changes in occupants' routines.

As mentioned, the approach predicts where an occupant will transition to and when. Specifically, for a given number of minutes, it predicts whether the occupant will stay in their current location for that duration, and if they will not, then it predicts where they will go. When asked whether the occupant will transition, each classifier responds with 1(yes) or 0 (no). If the mean prediction is equal or greater than 0.5 the overall prediction is 1, otherwise it is 0. If it is 1, then the classifiers each vote on the location, and the majority vote is the overall prediction, with random selection used as the tiebreaker. As the temporal and spatial predictions are made separately, accuracy is calculated for a two step prediction. To be correct, the algorithm must first correctly predict whether the occupant will stay where they are. If the occupant leaves, it must also correctly predict where they go. The approach was tested for look-aheads of 10 to 90 minutes, achieving over 90% accuracy for 10 minute look-ahead, dropping monotonically to just below 80% for 90 minutes.

2.1.3. Next Location Prediction

Further approaches have been applied to the problem of predicting only the occupant's next location based on their recent movements. In (Petzold et al. 2004) a state predictor and frequency predictor are evaluated. The frequency predictor is similar to a variable order Markov model in that for each sequence of movements it records the frequencies of the following locations, predicting for a given sequence the most frequent follow up location.

The state predictor is an alternative designed to retrain faster when the occupants' routines change. The prediction for any given sequence is a one of a set of possible states. There are two states for every location which could be transitioned to after the sequence, a higher and lower state. When the sequence is observed, the state is updated as follows: if the state matches the observed location, it transitions to the higher state for that location, or remains in it if already there. If the observation contradicts the state, it either transitions from the higher state to the lower state, or transitions from the lower state for the 'wrong' location to the lower state for the 'correct' location. In this way the state predictor requires only two contradictory observations to revise its prediction, whereas the frequency predictor will need enough contradictory observations to outweigh the existing training data.

However, since a single contradictory observation will not change the prediction, it should tend to settle on the most frequent location.

For both approaches to making the actual prediction, the sequence on which to base the prediction is selected in a manner based on Prediction by Partial Matching (PPM): the last n locations are considered as the input sequence with n starting at some maximum order. If the sequence of length n is not known to the predictor, then the last $n-1$ locations are used, and so on until a match is found. The authors tested a number of configurations, and achieved highest accuracies in excess of 90% (predicting next location when the occupant is not in their office only) using both the frequency and state predictor with PPM with a maximum order of 5.

In (Vintan et al. 2004) an Elman net is used for next location prediction. An Elman net is a neural network that includes a context layer which holds a copy of the values in the hidden layer from the previous time step. This allows the network to essentially record its previous state, so that its output is based both on the current inputs and the inputs one time step back, which improves performance when the neural network is being used to predict sequences.

The input to the neural network in this case is a sequence of the recently visited locations of some fixed length, along with an identifier for the occupant in question if the network is being trained on all occupants (training one network per occupant is also considered). The locations are encoded as binary numbers, either 4 or 5 bits long depending on how many locations must be encoded (individual occupant networks do not encompass all rooms), while occupants are 2 bit numbers. The output is then the binary encoded prediction of the occupant's next location.

The authors test a number of parameters of this approach to determine the optimal setup, eventually settling on training a separate network for each occupant, each with a hidden layer containing one more neuron than the input layer and a single context layer, with the last two locations visited by the occupant as the input. With this configuration they achieved an overall 90% accuracy predicting occupants' next location when they were not in their office.

In (Gellert et al. 2006) a Hidden Markov Model (HMM) is applied to the problem of next location prediction. A HMM is a Markov model in which the states are not directly observable, rather each state has, as well as state transition probabilities, a probability distribution over a set of possible outputs which are observable. In this context the outputs are the locations which the occupant can occupy, while the hidden states are states the occupant can be in which influence their choice of next location.

The authors evaluate a number of possible configurations, varying the number of hidden states as well as considering a higher order HMM. The best configuration was a first order one state model, which is essentially equivalent to a zeroth order standard Markov model, achieving 80% overall accuracy predicting occupants' next location when not in their own office.

2.1.4. Other Approaches

Other work includes (Voigtmann et al. 2011), in which sequence based prediction is improved by considering other occupants' histories when new behaviour is encountered. In (Eagle et al. 2009) occupants' patterns are broken down into a small number of primary patterns called eigenbehaviours, and as each occupant's behaviour on any given day is likely to be only a slight deviation from one of their eigenbehaviours, their movements can be predicted once it is determined which behaviour pattern they are following. Markov models are used for next location prediction in (Ashbrook et al. 2003) and (Song et al. 2006), with the latter comparing their performance against predictors based on the Lempel-Ziv (LZ) family of data compression algorithms.

Predicting the occupancy of locations is related to the locations of occupants, and is applied in smaller scale contexts such as a home or a single office, using similar methods to location prediction. In (Manna et al. 2013) Markov models are used for office occupancy prediction, with multiple first-order Markov models at different time lags applied to predict based on the occupancy state at several recent time points. A number of methods have been applied to occupancy prediction in smart homes, including neural networks (Mozer 1998) and LZ family predictors (Das et al.

2002). In (Scott et al. 2011) the occupancy state of the home was predicted for the rest of the current day by comparing the known occupancy states against the past occupancy data using the Hamming distance of the occupancy records, and averaging the five most similar days.

Another area of research centred on the home is the detection, recognition and prediction of occupants' activities. Basic activity recognition can be achieved with simple wearable sensors such as those build into smartphones. Examples of such approaches include (Kwapisz et al. 2011) and (Anguita et al. 2012). In both, the accelerometer data from worn smartphones is recorded as experiment participants perform a number of basic activities such as sitting, walking, standing, and going up or down stairs. A number of machine learning techniques are then applied to the data to allow recognition of these activities in real time. Other approaches such as (Kasteren et al. 2008) use sensors throughout the home to detect a wider variety of events, allowing the recognition of a wider range of activities including Showering, Sleeping, Breakfast and Dinner. To allow further detail in the detected events, the model of the environment can be made more detailed. In (Chen et al. 2012) the activities which can take place in the home are modelled in a hierarchical fashion so that, for example, the activity of making tea is a child of the more general activity of making a hot drink. If the required sensor data is available the specific activities can be recognised; if less sensor data is available, the more general parents are recognised instead. Without complete annotation of which sensor events correspond to which activities, there can be difficulty in determining which sensor events correspond to which activities, especially if activities overlap or are simultaneous. (Wan et al. 2015) deals with recognising which sensors relate to which activities by correlating which sensors tend to activate together and the time spans across which they activate, in order to segment a stream of incoming sensor events into discrete activities.

Location prediction is also applied in larger scale contexts. In (Noulas et al. 2012) the next venue that an individual will check into using Foursquare is predicted using linear regression. Low accuracy was achieved unless tens of locations were returned as answers for each prediction; due to the possible answers numbering in the thousands, precise prediction was not possible. In (Mathew et al. 2012) a hidden Markov model is used to predict the location of individuals in a dataset comprising 1.2 million km of movement. Again, the immense variety of possible locations resulted in low prediction accuracy.

2.1.5. Summary

Existing work on occupant location prediction is focused on next location prediction based on occupants' recent locations. Relatively few works are concerned with predicting the occupants' locations later on the same day or on future days, or with using extra context such as time of day, day of week, or ancillary data such as timetables.

Of those approaches which do address future location prediction, only two answer the question "in which location will occupant X be at time Y tomorrow", where the location can be a precise location (i.e. a particular room in a building versus which building) and the time is a precise timestamp (i.e. 3pm versus 'afternoon'). These approaches, the first-order Markov model in (Burbey 2011) and the naïve Bayes classifier in (Vu et al. 2011), predict on a similar basis to OccApriori; the frequency with which individuals were in a location in a given timeslot in the past. However, these models consider limited amount of context, with the Markov model in particular having difficulty adding any context. Furthermore neither approach can take into account occupant movements on the same day if the prediction is being made shorter-term, such as several hours later on the same day.

Other approaches discussed can use more elements of context when making predictions, in particular Indoor-ALPS (Koehler et al. 2014) can decide which elements of context to use independently for each of its four underlying classifiers. However, the approach only considers next location prediction, and the set of context elements available to choose from are determined in advance. No approaches automatically determine which elements of context to use or make their choice of context at prediction time.

Given that both additional context and pure historical location frequency based prediction of future locations have been shown to work independently, an obvious avenue to explore is an approach which combines these features, which motivates the selection of association rule mining as the basis of the approach presented in this thesis. Association rule mining has the flexibility to predict next location and near future locations based on recent locations, predict further in the future based on historical location frequencies, and will make use of any relevant context data available automatically.

2.2. Association Rules

This thesis uses association rule mining as the basis of its approach to occupant location prediction. A number of other machine learning approaches have been used, however as already discussed most existing work focuses on next location prediction or prediction with limited context. Two approaches in particular which are potentially suitable for future location prediction are classification and clustering, which will both be briefly compared to association rules before moving on to a detailed discussion of association rules themselves.

Mostly closely related to association rules are classifiers. Classifiers use the values of various attributes in the dataset to predict the value of a single attribute (the 'class'). In the context of occupant prediction, a classifier could learn which attributes relate to occupants' locations in a certain timeslot, to predict their location at that time. An example of such a use of a classifier is (Vu et al. 2011). However, as previously discussed, due to classifiers being intended to predict a single attribute, it is difficult for such an approach to relate timeslots to each other in order to use recent movements to increase prediction accuracy. The aim of this thesis is a prediction approach which can predict future locations, but is also capable of using any supplied data to improve prediction accuracy, including recent location data, making classifiers unsuitable.

Clustering is a method in which similar instances are grouped together. In occupant location prediction, this would allow for the recognition of daily patterns by grouping together days which share similar patterns of locations. This is similar to some previously discussed work on location prediction such as (Eagle et al. 2009).

However, this type of approach recognises entire days of movement, rather than learning more specific patterns which relate to individual time slots. In existing work this type of approach is applied to very general locations, such as home and work, or simply to a binary occupancy value (whether an occupant is present or not). As this thesis aims to predict exact locations in datasets which may have a larger set of possible locations, clustering and similar approaches are unlikely to work with the level of specificity required.

Association rule mining is designed to find any patterns which occur in a dataset, involving and predicting any attributes. Compared to classifiers, association rules are more amenable to predicting any attribute using any other attributes as the basis of the prediction, which is required for an approach intended to cover everything from next location prediction to future locations as far ahead as following days. As they find more specific patterns which relate to particular attributes, they allow for more precise prediction than approaches such as clustering which rely on recognising an entire day of patterns. Association rules are thus the most appropriate starting point for making the types of prediction which are the objective of this work.

2.2.1. Association Rule Mining

Association rule mining is an unsupervised approach to finding patterns in large datasets. They were introduced in (Agrawal et al. 1993), the original application being finding patterns of items which were purchased together in datasets of transactions. In that application items were simply strings which were either present or absent in each transaction, however the approach can be applied without modification to sets of attribute/value pairs. The Apriori algorithm, introduced in (Agrawal et al. 1994), is one of the original association rule mining algorithms. While there are newer algorithms which are more efficient, this thesis uses Apriori as the basis of its approach as it is the least complex algorithm and thus the simplest to modify, hence the name OccApriori.

In general association rule mining works as follows. 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 and support of X are defined as:

$$freq(X) = |\{I | I \in \{I_1 \dots I_n\}, X \subseteq I\}| \quad (2\text{-}1)$$

$$supp(X) = freq(X)/|D| \quad (2\text{-}2)$$

An association rule is an implication typically written in the form $X \Rightarrow Y$, where for some instance : $X \subseteq I, Y \subseteq I,\ X \cap Y = \emptyset$. What the rule actually implies is $X \subseteq I \Rightarrow Y \subseteq I$; that instances which contain X tend to also contain Y. We measure how often a rule applies using the support, and how reliable the rule is using the confidence, defined as follows:

$$supp(X \Rightarrow Y) = supp(X \cup Y) \quad (2\text{-}3)$$

$$conf(X \Rightarrow Y) = supp(X \cup Y)/supp(X) \quad (2\text{-}4)$$

The goal of an association rule mining algorithm is to find a set of rules which are above user-specified thresholds of support and confidence. The first step is to find all itemsets which are 'frequent' –the itemsets whose support is above the threshold. Association rules are then generated from the frequent itemsets, with any rules which fall below the confidence threshold being discarded. Enumerating the frequent itemsets is the more difficult step, as the desired itemsets must be found among the total $2^{|U|} - 1$ itemsets which can be generated.

2.2.2. Apriori

```
$C_1 = U$
$F_1 = \{X \in C_1 | supp(X) > minsupp\}$
$for(i = 2;\ F_{i-1} \neq \emptyset;\ i++):$
  $C_i = \{A \cup B | A, B \in F_{i-1}, |A \cup B| = i\}$
  $F_i = \{X \in C_i | supp(X) > minsupp\}$
$return \bigcup_i F_i$
```

Figure 2-3 – Apriori Algorithm

The set of all itemsets which can be generated from a given dataset can be arranged into an itemset lattice, in which itemsets are linked by subset/superset relationships. The Apriori algorithm uses breadth-first search to find all the frequent itemsets in the lattice. First all itemsets of size 1 are enumerated. Itemsets whose support is below the threshold are discarded, as any superset of an infrequent itemset will also be infrequent. Candidate itemsets of size 2 are generated from the remaining size 1 itemsets, after which infrequent itemsets of size 2 are discarded. This process then continues, finding frequent itemsets of size n by generating candidates from the frequent itemsets of size $n-1$ and discarding infrequent itemsets, until an n where no frequent itemsets exist is reached.

Transactions
A, B, C, E
A, C
B, D, E, F
B, C, E
A, D, E

Table 2-1 – Example dataset

Table 2-1 – Example dataset contains an example dataset to which we could apply Apriori to find the frequent itemsets. The items in this dataset are the letters A-F, grouped into transactions, with each transaction being a row of the table. Assuming we want to find itemsets with a minimum support of 3, the algorithm starts with the frequent 1-itemsets {A}, {B}, {C}, and {E}. There are no 1-itemsets for items D or F as they only appear in two transactions each, putting them below the support threshold i.e. infrequent.

The candidate 2-itemsets are now generated from the frequent 1-itemsets by combining them, giving us the itemsets {A,B}, {A,C}, {A,E}, {B,C}, {B,E} and {C,E}. The dataset is then scanned to determine the frequencies of these 2-itemsets, for example {A,B} is a subset of only the first transaction, narrowing the list which meets the minimum support to only {B,E}. As there are no candidates to generate from a single 2-itemset, the algorithm terminates.

Rule generation comes after itemset mining is complete. As a brief example, we may wish to generate the rules B => E or E => B based on the itemset {B,E}. These rules state that if one of the two items appear in a transaction, the other likely does as well. The rule B => E has a confidence of 100%, as every transaction which contains B also contains E, whereas the rule E => B has a confidence of 75%, as E appears in four transactions, while B appears in only three of those transactions.

While Apriori's approach is simple, requiring only the union of sets of items and scans through the set of instances to check support, and somewhat efficient, since it prunes itemsets with infrequent subsets, there are significantly more efficient approaches, two of the better known examples being Eclat (Zaki 2000) and FP-Growth (Han et al. 2000). Two of the main problems with Apriori are the breadth-first search resulting in a combinatorial explosion, and the repeated scans though the set of instances to determine the support of newly generated candidate itemsets. A depth-first-search based approach can avoid both of these issues.

2.2.3. Eclat

In (Zaki 2000) several algorithms based on a depth first approach are introduced, the best known of which is Eclat. Here the dataset is converted from its standard database representation of a set of instances each of which is a set of items contained by the instance, to a set of items each of which having an associated set of instances in which the item appears. The support of an item is then the cardinality of the set of instances in which it appears, and the set of instances which support a newly generated itemset is the intersection of the sets of instances supporting the itemsets from which it was generated. This is referred to as "TID list intersection" (i.e. transaction ID list) and means that the support of newly generated itemsets can be determined without rescanning the database as in Apriori, with the only cost being the computationally cheap intersection operation and maintaining the sets of supporting instances in memory.

For an Apriori-like approach this would involve maintaining many sets of instances in memory, however this is paired with a depth first approach to the itemset lattice. Each 1-itemset is examined in turn, generating all frequent 2-itemsets containing that 1-itemset. For each of these 2-itemsets in turn, the process is repeated, and so on. As new itemsets are generated, their supporting instances are found via set intersection, and old sets of instances from lower in the lattice can be discarded, saving memory.

Starting with the 1-itemsets and working upwards to find the point where itemsets become infrequent is a bottom-up approach; the paper also considers a top-down approach. This starts with the largest possible itemset and works downwards through infrequent itemsets. When a frequent itemset is encountered, all of its subsets are marked as frequent as well. This is similar to how, in the bottom-up approach, the supersets of any infrequent items are marked as infrequent, and relies on the same anti-monotonicity property. Since the algorithm can stop when it runs out of infrequent itemsets, starting from the largest itemset and working down may be faster if the largest frequent itemsets are high in the lattice (i.e. closer in size to the largest itemset than to a 1-itemset).

The algorithms presented in the paper mix and match bottom-up and top-down approaches, combining the two in a single hybrid search, along with the use of equivalence relations and pseudoequivalence relations (relations which are reflexive and symmetric but not transitive, allowing the possibility of overlap in the resulting subsets) to determine the most efficient way to divide the itemset lattice during depth-first traversal. In their tests their most efficient algorithm outperforms Apriori by an order of magnitude.

2.2.4. FP-Growth

In (Han et al. 2000) an approach is presented which avoids candidate generation completely. This approach builds a frequent pattern tree (FP-tree) which records the frequency of every pattern of items. As the nodes in the branches of a tree have an order, these patterns are ordered sets of items, rather than the unordered sets of items which make up the underlying dataset. The tree can be traversed and manipulated to enumerate the frequent itemsets, without ever generating infrequent itemsets which must be discarded. The tree can also represent the frequency data in a form more compact than the original dataset.

Each branch of the tree represents a frequent pattern. Each node in the tree refers to a single item, and records the frequency of the pattern which is formed by it and its ancestor nodes, starting with the node which is a child of the root, and ending with the node in question. As an item can appear in multiple patterns, there can be multiple nodes referring to each item, with the total frequency of the item being the sum of the frequencies recorded in all of that item's nodes. Nodes which refer to the same item also reference each other via a linked list, allowing for efficient traversal of all nodes which refer to the same item.

There is no 'correct' order to apply to the items when building the tree. An ordering is decided upon before building the tree with the aim of minimizing the size of the resulting tree. The order used in this work is the frequency with which each item appears in the dataset: the most frequent item in a given instance is first, the next most frequent is second, etc. Thus when an instance is added to the tree, the most frequent item in that instance becomes a child of the root node, the next most frequent becomes a child of the first item, etc. When patterns share a prefix they overlap in the tree, incrementing the frequency recorded in the existing nodes until the patterns diverge, at which point the new pattern's branch splits off from the existing branch. Along with discarding infrequent items before constructing the tree (due to the same property Apriori relies on), this is what allows the tree to be a smaller representation of the frequencies than the original dataset; overlapping patterns reduce duplication and save space.

Enumerating the frequent itemsets involves traversing the tree in a recursive manner. The frequent items are selected in reverse order of frequency (i.e. the reverse of the order used when building the tree). For each item, a conditional FP-tree is constructed, which is effectively the FP-tree built from only from those instances which contain the target item. This is done by traversing the main tree and extracting those patterns which include the target item, then building an FP-tree from the extracted patterns. Items which are frequent in the conditional FP-tree are those which occur frequently with the target item, thus providing frequent 2-itemsets. The conditional FP-tree process is repeated recursively for each of these frequent itemsets to find frequent itemsets involving those itemsets, until no further frequent itemsets appear. The next item from the original reverse-order list is selected, and the process repeats, although ignoring the item which was already processed.

In this way, using the structure of the FP-tree, all the frequent itemsets are enumerated without candidate generation and with only two scans of the source dataset required, while representing the dataset in a more compact format. The lack of candidate generation allows FP-Growth to avoid the combinatorial explosion encountered by Apriori as the support threshold decreases and the size of the dataset increases, making FP-Growth significantly more scalable.

2.2.5. Algorithm Selection

This thesis uses Apriori as the basis of the occupant location prediction approach. While the algorithms function differently, they all produce the same output for the same parameters, which means the choice of algorithm generally comes down to speed. However, the purpose of the work presented in this thesis is to experiment with modifications to the criteria applied to itemsets when considering whether to discard them, and so the ease of conducting such experiments is a factor as well.

The basic representation of the space of itemsets in a dataset is the itemset lattice. Apriori and Eclat work directly on this representation, whereas FP-Growth constructs its own, more efficient, data structure, the FP-Tree. As the FP-Tree and in particular the algorithm which constructs and traverses it is more complex than Apriori's equivalent, experiments which involve changing the area of the lattice traversed would be significantly more complex to implement while providing the same results in terms of the rules generated. Furthermore, since its efficiency depends on the construction and traversal of the FP-Tree, altering the process may impact that efficiency, somewhat negating its advantage over Apriori.

While Eclat is much closer to Apriori, similar concerns apply as choices must be made as to the order that itemsets are processed in during the traversal of the lattice, which again means that modifications could affect any efficiency advantage. These are just two of the better known association rule mining algorithms – but the same concerns apply to any other algorithm. Apriori, being little more than enumeration of the itemsets with some pruning, provides the best starting point to build a new algorithm to mine a new type of dataset. As the approach in this thesis is built on Apriori, it shall hereafter be referred to as OccApriori – Apriori for Occupant (location) data.

2.3. Extensions of Association Rules

2.3.1. Generalised Rules

As well as making the association rule mining process more efficient, there is also work on extending the types of association rules which are found. The most relevant to the work in this thesis is (Srikant et al. 1995), which introduces the idea of generalised association rules, rules whose items are members of a taxonomy. The standard items in the dataset become the leaves of the taxonomy tree, and the goal is to mine itemsets which contain items from any level in the taxonomy.

The simplest approach is to simply add the taxonomy ancestors of each item to the transactions. Once added to the dataset, they can be treated as normal items, and Apriori or any other algorithm can be run as normal. A number of potential enhancements to make the enumeration of the generalised itemsets are presented in (Srikant et al. 1995). These include simple implementation enhancements, such as generating a lookup table of the ancestors each item has rather than traversing the taxonomy tree every time. They also include algorithm modifications, primarily focused on pruning unnecessary items and itemsets. Itemsets containing and item and an ancestor of that item can be pruned, as the support is necessarily the same as that itemset less the ancestor item. When checking the support of itemsets of a given size, more general itemsets (i.e. those containing items from higher in the taxonomy tree) can have their support checked first. If the general itemset is not frequent, the more specific itemsets will necessarily also not be frequent, and so they can be discarded without checking their support. Chapter 6 examines how generalising association rules in this manner can be applied to occupant location prediction to yield useful new types of predictions and more flexible occupant pattern recognition.

2.3.2. Inter-transactional Rules

There are several other extensions of association rules in general, and Apriori specifically, which may be useful for occupant location prediction. Inter-transaction association rules (Lu et al. 2000) allow the mining of patterns which involve multiple instances. For example, if each instance is a set of data for a single day, standard association rules will find patterns such as 'if X happens on a day, Y will as well', whereas inter-transaction rules allow for patterns such as 'if X happens on a day, Y will happen on the following day'.

2.3.3. Quantitative Association Rules

Quantitative association rules, introduced in (Srikant et al. 1996), address mining rules where the values in the dataset are numeric. This is problematic for standard association rules as each number is treated as a distinct value unrelated to any other, resulting in a large number of low support rules. By breaking the range of values down into a smaller number of intervals and mining those intervals instead, this task reduces to the standard association rule mining problem.

2.3.4. Temporal Rules

Temporal association rules consider the issue of rules which are supported in limited time periods within the dataset, and temporal relationships between the items in those rules. Several formulations exist, addressing different parts of the temporal aspect of datasets. Examples include (Ale et al. 2000), wherein the period that an itemset appears in is addressed. Taking the example of market baskets, a product which was available for a limited time will appear in transactions more frequently during that specific time period than it will across the entire timespan of the dataset. In (Y. Li et al. 2001) recurring intervals are addressed by including time attributes which represent calendar periods. This allows for rules which apply in these periods, for example products which are frequently sold together only in December. The application of temporal semantics to association rules, discussed in (Rainsford et al. 1999), allows for the expression of temporal relationships between items, stating not only that A and B are associated but that, for example, B occurs during A, or before A, or after A.

2.3.5. Rule Updating

In practice a dataset will likely have new data added to it, requiring an update of the association rules based on that dataset, an issue first addressed in (Cheung et al. 1996). In this work every time new data is available, the itemsets which need to be re-evaluated are determined based on the instances being added to the dataset. This avoids the need to mine the dataset from scratch every time new data is added. Chapter 7 discusses potential future work in this area which can make use of these extensions of association rule mining.

2.3.6. Summary

Algorithm	Summary
Apriori	Relies only on anti-monotonicity property Inefficient compared to newer algorithms
Eclat	TID-list intersection avoids rescanning dataset Depth first search allows efficient TID-list intersection Choice of search order affects efficiency
FP-Growth	Avoids candidate generation completely Structure of pattern tree affects efficiency

Table 2-2 – Summary of association rule mining algorithms

Association rule mining is a mature field with several core algorithms for the association rule mining process itself, using a variety of optimisations to improve efficiency, and numerous extensions designed to find new types of rules. Table 2-2 – Summary of association rule mining algorithms lists the most notable features of each of the discussed algorithms. With a base approach to occupant location prediction using association rule mining in place, these extensions can be used to find new types of patterns in occupant location data. As previously discussed, this thesis uses Apriori as the basis of the approach as it is the least complex, making it the simplest to modify when designing experiments.

2.4. Occupant Modelling

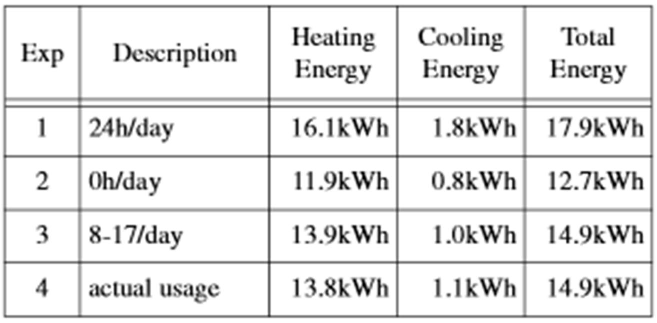
In this section occupant modelling approaches relevant to the dataset generator which will be presented in Chapter 3 are discussed. These approaches take two basic approaches to modelling occupants: either the parameters for the modelled occupants are set manually by the author of the model (for example defining the activities the occupants engage in during the day) or the model is given historical movement data for the occupants, with the model being automatically generated based on this data. These will be referred to as parameter-based and schedule-calibrated models respectively, each discussed in their own section.

2.4.1. Parameter Based Models

In (Zimmermann 2007) occupants and their environment are modelled as part of a multi-agent simulation. The model is intended to simulate energy consumption in buildings, and so there are components which model the physical layout and properties of the building and the heating, cooling and lighting systems. Occupants are modelled as agents whose movements are governed by roles and tasks. Occupants take on a role at the beginning of a day, and the role then dictates the scheduling of tasks for that occupant for that day, with the tasks then dictating the occupant's activities throughout the day. Roles can also be assigned to groups, of which individual occupants are members, allowing for tasks which are shared between multiple occupants.

Occupant activities in this model are broken down into several classes. The basic background activity of an occupant, such as working at their desk, is a continuous activity. This is the fall-back activity which an occupant does when not engaged in any other task. Regular activities are those which occur at fixed times with fixed durations, scheduled at the beginning of the day based on the occupant's role. Irregular activities are also scheduled activities, but are scheduled in between the higher priority activities, with their start time dependent on when the occupant is available (e.g. ad hoc meeting). Finally, secondary activities are ancillary activities which have a probability of occurring when the occupant is engaged in some other activity. These allow for activity interrupts such as visiting the bathroom.

The model is built using Specification and Description Language (SDL-92) graphical modelling language, whose modelling environment includes a code generator. This means that once the models of the building, occupants, etc. have been specified, the executable code to simulate that model is generated automatically. When the simulation starts, the occupants arrive at the workplace and move around the building according to their role and various tasks. Using the model of the physical layout of the building, the time taken to move between locations is modelled as the occupants change tasks.



Exp	Description	Heating Energy	Cooling Energy	Total Energy
1	24h/day	16.1kWh	1.8kWh	17.9kWh
2	0h/day	11.9kWh	0.8kWh	12.7kWh
3	8-17/day	13.9kWh	1.0kWh	14.9kWh
4	actual usage	13.8kWh	1.1kWh	14.9kWh

Figure 2-4 – Experimental results from (Zimmermann 2007)

The evaluation of the model simply provides the results of the simulation of a number of configurations in terms of the energy usage of the building, as shown in Figure 2-4. The evaluation shows that it is feasible in practice to build a model as outlined in the paper, generate the code and run the simulation to get results in a reasonable amount of time. However, it does not evaluate how closely the activities of the simulated occupants match the real occupants the model was based on.

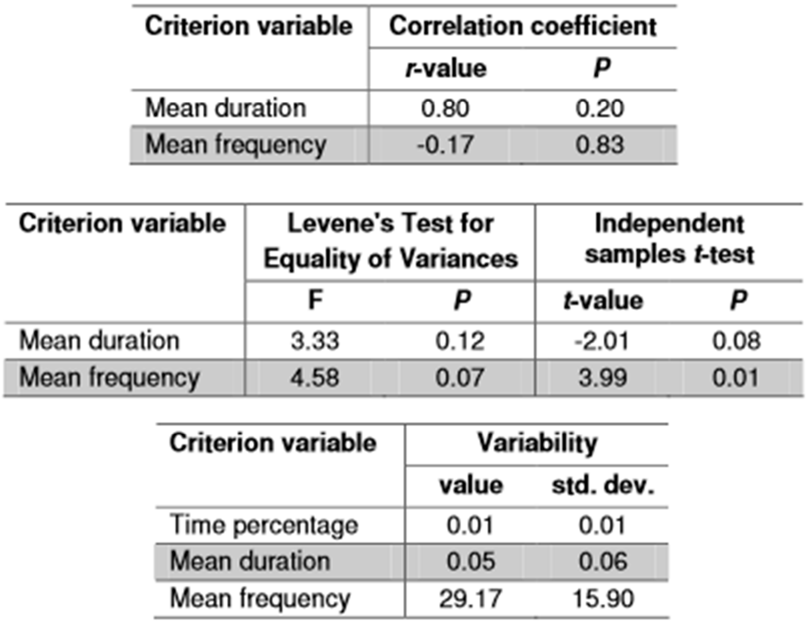
A similar breakdown of the tasks that simulated occupants undertake is used in (Tabak 2008), in which occupant activities are broken down into skeleton activities and intermediate activities. Skeleton activities are those which are determined by the work being done by the occupant, and are so named as they provide the basic structure of the occupants’ movements. These activities are broken down into planned and unplanned activities, where the former occur at fixed times, while the latter fit in around the planned activities (similar to regular and irregular activities in (Zimmermann 2007)). However, rather than modelling planned and unplanned activities as distinct types of activity, skeleton activities instead have a priority attribute, which determines how likely the activity is to be interrupted or rescheduled. Lower priority tasks fit in the schedule around higher priority ones.

Intermediate activities are those which are not determined by the workflow and are determined either by the occupant themselves, e.g. going for a break, or by the environment, e.g. an unexpected visitor arriving. Some of these activities have a simple chance of occurred at any given time based on, for example, the occupant's role, however the probability of others occurring is based on an S-curve due to the type of activity. For instance, as time passes an occupant is more and more likely to take a break, until they eventually do. Thus for these types of activities the time since last occurrence is a major factor in when they occur, rather than a simple random selection of an available time. Whether an intermediate activity interrupts a skeleton activity depends on the utility of the intermediate activity, and the importance of the skeleton activity.

Shared tasks are modelled as groups of skeleton tasks. The example given is that of a presentation, where one occupant will have a 'give presentation' task. This task will require that this initiator occupant is able to perform the task, but also that a stated number of other occupants can attend. This takes the form of other occupants having the 'attend presentation' skeleton task in their set of tasks, as well as their being available at the appropriate time. If these criteria are met the corresponding tasks are scheduled for the initiator and additional occupants. When selecting which of the occupants who could attend will actually attend, a random selection weighted by how close the occupants are to the initiator in the structure of the organisation is used.

The model also includes the physical layout of the building, the available facilities in the building which are used when completing tasks, the organisational units which make up the organisation and the roles the occupants can take on. Similar to (Zimmermann 2007), occupants are given their tasks by having both the occupants and tasks be assigned to roles and organisational units, thus having occupants' schedules defined by their position in the organisation and the role they fulfil. Occupants can be assigned multiple roles and organisational units. A series of components were implemented to execute the model, each handling the execution of a specific part of the model. For example, there are dedicated schedulers for skeleton tasks, shared tasks, intermediary tasks, as well as schedule overlap and gap removers and a dedicated scheduler for handling the time required to move between locations.

An evaluation model was calibrated using input data including a list of occupants of the building, their roles according to the university's job classification system, data from a system in which occupants had previously recorded the time they spent on various activities, and finally a web based survey to determine how occupants in various roles prioritised differed tasks. The output of the model was compared to two datasets, one in which occupant movements were recorded using RFID, and one in which the occupants recorded their own movements using an online diary system, both of which were recorded in parallel.



Criterion variable	Correlation coefficient	
	r-value	*P*
Mean duration	0.80	0.20
Mean frequency	-0.17	0.83

Criterion variable	Levene's Test for Equality of Variances		Independent samples *t*-test	
	F	*P*	*t*-value	*P*
Mean duration	3.33	0.12	-2.01	0.08
Mean frequency	4.58	0.07	3.99	0.01

Criterion variable	Variability	
	value	std. dev.
Time percentage	0.01	0.01
Mean duration	0.05	0.06
Mean frequency	29.17	15.90

Figure 2-5 – Evaluation of similarity of workspace usage between simulated occupants and data from the POPI+ system from (Tabak 2008)

The model output was tested for similarity to the collected datasets for the total time percentage, mean duration and mean frequency for the usage of different location types, time spent in intermediate tasks, time spent in task types as well as, for the RFID dataset, mean walking distance. Figure 2-5 shows an example of evaluation of the similarity, in this case for the usage of workspaces as compared to the online diary system. Overall the model proved to be a highly accurate recreation of the modelled occupants, with notable differences only in the frequency of task-types being selected and (possibly as a consequence) the frequency of usage of workspaces (seen in Figure 2-5, primarily from the high variability for the mean frequency). However, knowing that tasks and locations are selected the same number of times and used for the same durations, does not tell us whether the simulated occupants were in fact doing the same things at the same times as the real occupants; only that they did them equally often and for equally long.

2.4.2. Schedule Calibrated Models

In the following models, rather than the model author specifying the occupants' activities to determine their movements in the model, historical data of actual occupants is used to generate a model.

In (Page 2007) the occupancy of locations (rather than the location of occupants) is modelled using a time inhomogeneous Markov chain. For each location occupancy is either a 1 (location is occupied) or a 0 (location is vacant). Being a Markov chain, the probability of a location being occupied is based solely on whether the location was occupied in the previous time step; however the transition probabilities can be different for each time step, essentially making the probability dependent on the time of day as well.

The input when calibrating the model is the profile of probability of presence, the frequency with which the location was historically occupied in each time step. This alone is not enough to determine the transition probabilities, as there can be multiple values for the transition probabilities that agree with the probability of being occupied and vacant in each time step. To alleviate this, a parameter of mobility is defined as a function of the transition probabilities. This parameter essentially quantifies how likely the occupant is to move and this, combined with the profile of probability of presence, allows for the determination of unique values for each transition probability for each time step.

The model thus far cannot reproduce periods of extended vacancy such as the location's occupant being on vacation (or rather, such a pattern is exceptionally improbable). Instead, the frequency and probability distribution of duration of extended vacancies are additional inputs to the model, which then has an explicit chance of beginning an extended vacancy each generated day according to those inputs. These days of total vacancy are then ignored when providing the model with the profile of probability of presence, so that the Markov chain is only modelling the probability of presence when the location is in fact occupied.

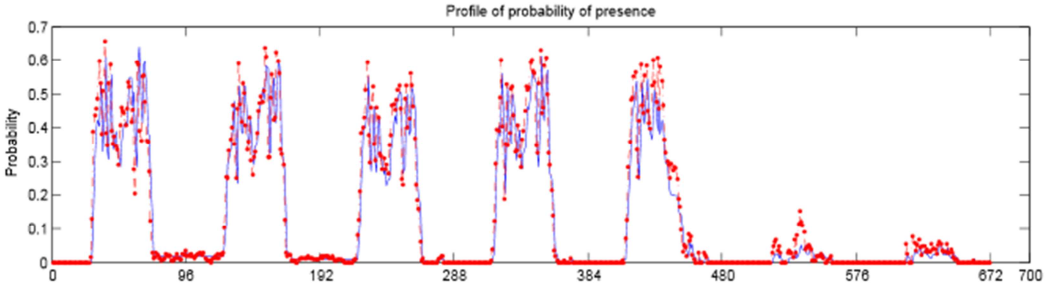


Figure 2-6 – Profile of probability of presence for office no. 3 in the actual occupancy data (blue line) and the simulation (red line) from (Page 2007)

The model is evaluated in terms of its ability to reproduce patterns of probability of presence, total occupancy time per day and per week, first arrival time, final departure time, duration of periods of presence and absence during the day, and frequency of change of state (occupied versus vacant). Figure 2-6 compares the simulated and real profile of probability of presence for one of the offices, demonstrating that the model successfully captures the frequency with which the location is occupied across the times of day. While the model had some difficulty correctly reflecting the frequency which with locations were completely vacant and the frequency with which the occupancy state changed, overall it provided an accurate recreation of the source data.

A similar approach is used in (Goldstein et al. 2010b) to model the tasks occupants engage in rather than the occupancy of locations. In this case the objective is to learn the probabilities of the occupant taking each of a set of tasks, based on the task the occupant just completed, the time of day, and a task suspension interval. The task suspension interval measures the time since a task last occurred, in order to recreate intervals between occurrences of tasks (for example spreading out breaks).

The calibration process involves populating a set of bins which record the frequency with which the occupant transitioned from one activity to another at a given time of day with given task suspension intervals. Once populated, these histograms are processed so that they record the probability of a transition, rather than the absolute frequency. They are then analogous to the transition probabilities in (Page 2007), in that for a simulated occupant in a given state (where the state consists of the time of day, previous task, and task suspension intervals), the probability of transition to each possible next state (which would be the possible tasks) can be looked up.

Once an occupant's next task is selected, the model considers the number of occupants who will be engaged in that task, and the duration of the task. When calibrating the number of occupants, a similar histogram method is used to determine the mean and variance of the number of occupants, given the task in question and the time of day, which are then used to select the number of occupants during task generation. The process for duration generation is the same, except that the determining factors are the task, time of day and additionally the number of occupants.

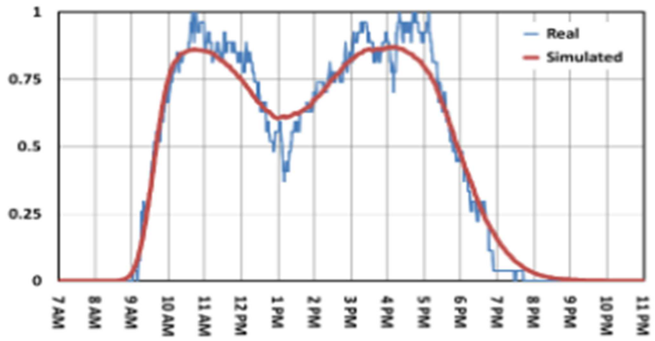


Figure 2-7 – Desk work probability profile from (Goldstein et al. 2010b)

The approach is evaluated in terms of its ability to recreate the desk occupancy profile of a real occupant whose schedules were used to calibrate the model, which it does with a fair degree of accuracy, as shown in Figure 2-7. It is also noted that the approach recreated some of the occupant’s other patterns, such as never leaving and returning twice in the same day. The model did have difficulty properly spreading out breaks which took place in different locations, as they were different tasks, and so the task suspension interval did not apply.

As all occupant schedules are generated independently, the number of occupants which is generated for a task is simply a number; it does not represent the number of simulated occupants involved in the task as the occupants do not interact. This is addressed in (Goldstein et al. 2010a), where the model is modified such that when a task which involves more than one occupant is generated, other occupants in the simulation are chosen to participate.

When a number of participating occupants greater than one is generated the initiator, the simulated occupant for whom the task was generated, is assigned the task as the first participating occupant. Additional occupants in the simulation are then selected at random, until the generated number of participating occupants is reached. To address the question of whether additional occupants should immediately quit their current task to attend the shared task, a randomly selected half of the additional occupants will quit any task they are engaged in, while the remaining half attend only when their current task is finished.

The work is further extended in (Goldstein et al. 2011) to consider the physical layout of the simulated building when selecting a location for a task to occur in, and when selecting additional occupants for a shared task. A task may have a fixed location in which it must occur (for example, breaks in which people leave the building must occur outside the building), or it may have a randomized location. In the latter case there will be a set of locations in the model of the building in which the task can take place. The selection of one of these locations is based on a cost function which incorporates both the distance between the candidate location and the initiating occupant’s base location, and the utilisation of the room in terms of the number of occupants, i.e. not choosing a three person room for a ten person meeting or vice versa. Additional occupants for shared tasks are selected based on the distance component of this cost function, with occupants whose current location is closer to the base location of the initiator occupant being more likely to be selected.

These two extensions of the work are concerned with combining personas, which are specifications of fictional occupants in terms of their arrival and departure tendencies, numbers of breaks per day, etc., with schedule calibration and the occupant and location selection methods presented in the paper. Thus the work is primarily evaluated in terms of its effect on the attributes of the personas, testing whether these more advanced methods of occupant and location selection can be used without unintentionally altering the behaviour of the fictional occupants. Simulating with the added location and occupant selection features reproduced the personas accurately with the exception of meeting frequency, where personas with a high probability of meeting could summon as additional occupants nearby low meeting probability personas, thus raising their meeting probability above the intended value.

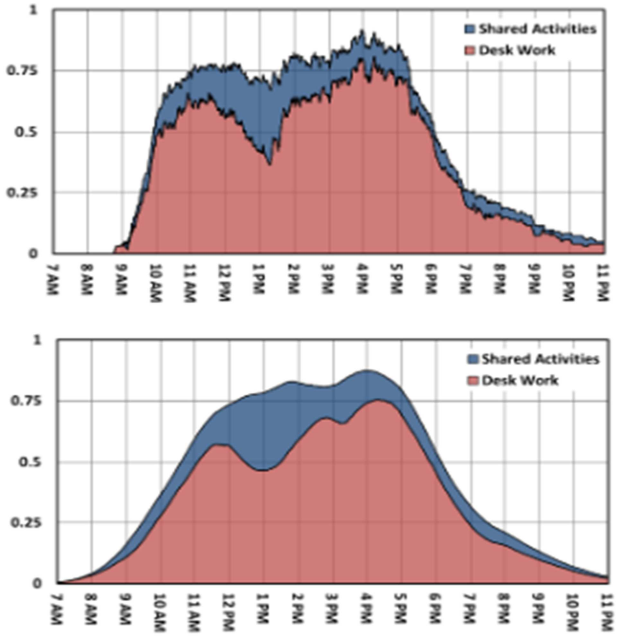


Figure 2-8 – Overall Desk work and Shared Acitivites probability profiles across six real occupants (top) and simualted occupants (bottom) from (Goldstein et al. 2011)

While the original schedule calibrated model is evaluated in terms of its ability to reproduce the occupant's location throughout the day, this is only done for the occupant's probability of being at their desk. Furthermore the evaluation is carried out on only a single occupant, and doesn't manage a perfect recreation of that one occupant. The follow up work models a larger number of occupants as part of the evaluation; however the evaluation is focused on how the persona attributes are affected. All that is added to the earlier evaluation in terms of whether the simulated occupants' movements match those of the real occupants are statistics on meeting room utilisation and overall probability of being involved in shared activities or desk work throughout the day across multiple occupants, as shown in Figure 2-8. While it shows that the model is very accurate in recreating the overall frequency with which the group of occupants was either at their desks or in shared activities, it does not verify whether the individual simulated occupants match the movement patterns of the original occupants.

2.4.3. Summary

Approach	Summary
Parameter Calibrated	All occupant parameters are set as desired All parameters must be set manually even if based on real data
Schedule Calibrated	Model automatically calibrated using real data as input Applying custom occupant parameters requires additional steps

Table 2-3 – Summary of occupant modelling approaches

In the area of occupant modelling, there are several approaches which capture the level of detail needed to create test datasets to evaluate specific features. However, the evaluations of these models do not confirm that they accurately recreate occupants in this level of detail. As the goal is to predict precise occupant locations based on historical location frequencies, any prospective model must be able to recreate occupants' precise locations and the times at which they visit them. The model in (Page 2007) is evaluated in terms of numerous metrics, including profile of probability of presence across the day. This is the level of detail required, unfortunately this model is for occupancy of locations rather than locations of occupants, and thus does not model the level of occupant location detail OccApriori is intend to work with. The most detailed evaluation aside from this is in (Tabak 2008), which confirms that the frequency, duration and total time spent in types of tasks matches the source dataset, but does not confirm that they actually occur at the same times. However, aside from this point, it is a detailed evaluation which confirms that the model closely reflects the original occupants. Furthermore a very similar task structure is used in (Zimmermann 2007). This suggests that this type of model is a good starting point, but requires further evaluation to confirm it captures time-of-day related patterns which are required.

2.5. Summary

Existing work on occupant location prediction is focused on next location prediction. The few existing approaches to future location prediction use limited elements of context, with such context being used to a greater degree in next location prediction. The aim of the approach developed in this thesis, OccApriori, is to combine the advantages of these different approaches into a single algorithm. Thus the algorithm must be capable of predicting occupant locations in the next timeslot or near future timeslots, while also being able to predict further into the future (for example next day predictions), and be able to incorporate any available elements of context to increase prediction accuracy.

Association rules are the most suitable approach to achieve this aim, as they are designed to find any patterns which exist in a dataset. Since they find any patterns which are present, they will find patterns which relate to next location, later locations, and distant future locations. As association rule mining operates in an unsupervised fashion, any elements of context which are in the dataset and correlate with occupant locations will automatically be included in the resulting association rules. Furthermore association rules have a number of existing extensions to the types of rules which can be mined, which can be leveraged for occupant prediction once the base algorithm is in place.

Evaluating the algorithm requires test datasets which resemble the type of datasets the finished algorithm would be expected to be applied to. Real world data is essential to test the algorithm performs in practice, however to generate specific test cases an occupant model capable of producing custom, but realistic, datasets is also useful. There are several existing approaches to modelling occupants which have been evaluated to show that they can produce realistic datasets. These evaluations do not comprehensively cover all the attributes of the generated dataset which are important for occupant prediction. However there are commonalities in the occupant models in multiple approaches, suggesting that such models are the ideal starting point for developing a purpose-built dataset generator for testing features of the occupant prediction algorithm.

3. Occupant Location Data Collection and Modelling

This chapter describes the sources of data used to evaluate OccApriori. This consists of three sources: an existing published dataset which has been used for occupant location prediction evaluation in the past, a dataset collected internally as part of this research with the aid of several volunteers, and an occupant model also developed as part of this research which consists of a dataset generator and a model of specific occupants, in order to produce datasets containing specific test cases.

Section 3.1 outlines the approach which is developed in this thesis in order to explain the features required of an evaluation dataset. Section 3.2 describes the existing dataset and the process of converting it into the data format used by OccApriori. Section 3.3 describes the internally collected dataset, and the process of collecting it. Section 3.4 develops a synthetic dataset generator, and in section 3.5 describe the process of modelling the occupants in the internally collected dataset in order to produce a synthetic version of that dataset. Finally section 3.6 evaluates the similarity of the data on which the model was based and the synthetic dataset output by the generator using that model, demonstrating that it can accurately model realistic occupant movements.

3.1. Location Prediction Outline

The data required to predict occupant locations depends on the functioning of the prediction algorithm. While the operation of the algorithm and the exact structuring of the dataset will be discussed in Chapter 4, this section will provide an outline of the algorithm and the types of predictions it makes.

As discussed in the previous chapter, association rule mining algorithms operate on datasets of transactions, with each transaction being a set of items. This means that occupant location data must first be broken down into transactions, which would represent large blocks of the time and items which are smaller blocks of time. The first implication of this is that the algorithm will always operate on periods of time, and predict where the occupant will be over a period of time.

The actual division of the dataset used in this thesis is to have a transaction represent a single day, and to break days down into equal-sized periods (usually half an hour), with each item representing one of those periods. Thus the approach learns patterns based on associations between the periods during which locations appear across days, for instance learning that the location ‘Office’ appears every day at 10:00am. Associations may of course also relate to multiple items (periods) in single transaction (day), for example ‘Office’ being the location at 11:00am if it is the location at 10:00am. This results in a rule of the form 10:00=Office => 11:00=Office.

Based on this, the data required to develop and evaluate the approach is the location of sample occupants throughout the day. Precise times of arrival and departure are not important as the algorithm works on time periods – for each period, the location that the occupant spent the majority of that period in is sufficient. However precise locations are important – if we only know whether the occupant is in or out of the building for instance, only similarly coarse predictions will be possible. Ideally the locations should be room-level.

Based on these requirements, the following sections describe the conversion of an existing dataset to the required structure, the manual collection of a new dataset with these specific requirements in mind, and the development of an occupant model capable of generated synthetic datasets in the appropriate format.

3.2. Augsburg Indoor Location Tracking Benchmark

As the objective of this thesis is to predict occupant locations within smart buildings, the ideal test dataset would be data from an actual tracking system in such a building. Datasets which were considered include (McNett et al. 2003) and (Henderson et al. 2008), in both of which location datasets are collected on a college campus using Wi-Fi localisation. However, both of these datasets track people's movements between multiple buildings and, due to relying on wireless access points for localisation, have difficulty pinpointing exact locations. In particular in the 2008 work, a user is only considered to have moved if they connect to access points more than 50m apart, as even stationary occupants could appear to be connecting to and disconnecting from APs in different buildings. A well known dataset is the Reality Mining dataset (Eagle et al. 2006), in which occupants were tracking using a combination of cell towers and Bluetooth sightings. As some of the Bluetooth devices sighted in the dataset are static, such as desktop pcs, part of the dataset could provide a location trace of the participants, however this would be limited to those locations demarked by static Bluetooth devices.

The Augsburg Indoor Location Tracking Benchmarks (Petzold 2004) is a location dataset in which the exact room-level location of a set of occupants is recorded throughout the working day in a university department. While the data was collected manually, it is exactly the type of data we would expect from an occupant tracking system using one of the technologies discussed in section 1.1. The approach developed in this thesis could be applied to the other datasets referenced above, including predictions across multiple buildings, however as the Augsburg dataset provides the type of data the approach is intended for it is the external dataset used when evaluating the approach.

The Augsburg dataset was collected by 4 occupants in the University of Augsburg via a PDA application which they used to record their movements. Every time an occupant changed rooms, they tapped their new room on the PDA, and the time was logged, resulting in a timestamped trace of the occupant's movements. The occupants gathered several weeks of data in two time periods, the summer season and fall season.

Occupant	*Duration*	*Locations*
A	5 weeks	4
B	7 weeks	9
C	6 weeks	8
D	11 weeks	9

Table 3-1 – Total amount of data collected by each occupant in the Augsburg dataset (5 day weeks)

OccApriori predicts occupants' locations during specific timeslots, as opposed to approaches which predict what the occupant's next location will be whenever they move. As the Augsburg dataset collected the exact times when occupants moved, it had to be converted into a dataset which would state the occupant's location in each timeslot. The evaluations in this thesis use half-hour timeslots, and consider an occupant's location in a timeslot to be the location they were in for the majority of that timeslot. The arrival times recorded in the Augsburg dataset were used to determine each occupant's location in each timeslot and record it in a timeslot based format. Table 3-1 shows the total duration of the data for each occupant, and the number of locations per occupant, in the Augsburg dataset. The number of locations is lower than in the original dataset as some locations, for example 'kitchen', were visited for such short durations that they did not become any occupant's location in any timeslot.

While the Augsburg dataset is the most suitable existing dataset for evaluating OccApriori, several attributes make it less than ideal. First and foremost it features a small number of occupants who collected a relatively small amount of data. While occupant D collected nearly 3 months of data, the other three occupants collected less than 2. It has been shown that the amount of data is available is sufficient for good next location prediction using some approaches, for example (Petzold et al. 2006), however other approaches may prefer more training data to make these or other types of predictions. The data is also collected in two different seasons, rather than being a single contiguous dataset, requiring approaches to make predictions on movements in fall based in whole or in part on occupant movements in the summer. Ideally predictions would be based on occupants' recent patterns of movement, not those from several months before. The occupants in the dataset also do not tend to move in patterns which relate the time of day, which is the basis on which OccApriori predicts their location in the future. While it is useful to evaluate how OccApriori performs on datasets that do not have occupants moving in ideal types of patterns, it is of course also important to evaluate it on occupants who do have such patterns. Finally, the Augsburg dataset contains no extra data which may be useful for predicting the occupants' locations. OccApriori is designed to be able to incorporate any extra data which can be provided, the most obvious example of which is schedule data for occupants who have scheduled events. While this extra data is not essential, it would be preferable to evaluate its effect.

3.3. Internally Collected Dataset

As part of this research a group of volunteers in UCC collected a dataset which addresses the shortcomings of the Augsburg dataset with regard to OccApriori. Six occupants volunteered to record their movements using Google Calendar. The occupants were instructed to record their location throughout the day using entries on the calendar which stated their location for the duration of the entry, for example, an entry from 09:00 to 12:00 with the location set to their office, or an entry from 14:30 to 15:00 with the location set to a colleague's office, representing a short meeting. Occupants were to record their location when in the building from 08:00 to 18:00 on weekdays.

The occupants were members of UCC's Department of Computer Science. They included a lecturer, a software developer, and postgraduate students. Two occupants had their own office, three shared an office, and one was seated in an open plan area. As they were all involved in research on intelligent buildings, the occupants frequently visited each other offices, and also had a regular group meeting all the occupants in the dataset attended. The building in which the data was recorded is an off-campus UCC building which houses single and multi-occupancy offices and open-plan offices for several departments, as well as lecture halls and labs for undergraduates.

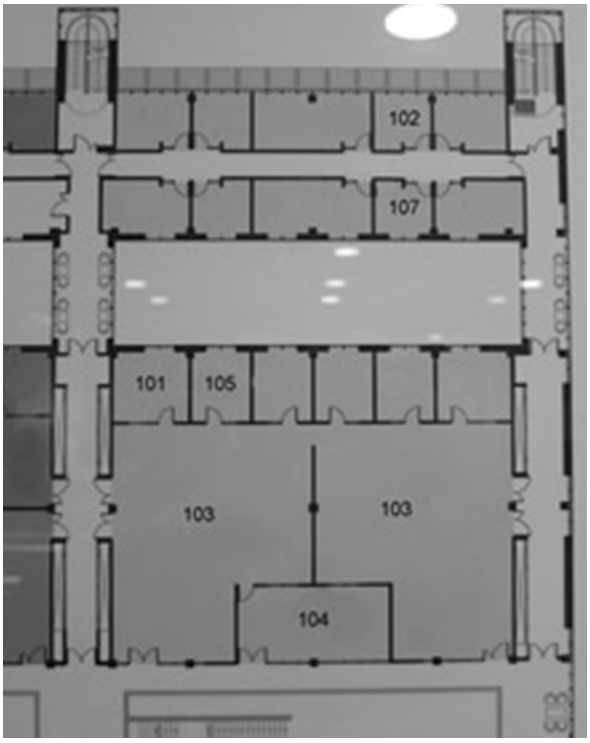


Figure 3-1 – The offices in the UCC dataset

Figure 3-1 shows the floor plan of the area in which the occupants' offices were located, with the room numbers marked. 103 is the open plan area, while 104 was the group's main meeting room. The other rooms in the dataset were lecture halls, labs, and less frequently used meeting rooms, which were located on other floors. A larger number of occupants would have been preferable, however the data gathering required manual recording of one's location throughout the day for a protracted period, and so the volunteers ultimately came from the pool of people who would have practical use for such a dataset.

The basic assumption made in this thesis for a tracking system which would provide the data for the predictor is that it would be able to track occupants with room-level granularity (although OccApriori does not require this), so occupants were instructed to record their location as the room code of the room they occupied. As the aim was to produce a dataset with half-hour timeslots, locations were to be recorded with half-hour granularity, with events beginning and ending on the hour or on the half hour. Of course, for stays in a single location longer than half an hour, a single event could cover the entire stay; a calendar entry was not required for every single time slot, as in the example above from 09:00 to 12:00. Locations outside the building were all simply recorded as ‘Away’, as the hypothetical tracking system the data collection was based on would not track occupants outside the building. Finally, occupants were instructed to record their scheduled movements as well, in the same format as they recorded their actual movements. Whenever they were scheduled for a meeting, lecture or any such event, they recorded the time and location of the event, as well as where they actually ended up being at that time. Table 3-2 shows the resulting location trace for one day for occupant B.

Time	*Location*
08:00	Away
08:30	102
09:00	108
09:30	108
10:00	108
10:30	108
11:00	106
11:30	106
12:00	106
12:30	106
13:00	102
13:30	102
14:00	102
14:30	102
15:00	102
15:30	102
16:00	102
16:30	102
17:00	102
17:30	Away
18:00	Away

Table 3-2 – A location trace for occupant B

As mentioned, the occupants used Google Calendar to record their movements. Each occupant used two calendars of their own, one for their actual movements, and one for their scheduled movements. Google Calendar has a feature whereby a private link can be provided to a calendar to allow access to anyone who is given the link. Using this method, the data could be collected without making any of the calendars publically available. The data was downloaded by exporting it from Google Calendar in the iCalendar (ICS) format. A script was written which could convert the iCalendar events from each pair of calendars (actual and scheduled movements) into a dataset in the Attribute-Relation File Format (ARFF) for each occupant, after which combining each occupants' individual datasets gave a complete dataset. ARFF is the file format used by the WEKA machine learning toolkit (University of Waikato 2015), which includes an association rule miner used for the earliest testing of applying association rules to occupant prediction. As the format proved suitable for representing the occupant location datasets it is also used as the input format for OccApriori.

While occupants were instructed to record their location over time with half-hour granularity, longer events were allowed considered, and the script does not assume that all occupants actually did record their locations with events that were a minimum of a half-hour long and started and ended on either an hour or a half hour. As with the converter for the Augsburg dataset, the ICS to ARFF converter uses the calendar data to determine the location in which each occupant spent most of each timeslot, thus allowing shorter events and events that start at smaller fractions of the hour to be properly accounted for.

The occupants were given a specific format in which to record the location based on the room code of the room they were in. The conversion process does not assume that all occupants recorded the location in this exact format at all times, instead correcting the formatting of the room codes in the recorded data to ensure that each location is referred to by a unique identifier. Part of the conversion script lists every unique location which appeared in the dataset, allowing for examination of the location codes recorded. The total number of locations in this dataset was small enough that it was possible to manually go through the list to find locations which were formatted incorrectly or which were external to the building. The calendars owners were consulted for any entries which were too vague to be certain what location they were referring to. The corrections which were made to the locations were written into the script which went through the dataset and made the appropriate adjustments to the location entries. The dataset was occasionally exported before data collection was complete to use for evaluations during the research; by storing the data cleaning in the script, existing fixes could be reapplied whenever the data was re-exported and re-converted.

Occupant	*Duration*	*Locations*	*Unique Locations*	*Base Location*
A	72 weeks	10	1	103
B	63 weeks	31	18	102
C	58 weeks	10	1	101
D	29 weeks	8	2	101
E	20 weeks	7	1	105
F	63 weeks	9	2	107

Table 3-3 – Total amount of data collected per occupant in the UCC dataset (5 day weeks)

Table 3-3 shows the timespan during which each occupant collected data, the number of locations they visit in the dataset, the number of unique locations, and each occupant's base location. The unique locations for each occupant are those locations which were only visited by that occupant. As the occupants were situated close to each other and were working on related projects, they generally visited the same set of locations at some point in time. The main exception is occupant B, whose unique locations are mostly lecture halls / computer labs. While most of the occupants shared most of their locations with one or more other occupants, the only locations shared among all occupants in the dataset are 'Away', the meeting room in which this group met (104) and occupant B's office (102).

As with the Augsburg dataset, the UCC dataset has some shortcomings. While the dataset provides a large amount of data for each occupant, with the lowest being 20 weeks of data, it is limited to only 6 occupants. The dataset provides schedule data as well as actual location data, however only occupant B has an extensive schedule with many regular events per day. The other occupants used the schedule for a small number of weekly events, and small numbers of once-off events. An ideal dataset would have a larger number of occupants, with different types of patterns, and more extensive use of the schedule or other extra data.

Section 3.6 uses the collected data as part of an evaluation of a synthetic dataset generator presented in the following section. As the collected data appears in the evaluation, it also serves as an overview of the patterns of the occupants in the collected dataset.

3.4. Synthetic Dataset Generator

The development of OccApriori approach required datasets which contained specific test cases to evaluate the behaviour of the predictor. To this end this thesis includes an occupant model based on the type of task structure from (Zimmermann 2007) and (Tabak 2008). The tasks the occupants engage in during the day are modelled in order to set their location in each timeslot when a dataset is generated. Tasks have properties such as start time, duration, probability and finally location, and are iterated through to decide on the occupants' locations. The model can also be viewed as a Markov chain, where each state is a set of generated location entries, and each transition adds a new entry to the set. This work consists both of a general occupant model, i.e. an actual method of modelling occupants, and a specific occupant model, i.e. a model of some specific real occupants, so for clarity the former is referred to as a synthetic dataset generator. The term model hereafter refers only to a specific model of specific occupants, with such models being provided as data to the dataset generator to produce synthetic datasets. The rest of this section describes the dataset generator.

Assume we have a set of agents A for whom we want to generate location data for a period of time. To determine which locations are set in which timeslots, we have a set of tasks T. These tasks are assigned to roles, which describe general patterns of behaviour. From the set of roles R, we then assign roles to agents. The specification of the agents, roles and tasks in the generator follows.

An agent is defined as $A_i =< o_i, R_i >$ where o_i is the name of agent A_i (i.e. which occupant this agent will be in the dataset), and R_i is a set of roles and probabilities of the form $< \{r_n, p_n\} \dots \{r_m, p_m\} >$, where p_n is the probability of r_n being chosen. As R_i is a probability distribution over the possible roles, the probabilities $p_n \dots p_m$ must sum to one. The same representation is used for the probability distributions in the rest of the generator.

Roles are defined as $R_i =< A_i, F_i, D_i, W_i, m_i, P_i, S_i, V_i >$, where A_i and F_i are probability distributions over the possible arrival and finish times of the occupant's day. Outside of these times the occupant is set to be 'Away' regardless of tasks. D_i is a probability distribution over the possible durations of the role. Given a duration d, a new role will not be chosen for the occupant until d days have passed in the simulation. W_i is the set of days of the week on which the role is allowed to be chosen. m_i is a flag indicating whether or not the occupant can attend meetings while in this role. Meetings are a special subset of tasks to which agents are assigned rather than the other way around. P_i, S_i and V_i are sets of primary tasks, secondary tasks and tertiary task respectively, each task type representing a different type of task.

Roles encapsulate occupants' movements throughout a day (or several consecutive days if the duration property is greater than one). They are essentially a collection of tasks the occupant can engage in on a day, along with the probabilities of when they first arrive and last leave. During evaluation, when a day starts, choosing roles for each occupant is the first step. The occupants' arrival and departure times are then generated based on the A_i, F_i properties of the role. Throughout the rest of the generation process, when each type of task is being evaluated, it is the P_i, S_i and V_i properties of the role selected for each occupant which dictates the tasks which will be evaluated for that occupant.

Primary tasks are defined as $P_i =< T_i, D_i, L_i, W_i, p_i >$, where T_i is a probability distribution over the possible start times, D_i is a probability distribution over the possible durations, L_i is a probability distribution over the possible locations, W_i is the set of days of the week on which the task can occur and p_i is the probability of the task occurring given that the task can occur and whether it occurs is currently being evaluated. When a primary task is being evaluated, the generator first checks if the current day in the simulation is a day on which the task can occur. If not, the task does not occur. If so, the values t, d and l are chosen from T_i, D_i and $L_i$based on the associated probability distributions. The task is thus set to occur at time t for duration d (measured in timeslots) in location l. If the occupant already has a set location in any of the timeslots that the task occurs in based on t and d then the task does not occur, otherwise the probability of it occurring is p_i.

Secondary tasks are similar to primary tasks except in how the time is chosen. Secondary tasks are defined as $S_i =< T_i, D_i, L_i, W_i, p_i >$. D_i and L_i are probability distributions over duration and location as with primary tasks. W_i specifies the days on which the task can occur. T_i is again the set of times at which the task can occur, however for secondary tasks this is an ordered list of times at which the event can occur. When the task is being evaluated, if the simulation is on a day on which the task can occur, the times are checked in order and the first time at which the event can occur, i.e. the first time t for which the time slots the task will occur in according to t and d are empty, is selected as the time that the task will occur. If the task cannot occur in any of the times due to time slots already being filled, then the task does not occur. If the task can occur and a time is chosen, then as with primary tasks p_i is the chance that the task actually will occur.

The set of tertiary tasks is actually a probability distribution over a set of locations $V_i =< \{l_n, p_n\} \ldots \{l_m, p_m\} >$. Tertiary tasks have no properties other than a location and a probability. Tertiary tasks are evaluated after all other tasks to fill in otherwise empty timeslots with low-priority or default tasks. As such, when the tertiary tasks V_i are being evaluated, every remaining empty timeslot will take location l_j with probability p_j for $n \leq j \leq m$.

There are also types of tasks called meetings, which are tasks which involve more than one agent. These are similar to the existing types of tasks, they simply have some extra properties and, since they are not associated with any single agent, are evaluated at a different point in the generation process, which is discussed below.

Primary meetings are defined as $PM_i =< T_i, D_i, L_i, W_i, p_i, R_i, O_i >$. T_i, D_i, L_i, W_i and p_i are the same properties as in primary meetings. R_i is a set of required agents, while O_i is a set of optional agents. As with primary tasks, the simulation must be on a day allowed by W_i for the meeting to occur, and the time, duration and location of the meeting are decided using the corresponding probability distributions. Similar to primary tasks, the meeting cannot occur if any of the required agents are unavailable due to the slots that the meeting occurs in already being filled, or one of the required agent's role does not allow meetings. If all of the required agents are available and the simulation is on an allowed day, then the probability of the meeting occurring is p_i. If the meeting occurs, any optional agents who are available will attend.

Secondary meetings, defined as $SM_i =< T_i, D_i, L_i, W_i, p_i, A_i, m_i >$, share the properties T_i, D_i, L_i, W_i and p_i with secondary tasks. In addition there is A_i, the set of agents who may attend the meeting, and m_i, the minimum number of agents required for the meeting to occur. The allowed days and selection of time, duration and location are handled in the same manner as for secondary tasks. Checking whether agents from A_i are available is done in the same manner as for primary meetings. If sufficient agents, as defined by m_i, are available for the meeting and the simulation is on an allowed day then the probability of it occurring is p_i, if not enough agents are available the meeting does not occur.

The following outlines the process of generating a dataset:

```
For each day in the simulation:
  For each agent in the simulation:
    If current role has no duration left:
      choose new role
    Choose arrival and finish time based on role

  For each primary meeting:
    Check that meeting can occur on this day
    Choose time, duration and location
    Check all required agents are available
    Choose whether meeting occurs
    If all conditions met:
      Schedule meeting for required agents
      Schedule meeting for available optional agents
```

```
For each agent:
  For each primary task of this agent:
    Choose time, duration, location
    Check if agent is available
    Choose whether task occurs
    If all conditions met:
      Schedule task for agent

For each secondary meeting:
  Choose duration and location
  For each possible start time:
    Check how many attending agents are available
    If available > minimum agents:
      Select this time
  Choose whether meeting will occur
  If all conditions met:
    Schedule meeting for available agents

For each agent:
  For each secondary task of this agent:
    Choose duration, location
    For each possible start time:
      If agent is free:
        Select this time
    Choose whether task will occur
    If all conditions met:
      Schedule task for agent

For each agent:
  For each timeslot in this agent’s schedule:
    If timeslot has nothing scheduled:
      Choose tertiary task based on role
      Schedule tertiary task
```

3.5. Model of Internal Dataset

To evaluate the dataset generator’s ability to model realistic occupants, and to provide a base model which can be altered as required for other evaluations, this thesis models the occupants in the internally collected dataset. The most recent 80 instances (i.e. 16 weeks) for each occupant form the dataset used as the basis of the model. This represents a long enough period that the model will not ‘over fit’ to a small time period, i.e. the model will not exclusively model the very specific recent behaviours of the occupants; it will represent more general patterns of movement over a protracted period. The period is also short enough that the model will not be confused by changes in behaviour over different time periods – occupant behaviour patterns may well change over long time periods, and building a single model using data with different patterns will result in a model which is a compromise between the two rather than properly modelling either. In the case of occupant B the period is cut down to 65 instances (13 weeks), as the occupant had a change in routine during the final 80 instances, which would have caused the issue just discussed. Following is the process of modelling these occupants based on this data.

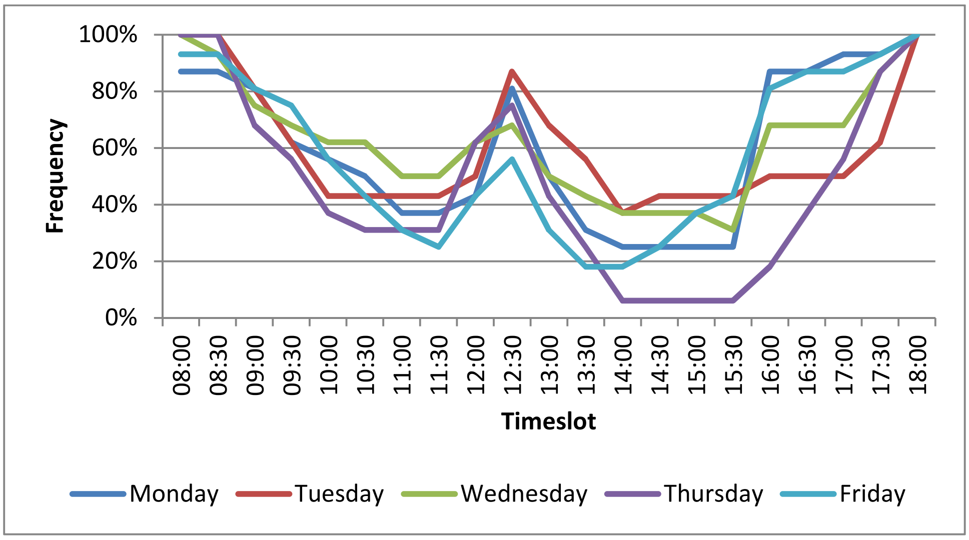


Figure 3-2 – Frequency of absence per timeslot for each day for occupant C

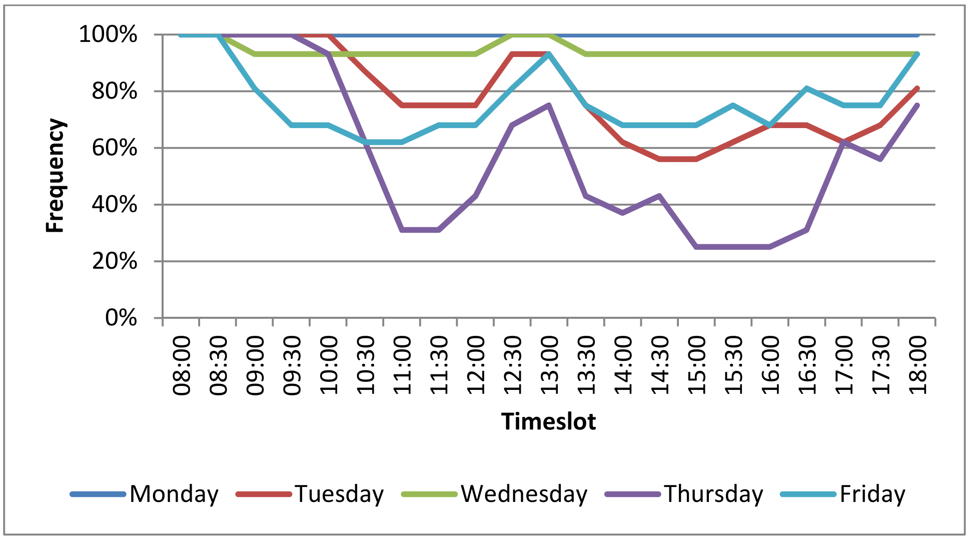


Figure 3-3 – Frequency of absence per timeslot for each day for occupant F

For most of the occupants, the majority of the model is concerned with when the occupant is 'Away' or not, as they are mostly in their office when they are in. We start by identifying patterns in the occupants' absence. We break the analysis down by day, as for some occupants their pattern of absence depends on the day. Figure 3-2 shows occupant C, whose frequency of absence varies by day, but always follows the same general pattern. The biggest deviation is on Thursday afternoon, at which time the occupant is less likely than usual to be absent – this is due to the group meeting at this time which all occupants in the dataset attend. Figure 3-3 shows occupant F, whose frequency of absence depends heavily on the day – on Monday and Wednesday they are almost guaranteed to be absent, while they are often present on Thursday. We group similar patterns of absence to model together via an in role and an out role, for occupant C that results in one pair of roles for the entire week, for occupant F we require three pairs of roles.

Each pair of roles consists of an in role and an out role for the days the roles cover – the out role being the role which states that the occupant is simply absent all day. To determine the frequency with which this role should be selected, we simply check the dataset to determine how frequently the occupant was absent all day on the corresponding days. All remaining absences, where the occupant was absent temporarily on a day on which they were in the building, are modelled as part of the corresponding in role. Continuing with the example of occupant C, we find that they are absent all day in 22.5% of instances. Table 3-4 shows the resulting role attributes for occupant C in the model.

Attribute	*Value*	*Probability*
Role	Occupant_C_In	77.5%
Role	Occupant_C_Out	22.5%

Table 3-4 – Values of the 'role' attribute and associated probabilities for occupant C

The frequency of the out role gives us a minimum frequency with which the occupant is absent in any given timeslot over the entire dataset for the relevant days. The actual minimum in the dataset is generally a bit higher, which we model as being due to essentially random absences which occur during the day. The specification of the generator discussed tertiary tasks, which are used to fill in timeslots which are empty once all the main tasks have been evaluated. We set the tertiary task for the in role to put the occupant in their office, but with a probability of being absent from the building based on the difference between the frequency of all day absences and minimum total frequency of absence in any timeslot. For occupant C, they are absent with a minimum frequency of approximately 30% across the entire week. Given a probability of being absent all day of 22.5%, a probability of randomly being absent in any timeslot of 9.7% in the in role results in the desired 30% overall probability. Table 3-5 shows the location attributes of the tertiary task in the role Occupant_C_In from Table 3-4.

Attribute	*Value*	*Probability*
Location	101	90.3%
Location	Away	9.7%

Table 3-5 – Values of the 'location' attribute and associated probabilities for the tertiary task of the Occupant_C_In role

The final cause of absence we model is the time of arrival and departure. In a role's attributes we model these times as a probability distribution over the possible arrival and departure times. The dataset, once we account for all day and random absences, gives us the cumulative probability of the occupant being absent at each timeslot, which we must convert into a probability distribution. Accounting for the existing chances of being away is simply an application of Bayes theorem:

$$P(In|Role_{In}) = \frac{P(Role_{In}|In) * P(In)}{P(Role_{In})}$$

We know the overall probability of being in $P(In)$ from the data, the probability of being in the in role $P(Role_{In})$ is set by us, and $P(Role_{In}|In)$ is simply 1 as the occupant cannot be in without being in the in role. Thus we can calculate for each timeslot the probability that the occupant should have set for being in within the in role in order to match the actual probabilities. For these calculations 'in' is a non-existent location which simply means 'not away'. Having the total probability of being in (and therefore, away) the occupant must have in the in role, we can then repeat this calculation to factor out the random chance of being away that we have set in the tertiary task. This final probability of being away is the probability of being away due to the occupant not having arrived yet at this time slot, i.e. the cumulative probability of not having arrived by this time. This process can be repeated for all timeslots until the points are reached where the probability of being away is 1 (very early, occupant never arrives), or 0 (occupant always in by now if they come in at all). The cumulative probability of not arriving also gives us the cumulative probability of having arrived by each timeslot, from which we can calculate the original distribution as $P(Arr_T) = P(In_T) - P(In_{T-1})$ where $P(Arr_T)$ is the probability of arriving at time T and $P(In_T)$ is the cumulative probability of having already arrived at time T (with all other factors removed). The same process can then be applied for departure times.

For most of the occupants there is a spike in the frequency of absence in the middle of the day due to going for lunch. The start time and duration for lunch can vary, so it is not as simple as calculating a single probability that the task will occur. The same process used for arrival and departure times is used to determine the probability of being absent due to going to lunch required in each timeslot to match the frequencies of the original occupant. Examination of the frequencies will give a rough idea of probabilities required to generate a similar pattern.

Timeslot	*F(Away)*
12:00	0.449275233
12:30	0.710144859
13:00	0.637681074
13:30	0.173912849
14:00	0.086956307

Table 3-6 – Frequency with which occupant E is absent due to being at lunch in each timeslot, given that they are in the office to begin with

Table 3-6 shows the frequency with which occupant E is away around lunchtime, once the already modelled factors are taken into account. The probability goes up at 12:30, and then decreases monotonically. A lunch task with a duration of two timeslots and a chance of starting either at 12:00 or 12:30 (or not at all) could explain most of these frequencies – the probability at 12:00 is simply the probability of the task starting at 12:00, at 12:30 it is the cumulative probability of it occurring at 12:00 or 12:30, as the occupant would be out at 12:30 either way, and 13:00 is the probability of it occurring at 12:30, as it is only that start time which would affect 13:00. Using this as a starting point we can calculate the required probabilities, resulting in a task with the following probabilities:

Attribute	*Value*	*Probability*
Time	12:00	63%
Time	12:30	36%
Duration	2	55%
Duration	3	44%
Probability	71%	-

Table 3-7 – The values of the time, duration and probability attributes and associated probabilities for occupant E's lunch task

Matching the probabilities as closely as possible required a task which has a probability of occurring below one, and a probability of having duration 3. Note that the probabilities for the Time attribute are the probabilities of that time being select and thus sum to one; the actual probabilities of the task occurring at either time also depends on the overall probability of the task occurring at all (and on the previously modelled factors such as the occupant being in).

Timeslot	*P(Away)*
12:00	0.45
12:30	0.71
13:00	0.62
13:30	0.208
14:00	0

Table 3-8 – Probability with which occupant E will be absent due to being at lunch in each timeslot, given that they are in the 'in' role

Table 3-8 shows the probabilities which result from the task in Table 3-7, again these being the probabilities without taking into account the previously modelled factors. There are close matches for 12:00, 12:30, 13:00, some error on 13:30, and the task ignores the small away frequency at 14:00 entirely. To properly model these trailing probabilities would require a significantly more complex task as changing, for example, the duration to allow a small chance of the task covering 14:00 as well would affect most of the other probabilities.

Attribute	*Value*	*Probability*
Time	09:00	1
Duration	4	1
Location	108	1
Probability	0.8	-
Day	Tuesday	-

Table 3-9 – A typical primary task for a fixed event such as a lecture, this one belonging to occupant B

The remaining elements, the primary tasks for fixed events such as lectures and the group meeting which the occupants all attend are trivial in comparison, involving no more than stating the start time, duration and day to which the task applies. Table 3-9 shows the properties of such a task belonging to occupant B.

To determine if the model has reached a point where it recreates the occupants' location frequencies to a sufficient degree, datasets generated from it are evaluated in the following section.

3.6. Evaluation of Model

In order to evaluate the similarity of the synthetic version of the internally collected dataset, this section compares statistics on occupant location throughout the day. The metrics used are location frequency per timeslot, arrival and departure frequency per timeslot, stay duration frequency per location, and finally location change frequency, comparing the frequencies to the corresponding frequencies from the original dataset to determine similarity

Location frequency per timeslot is the most fundamental point of comparison as it is the basis of both the occupant model and the occupant location prediction algorithm. Any error here would indicate that the model is severely lacking. Arrivals and departures are covered by this evaluation via the 'Away' location, however the location frequency per timeslot for the 'Away' location essentially gives the cumulative probability distribution of the occupant having arrived or departed. The model attempts to capture the underlying probability distribution over the time of arrival or departure, thus the evaluation also includes a comparison of frequency of arrival or departure in each timeslot. Stay duration frequency is considered as the frequency of location in each timeslot could be matched independently of each other without properly capturing the occupants' behaviours. For example, an occupant with an hour long meeting – the frequency of being in the room in each half hour timeslot could be matched with disjoint half hour visits in each time slot, but this would not properly capture the occupant's behaviour. This is important as next location prediction in particular relies on the occupants' recent locations. Location change frequency is essentially the flip side of this same metric – the longer the occupant tends to stay in each location, the less often they will change location and vice versa. The evaluation includes this metric to complement the previous one. Combined, these metrics can tell us whether the model is capturing the specific aspects of the source data which were modelled as well as the general features which are not explicitly modelled.

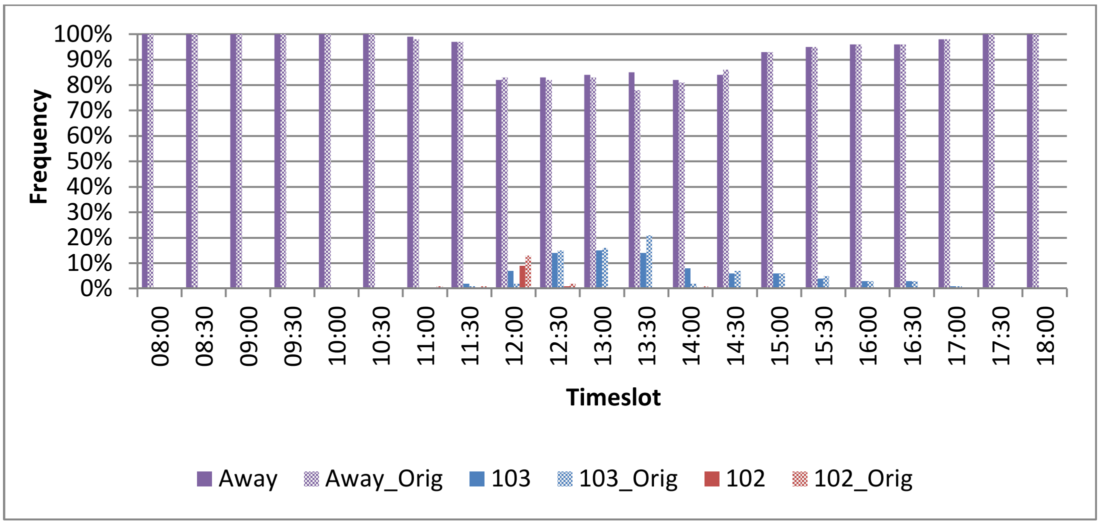


Figure 3-4 – Location frequency per timeslot for occupant A in the original and synthetic datasets. The bars representing each collected/synthetic frequency pair are adjacent and share a color.

Figure 3-4 shows Occupant A's location frequency per timeslot. The bars in the graph are paired such that for each timeslot/location combination, the original dataset frequency and generated dataset frequency are displayed next to each other in the same colour. Thus if the bars of the same colour are the same height, the frequencies match. In Occupant A's case, 'Away' is by far the most frequent location so the frequencies easily match, with the other locations which appear also closely matching the original dataset. The RMSE (Root Mean Square Error) across all locations is 1.75.

Figure 3-5, Figure 3-6 and Figure 3-7 show the location frequencies per timeslot for occupants C, D and E respectively. All three of these occupants visited three main locations: the 'Away' location, their own office, and the group meeting room. The frequencies are therefore dominated in all three cases by 'Away' and their own office. All three occupants follow a general pattern of increasing frequency of being in their office as the morning progresses, followed by sharp brief drop at lunchtime, and then a slow decline in the evening, with the 'Away' frequencies displaying the inverse pattern. While all three occupants follow this general pattern, the exact shape of the frequency pattern is different for each occupant, and the model provides an accurate recreation in each case, with RMSE values of 2.9, 2 and 5.5 respectively. The largest error is in the occupant E's model, which loses track of the pattern in the second half of the day, resulting in a 7.8 RMSE for location 105 and a higher overall RMSE than the preceding two occupants. This indicates that occupant E has an extra factor affecting their movement from 13:30 to 16:00, which would require a more detailed model to capture.

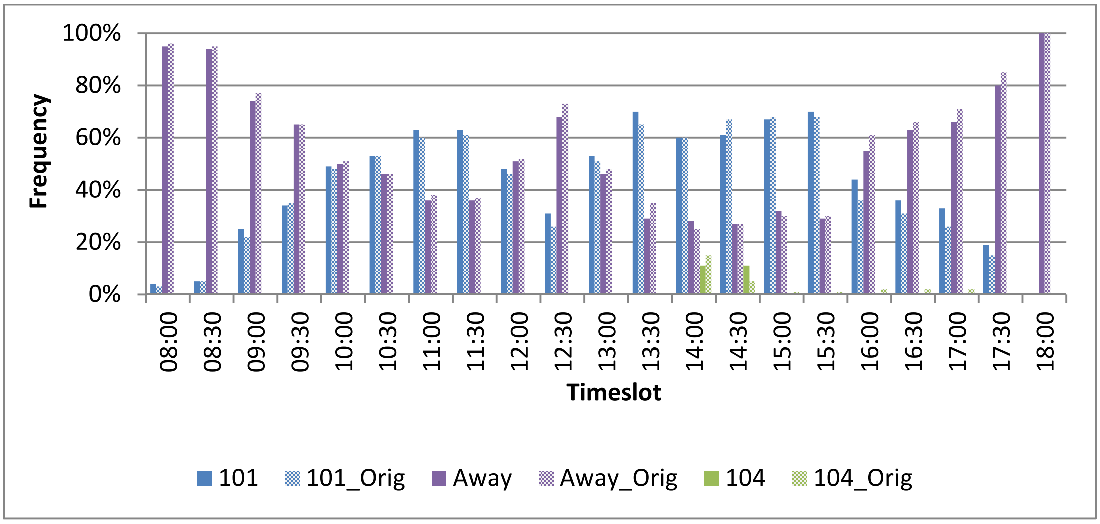


Figure 3-5 – Location frequency per timeslot for occupant C in the original and synthetic datasets

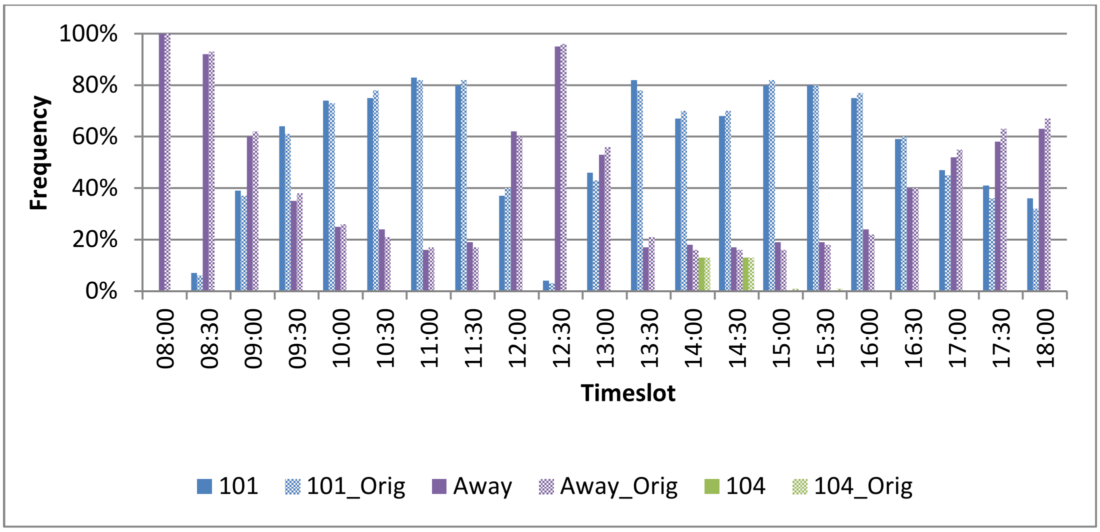


Figure 3-6 – Location frequency per timeslot for occupant D in the original and synthetic datasets

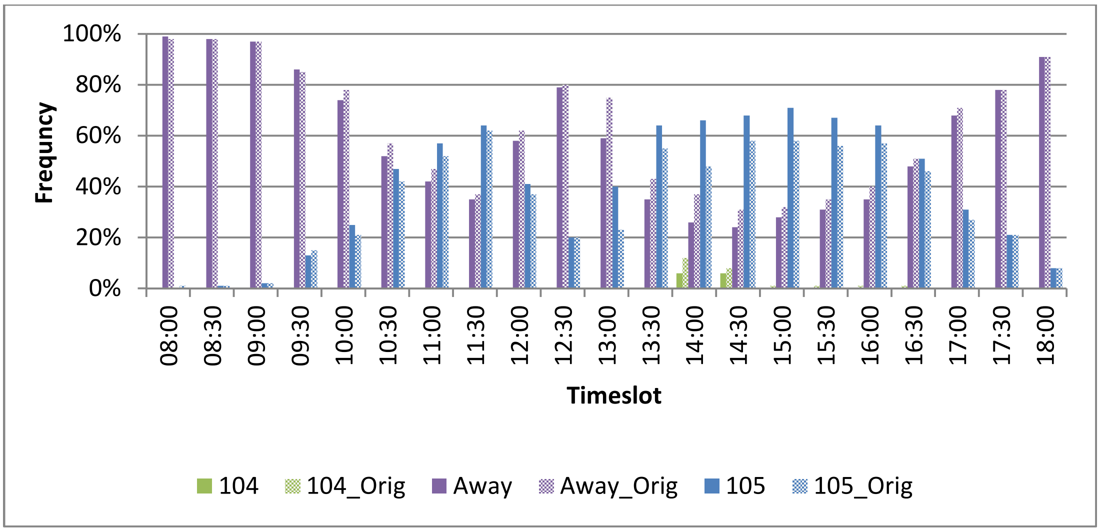


Figure 3-7 – Location frequency per timeslot for occupant E in the original and synthetic datasets

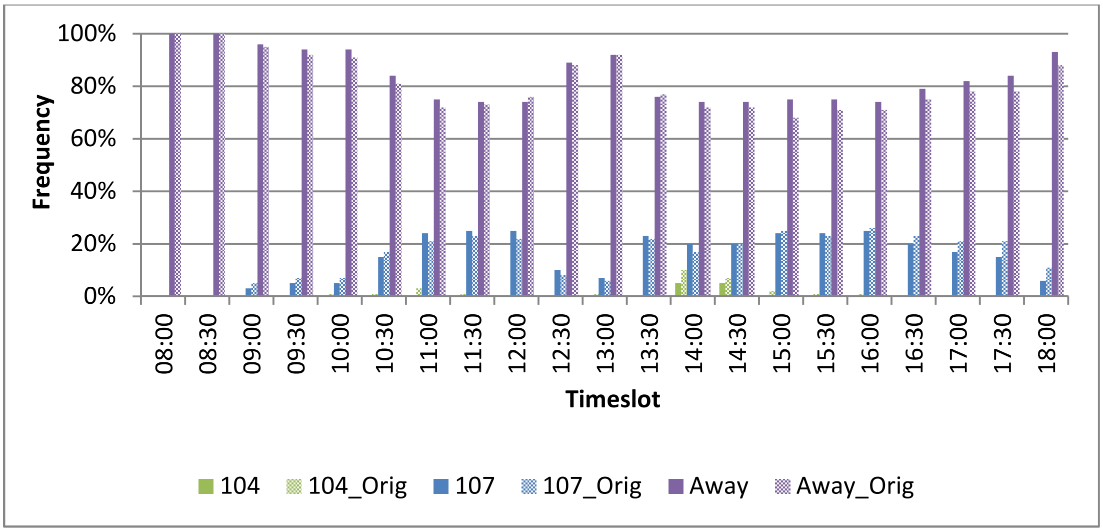


Figure 3-8 – Location frequency per timeslot for occupant F in the original and synthetic datasets

Figure 3-8 shows the location frequencies for occupant F, who similar to occupant A is most often 'Away', but who makes more regular appearances in the office. Occupant F's office frequencies show the same general pattern as occupants C, D and E, but with much lower overall probability of being in. Again the model is able to correctly recreate these location frequencies, with an overall RMSE of 2.5.

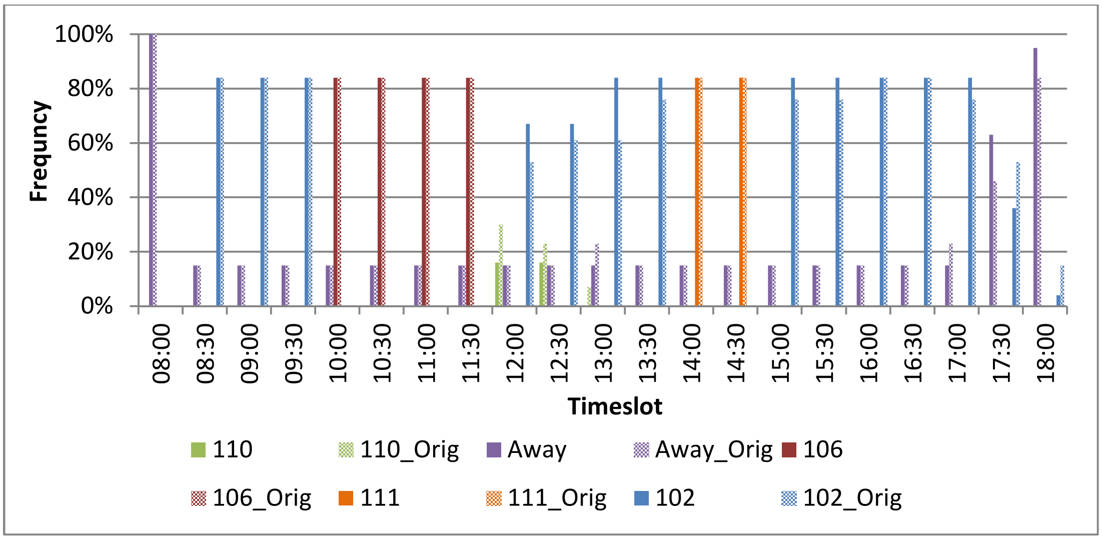


Figure 3-9 – Location frequency per timeslot for occupant B on Mondays in the original and synthetic datasets

Figure 3-9 show the location frequencies for occupant B on Mondays. As occupant B has many scheduled events which cause them to visit a larger number of locations and to have much more distinct patterns on different days, only one day's frequencies are shown for clarity. This occupant displays a much more rigid pattern of frequencies, where their schedule dictates a high frequency of being in specific locations at specific times. There is a constant low frequency of being away, due to days when the occupant does not come into the office and the day's events do not occur. The general pattern adhered to by the previous occupants is not seen in this occupant's location frequencies. Using day specific tasks to model the occupant regular schedule accurately recreates the location frequencies, with an RMSE of 4.6. This occupant's frequencies for Tuesday through Friday follow a similar type of pattern, with a similar level of accuracy in the model's recreation.

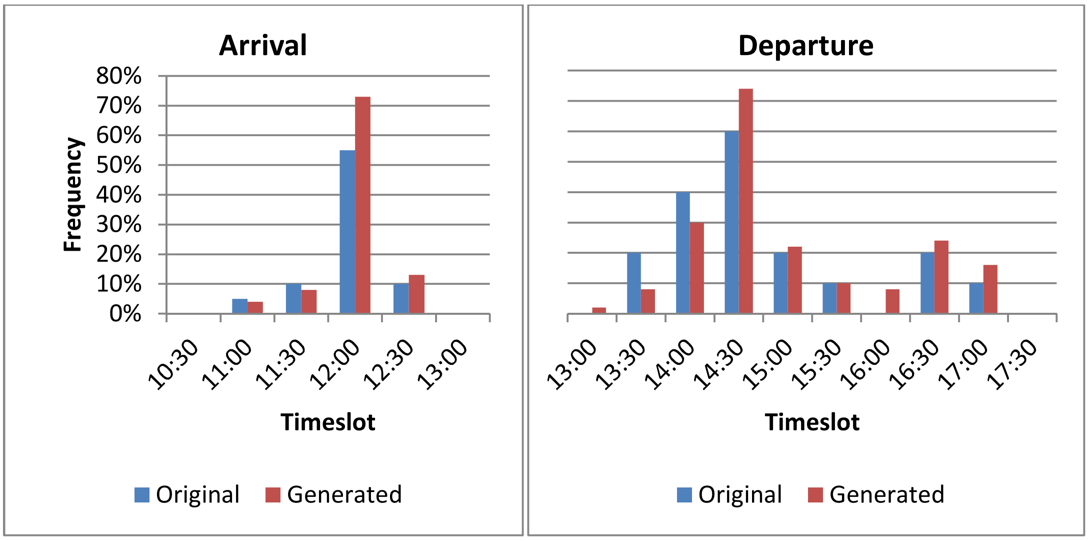


Figure 3-10 – Arrival and departure time frequencies for occupant A in the original and synthetic datasets

Figure 3-10 shows the frequency with which occupant A arrives at and departs from the building in each timeslot, for both the original and synthetic version of the dataset. The similarity in the frequencies shows that the synthetic occupant captures the overall pattern of the original occupant's arrivals and departures, although the synthetic occupant's frequency peaks higher for both.

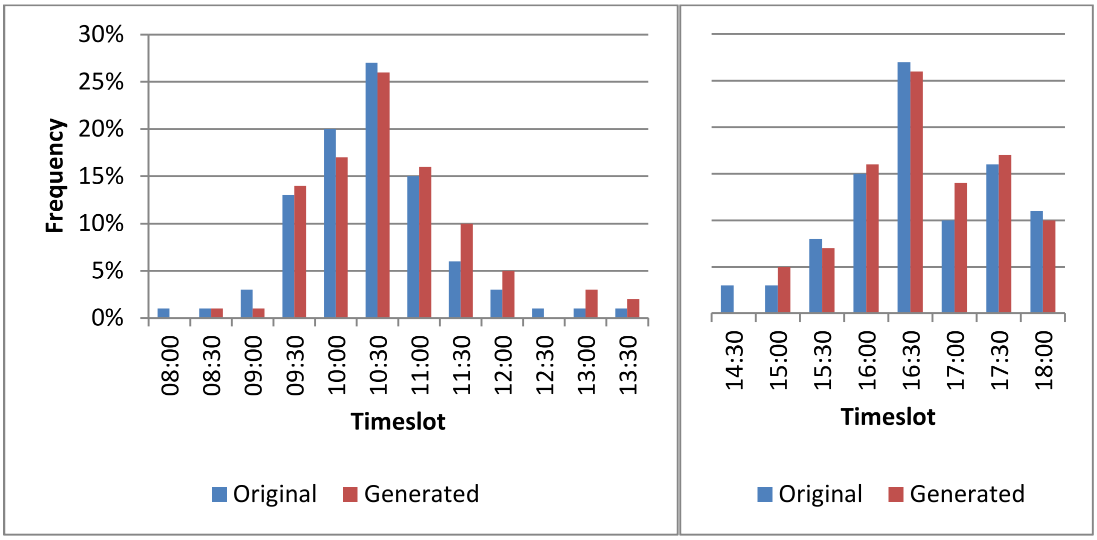


Figure 3-11 – Arrival and departure time frequencies for occupant E in the original and synthetic datasets

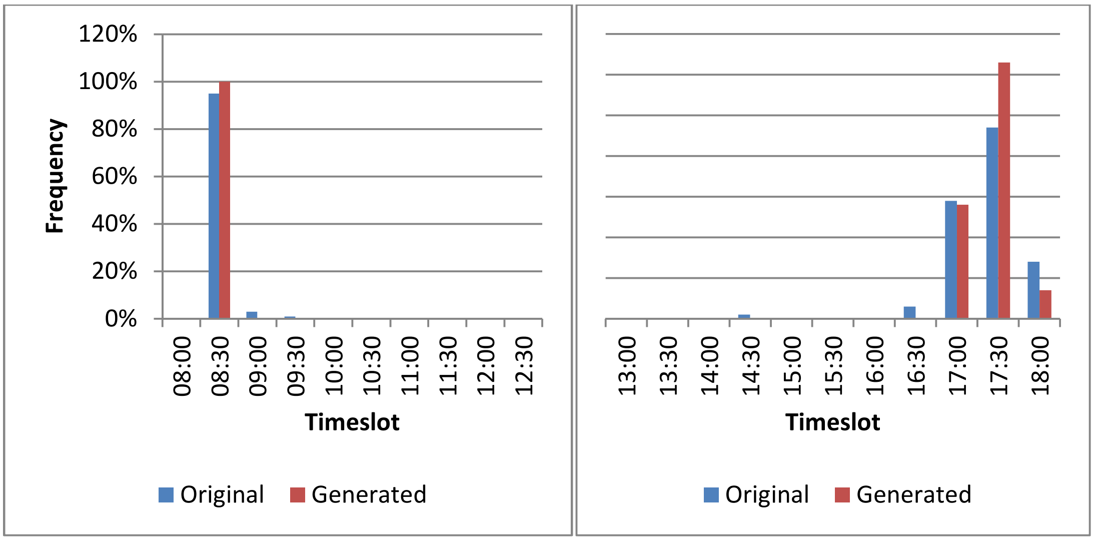


Figure 3-12 – Arrival and departure time frequencies for occupant B in the original and synthetic datasets

Figure 3-11 and Figure 3-12 show the arrival and departure time frequencies of occupants E and B respectively. These distributions show the extremes of the variety of arrival and departure patterns in the dataset. Occupant B, being the occupant with the most scheduled events, has extremely rigid arrival and departure times, in particular they arrive in the same timeslot almost every day. Occupant E displays a similar pattern to occupant A, but at different times and with a wider distribution. The synthetic dataset frequencies show that the model is able to capture both of these patterns.

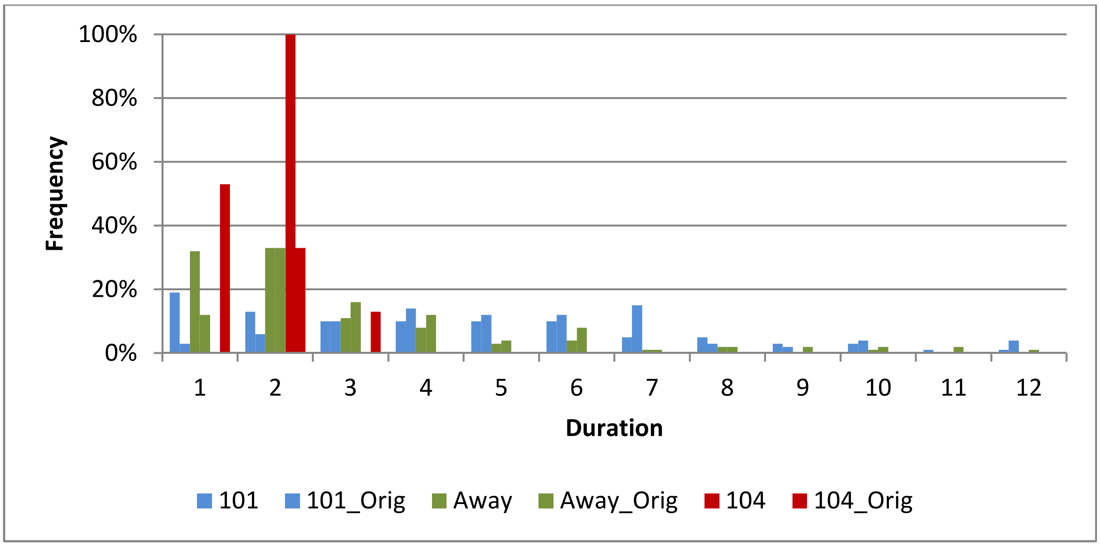


Figure 3-13 – Stay duration frequency per location for occupant C in the original and synthetic datasets

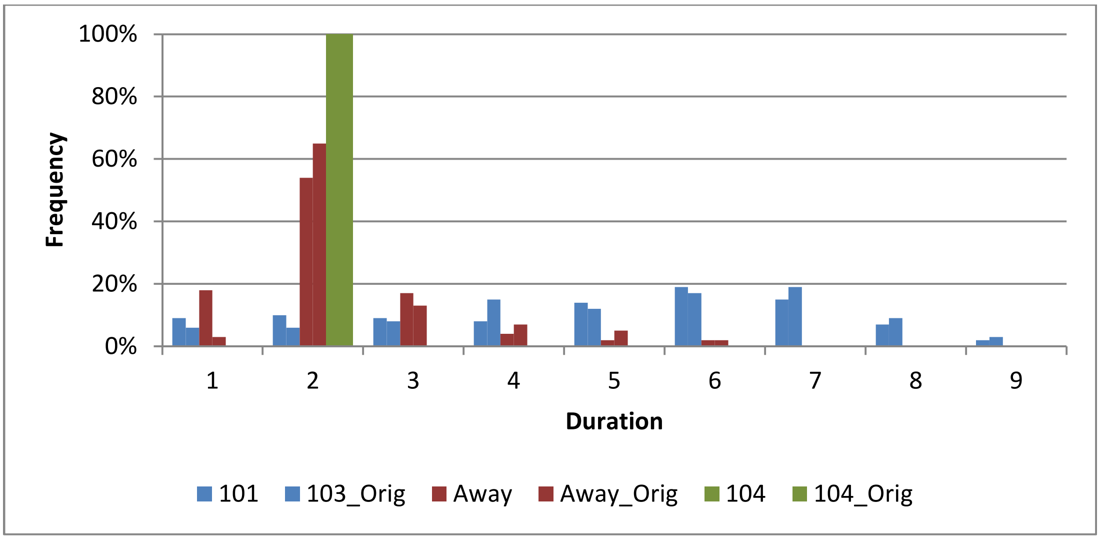


Figure 3-14 - Stay duration frequency per location for occupant D in the original and synthetic datasets

Figure 3-13 and Figure 3-14 show, for occupants C and D respectively, the frequency with which the occupant remained in each location for each duration. Overall the patterns are similar, although occupant D tends to remain in their office for longer durations with higher frequency, whereas occupant C's frequency remains the same or decreases with increasing duration. The model captures most the details of both occupants accurately, with the largest exception being the group meeting, the only non-office non-away location which appears for either occupant. The group meeting is modelled using the synthetic dataset generator's meeting tasks, which state that the meeting will only occur when the appropriate agents are available. This means the meeting must have the same duration for all agents. However, the actual collected data for each occupant indicates a different duration, with occupant C often remaining for half an hour, and occupant D always remaining for an hour. This type of discrepancy could be caused by occupants leaving early, however in this case it is actually because the parts of the year covered by the occupants is different – in the months covered by both occupants, the meeting tended to last an hour, whereas in the months covered only by occupant C, the meeting tended to be shorter.

Occupant B again displays a distinctly different pattern to the other occupants, as shown in Figure 3-15. Aside from a small and decreasing frequency of staying in their office for each duration, the occupant has essentially fixed durations for their stays in every other location. These durations are easily matched by the model, as they are simply stated as the durations of the relevant tasks.

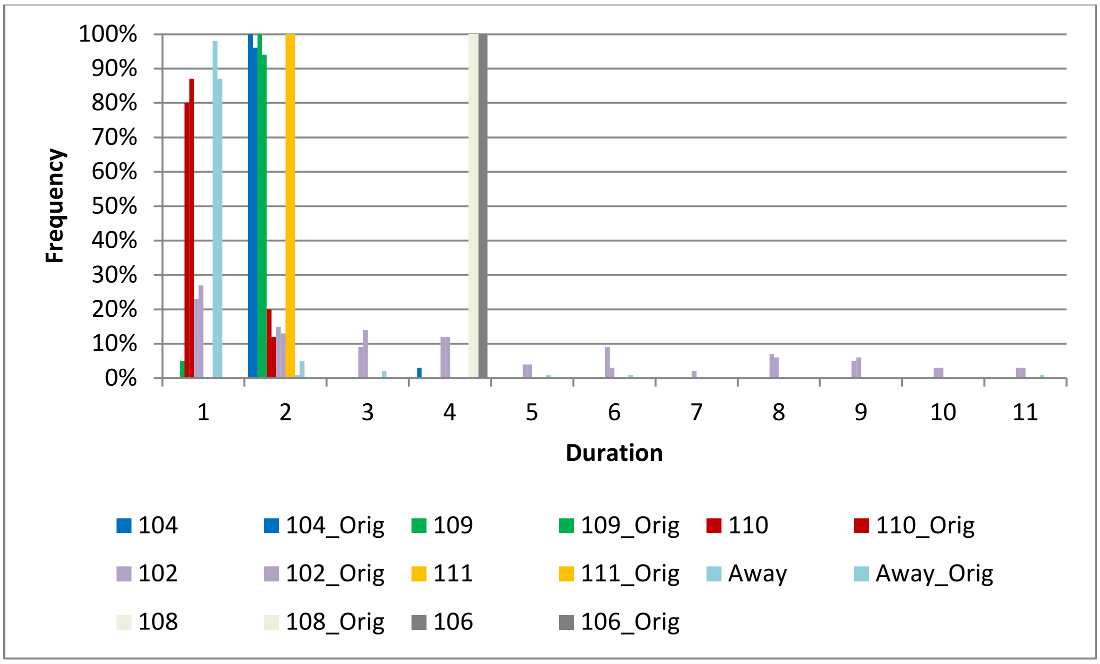


Figure 3-15 - Stay duration frequency per location for occupant B in the original and synthetic datasets

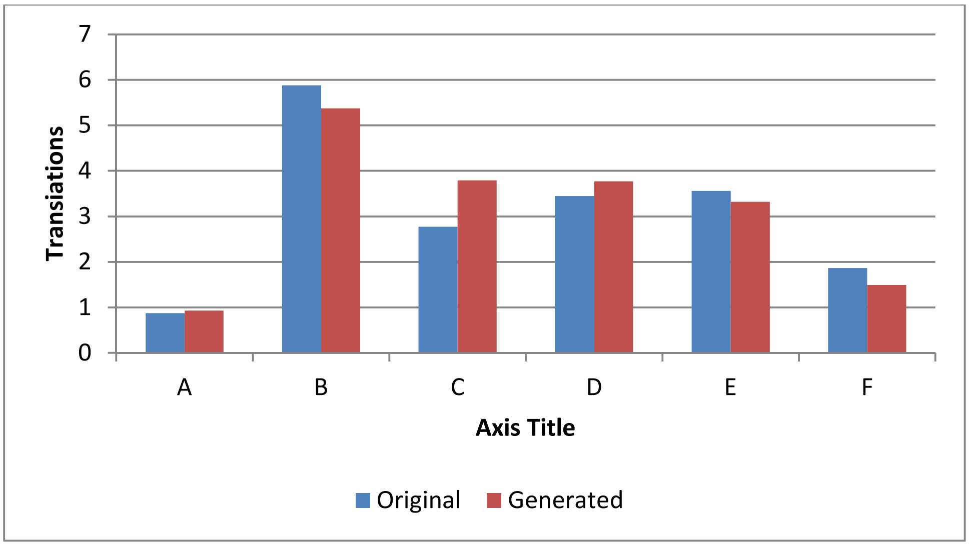


Figure 3-16 – Average location changes per day for each occupant in the original and synthetic datasets

Figure 3-16 shows the average number of changes in location per day for each occupant. Combined with the previous comparison which shows that the synthetic occupants remain in each location for the correct amount of time, this graph shows that they also change location with the correct frequency, providing an accurate model of the original occupants.

	Orig	*Gen 0*	*Gen 1*	*Gen 2*	*Gen 3*	*Gen 4*	*Gen 5*	*Gen 6*	*Gen 7*
NT	83%	82%	83%	82%	83%	82%	82%	82%	82%
ND	70%	70%	70%	70%	70%	70%	70%	70%	70%

Table 3-10 – Percentage of correct predictions, predicting every occupant's exact location in every timeslot, predicting location in the next timeslot (NT) and predicting their locations on the next day (ND) for the original and 8 generations of the synthetic dataset

Since the purpose of the generated dataset is to be the basis of an evaluation dataset for the predictor which will be presented in Chapter 4, a comparison of the accuracy of the predictor on the original and generated datasets is also provided. This cannot tell us anything in particular about the similarity of the dataset, but will highlight if there are any thus far unidentified issues with the model which impact its output's predictability compared to the original dataset.

The predictor was trained on the entire dataset preceding the segment used as the basic for the model. The rules resulting from this training were then used to predict both the original data the model was based on, and the data generated by the model. Table 3-10 shows the overall accuracy of the predictions of each occupant's location in the next timeslot (NT) and their location throughout the day the next day (next day). The test includes eight runs of the synthetic dataset generator to test the level of variance in the resulting dataset. The results show that the generated data has almost exactly the same level of predictability as the original dataset, and that the predictability varies only slightly when the dataset is regenerated.

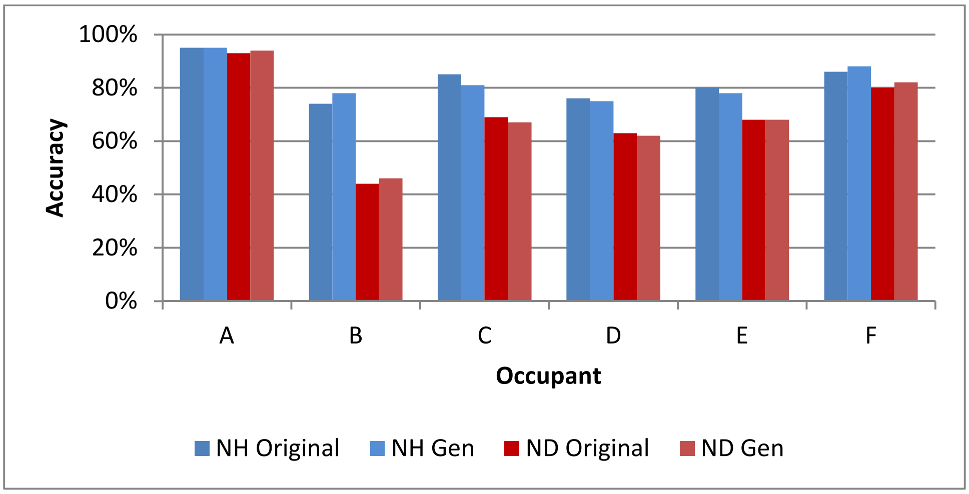


Figure 3-17 – Prediction accuracy from Table 3-10 broken down by occupant for original dataset and synthetic dataset 'gen 0'

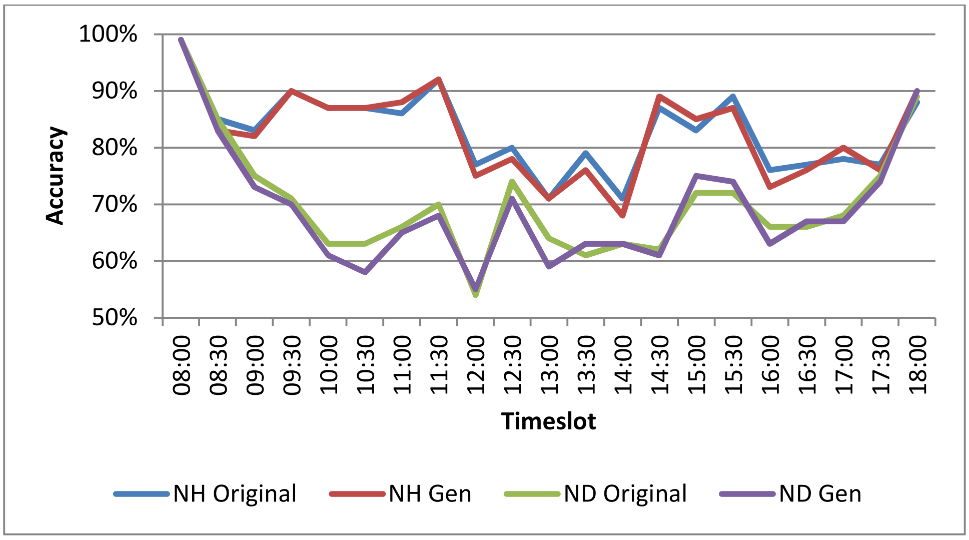


Figure 3-18 – Prediction accuracy from Table 3-10 broken down by timeslot for original dataset and synthetic dataset 'gen 0'

Figure 3-17 and Figure 3-18 break down the accuracy of the predictions on the original dataset and the generated dataset 'Gen 0' by occupant and timeslot. Along with the overall accuracy being the same, these figures show that the accuracy levels are similar across all occupants and timeslots as well.

3.7. Summary

This chapter discussed the two collected datasets used to evaluate OccApriori. The internally collected UCC dataset contains data of a similar type to the existing Augsburg dataset, but improves upon it in terms of number of the occupants, duration of data collection and in the inclusion of timetable data.

The chapter also presented a synthetic dataset generator based on existing occupant modelling approaches. It detailed the process of modelling the occupants in the collected dataset, and performed a detailed evaluation to show that the synthetic datasets generated from that model closely match the original data on a variety of metrics, as well as being equally predictable. This model provides a base onto which specific test cases can be built to test particular features of OccApriori.

4. Occupant Location Prediction using Association Rules

Chapter 2 discussed the process of association rule mining, which involves mining frequent itemsets of ascending size from the dataset, then generating rules from the itemsets. As association rule mining is an unsupervised process it can be applied to occupancy data without any modification, although the occupancy data must be carefully structured so that the desired types of patterns will be found. The aim in this thesis is to find patterns which relate to specific time periods, in order to predict where occupants will be at those times in the future. To this end the dataset is arranged such that an occupant’s movements for a day becomes an instance, with the time periods and the occupant’s location at those time periods being the main attributes and values respectively. The occupant and the day that each instance applies to are present as values in the instance to allow patterns to be related to these variables. With this type of dataset structure the unmodified algorithm can then find patterns which repeat across multiple days, similar to how in the traditional usage it finds patterns which repeat across multiple market baskets. However, the patterns required for occupancy prediction are not the same patterns which the algorithm was designed to find in market baskets. The original application is interested in patterns which apply across all transactions, whereas for occupant prediction it is necessary to predict every occupant individually, so while overall patterns which apply to all occupants are useful, more specific patterns which can predict specific occupants are required. Unmodified, the algorithm will mainly or exclusively find the former type of pattern, resulting in low prediction accuracy. OccApriori consists of a set of components which work together to focus the algorithm on the types of patterns required for occupant prediction. These will be summarised briefly before each is discussed in detail.

The components of OccApriori are the dataset structure, iterative support reduction, support modification, windowing, candidate pruning, rule pruning and finally rule selection. As already mentioned, the dataset is structured specifically to allow the required types of patterns to be found. Iterative support reduction addresses the issue of choosing a support threshold; as each occupant may have different patterns of movement, different support thresholds are needed to reach the useful patterns while avoiding a combinatorial explosion. Support modification involves changing the definition of the support metric as the occupant specific patterns required for occupant specific predictions can have extremely low support, as the occupant's data as a whole has limited frequency within the dataset. The windowing component targets short time windows in the data with a lower support threshold in order to find patterns which are distinct but rare, and therefore low-support. Candidate and rule pruning change the generation of itemset candidates and association rules to avoid generating itemsets and rules which are not useful for occupant prediction. Finally, rule selection is simply the process of choosing a rule to be the output when a prediction is requested.

Sections 4.1 through 4.6 discuss in detail the main components of OccApriori and how they assist the algorithm in finding occupant location patterns. Following this OccApriori's performance is tested on the internally collected UCC dataset and the existing Augsburg dataset, evaluating how the algorithm performs at different times, on different occupants, and with different data available. These results are compared against three existing approaches on both datasets. Finally the contribution of each component of OccApriori to increasing prediction accuracy is examined.

4.1. Dataset Structure

In order to apply association rule mining to occupancy data, the dataset must be suitably structured. The aim of OccApriori to predict occupants' locations at times throughout the day based on their historical locations at those times, and so each instance in the dataset is defined to be a single day for a single occupant. Since the time at which each location appears in the day is required for predictions, and not merely which locations appear in the same day, the items in the instances are attribute/value pairs, with the initial attributes and values being times and locations respectively. The basic set of attributes also includes the occupant's identity, the day of the week, and the occupant's scheduled location during the day if they have timetable data or similar available.

The occupant and day are direct attributes whose values simply state which occupant and day the instance relates to, allowing the algorithm to find patterns relating to specific occupants and/or days. For the location data, actual and scheduled, the day consists of a set of equal-length timeslots. For each timeslot there is an attribute for actual location during that period and another for scheduled location during that period, with the values stating the location. Thus the full set of basic attributes in the dataset is:

$$A = \{d, o, l_i, \dots, l_j, s_i, \dots, s_j\} \tag{4-1}$$

Here d is the day, o is the occupant, l_n is the occupant's location at timeslot n, and s_n is the location they were scheduled to be in at timeslot n. Of these attributes, $l_i \dots l_j$ are the ones which the algorithm aims to predict. 'Away' is treated as an actual location which is recorded when the occupant is not in the building. The value for any attribute can be empty, however if absences are not explicitly recorded in the data then any patterns involving absences, such as departing the building at the end of the work day, will not be found. Table 4-1 below shows a pair of sample instances of this form, with the attributes listed in the table header.

Occupant	*Day*	*12:00*	*13:00*	*12:00 (S)*	*13:00 (S)*
Bob	Monday	Bob Office	Canteen	Bob Office	-
Bill	Monday	Bill Office	Away	-	-

Table 4-1 – Example instances using OccApriori's dataset structure

While the items in the traditional market basket dataset are all simple strings which are either present or absent, the attribute/value pairs in an occupant location dataset have some additional properties which must be considered. The timeslot attributes $l_i \dots l_j$ represent an ordered list, the ordering being the time at which they occur. This ordering is important as only predictions of later times will be useful. The attributes $s_i \dots s_j$ are also ordered, however this ordering is not important. This is because predicting some l_n using data for some l_m where $m > n$ means the time n must have passed already since there is actual location data for time m, thus making a prediction of l_n unnecessary, whereas because schedule data is by definition available ahead of time, any slot in $s_i \dots s_j$ can be used to predict any slot in $l_i \dots l_j$.

The non-timeslot attributes are special in that they are actually data about the location data, i.e. meta-data. This has consequences for the frequency of the possible values as while a timeslot can, in principle, take on any location value, each day of the week occurs with fixed frequency, and each occupant's data represents a fixed fraction of the total dataset. This in turn changes the support threshold which must be used to delineate frequent itemsets. Taking the example of the UCC dataset with 5 days and 6 occupants, assuming an even distribution of days and occupants across the instances, an itemset which includes a value for occupant and day will have a maximum support of ~0.03, which is extremely low. This will be addressed in section 4.4.

As association rule mining is an unsupervised approach designed to find any patterns present in a dataset, any of the attributes listed above can be absent and the algorithm will still function. An obvious example is that the schedule data may be incomplete or empty simply because occupants may not have any scheduled activities, and the evaluation includes examining the effect of removing the timetable data where it is available. Instances missing actual location data or even the occupant and day labels can also be handled, but will obviously be of limited use in terms of making predictions. New attributes can also be freely added to the dataset, and will be picked up by the algorithm without further intervention. Examples of attributes which could be added include month, date, academic term, weather data, etc. As with the basic attributes it must be considered whether a new attribute is a meta-data attribute or not. Table 4-2 shows the example dataset from Table 4-1 with the Month and Weather attributes added (and the schedule attributes remove due to space limitations).

Occupant	*Day*	*Month*	*Weather*	*12:00*	*13:00*
Bob	Monday	July	Rain	Bob Office	Canteen
Bill	Monday	July	Rain	Bill Office	Away

Table 4-2 – Example instances using OccApriori's dataset structure with two new attributes added – Month and Weather

4.2. Iterative Support Reduction

In choosing a support threshold for frequent itemset generation, the two concerns are eliminating itemsets which represent patterns which are unreliable due to their infrequency, and avoiding the combinatorial explosion which occurs when trying to enumerate every itemset that appears in the dataset. In occupancy data the threshold at which itemsets become unreliable or difficult to enumerate depends on the occupant's patterns. To illustrate this we can look at two hypothetical occupants who represent the extreme of pattern frequency.

Occupant	*09:00*	*10:00*	*11:00*	*12:00*	*13:00*
Occupant 1	A	A	A	A	A
Occupant 1	A	A	A	A	A
Occupant 2	B	C	D	E	F
Occupant 2	G	H	I	J	K

Table 4-3 – Example instances for two occupants, one of whom is always in location A, the other going to a brand new location in every time slot

In this example, Occupant 1 is in the same location in every timeslot in every instance, whereas Occupant 2 is always in a new unique location. Due to the

Occupant 1 staying in the same location, every itemset which includes the 'Occupant: Occupant 1' item will have the same support, specifically 0.5. Above this support threshold we will find no frequent itemsets for Occupant 1, below it we will find every itemset which occurs. Contrast this with Occupant 2, whose itemsets will all have the minimum possible support of $1/|D|$, in this case 25%. Again, above this threshold we find no itemsets for this occupant, and below we enumerate them all.

These occupants represent the extreme edge cases, however any occupant is going to fall somewhere between completely consistent and completely inconsistent movements. The degree of their consistency determines at what support threshold the combinatorial explosion will occur. Thus, the algorithm may need to use different support thresholds for different occupants in order to efficiently find useful itemsets. To address this OccApriori replaces the fixed support threshold with a minimum support threshold, and iteratively drops the support threshold from a higher value during candidate generation as follows:

```
$for \begin{pmatrix} csupp = startsupp; \\ csupp \geq minsupp; \\ csupp -= step \end{pmatrix}:$
  $C_1 = U$
  $F_1 = \{X \in C_1 | supp(X) > csupp\}$
  $for(i = 2;\ F_{i-1} \neq \emptyset;\ i++):$
    $C_i = \{A \cup B | A, B \in F_{i-1}, |A \cup B| = i\}$
    $F_i = F_i \cup \{X \in C_i | supp(X) > csupp\}$
    $if\ timeLimiter()\ or\ memoryLimit(C_i)$
      $return \bigcup_i F_i$
  $return \bigcup_i F_i$
```

Figure 4-1 – Apriori Algorithm with Iterative Support Reduction

The algorithm starts by finding frequent itemsets with a high support threshold. If the process finishes, i.e. it has enumerated all itemsets this frequent, it reduces the support threshold and starts finding frequent itemsets again. New frequent itemsets that are found are added to those found in previous iterations. The algorithm has configurable time and memory limits which are checked when it steps up the length of itemsets it is searching for. If either of the limits has been violated, the process finishes early. If the limits are never violated, the process ends with a final iteration with the support threshold equal to the minimum support threshold. Figure 4-1 shows the algorithm with iterative support reduction in place, which is the main change to the flow of the algorithm; the following components alter the criteria under which candidate itemsets or rules are considered in this algorithm.

This approach allows the algorithm to set a general minimum support threshold across all occupants based on the minimum frequency the user considers useful, while avoiding getting caught in a combinatorial explosion. While the example given is occupants, this issue and solution apply to all metadata attribute combinations, for example days and the occupant/day combination. A downside of reducing the support iteratively is that, due to the process of generating candidates of length n process requiring all frequent itemsets of length $n-1$, each iteration regenerates the frequent itemsets already found in previous iterations. This makes the process take longer than if the final support threshold was instead set as a fixed threshold, avoiding the iteration.

4.3. Candidate and Rule Pruning

Not all associations in the data will be useful for occupant location prediction. The data may contain associations that do not relate to occupant locations at all, i.e. itemsets which do not contain any of $l_i \dots l_j$, and may produce rules which predict the past, i.e. predicting l_n using l_{n+m} for some $m > 0$. As association rule mining is designed to find all associations within the support and confidence limits, by default it will find these associations along with the useful ones. To address this OccApriori alters both candidate and rule generation.

In candidate generation the algorithm discards any itemset of length 2 upwards which does not contain a timeslot attribute. The algorithm should predict only occupant locations, so any itemset lacking a timeslot item cannot produce a rule it will ever use, thus it aims to remove them as early in the process as possible. The initial generation of 1-itemsets is performed as normal because removing the itemsets lacking a timeslot attribute at this point would be equivalent to simply removing the non-timeslot data from the dataset. The no-timeslot itemsets of length 2 can be removed because for any no-timeslot 2-itemset (x, y), if this itemset will ever be a part of a frequent itemset containing a timeslot (x, y, l_n) then the 2-itemsets (x, l_n) and (y, l_n) must also be frequent. (x, l_n) and (y, l_n) will not be removed due to containing timeslot attributes and will generate the (x, y, l_n) itemset in a later iteration of candidate generation. Thus, removing non-timeslot itemsets of length 2 upwards does not remove any itemsets the algorithm is intended to find. This modification to candidate generation makes candidate generation more efficient, potentially allowing for a lower support threshold for any occupants whose non-timeslot data (e.g. timetable data) caused a combinatorial explosion, and prevents the generation of the useless rules non-timeslot itemsets would have resulted in.

In rule generation, from each itemset OccApriori only generates the rule whose consequent is the final timeslot item in the itemset, i.e. for any itemset I it only generates the rule $I\backslash\{l_n\}$ => $\{l_n\}$, where $l \subset I$ is the set of timeslot attributes in I and n is the latest time which appears I. As before, the aim is to predict occupant locations, so only rules which predict a location are required. Furthermore, predicting a past location based on a later location is not useful in practice. Finally, in order to be able to meaningfully compare the confidences of the rules when predicting, the rules should predict only a single timeslot at a time. So the only rules from any itemset I that are useful are the rules $J\backslash\{l_n\}$ => (l_n) where J is any subset of I such that $|J| > 0$ and n is the final timeslot in J. However if I is frequent, then any itemset J is also frequent, so to get the full set of rules required the algorithm needs only generate the single rule $I\backslash\{l_n\}$ => (l_n) from every frequent itemset. This increases the efficiency of rule generation and rule selection during prediction by reducing the number of rules.

4.4. Support Modification

As discussed in section 4.1, occupancy datasets include a number of attributes which are actually metadata, and so cannot ever reach the maximum support value of 1. This means that these itemsets are less likely or impossible to appear except with very low support thresholds. Part of the purpose of the support threshold is to eliminate itemsets whose frequency is so low that they will result in unreliable rules, and this applies to occupancy datasets as well. However, the itemsets which include metadata attributes are also of more interest at lower support values than 'normal' itemsets. Going back to the example in Table 4-3, consider the itemsets {10:00: A, 11:00: A} and {Occupant: Occupant 1, 10:00: A, 11:00: A}. Both of these itemsets have a support of 0.5 as they both occur in two of the four example instances. However one of these is a general pattern which does indeed occur half of the time, whereas the other is a pattern which happens in every instance which refers to Occupant 1; but the fact that Occupant 1 only appears in half the instances in the dataset results in the same support of 0.5. Since Occupant 1 follows this pattern 100% of the time, it would be more accurate the give it a support value of 1. The lower support becomes an issue if our support threshold is, for example, 0.6 as this pattern which Occupant 1 follows every day will now be discarded along with the less reliable general pattern. We may wish to discard the latter, but we must keep the former if we are to allow any occupant-specific patterns in our generated rules. To this end, OccApriori redefines support as:

$$supp(X) = freq(X)/max(freq(X)) \qquad (4\text{-}2)$$

This means that rather than measuring the frequency of an item against the size of the dataset, it is measured against the maximum frequency the itemset could possibly have. In principle any timeslot items could appear in all instances, whereas the frequency of day or occupant items is subject to how much data is in the dataset for each day or occupant. Thus, for itemsets which consist only of normal items the support value does not change, as their maximum frequency is equal to the size of the dataset. For itemsets which include metadata items, the support is scaled such that the frequency of the metadata combination itself gives a support value of 1. Going back to the example above, the timeslot-only itemset would keep its support value of 0.05, whereas the occupant-specific itemset would have a support of 1, as the timeslot values appear every time the metadata attribute appears. By setting a support threshold of for example 0.5, we can include patterns which appear 50%+ of the time overall, or which hold true for Bob in 50%+ of his instances, or hold true for Bob on Tuesdays in 50%+ of those instances.

Before incorporating this redefined support metric into the algorithm its effect on the anti-monotonicity property, which Apriori relies on, must be considered. The anti-monotonicity property means that every subset of a frequent itemset must also be frequent, i.e. that the support of a subset of a set must be at least as high as the support of the set. However, if we remove a metadata item from an itemset, we can get a support value which is lower, which is demonstrated by the example itemsets above. Thus if we simply modify the support definition, the algorithm will discard itemsets which would later form part of a frequent itemset when combined with metadata attributes in candidate generation.

Since this issue arises when metadata attributes are added to items, it can be avoided simply by initialising candidate generation with all the possible metadata itemsets. For every combination of metadata items C, OccApriori initializes candidate generation with all itemsets of size $|C|+1$ which are a superset of C. Within each metadata combination the anti-monotonicity property is preserved, so candidate generation can continue as normal from this point. This also means that the algorithm can do a separate pass of frequent itemset mining for every metadata combination, which it does in order to reduce memory requirements, as this redefinition of support greatly increases the area of the itemset lattice which is explored for any given support threshold.

4.5. Windowing

Some patterns that the algorithm needs to find have such low support that reducing the support threshold low enough to find them is not feasible, due both to the combinatorial explosion and including unreliable low-support patterns the algorithm should not generate rules from. Table 4-4 shows the example of an all-day absence, represented by the sequence of location A's.

Occupant	*09:00*	*10:00*	*11:00*	*12:00*	*13:00*
Occupant 1	A	A	A	A	A
Occupant 1	B	C	D	E	F
Occupant 2	G	H	I	J	K
Occupant 2	L	M	N	O	P

Table 4-4 – Example instances in which two occupants go to a brand new location in every time slot, except for one instance where Occupant 1 is absent all day

This is quite an obvious type of pattern as the locations are all identical, however if occupants are absent all day infrequently enough, its support will be too low for it to be found. In this example the support of the absence instance is 0.25; if our minimum support cannot be lower than 0.3 this pattern will be discarded, and the predictor will have no rules with which to deal with future absences. The next time the predictor is presented a pattern such as {Occupant: Occupant 1, 09:00: A, 10:00: A, 11:00: A} it will not know to predict {12:00: A}, despite the pattern being so distinct and obvious.

To deal with this OccApriori takes advantage of the temporal relationship of the timeslot items in the dataset, a feature not found in the original market basket application. Existing work on occupant location prediction shows that the occupants' most recent movements are a reliable predictor of their next location. Thus for itemsets whose timeslot items occur over a short time period the algorithm can use a lower support threshold without generating unreliable rules. By targeting only itemsets with close timeslot items, it also avoids the combinatorial explosion which would occur the overall support threshold was lowered.

To do this it mines subsets of the location attributes which form consecutive temporal windows over the location data. A window is defined as:

$$win(n,m) =< d, o, l_{n \dots n+m}, s_{i \dots j} > \qquad (4\text{-}3)$$

Here 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 the algorithm searches within every window of the chosen length, i.e. for every n such that $n \geq i$ and $n + m \leq j$. In candidate generation for each window, it does not include any timeslot items which fall outside the current window in the initial itemsets. With the extraneous items excluded, frequent itemset mining and rule generation can continue as normal.

Windowing results in rules which describe low-frequency short-duration patterns, going back to the example of an all-day absence a possible rule is {9:00=Away} => {10:00=Away}. No single itemset or rule found by windowing can describe a long pattern such as every timeslot item having the value ‘Away’ due to the window constraint applied, however in combination a set of shorter rules can describe a longer pattern. For example a set of rules {n=Away}=>{n+1=Away}, where n is the timeslot, can be found to describe an all-day absence, allowing the predictor to recognise the pattern throughout the day and correctly predict that the occupant will remain absent.

4.6. Rule Selection

To predict occupants’ future locations OccApriori includes a mechanism to choose the appropriate rules. When a prediction is required, values for any subset of the dataset’s attributes can be supplied in the forum of an itemset V. The target for the prediction l_t is also given. The algorithm searches the generated rules for all rules $X => Y$ where $X \subseteq V$ and $Y = \{l_t\}$; any rule which matches the data available to make the prediction which predicts also the desired time slot. From these it selects the rule with the highest confidence to make the prediction. A number of other metrics were evaluated for rule selection; however none provided a consistent benefit over confidence across all predictions.

4.7. Evaluation of Performance

In all of the following evaluations the exact location of every occupant in every timeslot is predicted. Accuracy is defined as the percentage of predictions made which were correct. Failure to make a prediction is considered an incorrect prediction. Several types of prediction are evaluated, which allow different attributes from the set $\{d, o, l_i \dots l_j, s_i \dots s_j\}$ to be used when making the prediction. Next-Slot predictions predict an occupant's next location, they are based on any available data up to and including the previous timeslot, i.e. $\{d, o, l_i \dots l_{n-1}, s_i \dots s_j\}$ where n is the current timeslot. Next-Day prediction is a general term for any prediction which doesn't use recent location data, instead being based on historical location frequency. For these predictions $l_i \dots l_j$ are unavailable as the prediction is made before even time i has passed. A prediction that does not use $l_i \dots l_j$ could be made an arbitrary number of days into the future, or could make predictions for later on the same day without recent locations available (e.g. a tracking system whose data is not accessible in real time), but the intended use case for this type of prediction is providing data to overnight heating systems, so the evaluation considers next day predictions in particular. Both of these types of predictions can then be made with no timetable data (marked as 'No TT'), in which $s_i \dots s_j$ are not used. In practice this would only be done if the timetable data were unavailable, but it is included here in order to evaluate the difference the timetable data makes.

The evaluation uses two datasets, the UCC dataset and the Augsburg dataset. The UCC dataset contains 1524 instances distributed across 6 occupants and includes timetable data. The Augsburg dataset contains 109 instances distributed across 4 occupants, and does not feature timetable data.

In evaluating the performance of the complete algorithm the minimum support and confidence are set to 0.2 and 0.5 respectively. The initial support value before iterative reduction is 0.8, and windowing is set to a 0.05 support threshold and a window length of 3.

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

Table 4-5 – OccApriori's accuracy for different prediction types on the UCC dataset, predicting all occupants at all time slots

Table 4-5 shows the accuracy of the complete OccApriori 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 the existing 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 rely solely on the occupants' historical movements without any knowledge of their movements during the day. However, for these predictions, access to the timetable data does improve the accuracy.

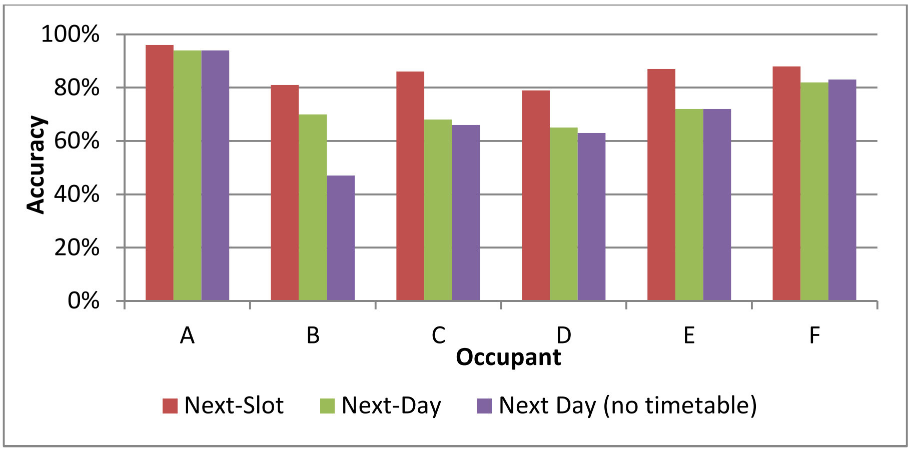


Figure 4-2 – OccApriori's accuracy for different prediction types on the UCC dataset, across all times, broken down by occupant

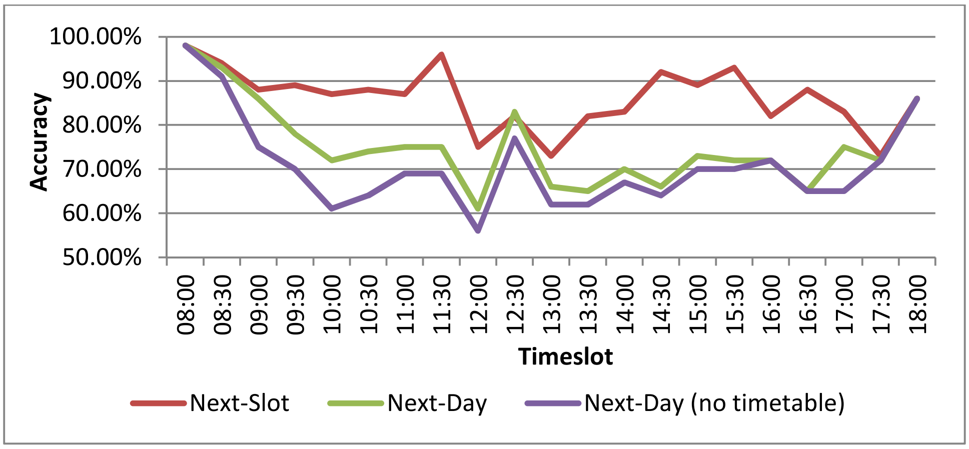


Figure 4-3 – OccApriori's accuracy for different prediction types on the UCC dataset, across all occupants, broken down by time

Figure 4-2 breaks the Next-Slot, Next-Day and Next-Day with no timetable accuracies down by occupant. Occupant A has the most homogenous movements of all the occupants, and so their movements are easily predicted in all configurations. Occupant B has the most varied movements, and those movements are largely determined by their timetable. This makes them the hardest to make accurate Next-Day predictions for, however the ability to use the timetable data in predictions increases the accuracy significantly. For the other occupants, whose movements follow general time-dependent 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 predictions and a slightly lower one for Next-Day predictions.

Figure 4-3 breaks the prediction accuracies down by timeslot, averaged over all occupants, showing how they perform at different times of day. In general the accuracy levels match at the beginning and end of the day and around lunch, as these are the times at which the occupants are most predictable. Outside those times the Next-Day prediction accuracy drops 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. At 17:00 and especially 17:30 Next-Slot accuracy drops to Next-Day levels, before both types of prediction accuracy rises at 18:00. The exact departure time of most of the occupants in the dataset is somewhat random, which means that the additional same-day data used when making Next-Slot predictions doesn't help in determining whether they will be in the office or not at 17:30 – hence the accuracy drops to Next-Day levels. These occupants are much more likely to be out of the office past 18:00, thus the accuracy rises for all prediction types in this timeslot.

	Occ. B	*Office*	*Meeting*	*Building*	*Away*
Actual Location	Office	304	0	1	66
	Meeting	11	19	0	2
	Building	21	0	145	7
	Away	37	1	0	214

Table 4-6 – Confusion matrix for the UCC dataset's Occupant B when making Next-Slot predictions

	Occ. E	Office	Meeting	Building	Away
Actual Location	Office	148	1	3	65
	Meeting	1	6	0	0
	Building	3	0	3	0
	Away	29	0	0	581

Table 4-7 – Confusion matrix for the UCC dataset's Occupant E when making Next-Slot predictions

Table 4-6 and Table 4-7 show the confusion matrices for occupants B and E respectively, classifying the location as in their own office (Office), in one specific group meeting which all these occupants attend (Meeting), any other room in the building (Building) or away (Away). All occupants except B show the same pattern in their matrices as E. The matrices show that the primary source of errors is reversing Office and Away, which is caused by being uncertain whether the occupant will actually be in. In the case of Occupant B, who has the most varied movements, there are also instances of failing to predict that the occupant would leave their office for the group meeting or events elsewhere in the building, due to lower certainty about their movements. For the group meeting and events elsewhere in the building, the occupants tend to be predicted to be in the correct location or in their office; recognizing that the occupant is in, but not always that they will be leaving their office.

Following is the evaluation on the Augsburg dataset:

Seasonal	*Fall*	*Max Training*
56%	63%	68%

Table 4-8 – OccApriori's accuracy for Next-Slot predictions on three different training/test splits of the Augsburg dataset, predicting all occupants in all timeslots

Table 4-8 shows the accuracy on three evaluations on the Augsburg Dataset. The 'Seasonal' evaluation trains on the summer data and tests on the fall data. In this split of the dataset, there is a maximum of two weeks of training data for each occupant, with over a month of data in the test set. The 'Fall' evaluation splits the fall data only, such that the final week is used as the test set for each occupant, leaving approximately a month as the training set. Finally the 'Max Training' evaluation splits the entire dataset such that the final week of the fall data is used as the test set for each occupant, with the remainder of the dataset used for training.

The 'Seasonal' split of the data was used as an initial comparison to the evaluations performed in (Petzold et al. 2006), and shows that OccApriori has difficulty on this dataset, failing to provide the level of accuracy it achieves on the UCC dataset. Two possible causes of this low accuracy are the small size of the training set, and the fact that the training and test sets are from two different times of year. The two extra splits of the data allow testing of the effect of these factors on OccApriori's accuracy.

Training and testing on a split of the fall data only, in the 'Fall' evaluation, does show an increase in accuracy. However, training on both the summer data and a split of the fall data, in the 'Max Training' evaluation, shows an even greater increase in accuracy. Since using the summer data allows the algorithm to predict movements in fall with greater accuracy, the fact that some data is from summer and some is from fall does not appear to be an issue. This suggests that the gain in accuracy is due to the increased size of the training set, showing that OccApriori requires more training data than the original split provides. However, even with more training data, the accuracy is still low.

This low accuracy appears to be due to the movement patterns of the occupants in the dataset. While the occupants adhere to regular patterns, the patterns do not relate to specific times, i.e. the occupants visit rooms in particular sequences, but the sequences can occur at varying times of day. OccApriori cannot learn a sequence independent of the time it occurs, and thus has difficulty with these occupants.

Day	*Start*	*Finish*
Tuesday	10:00	12:30
Thursday	14:30	15:00
Tuesday	9:00	11:30
Thursday	13:00	15:00
Tuesday	15:00	15:30
Monday	16:00	16:30
Tuesday	9:30	10:00
Tuesday	14:00	16:00

Table 4-9 – The days and times that occupant B visits room 409 in the Augsburg dataset

Table 4-9 shows an example of one of these patterns in the dataset. Occupant B occasionally visits room 409, and these visits are either short visits which last half an hour, or long visits which last 2-2.5 hours. It should therefore be possible to predict that the occupant will remain in room 409 for at least another hour if they have been there for an hour. However, these visits all occur at different times, and so OccApriori has no patterns it can call on to make the correct predictions.

4.8. Comparison to Existing Approaches

This section compares OccApriori to three existing approaches which have previously been evaluated on the Augsburg dataset: a Hidden Markov Model (Gellert et al. 2006), an Elman net and a frequency predictor, the latter both evaluated in (Petzold et al. 2006). The HMM and Elman net were evaluated using the respective tools in MATLAB, while the frequency predictor was implemented based on the frequency analysis context predictor in (Petzold et al. 2004). The evaluation compares on Next-Slot accuracy, as these approaches only predict occupants' next locations.

A variety of settings were tested for each approach in order to find the one which achieved the highest accuracy. The frequency predictor uses a maximum order of 2 as shorter sequences proved to be more reliable predictors. The HMM uses 7-8 hidden states depending on the dataset. 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; after each prediction the frequencies are updated based on the correct answer. In this evaluation the Elman net and the frequency predictor were allowed to retrain after each whole day of predictions, with the predicted day added to the training set, in order to maximize their accuracy. The HMM and OccApriori predict with only the initial training run, as the time taken to train makes retraining after each day unfeasible during evaluation for these approaches.

HMM	*Elman Net*	*Freq. Predictor*	*Mod. Apriori*
77%	86%	87%	86%

Table 4-10 – All approaches accuracy for Next-Slot predictions on the UCC dataset, predicting all occupants at all time slots

Table 4-10 shows the accuracy of each approach for next-slot predictions on the UCC dataset. The HMM performs worse than the others due to being unable to retrain. The other three approaches achieve approximately the same accuracy, including ours despite it also being unable to retrain.

HMM	*Elman Net*	*Freq. Predictor*	*Mod. Apriori*
74%	76%	77%	56%

Table 4-11 – All approaches accuracy for Next-Slot predictions on the Augsburg dataset, predicting all occupants, at all time slots

Table 4-11 shows the accuracy of each approach for next-slot predictions on the Augsburg dataset. As discussed previously, OccApriori performs poorly on this dataset due to the type of movement patterns the occupants follow. As the other approaches are sequence based and do not take into account the time at which movements occur, they are easily able to learn and recognise the movement patterns of these occupants and thus perform much better. Going back to our example pattern, once the sequence-based approaches see any instance of occupant B visiting room 409 for at least 2 hours, they can predict that the occupant will remain that long regardless of when the visit begins. The accuracies for all approaches are lower on the Augsburg set due to the greater unpredictability of the occupants. On the whole the 4C occupants change location less and go to fewer different locations, leading to them being easier to predict in general for all approaches. The next chapter addresses extending OccApriori to be able to learn these types of patterns in order to improve its accuracy on these types of occupants.

4.9. Evaluation of Algorithm Components

This section examines the contributions of OccApriori's components to its prediction accuracy advantage over Apriori. The evaluation on the UCC dataset is repeated in several configurations, with the types of predictions being made and the information available to the predictor are as before. Not all components of OccApriori directly affect accuracy; it is windowing and support modification specifically which expand the area of the itemset lattice which is searched, and so the change in accuracy is evaluated via the enabling and disabling of these two components.

Configuration	*Next-Slot*	*Next-Day*	*Next-Day No Timetable*
Apriori (Baseline)	61%	61%	61%
OccApriori - Windowing	76%	61%	61%
OccApriori - SupportMod	80%	73%	71%
OccApriori - Windowing + SupportMod	86%	75%	71%

Table 4-12 – Next-Slot, Next-Day and Next-Day with no timetable prediction accuracy on the UCC dataset for unmodified Apriori, and OccApriori's accuracy-affecting components

Table 4-12 shows the overall accuracy of each component of the algorithm. The unmodified Apriori algorithm is included as a baseline. Without the accuracy-affecting Windowing and Support Modification components, OccApriori gets the same accuracy as the unmodified Apriori, as it is performing the same search of the itemset lattice.

What the baseline finds in that search, due to two of the occupants in the dataset primarily working from home, the remaining occupants having periods of extended absence, and the 'Away' location being one of the few which is shared between all occupants, is that 'Away' is the most frequent location in every timeslot. Thus for any given time it predicts this most frequent location. This is essentially the same as a first order Markov model using the time as the context.

Windowing substantially increases Next-Slot accuracy due to finding low support patterns which relate to particular time periods throughout the day. It does not help with the Next-Day predictions, as rules which predict an occupant's location based on their recent locations, which windowing is intended to find, do not apply to Next-Day predictions. The timetable makes no difference with windowing alone, as timetable data applies only to a specific occupant on specific days – too specific to be counted as frequent with windowing alone.

Support modification alone brings the accuracy values close the final values produced by the complete algorithm, as all patterns related to specific occupants, specific days, and specific day/occupant combinations are searched for. Timetable data makes some difference to the Next-Day accuracy here, as individual occupant/day patterns are mined and the timetable is found to be relevant.

Finally the combination of windowing and support modification yields the accuracy values examined in section 4.7. There is overlap in the extra correct predictions made when support modification and windowing are applied independently, so the total increase is not the sum of their individual contributions, however the final accuracy achieved on Next-Slot predictions in particular is significantly higher than either modification achieves alone. More timetable related patterns are found and used than before, as specific occupant/day patterns are mined while paying attention to specific periods of the day.

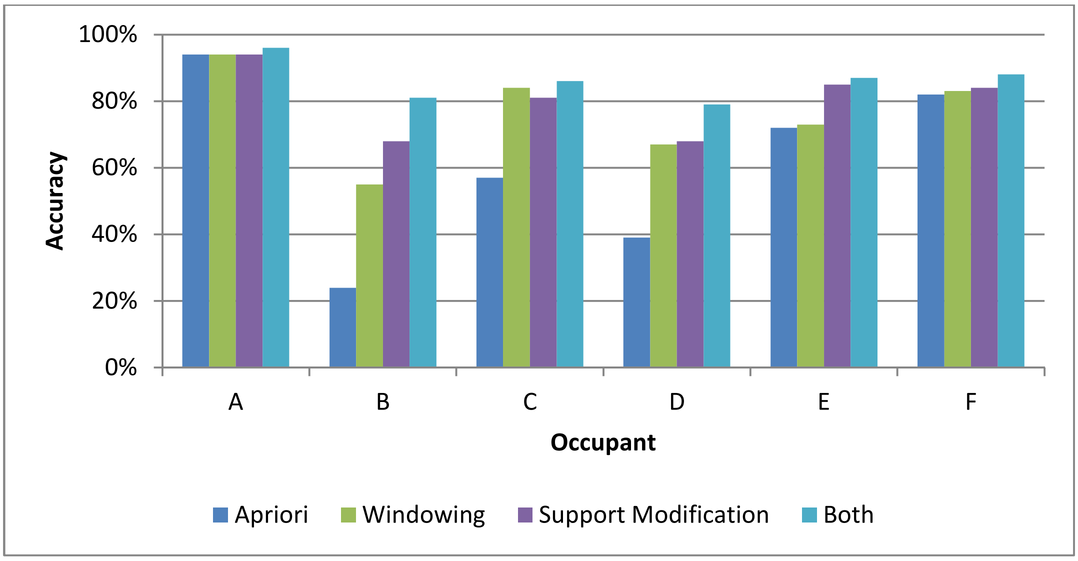


Figure 4-4 – Next-Slot accuracy on the UCC dataset for unmodified Apriori and OccApriori with Windowing, Support Modification, and both, broken down by occupant

Figure 4-4 shows the accuracy of each component making Next-Slot predictions for each occupant in the dataset. Occupants A and F primarily work from home and are absent most of the time, as such Apriori's default prediction of Away produces high accuracy, and there is not much improvement to be made OccApriori. Occupant A shows a slight increase in accuracy with both components enabled despite no gain in accuracy when they are applied individually; this is because windowing and support modification work together to find, for example, patterns which apply to a particular period of time for a particular occupant on a particular day. In the case of occupant A, this allows a couple of percentage points greater accuracy on those occasions when the occupant does come in. Occupant F sees a 6 percentage point increase in accuracy (correcting 33% of errors) going from unmodified Apriori to the OccApriori with both components enabled. While the percentage point increase in accuracy for these occupants is low, and the individual components make little or no improvement on them, the consistent performance of the algorithm even when the search of the itemset lattice is broadened by support modification and windowing shows that it is successfully avoiding the combinatorial explosion which is associated with highly repetitive patterns in the data.

The largest difference is seen in occupant B, who has the most complex movements in the dataset. The default prediction of Away does not perform well as this occupant is in the office most of the time. As there are a significant number of useful patterns to be found, both windowing and support modification provide substantial accuracy increases individually, and a further increase when combined.

Occupants C, D and E show that for different occupants, different features are important for making accurate predictions. For occupant C both windowing and support modification come close to the maximum achieved accuracy, whereas most of the extra accuracy comes from only from support modification for occupant E. For occupant D the combination of features performs significantly better than either individually.

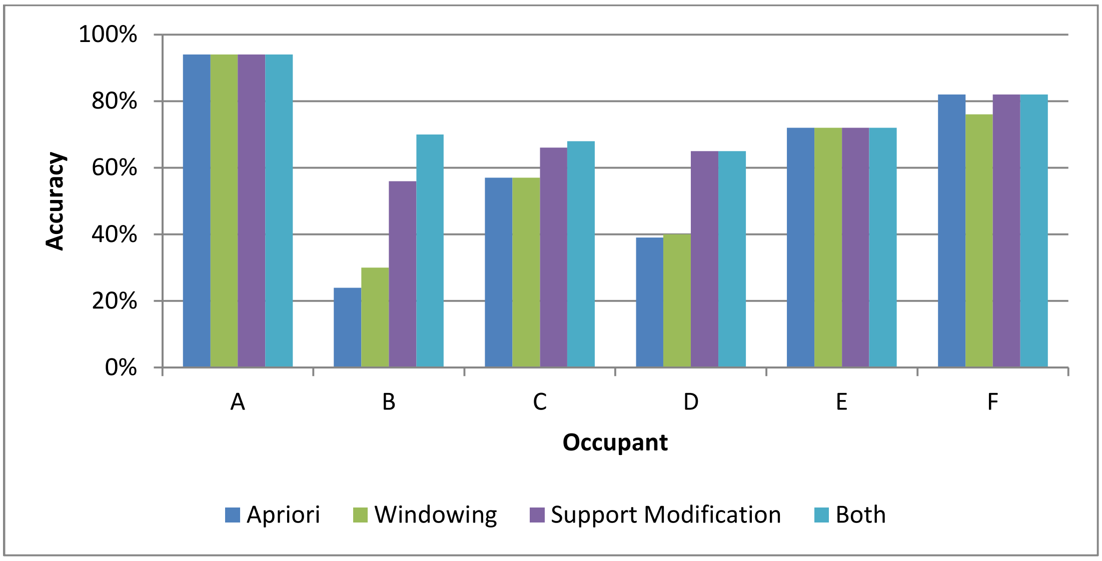


Figure 4-5 – Next-Day accuracy on the UCC dataset for unmodified Apriori and OccApriori with Windowing, Support Modification, and both, broken down by occupant

Figure 4-5 shows the same results for Next-Day predictions. The degree to which the OccApriori improves accuracy on each occupant over Apriori is roughly the same as for Next-Slot predictions, although the gap between the two algorithms is smaller as only a subset of the extra patterns found are useful for Next-Day prediction. Occupants A and F show no improvement over Apriori, as these occupants are absent so often that Apriori's prediction of always absent remains the best possible prediction when predicting a day in advance. Again however, aside from a small anomaly when windowing is used alone on occupant F, the algorithm expands the search of the itemset lattice while avoiding problems associated with the repetitive patterns of these occupants.

Occupant E shows no increase in accuracy, however the predictions made by the OccApriori are different from those made by Apriori – it predicts a pattern of being in the office based on the occupant's most frequent patterns, however due to the occupant's frequent and unpredictable all-day absences, these predictions (in this specific case) end up being almost exactly as accurate as Apriori's always-absent predictions.

In general windowing does not help for the Next-Day predictions because, as mentioned previously, it is designed to pick up patterns useful for shorter term predictions. In the case of occupant F windowing actually hurts accuracy slightly, due to finding less supported patterns which the occupant did not adhere to in the test set. This negative effect is nullified with both components enabled however.

Figure 4-6 and Figure 4-7 show the accuracy of each modification making Next-Slot and Next-Day predictions respectively, broken down by timeslot. Windowing follows the Apriori baseline closely in Next-Day predictions due to not helping prediction accuracy, though the slight deviations show that it does impact the predictions, but it harms as much as it helps.

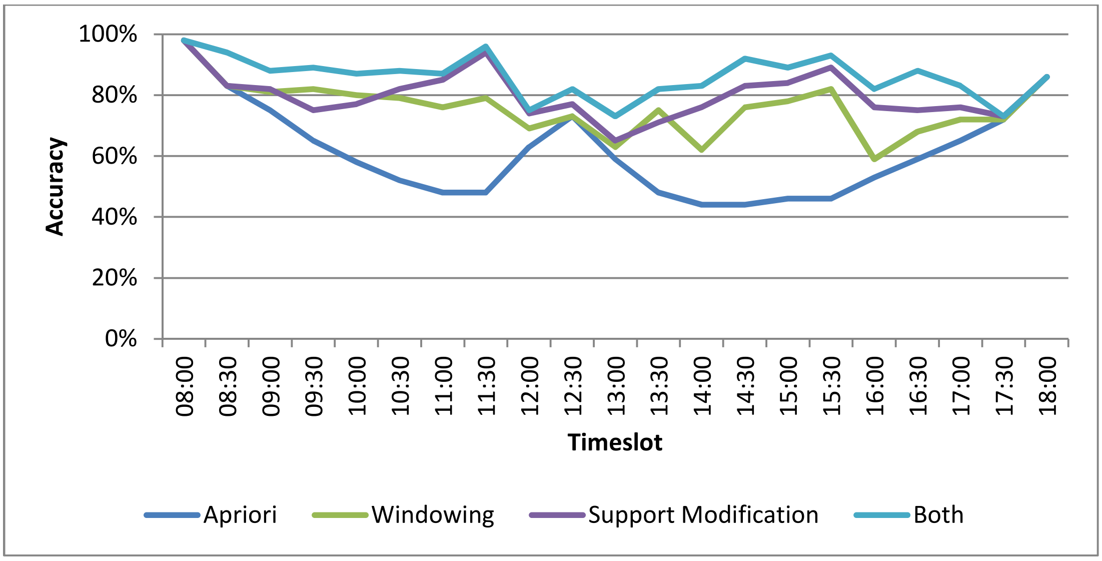


Figure 4-6 - Next-Slot accuracy on the UCC dataset for unmodified Apriori and OccApriori with Windowing, Support Modification, and both, broken down by timeslot

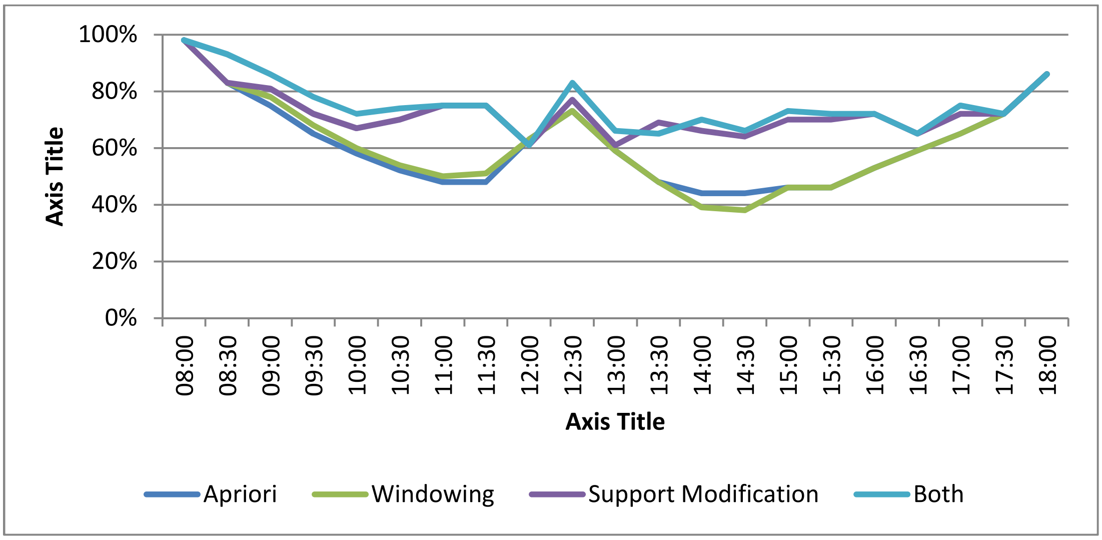


Figure 4-7 - Next-Day accuracy on the UCC dataset for unmodified Apriori and OccApriori with Windowing, Support Modification, and both, broken down by timeslot

For Next-Day predictions both support modification and both components combined manage to maintain a base accuracy of about 70%, with spikes higher at the beginning and end of the day, as well as at lunchtime, as these are the times at which the occupants are most predictable in general. Windowing roughly follows this same baseline when making Next-Slot predictions, with support modification generally doing better and the complete algorithm beating both individual modifications at most times.

4.10. Summary

This chapter presented OccApriori, which adapts association rule mining for occupant location prediction. The first step was to structure the dataset such that the mining process would find the required patterns. By making each instance a single day for a single occupant, with the timeslots as attributes and locations as values, it was possible to start mining rules and predicting even with the unmodified algorithm.

Simply applying the existing algorithm doesn’t yield good accuracy however, as it does not take account of the structure in occupancy data. This chapter identified several key issues in the patterns the algorithm failed to find, as well as useless patterns it did find but should instead ignore. It described solutions to these issues which involved modifying Apriori to suit occupant prediction, and demonstrated that these modifications allow the resulting algorithm OccApriori to find more useful patterns and fewer useless ones, increasing accuracy. While OccApriori uses Apriori as its base, its differences involve modifying the support of or discarding itemsets based on specific criteria, and so the algorithm can potentially use any association rule mining algorithm as its base.

The evaluation shows that OccApriori is very flexible in the predictions it can make, predicting from the occupant’s next location up to predictions of their location across the following day, while making use of ancillary data such as timetable data. It matches existing approaches on the UCC dataset for next-location prediction, while being able to make a wider variety of predictions. It does have difficulty with the occupants in the Augsburg dataset, which will be addressed in the following chapter.

5. Location Sequence Prediction using Association Rules

This chapter discusses the type of movement pattern which proved difficult for OccApriori to predict in the Augsburg dataset. The occupants in the dataset move in patterns that do not strongly relate to the time of day, violating the assumption OccApriori has thus been based on. As sequence predictors have been shown to be successful on this type of pattern, this chapter describes a new type of itemset and rule which can capture location sequences without referring to the time of day, and modifications to Apriori to mine these new itemsets. These new rules are evaluated on the UCC and Augsburg dataset in comparison with OccApriori as described in the previous chapter, and repeat the comparison with the existing approaches, to show that the new rules fill the gap in the capabilities of the algorithm.

5.1. Predicting time independent sequences with timeslot based rules

The last chapter showed that there are some patterns of movement which OccApriori has great difficulty learning compared to some existing approaches. This is because OccApriori is timeslot-specific; it correlates the occupant’s behaviour with a specific time of day. If an occupant’s patterns do not relate to specific times of day then OccApriori will require an example of the pattern for every timeslot in which it may need to be predicted. This is not the case for approaches which rely only on the sequence of recent locations, and not on the specific time at which those locations are visited.

09:00	*10:00*	*11:00*	*12:00*	*13:00*	*14:00*	*15:00*	*16:00*
A	A	A	A	A	A	A	A
A	A	A	A	A	**B**	**B**	A
A	A	**B**	**B**	A	A	A	A

Table 5-1 – An example training set with a sequence 'BB' that occurs once each in two of the instances at different times

09:00	*10:00*	*11:00*	*12:00*	*13:00*	*14:00*	*15:00*	*16:00*
A	A	A	A	**B**	**B**	A	A

Table 5-2 – An example test instance with the sequence 'BB' occurring at a different time than in the training instances

Table 5-1 shows example movements of an occupant who remains in location A except when visiting location B. These visits to location B last two hours but occur at different times. The ideal patterns the algorithm should find here are that the occupant will stay in B if they are there, unless they have been there for two hours. Table 5-2 shows an imaginary test set, with a single instance we would like to make next-slot predictions for, which includes another visit to B at a different time again.

Timeslot	*Rule*	*Confidence*	*Correct?*
13:00	{} => {13:00 = A}	100%	Incorrect
14:00	{} => {14:00 = A}	66%	Incorrect
15:00	{14:00 = B} => {15:00 = B}	100%	Incorrect

Table 5-3 – The rules which would be used to predict by Timeslot-OccApriori in the example test instance

Table 5-3 shows the rules that would be used to predict timeslots 13:00, 14:00 and 15:00 if we used association rules to predict the instance in Table 5-2 based on the instances in Table 5-1. In the case of timeslot 13:00, in the training data the occupant is always in A, so any applicable rules will have a 100% confidence that the occupant will be in A. This means that it predicts incorrectly, but since we are trying to predict a visit with a random start time using only recent movements, we expect to fail to predict the beginning of the visit. For timeslot 14:00 the best that can be achieved is predicting based on the fact that the occupant has been in A in 66% of the instances at 14:00, resulting in an incorrect prediction. When predicting we know that the occupant was in B at 13:00 in our test instance, but we have no prior example of {13:00 = B}, so we cannot use this data. However, we do have a prior example for {14:00 = B}, and so we predict with 100% confidence that the occupant will remain in B at 15:00, as this is what they did at 15:00 in the only instance which features {14:00 = B}. Of course, this is incorrect.

Timeslot	*Sequence*	*Confidence*	*Correct?*
13:00	A => A	88%	Incorrect
14:00	A, B => B	100%	Correct
15:00	B, B => A	100%	Correct

Table 5-4 – Sequences which could be used to predict in the example test instance

Table 5-4 shows the sequences that might be used if the same predictions were made with a sequence-based approach. The choice of sequence used to predict depends the approach, in this example we keep the confidence metric used by association rules: the proportion of times that the body of the sequence is followed by the final element is the confidence. We begin again at timeslot 13:00, and the prediction is made based on the fact that location A occurs 20 times in Table 5-1, of which it is followed by another A fifteen times and followed by B twice. It is also followed by nothing three times due to being at the end of the dataset, we ignore these cases for the confidence calculation. Fifteen times out of seventeen results in ~88% confidence, essentially saying that if the occupant is in A they will stay in A. This is usually correct, but for timeslot 13:00 in Table 5-2 it is incorrect. As before, we don't expect to predict the beginning of the visit. However, for timeslot 14:00 we have the sequence A, B, B, which has 100% confidence since every instance of A, B is followed by a B. This gives us a correct prediction. The sequence B, B, A works the same way for timeslot 15:00, giving us another correct prediction.

By learning only the sequence of locations without any reference to specific timeslots, a sequence based method can correctly recognise that the occupant will remain in and then depart B, even though the pattern occurs at a new time in the test. Timeslot-OccApriori could also correctly predict this with enough examples, but it would need one for every time at which the pattern occurs in a test, with sufficient support that rules are generated. In cases where such an event had limited examples, the algorithm would achieve higher accuracy if it could find these kinds of timeslot-independent patterns as well. Even in a dataset where the events occur at regular times, and Timeslot-OccApriori would perform well, it would be useful for it to be able to find timeslot-independent patterns, as it would then still be able to recognise the events even if their times changed. To this end, this chapter presents a modification of Apriori such that it mines sequences of locations from which timeslot-independent location-sequence-based association rules can be generated.

5.2. Ordered Itemsets

Mining sequences of locations requires a set of attributes which can represent such sequences in itemsets. The aim is to mine consecutive sequences of locations, as existing approaches demonstrate that recent locations are a reliable predictor of an occupant's next location. The representation which follows could also allow for non-consecutive sequences, however the itemset lattice for such sequences is much larger without necessarily providing an increase in accuracy, so the algorithm focuses on consecutive sequences.

In order to represent a sequence of locations, the attributes in a sequence must indicate the order of the items, without referring to specific time slots. To this end the itemsets use a new set of location attributes $\{q_0 \dots q_j\}$ which represent an ordered list. These attributes are similar to the location attributes $\{l_i \dots l_j\}$, but rather than a set of time/location pairs, $\{q_0 \dots q_j\}$ is a set of order/location pairs, the attribute indicating which position in the sequence the value occupies. The individual attributes are numeric, as with the timeslot attributes, with 0 representing the start of the sequence, and each following attribute being the offset from the start, so the second location has attribute 1, the next has attribute 2 and so on. As the algorithm only deals with consecutive sequences, for any sequence of length $j + 1$ the attributes will be $0 \dots j$.

Ordered itemsets are itemsets of the form $\{q_0 \dots q_j\}$. Table 5-5 – An example of an ordered itemset; the attributes state each locations position in the sequenceshows an example of such an itemset. Each ordered itemset is essentially the set of itemsets $\left\{\{l_k \dots l_{j+k}\}: n \leq k < m - j\right\}$ represented as a single list, where n and m

are the minimum and maximum timeslots in the dataset respectively, k is the range of possible start timeslots for the sequence to occur, and j is the length of the sequence. An ordered itemset may be instantiated to a specific itemset by choosing a starting timeslot k and adding k to every attribute in the sequence, turning the itemset $\{q_0 \ldots q_j\}$ into the itemset $\{l_k \ldots l_{j+k}\}$ for the chosen timeslot k. Table 5-6 shows the sequence from Table 5-5 with 3 example timeslot assignments.

0	*1*	*2*	*3*
Bob Office	Bob Office	Lecture Hall	Bob Office

Table 5-5 – An example of an ordered itemset; the attributes state each locations position in the sequence

9:00	*10:00*	*11:00*	*12:00*
10:00	*11:00*	*12:00*	*13:00*
13:00	*14:00*	*15:00*	*16:00*
Bob Office	Bob Office	Lecture Hall	Bob Office

Table 5-6 – Example timeslot assignments to the itemset from Table 5-5

With the attributes which replace the timeslot attributes from Timeslot-OccApriori in place the metadata attributes can be added back in, functioning exactly as before. Thus the full set of attributes is now:

$$A = \{d, o, q_0 \ldots q_j\} \tag{5-1}$$

Here d and o are again the day and occupant, and $q_0 \ldots q_j$ is the representation of a sequence as described here. As before, any data which could be useful can be added to the set of attributes and any useful correlations will be found without further intervention. However, in the case of mining sequences, attributes which are expected to have useful correlations with certain times of day will not be useful, as sequences do not refer to any specific time of day. This is why the timetable attributes $s_i \ldots s_j$ are not present; to be useful for sequence-based prediction, the presence of a timetabled event would have to positively correlate with a certain sequence of movements occurring at all possible times.

0	*1*	*2*	*3*	*12:00*
Bob Office	Bob Office	Lecture Hall	Bob Office	Lecture Hall

Table 5-7 – An example of an ordered itemset with timetable data added

Table 5-7 shows the example from Table 5-5 ordered itemset with a timetable entry added. With timeslot-specific rules, the presence of this timetable data will likely increase the confidence of predicting a lecture at 12:00. With sequence based rules however, using this timetable data would mean we are more confident that the occupant will give a lecture after any and every two-hour period spent in his office, if he is scheduled to give a lecture at 12:00. Since this is not what the timetable entry means, we will not actually find such a correlation, rendering the timetable data useless[1].

As with standard itemsets, rules generated from ordered itemsets are implications of the form $X \Rightarrow Y$where X and Y are now ordered itemsets. These rules are generated in the same way as before; the antecedent contains every item except the last element of the sequence, which becomes the consequent.

5.3. Confidence and Support

As discussed, each ordered itemset essentially represents a set of timeslot itemsets, so the definitions of support and confidence must be considered. The definition of support must maintain both the anti-monotonicity property and correctly reflect the frequency with which patterns appear in the data. For timeslot itemsets the definition of support is $supp(X) = freq(X)/|D|$, while $freq(X)$, is the number of instances I in D for which $X \subseteq I$. This definition is clear for timeslot itemsets as each itemset either occurs or does not occur in each instance, either $X \subseteq I$ or $X \nsubseteq I$, but ordered itemsets can occur more than once per instance.

09:00	*10:00*	*11:00*	*12:00*	*13:00*	*14:00*	*15:00*	*16:00*
A	A	A	A	A	A	A	A
A	**B**	**B**	**C**	**C**	**B**	**B**	A
A	A	**B**	**B**	A	**C**	**C**	A

Table 5-8 – An example set of instances with a sequence 'BB' which occurs three times across two instances, and a sequence 'CC' which occurs twice across two instances

[1] It would be possible to modify timetable attributes in a similar manner to the modification of the timeslot attributes, this will be discussed in the final chapter as part of the future work.

Frequency could be defined in the same way as it is for timeslot itemsets, the number of instances in which the ordered itemset occurs. In this case frequency would be the number of instances I in D for which $\exists k\colon X_k \subseteq I$, where $X_k$is an instantiation of X such that the ordered items $\{q_0 \dots q_j\}$ become the timeslot items $\{l_k \dots l_{j+k}\}$. However, this definition ignores how often the sequence appears within instances. Consider the sequences BB and CC in Table 5-8, which would both have a frequency of two as they both occur in two of the instances, despite the fact that sequence BB occurs three times, while CC occurs only twice.

In order to reflect the true frequency of ordered itemsets, frequency for an ordered itemset X is defined as:

$$freq(X) = \sum\nolimits_{k=n}^{m+1-j} freq(X_k) \tag{5-2}$$

where n and m are the first and last timeslots in the dataset and j is the length of the sequence. This means that the frequency of an ordered itemset is the sum of the frequencies of all the possible instantiations of the sequence it represents. Going back to our example, this gives sequence BB a frequency of three compared to a frequency of two for CC, as BB occurs twice in one instance.

This definition of frequency can result in support values greater than one, as the number of occurrences of the pattern may be greater than the number of instances in the dataset. The maximum possible support also varies depending on the length of the sequence, as longer sequences cannot occur as often, for example a sequence which is the same length as the instance can only occur once per instance. However these support values correctly reflect the frequency of the patterns in the dataset, and the anti-monotonicity property is kept.

We can determine that the anti-monotonicity property is kept as follows. For a pair of ordered itemsets X and Y such that $X \subset Y$, $X_k \subset Y_k$ for every k for which both X and Y can be instantiated. Since X_k and Y_k are ordinary itemsets, $upp(Y_k) \leq supp(X_k)$ for the aforementioned values of k, as that is the established anti-monotonicity property, and it follows that the sums of the supports maintain this inequality. There will be one or more values of k for which Y cannot be instantiated but X can, as k must fall in the range $n \leq k < m - j$, where n and m are the first and last timeslots respectively and j is the length of the sequence, and j is larger for Y than for X as Y is the longer sequence. For these values of k, Y_k will have a support of 0, while X_k may have a nonzero support, maintaining the inequality above. Thus when we increase the length of a sequence, the support must be the same or lower.

Using this definition of frequency and support, confidence is calculated as before as $conf(X \Rightarrow Y) = supp(X \cup Y)/supp(X)$. This means that confidence of a rule based on ordered itemsets is the number of times it is correct compared to the number of times it occurs in the dataset, including multiple occurrences in a single instance.

5.4. Candidate Generation

To generate ordered itemsets the algorithm uses a modified form of Apriori's candidate generation. Candidate generation works by combining every pair of frequent itemsets of length n to generate candidate itemsets of length $n + 1$, for increasing values of n. However, because the ordered itemsets represent consecutive sequences, the attributes for any ordered itemset of length j are $\{q_0 \dots q_j\}$. This means that two itemsets of the same length cannot be combined, as the attributes are the same. Combining them would result in an itemset which contained two values for a single attribute, which will always have a support of zero as an occupant cannot be in two places at once. Table 5-9 shows an example of two such sequences. If we combined these with standard candidate generation, attributes '1' and '2' would both be set to both 'Bob Office' and 'Lecture Hall'.

0	*1*	*2*	*3*
Bob Office	Bob Office	Lecture Hall	Bob Office
Bob Office	Lecture Hall	Bob Office	Bob Office

Table 5-9 – Two example ordered itemsets that we wish to combine in candidate generation

Instead of simply combining the items into a single set, for ordered itemsets the algorithm first increments the attributes of one of the itemsets by one, shifting the sequence to the right by one timeslot, and then combines them. In order to deal with attribute/value conflicts like the one described above the algorithm requires that after shifting the sequence, every attribute and value of each sequence match the other, except the ones which are outside the range of the other sequence. This means that it combines two itemsets $\{q_0 \ldots q_j\}$ and $\{r_0 \ldots r_j\}$ when the following holds: $\forall n: 1 \leq n \leq j-1: q_{n+1} = r_n$. Table 5-10 shows this process on the example itemsets from Table 5-9.

0	*1*	*2*	*3*	*4*
Bob Office	Bob Office	Lecture Hall	Bob Office	
	Bob Office	Lecture Hall	Bob Office	Bob Office
Bob Office	Bob Office	Lecture Hall	Bob Office	Bob Office

Table 5-10 – The example itemsets from Table 5-9, with the second itemset shifted to the right, and the itemset which results from combining them

As this example shows, the algorithm is essentially combining sequences if one of the sequences appears to take place one timeslot later than the other, providing an extra location for the end of the 'earlier' sequence. In the example, this is an extra 'Bob Office' entry two timeslots after the lecture.

As with standard candidate generation the algorithm starts with ordered itemsets of length $n = 1$ and proceeds through successive values of n, eliminating candidates which fall below the support threshold at each step. Section 5.3 showed that the anti-monotonicity property holds, so the algorithm will not remove any desirable itemsets in the candidate elimination step. Another consideration is whether it will generate all possible sequences with this generation process.

Any frequent ordered itemset $\{q_0 \ldots q_j\}$ can be broken down into two subsets $\{q_0 \ldots q_{j-1}\}$ and $\{q_1 \ldots q_j\}$. These in turn can be generated from the subsets $\{q_0 \ldots q_{j-2}\}$, $\{q_1 \ldots q_{j-1}\}$ and $\{q_1 \ldots q_{j-1}\}$, $\{q_2 \ldots q_j\}$. This can be generalised as any sequence $X = \{q_0 \ldots q_j\}$ being generated from the set of sequences that are l items shorter than $\{q_n \ldots q_{j-m}\}$, for $\forall n, m: n + m = l, 0 \leq n, m \leq j$. As you continue increasing l, breaking down the sequences into shorter sequences, eventually you will reach the point where $l = j$ and therefore $n = j - m$. At this point the subsets

$\{q_n \dots q_{j-m}\}$ are all one item long. Since the algorithm begins the candidate generation with all itemsets of length one, and follow this process in reverse to generate candidates, any possible consecutive sequence will be generated as long as it is above the support threshold.

5.5. Rule Selection

Rule selection proceeds in the same manner as with timeslot-dependent rules, except that ordered rules must be instantiated to check their applicability. Given the prediction target l_t and the data to be used in making the prediction V, for each ordered rule $\{q_0 \dots q_{j-1}\} => \{q_j\}$ we set $j = t$ to get the rule $\{l_{t-j} \dots l_{t-1}\} => \{l_t\}$. For each of these instantiated rules we then check whether $\{l_{t-j} \dots l_{t-1}\} \subseteq V$. As before, the applicable rule with the highest confidence is chosen to make the prediction. Since instantiated ordered rules are just ordinary timeslot based rules, the two types of rules can be combined into a single rule set and used simultaneously. However on the test datasets simply combining them did not provide any advantage over selecting and using only the more appropriate rule set for the data.

5.6. Evaluation of Performance

As with the evaluation in the previous chapter, the exact location of every occupant is predicted for every timeslot. Accuracy is measured as the percentage of predictions made which were correct, failure to predict is considered an incorrect prediction. As this part of the approach deals with consecutive sequences of locations only, the predictions it makes are all next-hour, and do not use timetable data. The evaluation uses confidence and support thresholds just above zero in order to maximize the coverage of the resulting rule set. In place of a higher support limit, the length of generated rules is limited to 2 items as shorter sequences have proven to be more reliable predictors of occupants’ next locations.

Ordered-OccApriori	*Timeslot-OccApriori*
86%	86%

Table 5-11 – Ordered and timeslot approach accuracy on the UCC dataset, making Next-Slot predictions for all occupants in all timeslots

Table 5-11 shows the accuracy of Ordered-OccApriori against Timeslot-OccApriori from Chapter 4, making next-slot predictions for all occupants in the UCC dataset. On this dataset they achieve the same accuracy.

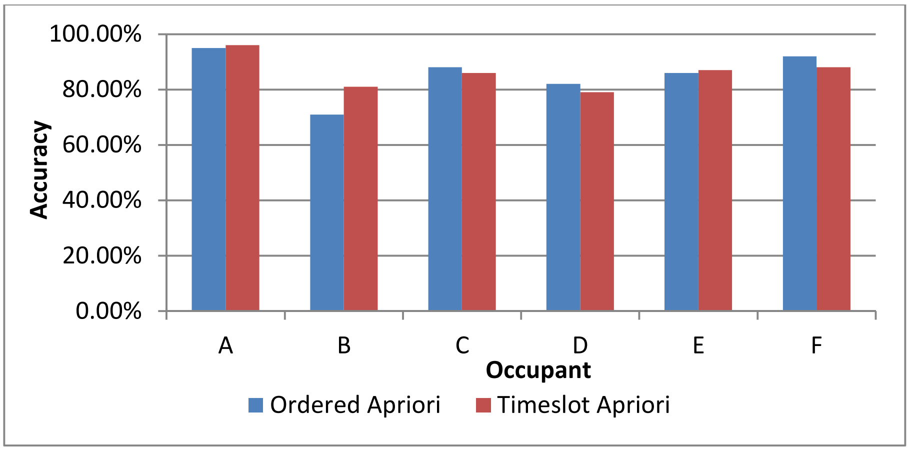


Figure 5-1 – Ordered-OccApriori and Timeslot-OccApriori's accuracy on the UCC dataset, across all times, broken down by occupant

Figure 5-1 breaks the accuracies for these approaches down per occupant. This shows that while the overall accuracy is the same, the two approaches in fact make slightly different predictions, resulting in slightly lower or higher accuracies for each occupant. The largest difference in accuracy is on Occupant B, where Ordered-OccApriori has noticeably lower accuracy than Timeslot-OccApriori. This is the occupant on whom Timeslot-OccApriori gained the most from the availability of timetable data, as this occupant has the most scheduled events and most varied movements. This indicates that for occupants whose movements are dictated by their schedule, their recent movements are not necessarily the best way to predict even their next location. For all the other occupants the accuracies are approximately the same, with each occupant favouring one approach over the other very slightly.

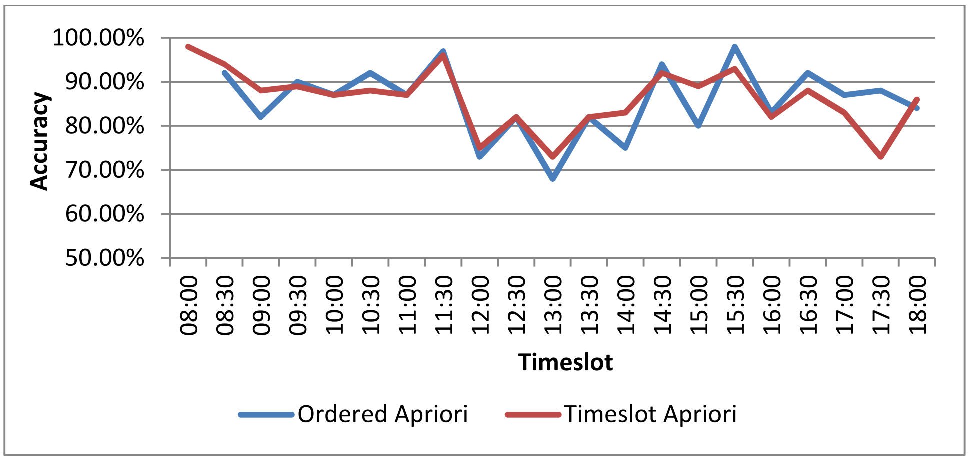


Figure 5-2 – Ordered-OccApriori and Timeslot-OccApriori's accuracy on the UCC dataset, across all occupants, broken down by time slot

Figure 5-2 breaks the accuracies for the two approaches down by timeslot, averaged over all occupants. As with the occupant breakdown there are some variations due to some different predictions being made, but overall the two approaches follow the same pattern of accuracy across the day quite closely.

Timeslot-OccApriori already performed well on the UCC dataset, and so there are only minimal gains over it from Ordered-OccApriori. By contrast, Timeslot-OccApriori performed poorly on the Augsburg dataset. Table 5-12 shows the accuracy on both approaches on the Augsburg dataset using the Seasonal split, wherein the summer data is used for training and the fall data used for testing.

Ordered-OccApriori	*Timeslot-OccApriori*
76%	56%

Table 5-12 – OccApriori's accuracy for Next-Slot predictions on the Seasonal split of the Augsburg dataset, predicting all occupants in all timeslots

Chapter 4 described how the occupants in the Augsburg datasets have movement patterns which recur, but recur at different times, making them difficult to learn for Timeslot-OccApriori and more suited to a sequence-based approach. Table 5-12 shows that the sequence based modification to OccApriori achieves a significantly higher accuracy on this dataset than Timeslot-OccApriori. In fact, even when given most of the dataset as training data, Timeslot-OccApriori maxed out at 68% accuracy (as shown in Table 4-8), still lower than the accuracy achieved by Ordered-OccApriori on the smaller amount of training data. This demonstrates both that it was OccApriori’s inability to learn time-independent patterns of movement that made predicting the Augsburg occupants difficult, and that Ordered-OccApriori can successfully learn these patterns.

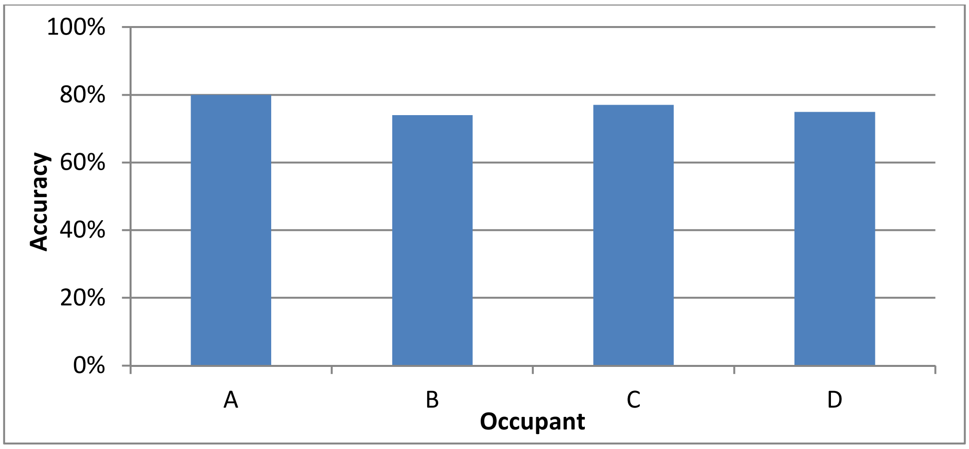


Figure 5-3 – Ordered-OccApriori’s accuracy on the Augsburg dataset, across all times, broken down by occupant

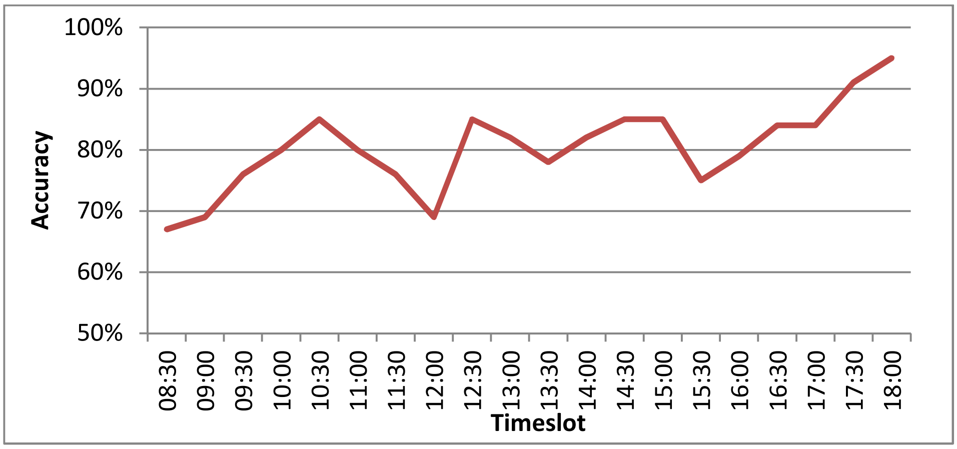


Figure 5-4 – Ordered-OccApriori’s accuracy on the Augsburg dataset, across all occupants, broken down by time

Figure 5-3 and Figure 5-4 show the accuracy of Ordered-OccApriori on the Augsburg dataset across occupants and timeslots respectively. All of the occupants are approximately equally predictable. Across the course of the day the accuracy increases and decreases, however the trend is an increase as the day goes on. With no reliable time-of-day based predictions possible the occupants’ arrival times are difficult to predict, making for low early accuracy, while their departures late in the day make it easier to predict that they will remain away, resulting in higher accuracy.

5.7. Comparison to Existing Approaches

This section again compares OccApriori to three existing approaches. The configuration of these approaches is the same as discussed in Section 4.8. That section explains that in order to achieve the best accuracy the frequency predictor and Elman net were allowed to retrain after each day of predictions, as they trained fast enough to this to be feasible. Ordered-OccApriori is also trains fast enough, and thus also retrains after each day of data to improve accuracy.

HMM	*Elman Net*	*Freq. Predictor*	*Timeslot-OccApriori*	*Ordered-OccApriori*
77%	86%	87%	86%	86%

Table 5-13 – All approaches accuracy for Next-Slot predictions on the UCC dataset, predicting all occupants across all time slots

Table 5-13 shows the accuracy of each approach for next-slot predictions on the UCC dataset. As mentioned in section 4.8, the HMM does worse as retraining it after every day was not feasible during evaluation. The other four all achieve approximately the same accuracy, despite Timeslot-OccApriori also not being able to retrain.

HMM	*Elman Net*	*Freq. Predictor*	*Timeslot-OccApriori*	*Ordered-OccApriori*
74%	76%	77%	56%	76%

Table 5-14 – All approaches accuracy for Next-Slot prediction on the Augsburg dataset, predicting all occupants, at all time slots

Table 5-14 shows the accuracy of each approach for next-slot predictions on the Augsburg dataset. The previous section showed that Ordered OccApriori performs much better on this dataset than Timeslot OccApriori; these results show that this improvement brings OccApriori in line with the existing approaches, achieving approximately the same accuracy.

It should be noted that while the approaches compared against are only capable of next-location prediction, they have the advantage of maintaining smaller, simpler models of the dataset. In particular, the Hidden Markov Model (though its training performance was poor) and the Elman Net both maintain a static model in which the values of a fixed set of variables (probabilities in the HMM and weights in the EN) are learned based on the training set. Thus the model remains the same size regardless of the data used as input.

Contrasting with this, the frequency predictor and Timeslot OccApriori both maintain the set of consecutive patterns they have seen, meaning that their model grows along with the number of such patterns which appear in the dataset. However, consecutive patterns are only a small part of the space of all patterns, and so these approaches are still lightweight and fast. Timeslot-OccApriori, which only eliminates those patterns which are not useful for predicting occupant locations and keeps all others, has to search more of the space of possible patterns and keep a larger list of learned patterns than the other approaches, resulting in greater resource demands. As previously discussed, this reliance on more specific patterns is also which this approach has difficulty with the Augsburg occupants.

This means that out of these approaches, Timeslot OccApriori is the one which is most likely to have scaling issues on larger datasets; however association rules are designed to operate on extremely large datasets and so there is extensive work on increasing the efficiency of the mining process. The future work section of the final chapter discusses the application of the design of OccApriori to more efficient base algorithms.

5.8. Summary

This chapter described a modification to association rule mining which allows it to mine pure location sequences. This allows OccApriori to learn and predict using patterns which occur at varying times of day. It then evaluated the performance of the ordered rules, and showed that they improve on the timeslot rules on the Augsburg dataset, matching existing sequence prediction approaches.

The ordered rules only predict the last in a sequence of locations, and thus cannot be used for prediction further into the future. Occupants whose movements do not relate to the time of day remain unpredictable without their recent movements to use as a basis. However, one of the goals is an approach with the flexibility to make a variety of types of prediction using whatever data is available. The addition of ordered rules fills a gap in the capabilities of OccApriori compared to existing approaches. The ordered rules and timeslot rules can be combined into one rule set, so for a dataset where only some occupants' movements relate to the time of day, or where each occupant's movements relate to the time of day some of the time, the algorithm can generate a combined set of rules at, making each prediction using timeslot or ordered rules as appropriate.

6. Occupant Prediction Using Generalised Locations

This chapter considers the application of generalised association rules to occupant location prediction, to increase the range of predictions OccApriori can make. By applying a taxonomy to the occupant locations, general classes of locations can be described. General locations can be predicted in cases where there are no high-confidence predictions of a specific location, and make specific predictions based on general locations, such as predicting that the occupant will not return to their office until later based on their distance from the office.

Section 6.1 details some example cases where generalised locations would allow the algorithm to make predictions it could otherwise not make. Sections 6.2 and 6.3 describe applications of a taxonomy that allow it to make the desired predictions. Section 6.4 takes the model of the occupants in the UCC dataset from chapter 3, and modifies the patterns of one of the occupants to incorporate test cases based on the previously discussed examples. Finally section 6.5 evaluates the performance of the taxonomy-enhanced approach versus timeslot OccApriori on these test cases, showing that the application of the taxonomy allows it to make extra predictions.

6.1. Increasing prediction flexibility with taxonomies

Thus far OccApriori has only been concerned with predicting occupants' exact future locations based on the exact locations in the past. However, there are cases where it would be useful to make predictions involving more general classes of locations. This includes being able to make a more generalised prediction about where an occupant will go, for example predicting what part of a building they will be in, whether it will be some location involved in a certain task, or even simply predicting that they will be somewhere other than their current location. It also includes making predictions based on patterns that do not reference specific locations, for example predicting an occupant's next location based on what part of a building they were in, or what task they were involved in.

These ideas are represented using a taxonomy. In a taxonomy of locations, the leaf nodes would be the specific locations which have been learned and predicted so far, i.e. specific rooms. Nodes higher in the taxonomy would represent a more general class of locations and so are referred to hereafter as 'generalised locations'. For example, if we classify rooms by their location, then rooms 'A', 'B' and 'C' might all be generalised as 'Building_1' as all these rooms are located in Building 1. Or, they may all be generalised as 'Office' as they are all offices. Using such taxonomies, it is possible to refer to occupant locations in a more general sense than exact locations, allowing for generalised rules and predictions, i.e. rules and predictions which use generalised locations instead of specific ones.

In some cases these generalised predictions are only added-value, the algorithm would predict the specific location and add the general prediction to provide extra information. However there are cases where the ability to learn more general patterns would allow it to make predictions it would otherwise be unable to make. This chapter examines two examples of this in particular. The first is difficulties predicting an occupant's location due to being unable to choose between several low-confidence options, i.e. knowing they will go somewhere but being unsure where. The second is being unable to recognise patterns which relate to classes of locations with few or no examples of those locations, for example an occupant who regularly has meetings of fixed duration, but with varying location.

09:00	*10:00*	*11:00*
Office	Office	A
Office	Office	B
Office	Office	C

Table 6-1 – An example occupant who remains in their office until 11:00, at which point they go to one of three locations with equal probability

Table 6-1 shows an example dataset where an occupant stays in their office until 11:00, at which point they go to one of three locations. As we have no data on what causes the occupant to go to one location rather than another, we are simply left with a 33% chance of the occupant going to each location. This will result in three rules of the form {09:00=Office, 10:00=Office} => {11:00=A}, one for each location, each of which will have 33% confidence.

We can see from the data that the occupant always leaves their office at 11:00 to go somewhere; the only doubt is where exactly they will go. However, due to the fact that we are trying to predict the exact location, all we can do is make a low-confidence prediction for one of the three locations. The low confidence is correct, as we don't know with any certainty where the occupant will go, but it is an issue if the rule is not chosen because of its low confidence. The simplest example of this is if the confidence threshold is too high to allow these rules; if our confidence threshold is 50%, then in the example above we will make no prediction for where the occupant is at 11:00.

09:00	*10:00*	*11:00*
Office	Office	A
Office	Office	B
Office	Office	C
Office	Office	Office
Office	Office	Office

Table 6-2 – An example occupant who remains in their office, with the possibility of leaving at 11:00 to go to one of three locations

Something similar would occur if we were to add two extra instances to the data in Table 6-1 as shown in Table 6-2; now with two examples of the occupant staying in the office at 11:00, we would predict (with 40% confidence) that the occupant will remain in the office, even though there is a 60% chance they will leave.

09:00	*10:00*	*11:00*
Office	Office	**Not-Office**
Office	Office	**Not-Office**
Office	Office	**Not-Office**
Office	Office	Office
Office	Office	Office

Table 6-3 – The same example occupant as Table 6-2, with the non-office locations replaced with a single generalised location

We can avoid these failures to predict by generalising the locations outside the office. Table 6-3 replaces the locations A, B and C with the generalised location 'Not-Office'. If we were to train on this modified dataset, we would end up with two possible predictions for where the occupant will be at 11:00, 'Office' with a confidence of 40% and 'Not-Office' with a confidence of 60%. This means that we can now correctly predict that the occupant is more likely to leave the office, without saying where specifically.

09:00	*10:00*	*11:00*
Office	Office	**Not-Office**
Office	Office	**Not-Office**
Office	Office	**Not-Office**

Table 6-4 – The example occupant from Table 6-1 with locations A, B and C generalised

Returning to the example in Table 6-1, generalising locations A,B and C results in Table 6-4. We can now learn a 100% confidence rule stating the occupant will leave their office, allowing us to make the prediction with any confidence threshold.

09:00	*10:00*	*11:00*
A	A	A
B	B	B
C	C	C

Table 6-5 – An example training set featuring an occupant who may be in one of three locations with equal probability, and who will remain in that location

Table 6-5 shows an example of the opposite issue. In this case the occupant stays in the same location in all three timeslots. For each of these locations we can learn that the occupant will remain in that location rather than returning to the office. However we do not learn the general pattern that there are locations which will keep the occupant away from their office.

09:00	*10:00*	*11:00*
D	D	D
E	E	E

Table 6-6 – An example test set in which the occupant is in two new locations

Table 6-6 shows an example test set which has two new locations which keep the occupant away from the office. We cannot predict anything regarding these locations as they do not appear in the training set. For example, if we wish to predict the occupant's location at 11:00, we can't as there is no data and hence no rules regarding locations D or E at 09:00 or 10:00. This is similar to the issue addressed in Chapter 5, where the algorithm needed to be able to recognise patterns even if the time at which they occur changed; in this case the time remains the same but the locations involved are changing. If we have data on these locations that tells us that A through E are all locations that will keep the occupant from their office, we would like to use that data to be able to make predictions for the instances in Table 6-6.

09:00	*10:00*	*11:00*
Not-Office	Not-Office	Not-Office
Not-Office	Not-Office	Not-Office
Not-Office	Not-Office	Not-Office
Not-Office	Not-Office	Not-Office
Not-Office	Not-Office	Not-Office

Table 6-7 – The example training and test sets, now shown together and with the locations generalised

Table 6-7 replaces all the locations in the example training and test sets with the generalised 'Not-Office' location. With this replacement it is trivial to learn the pattern that the occupant will remain out of the office if they start the day out of the office. If we wish to predict the occupant's location at 11:00 for example, the rule {09:00=Not-Office, 10:00=Not-Office => 11:00=Not-Office} will be available.

The following sections discuss applying taxonomies to the data used in OccApriori in order to learn general patterns like these alongside the specific patterns which it already learns.

6.2. Consequent-Only Taxonomy

As shown in the example in Table 6-1 above, some problems the algorithm solves by generalising predictions only require that the prediction itself be generalised; the antecedent of the rule does not need to be modified. This means that for this type of prediction, it can actually apply a taxonomy during rule generation, avoiding extra time or memory requirements for the itemset mining process. Instead the rule generation process generates two rules from every itemset, resulting in a rule set up to twice as large. The following is an example of this process.

09:00	*10:00*	*11:00*
Office	Office	A
Office	Office	B
Office	Office	C

Table 6-8 - An example occupant who remains in their office until 11:00, at which point they go to one of three locations with equal probability

Table 6-8 again shows the example of an occupant who has three locations they may go to when they leave their office. We wish to generalise these locations into a single non-office location, and Table 6-9 shows such taxonomy.

Location	Generalised
A	Not-Office
B	Not-Office
C	Not-Office

Table 6-9 – A single-level taxonomy which generalises three locations into one

As discussed previously, during the rule generation process every frequent itemset is converted into a rule with the latest timeslot item as the consequence. The generated rule is discarded if its confidence is below the confidence threshold. We now add an extra step where the location in the consequent is generalised according to the taxonomy above, i.e. the specific location is replaced with the corresponding generalised location in Table 6-9, if the specific location appears in the table. The confidence of the generalised version rule is checked separately from the original rule, and it too is discarded if its confidence is too low. Table 6-10 shows an example of a specific rule, and the generalised rule which would be generated from it based on the above taxonomy, with confidence values based on Table 6-8.

Rule	Confidence
{09:00=Office, 10:00=Office} => {11:00=>A}	33%
{09:00=Office, 10:00=Office} => {11:00=>Not-Office}	100%

Table 6-10 – Two rules which can be found using Table 6-8 as a training set, the first with a specific location, the second with a generalised location, and the corresponding confidences

Calculating the confidence for a generalised rule requires checking the support of the entire rule, which contains both specific and generalised locations. To accommodate this, the algorithm simply maintains a generalised version of the training set along with the unmodified version. If the antecedent is supported by an instance, and the consequent is also supported by the generalised version of the same instance, then the instance supports the rule.

The fact that the algorithm checks the confidence separately for the generalised rules means that they can be generated and retained even if their specific counterparts are not. For example, if the confidence threshold was 50% then the first rule in Table 6-10 would be discarded, but it would still generate and retain the second rule. However, as the itemset mining process is unmodified in this approach, the itemset on which these rules are based must have sufficient support to be considered frequent, otherwise neither rule will be generated. The following section considers generalising itemsets thus altering their support.

When the process is complete the result is two sets of rules, the standard set generated without regard to the taxonomy, and the set of generalised rules which can be used when a generalised prediction is desired. The evaluation section will discuss how predictions are made using the generalised rules to evaluate their accuracy.

6.3. Full Itemset Taxonomy

Table 6-5 and Table 6-6 presented an example where the aim was to predict based on a general pattern in the occupant's movements as well as making a general prediction. This requires that the algorithm allow for rules which contain generalised locations in the antecedent as well as the consequent. To achieve this, the taxonomy is applied at the candidate generation stage by adding the generalised items into the dataset. For a given dataset I:

$$\forall b : \exists a \subset I : gen(a) = b, I = I \cup b \qquad (6\text{-}1)$$

where a and b are 1-itemsets and $gen()$ is a function to generalise an itemset based on a given taxonomy. This essentially means that for every item in each instance which contains a location which can be generalised, an extra item is added with the same attribute but with the generalised location as the value.

Table 6-11 shows the example training set from Table 6-5 with the taxonomy in Table 6-9 applied, which is simply that every location other than 'Office' is 'Not-Office'. It also has an extra instance with two 'Office' locations, which has also been generalised using the taxonomy. Notice that for the items which were generalised, locations A, B and C, there is now a duplicate item in the instance with the same attribute but a different value. For the 'Office' locations, which are not in the taxonomy, there is no additional item, the lack of which is represented by a blank cell in the table.

09:00	*10:00*	*11:00*	*09:00*	*10:00*	*11:00*
A	A	A	Not-Office	Not-Office	Not-Office
B	B	B	Not-Office	Not-Office	Not-Office
C	C	C	Not-Office	Not-Office	Not-Office
Office	A	Office		Not-Office	

Table 6-11 – An example training set in which all locations in the dataset are generalised according to a taxonomy, adding the corresponding general locations to the dataset alongside the specific locations where applicable

Multiple items with the same attribute in the same itemset have come up before, as a consequence of candidate generation for timeslot itemsets and ordered itemsets. In those cases it would mean the occupant was in two places at once, and so would never appear in the data, and the itemsets would always have a support of 0. For example, the itemset {09:00=A, 09:00=B} could never actually happen, and will always be discarded if it is generated.

In this case the duplicate attributes are added into the data itself, which means itemsets which contain apparently contradictory values for an attribute can now be supported by the data, as long as one of the values is a specific location and the other is a generalised location. In some sense the occupant can now be in multiple locations at once, as we can refer to one physical location with multiple values. So while an itemset like {09:00=A, 09:00=B} will still never be supported, an itemset like {09:00=A, 09:00=Not-Office} can be, and indeed is in our example. However, itemsets with duplicate attributes are still discarded during candidate generation.

If we have a pair of 1-itemsets a and b such that $b = gen(a)$, and we have applied our taxonomy to the dataset as in equation (6-1), then it follows for every instance I that if $a \subset I$ then $b \subset I$. This means that the support of a and $a \cup b$ will be the same, and any rules with a and $a \cup b$ in the antecedent will apply in the same cases. For this reason itemsets which have duplicate attributes can be discarded. The aim is to have an itemset, and from that a rule, which either has item a and thus implies the presence of the generalised item b, or which has item b and thus encompasses the specific item a and other specific items. An itemset which contains both is redundant, but is a valid itemset which can be supported by the dataset, and so will consume time and memory in candidate generation if it is not discarded. Table 6-12 shows an example of this based on our example training set in Table 6-11. The first rule can make a general prediction from specific data, and the second rule can make a general prediction from general data. The third rule is actually the same rule as the first rule; it can only apply when 09:00=A, so the statement of 09:00=Not-Office is redundant. By not allowing itemsets with duplicate attributes, the generation of the itemset which is the antecedent of the third rule is avoided, as is the generation of the rule.

{09:00=A} => {10:00=>Not-Office}
{09:00=Not-Office} => {10:00=>Not-Office}
{09:00=A, 09:00=Not-Office} => {10:00=>Not-Office}

Table 6-12 – Three versions of the same rule, predicting using a specific location, a general location, and both versions of the location, respectively

As the generalised locations are now in the training data, candidate generation, support and confidence calculations, and rule generation all proceed as normal. The only addition is that, as with the consequent-only approach, the rules which predict specific and general locations must be separated, so that the appropriate type of rule can be looked up when trying to make a certain type of prediction. This is achieved simply by checking if the location in the consequent of each generated rule is specific or general, and storing it in the appropriate set.

The goal of this approach is to make predictions based on general patterns of occupant movement. Table 6-13 shows the generalised version of both Table 6-5 and Table 6-6, the first three instances being the training set and the latter two being the test set. All the instances have generalised items added to them per the taxonomy in Table 6-9.

09:00	*10:00*	*11:00*	*09:00*	*10:00*	*11:00*
A	A	A	Not-Office	Not-Office	Not-Office
B	B	B	Not-Office	Not-Office	Not-Office
C	C	C	Not-Office	Not-Office	Not-Office
D	D	D	Not-Office	Not-Office	Not-Office
E	E	E	Not-Office	Not-Office	Not-Office

Table 6-13 – The example training and test sets from Table 6-5, with the locations generalised according to the taxonomy in Table 6-9

When we encounter the occupant movements in the test set {09:00=D, 10:00=D} and {09:00=E, 10:00=E}, we are still unable to predict {11:00=D} and {11:00=E} respectively, as these locations have not been seen before. However, we can now learn the pattern {09:00=Not-Office, 10:00=Not-Office, 11:00=Not-Office}, which applies to every instance in the dataset, and thus can be used to predict that the occupant will not return to their office, even though we don't recognise the location they are in.

This approach to applying the taxonomy to the occupant location data also covers the capabilities of the consequent-only approach. While the consequent-only approach works on rules rather than itemsets, every rule it finds is essentially an itemset with a single generalised item. If such a correlation exists in the dataset, it will be found by the full-itemset approach along with all other sufficiently supported correlations involving generalised items.

6.4. Generated Dataset Modifications

As none of the available actual datasets have sufficient examples of patterns which could be learned only by applying a taxonomy to allow these approaches to significantly improve the accuracy, the evaluation instead uses a modification synthetic version of the 4C dataset which includes such patterns. The modified dataset incorporates the two types of patterns which were discussed in this chapter which cannot be learned without the application of a taxonomy. The following are the modifications made to Occupant B from the dataset, whose model was used to generate this dataset as they had the most suitable existing tasks.

In the model, Occupant B has 11 tasks across the five days of the working week which represent lectures or labs. Each of these tasks has a fixed day, start time, and duration, and most have an 80%-90% chance of occurring. In the original model each task had a fixed location, with some tasks occurring in the same location for a total of 5 locations. For taxonomy testing, the locations of all 11 tasks are replaced. The model defines a set of 10 fictional locations. Each of the 11 tasks occurs in these same 10 locations, and always has an equal (i.e. 10%) chance of occurring in each of the locations. This configuration is the same in the training and test set. Table 6-14 shows the properties of the tasks aside from the location.

Task ID	*Day*	*Time*	*Duration*
273	Monday	10:00	2hrs
274	Monday	14:00	1hr
275	Tuesday	09:00	2hrs
276	Tuesday	11:00	2hrs
277	Wednesday	02:00	1hr
278	Wednesday	15:00	1hr
279	Wednesday	16:00	1hr
280	Thursday	09:00	2hrs
281	Thursday	11:00	1hr
282	Thursday	16:00	1hr
283	Friday	15:00	1hr

Table 6-14 – Occupant B's tasks, showing the day they occur on, the time at which they commence and their duration

The result of this modification is that while the set of locations in which these 11 tasks can occur is known, which location will be chosen each time one of the tasks occurs is random, and so cannot be predicted. Since the set of locations is known however, it should be possible to able to apply a taxonomy which generalises these 10 locations into a single lecture-type location, where these 11 lecture-type tasks occur. The consequence-only approach should be sufficient to make the general prediction that the occupant will be in this lecture-type location at the times at which these tasks occur.

Occupant B has 5 existing roles which group their activities across the days of the week. To these 2 new roles were added which model the occupant being abroad. These 2 roles have a total of 11% chance between them of being selected as the role for any given day. When in one of these roles, the occupant will start the day in abroad one of 5 countries, randomly selected, with each country having an equal (i.e. 20%) chance of being selected. One of these two roles keeps the occupant in the country for the entire day. The other role has the occupant return to their office at 13:00, the concept being that they completed their return trip from being abroad that morning, and have returned to the office. Within the 11% chance of the occupant being abroad, two configurations are evaluated, one in which the occupant has a 75% chance of remaining abroad, and one in which the occupant has a 75% chance of returning. The configuration is the same in the training and test set except that the 5 countries are replaced with 5 new countries in the test set. Table 6-15 shows an overview of the configuration of these roles.

Role ID	*Location*	*Return Time*
Training Set		
296	Countries 1…5	N/A
300	Countries 1…5	13:00
Test Set		
297	Countries 6…10	N/A
301	Countries 6…10	13:00

Table 6-15 – The two roles added to Occupant B to model going abroad to a random country, with the first two roles being used when generating the training set, and the latter two being used when generating the test set

The result of this modification is that when the occupant goes abroad in the test set, their location is not recognised as it does not appear in the training set. As illustrated by the example in section 6.3, this means a prediction cannot be made without the application of a taxonomy. Since the pattern of staying abroad or returning is consistent, it should be possible to make a prediction with a taxonomy applied. The probability of the occupant returning during the day if they are abroad in the morning is varied in the model; whether the occupant actually stays abroad or returns is random, however in each configuration one or the other is far more likely, and so if the algorithm is correctly learning the occupants patterns it should be able to predict the more likely outcome.

The days that the occupant goes abroad are randomly selected, so it will not be possible to make accurate next day predictions, as there is no basis on which to predict that the occupant will be abroad. To deal with this there is an additional attribute similar to a timetable entry which states that the occupant will be abroad on the days that an abroad role is chosen, in order to allow next day predictions regarding the occupant staying abroad or returning. This extra attribute only states that the occupant will be abroad, not the specific location they will be in, and is only used when evaluating next-day predictions.

6.5. Evaluation of Performance

As with the previous evaluations, the accuracy of OccApriori is evaluated by having it predict the exact location of every occupant in every timeslot for every instance in the test set. In this case the prediction has two components, a prediction of the occupant's specific location, and a prediction of the occupant's general location. The general component is a second chance for the predictor to predict correctly; the base accuracy for the general prediction evaluation is the accuracy the approach achieves without using the taxonomy, i.e. if the specific prediction is correct the whole prediction is simply considered correct. Accuracy is increased in cases where the use of the taxonomy allows for the recognition of extra patterns, and correct general predictions are available to replace incorrect specific predictions.

When making a generalised prediction both the general and specific rules are included in the rule set used to predict. The evaluation taxonomy only has two locations, 'Lecture' and 'Abroad'; without additional specific locations such as the office and 'Away' to choose from, the predictor will simply guess one of these two general locations for every timeslot. Due to the inclusion of the specific rules as well, the predictor will only predict one of the general locations if it has a higher confidence than the specific prediction. If the generalised prediction is actually a specific location it has no effect on the accuracy, because any prediction where the specific component was correct is counted as a correct prediction anyway.

	Next Timeslot	*Next Day*
No Taxonomy	58%	41%
Taxonomy	64%	54%

Table 6-16 – Accuracy of OccApriori on the synthetic test set with and without consequent-only location generalisation

Table 6-16 shows the accuracy of the consequent-only approach versus using no taxonomy on the synthetic test set. It shows an increase in accuracy for both next timeslot and next day predictions when the taxonomy is used.

Timeslot	*Actual Location*	*No Taxonomy*	*Taxonomy*
08:30	102	102	102
09:00	FL5	102	Lecture
09:30	FL5	FL5	Lecture
10:00	FL5	FL5	Lecture
10:30	FL5	FL5	Lecture
11:00	FL3	NOPRED	Lecture
11:30	FL3	NOPRED	Lecture
12:00	FL3	102	Lecture
12:30	FL3	102	Lecture
13:00	102	102	102
13:30	102	102	102

Table 6-17 – An example instance from the test set, showing the next-timeslot predictions made with and without consequent-only location generalisation

Table 6-17 shows an example instance and the corresponding next-timeslot predictions where the use of the taxonomy improves prediction accuracy. The 'Actual Location' column shows that the occupant is in their office at the beginning and end of the time period, and visits two of the randomly selected locations in the intervening time. Without the use of the taxonomy, the predictor gets the time slots spent in the office correct. For the 09:00-12:00 event, it gets three of the timeslots correct. The predictor observes that the occupant moves to FL5 at 09:00 and has rules with sufficient support and confidence which predict that the occupant will remain there. The same is not true for FL3, for the 11:00-13:00 event it varies between predicting that the occupant will be in their office, and making no prediction at all. This is due to the rules which relate to this location having too low confidence. Applying the taxonomy, OccApriori correctly predicts the occupant's location in every timeslot, as the generalised rules have much higher confidence, similar to the example in section 6.2. As the specific location rules are available when making generalised predictions, it is still able to correctly predict that the occupant is in their office in the cases where the generalised locations do not apply.

Timeslot	*Actual Location*	*No Taxonomy*	*Taxonomy*
08:30	102	102	102
09:00	FL5	NOPRED	Lecture
09:30	FL5	NOPRED	Lecture
10:00	FL5	NOPRED	Lecture
10:30	FL5	NOPRED	Lecture
11:00	FL3	NOPRED	Lecture
11:30	FL3	NOPRED	Lecture
12:00	FL3	NOPRED	Lecture
12:30	FL3	NOPRED	Lecture
13:00	102	NOPRED	NOPRED
13:30	102	102	102

Table 6-18 – An example instance from the test set, showing the next-day predictions made with and without consequent-only location generalisation

Table 6-18 shows the same example instance and the corresponding next-day predictions. In this case the predictor fails to make any prediction for the two events with randomised locations. As it does not have the occupant's recent movements available, all the possible rules are too low confidence to be included in the rule set, and so it cannot predict anything. Again, the addition of the taxonomy allows the algoirhtm to make correct general predictions. As the events occur at fixed times, the occupant's recent movements are not required to make the prediction, and with the generalisation of the location the confidence of the rules is high enough for them to be included.

	Next Timeslot	*Next Day*
No Taxonomy	63%	48%
Taxonomy	93%	83%

Table 6-19 – Accuracy of OccApriori on the synthetic test set configured for a low return chance, with and without full-itemset location generalisation

Table 6-19 shows the accuracy of the full-itemset approach versus using no taxonomy on the on the synthetic test set configured such that the occupant has a 75% chance of remaining abroad on the days on which they are abroad. The random location lectures are still present in the dataset, and are predicted using generalised rules as with the consequent-only approach. In addition, with this application of the taxonomy, the algorithm can now predict that the occupant will remain abroad, increasing the accuracy further.

Timeslot	*Actual Location*	*No Taxonomy*	*Taxonomy*
09:00	country6	102	Abroad
09:30	country6	102	Abroad
10:00	country6	country1	Abroad
10:30	country6	country1	Abroad
11:00	country6	102	Abroad
11:30	country6	102	Abroad
12:00	country6	102	Abroad
12:30	country6	102	Abroad
13:00	102	102	Abroad
13:30	102	102	102
14:00	102	102	102

Table 6-20 – An example instance from the test set, low return chance configuration, showing the next-timeslot predictions made with and without full-itemset location generalisation

Table 6-20 shows an example instance in which the occupant is abroad and returns part way through the day, which in this configuration is the less likely occurrence, with a 25% chance given that the occupant was abroad in the first place. As this approach applies the taxonomy to the dataset itself, the instance also contains the generalised versions of the locations where applicable, which in this case is generalising ‘country6’ to ‘Abroad’. The table also shows the next-timeslot predictions with and without the use of the taxonomy.

Without the taxonomy the predictor primarily predicts that the occupant is in their office, as that is their most common location. As these are next-timeslot predictions, the predictor knows that the occupant is in ‘country6’, however there are no relevant rules as this location does not appear in the training set. The predictor also knows that the occupant is in the general location ‘abroad’, however due to the various countries it has to choose from in the training set, the best it can do with this information is to guess ‘country1’ in two timeslots.

When the taxonomy is applied the predictions correctly reflect that the occupant is abroad. As the entire dataset has the taxonomy applied to it, it is trivial for the predictor to make predictions such as {09:00=Abroad => 10:00=Abroad}. The occupant is predicted to remain abroad at 13:00 as in this configuration the occupant does in fact remain abroad more often than not, though they return in this example. This error is corrected in the next timeslot, as these are next-timeslot predictions and the predictor can see that the occupant has returned.

Timeslot	*Actual Location*	*No Taxonomy*	*Taxonomy*
09:00	country6	102	Abroad
09:30	country6	102	Abroad
10:00	country6	country1	Abroad
10:30	country6	country1	Abroad
11:00	country6	102	Abroad
11:30	country6	102	Abroad
12:00	country6	102	Abroad
12:30	country6	102	Abroad
13:00	102	102	Abroad
13:30	102	102	Abroad
14:00	102	104	Abroad

Table 6-21 – An example instance from the test set, low return chance configuration, showing next-day predictions made with and without full-itemset location generalisation

Table 6-21 shows the same example instance with the Next-Day predictions. The predictions without the taxonomy are the same as the next-timeslot predictions made without the taxonomy in Table 6-20. This is because the predictor knows based on the timetable attribute that the occupant is abroad, but again has no basis on which to choose between the countries in the training set, and so defaults to the occupant's most common location in most timeslots. With the taxonomy, it predicts that the occupant is abroad all day. In this configuration the occupant remains abroad more often than not, and so without data from previous timeslots on the same day to show that the occupant has returned, the more likely case is that the occupant remains abroad.

	Next Timeslot	*Next Day*
No Taxonomy	72%	59%
Taxonomy	94%	86%

Table 6-22 – Accuracy of OccApriori on the synthetic test set configured for a high return chance, with and without full-itemset location generalsation

Table 6-22 shows the accuracy of the full-itemset approach as in Table 6-19, except that in this configuration the occupant has a 75% chance of returning on days they are abroad. Other than the return probability, the occupant model is unchanged. The biggest change is an increase in both next-timeslot and next-day accuracy when not using the taxonomy. This is because the predictor defaults to predicting the occupant will be in their office when it does not have the help of the taxonomy, and since the occupant returns to their office more often in this configuration, this default prediction is correct more often.

Table 6-23 shows an example instance in which the occupant is abroad and returns part way through the day, which in this configuration is the more likely occurrence, with a 75% chance given that the occupant was abroad at all. As before the instance also contains the generalised locations where applicable. The predictions in the table are the next-timeslot predictions with and without the use of the taxonomy.

Timeslot	*Actual Location*	*No Taxonomy*	*Taxonomy*
09:00	country6	country4	Abroad
09:30	country6	country4	Abroad
10:00	country6	country4	Abroad
10:30	country6	country4	Abroad
11:00	country6	country4	Abroad
11:30	country6	country4	Abroad
12:00	country6	102	Abroad
12:30	country6	102	Abroad
13:00	102	102	102
13:30	102	102	102
14:00	102	102	102

Table 6-23 – An example instance from the test set, high return chance configuration, showing the next-timeslot predictions made with and without full-itemset location generalisation

Timeslot	*Actual Location*	*No Taxonomy*	*Taxonomy*
09:00	country6	country4	Abroad
09:30	country6	country4	Abroad
10:00	country6	country4	Abroad
10:30	country6	country4	Abroad
11:00	country6	country4	Abroad
11:30	country6	country4	Abroad
12:00	country6	102	Abroad
12:30	country6	102	Abroad
13:00	102	102	102
13:30	102	102	102
14:00	102	102	102

Table 6-24 – An example instance from the test set, high return chance configuration, showing the next-day predictions made with and without full-itemset location generalisation

As in Table 6-20, the next-timeslot predictions without the taxonomy vary between predicting an arbitrary country and the occupant's office, although a country happens to be chosen more often in this example. This is again because without the taxonomy, the predictor has no reliable way to select between the countries in the training set, and cannot predict the new countries in the test set anyway. The predictions using the taxonomy are similar to the predictions in Table 6-20, the only difference is that in this case it does not incorrectly predict the occupant is abroad for one timeslot after they return. This is because in this configuration they usually return at this time, so it predicts that they will return based on their historical patterns.

Table 6-24 shows the same example instance along with the next-day predictions with and without the taxonomy. Again there is an additional timetable attribute which indicates that the occupant will be abroad so that the predictor can make predictions regarding the occupant being abroad. The resulting predictions are the same as those in Table 6-23. In the case of the predictions without using the taxonomy, this is for the same reason as in Table 6-20 and Table 6-21, the availability or lack of same day occupant data makes no difference as the predictor, lacking reliable rules regarding the specific countries, must fall back on what are essentially guesses. The predictions with the taxonomy are the same because in both cases the predictor used historical patterns to predict the occupant would return, and as the occupant actually did return in this case, no correction was made based on the same day data available in Table 6-23.

6.6. Summary

This chapter described example cases where generalising the occupant locations in the dataset would allow the algorithm to make extra predictions. It went on to describe two approaches to using these generalisations, a simple consequent-only application for making generalised predictions alongside specific predictions, and a full-itemset application for using general patterns to make the prediction as well as to generalise the prediction itself. The model of a real occupant was modified to incorporate test cases, to evaluate whether the algorithm could learn these types of patterns in practice, and the evaluation demonstrated that where these patterns exist, the addition of the taxonomy does indeed allow OccApriori to make the extra predictions.

Generalised association rules are an established extension to association rule mining. This chapter demonstrated that, with the core OccApriori approach in place, they can be applied to enable new types of predictions of occupant locations to be made. The same is true of other existing extensions of association rules; building on OccApriori, they all offer new potential types of prediction which may be useful depending on the patterns in the occupancy data, increasing the flexibility of OccApriori even beyond what the core algorithm offers. The future work section discusses the potential applications of several of these extensions.

7. Conclusions and Future Work

7.1. Summary

The contributions of this work in terms of occupant prediction can be broken down into three components. Chapter 4 develops an algorithm based on Apriori which features superior accuracy in occupant prediction to the degree that it can match existing approaches at next location prediction, achieving 86% accuracy on the UCC dataset. OccApriori makes no hard distinction between next location prediction, location prediction on the following day, or anything in between (e.g. predicting 3 hours in advance); it is simply a matter of which data is available when predicting. Due to this, for occupants for whom it can make accurate next-location predictions, accurate next-day predictions are available from the same set of rules, achieving 75% accuracy in the case of the UCC dataset. These values compare to 61% accuracy achieved for both next-slot and next-day predictions by unmodified Apriori. Breaking down the modifications made to the algorithm, the results show that depending on the occupant, the individual modifications have varying degrees of effect, but that the complete algorithm is at least as accurate as any other configuration, and more accurate in most cases.

Chapter 5 examines the issue of occupants whose movements do not fit the algorithm's assumption that occupant movements will relate primarily to the time of day. For these occupants, existing approaches based on time-independent sequences of locations have proven effective. This chapter defines a new type of itemset and rule which can encapsulate these sequences, and develops a modification to OccApriori to mine them. The result is a set of rules which achieves accuracy of 76%, compared to 56% achieved by the time-of-day based OccApriori, matching existing sequence based approaches. The sequence based rules can be used along with the time of day based rules to leverage both types of pattern.

Chapter 6 examines the extra types of predictions which can be made by using one of the existing extensions of association rules, generalised association rules, in OccApriori. While OccApriori was concerned only with specific locations to this point, by applying a taxonomy to the dataset it can be made to use generalised locations alongside the specific locations. This allows for both extra predictions of specific locations by relying on general patterns and making higher confidence predictions of an occupant's general location alongside the prediction of their specific location. Evaluation on a synthetic test set based on a real occupant showed a significant increase in accuracy with the application of the taxonomy, demonstrating the algorithm's ability to find and use both types of pattern.

An additional contribution an occupant model based on the previous assumption that occupant movements relate to the time of day. Chapter 3 describes this model and the process of modelling actual occupants based on their historical location data. The chapter evaluates the output of the model compared to the data it was based on under a variety of metrics to show that an accurate reproduction of the original occupants is possible using this model.

7.2. Conclusions

The hypothesis defended in this work is:

> *Association rule mining can be applied to data on the past movements of a building's occupants, generating rules which predict their future locations. Such an algorithm can be modified to take account of the specific attributes and metadata that occupancy data includes in order to reach higher accuracy. Furthermore the algorithm can be extended to learn more general concepts and sequences in the data to extend the range of patterns which are found and can be used for prediction.*

The hypothesis states that association rule mining can be used to predict occupant movements, and with modification, can match or exceed the accuracy of existing approaches, while making a wider variety of predictions. This thesis has shown that OccApriori does in fact match existing approaches for next location prediction. With regard to the variety of predictions it can make, OccApriori is capable of next location prediction, prediction further into the future, and prediction of general locations, and when making those predictions can use time-of-day based location frequency or time-independent location sequences, incorporating whatever data is made available including recent location data, timetable data and instance metadata. The following section discusses further types of predictions that could be made with other extensions to OccApriori.

For the intended application, more efficient heating in buildings, next day prediction of occupant locations can be more important than next location prediction. The few existing approaches which are designed to predict that far into the future use very limited context when making their predictions. OccApriori on the other hand will use any extra data which is provided, the main example examined in this thesis being timetable data. On the UCC dataset, the use of timetable data increased accuracy to 75% for next-day predictions, from the 71% possible without using the timetables.

As illustrated by the Augsburg dataset, for some occupants the assumption that movements relate to time of day simply doesn’t hold. This undoes the predictive power of all current approaches which can predict further into the future, including OccApriori. One option to predict such occupants is to add any additional data which affects their movements, such as a timetable, which OccApriori can then use. On the most difficult to predict occupant in the UCC dataset OccApriori achieved a 23 percentage point gain in accuracy with the addition of timetable data. If there is no such data, if occupants decide on their movements on-the-fly throughout the day or decide on some basis unavailable to the predictor, then prediction of their specific location further into the future may not be possible. However, for the application intended in this thesis, a prediction of the number of hours each occupant is likely to be in each location may suffice, as such predictions would describe the total heat load on the building just as well as more specific predictions. This too is a potential extension of OccApriori which will be discussed in the future work section.

Compared to existing work, the approach presented in this thesis has a greater focus on the flexibility to learn a variety of patterns, to use a variety of data, and to make a variety of predictions. Association rules naturally provide a greater flexibility than the methods used in existing approaches; by building on this foundation and extending it with occupant prediction in mind, this thesis develops an approach which is able to make the different types of predictions offered by various existing approaches, with the same accuracy they achieve, while simultaneously being capable of making many more types of prediction, with significant potential for further extension in the future.

7.3. Future Work

Potential future work on this topic falls mainly into three areas: Further expanding the range of patterns which can be found and predictions which can be made, addressing issues which will arise when applying this approach in a real control scenario, and further development of the occupant model.

A limitation of OccApriori in its present form is that while it predicts based on the occupant's historical movements, it cannot predict based on specific past events. While it may learn an occupant's patterns on Wednesdays and Thursdays, it cannot predict an occupant's location on a particular Thursday based on the preceding Wednesday. Similarly it can't predict, for example, that one occupant won't attend a meeting due to the occupant who called the meeting being absent. There is existing work on extending Apriori to mine patterns between instances. Given the structure OccApriori applies to the occupant location dataset, finding these patterns would allow it to make these extra types of predictions.

Similarly, there is existing work on mining quantitative association rules, in which the values of attributes take on one from a range of values, so that patterns involving interval data can be learned. Applied to OccApriori, this could be used to learn duration of stay in the previously discussed case where there are occupants for whom next day predictions are particularly difficult. By extending OccApriori in this manner, and recording in the instances the duration of stay in each location as well as location in each timeslot, it could learn patterns in the duration each occupant says in each location. If, for example, it can't predict exactly when an occupant will be in their office, it may instead be able to predict how many hours they will spend in their office.

Another alterative type of prediction would be the occupants' locations in time periods. In this case rather than dealing only with specific timeslots, the algorithm would include attributes which record the occupants' location across more general periods such as the morning or afternoon. This could be achieved simply by adding attributes which record where the occupant was for the majority of each time period, then mining the dataset as usual. Specific timeslot and general period predictions could then be offered, with confidence values determining when period predictions should be taken over more specific predictions.

In a similar vein it may be desirable to predict arrival to and departure from the building. Several of the occupants in the UCC dataset displayed a pattern wherein they have a period of increasing chance of arriving in the morning, a similar period of increasing chance to leave in the evening, and a high probability break in the middle of the day for lunch. Aside from these, absences are short and fairly random. This is just one example of a case where predicting the initial arrival and final departure would likely be sufficient. As OccApriori already produces rules which describe the probability of arrival and departure across the day, these events could potentially be predicted in a number of ways, for example predicting arrival time as the earliest timeslot in which the confidence that the occupant is out drops below a given threshold. Such an approach could also be applied to other locations, allowing for predictions of first arrival in any location, similar to (Burbey 2011).

In the evaluation of the timeslot and ordered rules compared to each other on the UCC dataset, where both approaches are successful, the results showed that while they achieve virtually identical overall accuracy, there were variations on specific occupants and at specific times. This reveals that the two approaches are making slightly different predictions, with one or the other being more accurate in different cases. If the algorithm could identify the cases where one or the other rule set is better, it could explicitly select the better type of rule in every case, and achieve higher accuracy with the combined rule set than either rule set achieves alone.

Chapter 5 mentioned the possibility of modifying timetable attributes similarly to how the timeslot attributes were modified, in order to allow timetable entries to be used with sequence-based prediction. By having a parallel set of positions in the ordered itemsets for timetable entries, in the same manner as with the ordinary itemsets, the algorithm could learn sequences which state that an occupant is in their office for two hours and then scheduled to be at a meeting in the third hour. This would then match anywhere in the historical data where the occupant was in their office for two hours, and had a meeting scheduled in the following hour, regardless of what specific time of day was involved, and could be used to predict in similar matching cases. This would require a modification of candidate generation to generate sequences including the parallel timetable entry ordering.

Looking towards a real deployment of the system, the first thing to consider may be re-implemented OccApriori using newer, more efficient algorithms for mining association rules as the starting point. Since the main modifications are defined in terms of modifying the calculation of support and restricting which candidate itemsets are considered acceptable, they are in principle applicable to any of the other algorithms. However, Eclat and its related algorithms achieve their efficiency using pseudo equivalence relations to divide the dataset, while FP-Growth requires the selection of an ordering over the items. Since occupant location data has a natural ordering over the items that the algorithm is most interested in, there may be further efficiency gains to be had using modifications specific to these algorithms.

In a real system the building's tracking system will continue to provide new data on occupants' movements, requiring updates to the previously mined rules to incorporate it. OccApriori as presented would have to retrain from scratch every time, however there is existing work on updating an existing set of association rules based on the content of the new instances, without starting from scratch. This could be integrated into OccApriori, allowing it to integrate new data as it came in.

One issue that OccApriori currently does not address is that occupants' patterns change over time. If it produces a rule set based on months or years of data, it is probable that rules based on more recent data will be better predictors than rules which find their support in older patterns. To address this, methods for weighting rule selection based on the age of their supporting instances should be investigated. A simple approach would be to only train on the most recent n months of data – this guarantees that predictions are made based on recent patterns, however in the process is throws away any useful knowledge that may be contained in the original data. Rather than pure age, it may be worth considering dividing the dataset into time periods, and then selecting rules along those lines. For example, in an academic institution, occupancy patterns will change in and out of term time. If these time periods were explicitly identified in the dataset, term time predictions could be made using rules based on past term time, and similar for out of term time. Several formulations of temporal association rules exist, which address this issue from different angles, and could potentially be applied.

Finally there are some avenues to explore with the occupant model presented in chapter 3. First and foremost, some or all of the process of modelling occupants could be automated. The process of determining the basic patterns for each occupant, arrival and departure probabilities for each time, and probability of all day absence, are simply a step by step application of Bayes theorem to the source data, as described in Chapter 3. Breaking other occupant activities down into explicit tasks is more of an intuitive task when done by hand, but still essentially consists of analysing the frequency of locations across timeslots, thus full automation of the process is certainly a possibility.

In terms of the features this model possesses compared to existing occupant models, the most obvious gap is dependence between tasks on the same day. Currently the only way tasks interact in the model is that a task which is scheduled earlier may take the timeslot a later scheduled task wants, causing the latter to not be scheduled. Existing occupant models include various approaches to having tasks affect each other more directly – for example tasks which model taking a break have a higher probability of occurring as the time since the last break increases. While the assumption that occupant location patterns relate to the time of day primarily is important in this thesis as it underlies OccApriori, there are occupants for whom this doesn’t hold, and so developing the model toward including these extra relationships between tasks would potentially make it more generally applicable.

8. Bibliography

Agrawal, Rakesh, Tomasz Imieliński, and Arun Swami. 1993. "Mining Association Rules between Sets of Items in Large Databases." *ACM SIGMOD Record* 22 (2): 207–16. doi:10.1145/170036.170072.

Agrawal, Rakesh, and Ramakrishnan Srikant. 1994. "Fast Algorithms for Mining Association Rules." *Proc. 20th Int. Conf. Very Large Data Bases, VLDB* 1215: 487–99. doi:10.1.1.40.6757.

Ale, Jm, and Gh Rossi. 2000. "An Approach to Discovering Temporal Association Rules." *Proceedings of the 2000 ACM Symposium on Applied Computing - Volume 1*, no. c: 294–300. doi:10.1145/335603.335770.

Anguita, Davide, Alessandro Ghio, Luca Oneto, Xavier Parra, and JorgeL. Reyes-Ortiz. 2012. "Human Activity Recognition on Smartphones Using a Multiclass Hardware-Friendly Support Vector Machine." *Ambient Assisted Living and Home Care* 7657: 216–23. doi:10.1007/978-3-642-35395-6{\textunderscore }30.

Ashbrook, Daniel, and Thad Starner. 2003. "Using GPS to Learn Significant Locations and Predict Movement across Multiple Users." *Personal and Ubiquitous Computing* 7 (5): 275–86. doi:10.1007/s00779-003-0240-0.

Burbey, Ingrid E. 2011. "Predicting Future Locations and Arrival Times of Individuals Predicting Future Locations and Arrival Times of Individuals." Virginia Polytechnic Institute and State University. https://theses.lib.vt.edu/theses/available/etd-05052011-130912/unrestricted/Burbey_IE_D_2011_2.pdf.

Burbey, Ingrid E, and Thomas L Martin. 2008. "Predicting Future Locations Using Prediction-by-Partial-Match." In *MELT'08*, 1–6. doi:10.1145/1410012.1410014.

Chen, Liming, Chris D Nugent, and Hui Wang. 2012. "A Knowledge-Driven Approach to Activity Recognition in Smart Homes" 24 (6): 961–74.

Cheung, D.W., J. Han, V.T. Ng, and C.Y. Wong. 1996. “Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique.” In *Proceedings of the Twelfth International Conference on Data Engineering*, 106–14. New Orleans, Louisiana: IEEE. doi:10.1109/ICDE.1996.492094.

Corvee, Etienne, Slawomir Bak, and Francois Bremond. 2012. “People Detection and Re-Identification for Multi Surveillance Cameras.” In *International Conference on Computer Vision Theory and Applications*, 82–88. http://hal.archives-ouvertes.fr/hal-00656108/.

Das, Sajal K., Diane J. Cook, Amiya Bhattacharya, Edwin O. Heierman, and Tze Yun Lin. 2002. “The Role of Prediction Algorithms in the MavHome Smart Home Architecture.” *IEEE Wireless Communications* 9 (6): 77–84. doi:10.1109/MWC.2002.1160085.

Eagle, Nathan, and Alex Pentland. 2006. “Reality Mining: Sensing Complex Social Systems.” *Personal and Ubiquitous Computing* 10 (4): 255–68. doi:10.1007/s00779-005-0046-3.

Eagle, Nathan, and Alex Sandy Pentland. 2009. “Eigenbehaviors: Identifying Structure in Routine.” *Behavioral Ecology and Sociobiology* 63 (7). Springer-Verlag: 1057–66. doi:10.1007/s00265-009-0739-0.

EIA. 2015. “Monthly Energy Review February 2015.” U.S. Energy Information Administration. http://www.eia.gov/totalenergy/data/monthly/archive/00351502.pdf.

Eurostat. 2014. “Energy, Transport and Environment Indicators.” http://ec.europa.eu/eurostat/documents/3930297/6613266/KS-DK-14-001-EN-N.pdf/4ec0677e-8fec-4dac-a058-5f2ebd0085e4.

Furey, Eoghan, Kevin Curran, and Paul McKevitt. 2012. “HABITS: A Bayesian Filter Approach to Indoor Tracking and Location.” *International Journal of Bio-Inspired Computation* 4 (2): 79–88. doi:10.1504/IJBIC.2012.047178.

Gellert, Arpad, and Lucian Vintan. 2006. “Person Movement Prediction Using Hidden Markov Models.” *Studies in Informatics and Control* 15 (1): 17–30. http://webspace.ulbsibiu.ro/arpad.gellert/html/SIC_HMM.pdf.

Goldstein, Rhys, Alex Tessier, and Azam Khan. 2010a. “Customizing the Behavior of Interacting Occupants Using Personas.” In *SimBuild 2010: Fourth National Conference of IBPSA-USA, New York City, New York, August 11-13, 2010*, 252–59.

———. 2010b. “Schedule-Calibrated Occupant Behavior Simulation.” *Proceedings of the 2010 Spring Simulation Multiconference on - SpringSim ’10*, 1–8. doi:10.1145/1878537.1878725.

———. 2011. “Space Layout in Occupant Behavior Simulation.” *Proceedings of IBPSA-AIRAH Building Simulation Conference*, 1073–80.

Han, Jiawei, Jian Pei, and Yiwen Yin. 2000. “Mining Frequent Patterns without Candidate Generation.” *ACM SIGMOD Record* 29 (2): 1–12. doi:10.1145/335191.335372.

Henderson, Tristan, David Kotz, and Ilya Abyzov. 2008. “The Changing Usage of a Mature Campus-Wide Wireless Network.” *Computer Networks* 52 (14): 2690–2712. doi:10.1016/j.comnet.2008.05.003.

Kasteren, Tim Van, Athanasios Noulas, Gwenn Englebienne, and Ben Kr. 2008. “Accurate Activity Recognition in a Home Setting.” *: UbiComp’08: Proceedings of the 10th International Conference on Ubiquitous Computing, ACM, Seoul, Korea*, 1–9. doi:10.1145/1409635.1409637.

Kim, Hyunuk, and Ha Yoon Song. 2012. “Formulating Human Mobility Model in a Form of Continuous Time Markov Chain.” In *Procedia Computer Science*, 10:389–96. doi:10.1016/j.procs.2012.06.051.

Kim, Seung Yeon, and Ha Yoon Song. 2014. “Predicting Human Location Based on Human Personality.” *Lecture Notes in Computer Science* 8638: 70–81.

Koehler, Christian, Nikola Banovic, Ian Oakley, Jennifer Mankoff, and Anind K Dey. 2014. “Indoor - ALPS : An Adaptive Indoor Location Prediction System.” In *Proceedings of UbiComp 2014*, 171–81. Seattle, USA.

Kwapisz, Jennifer, Gary Weiss, and Samuel Moore. 2011. “Activity Recognition Using Cell Phone Accelerometers.” *ACM SIGKDD Explorations …* 12 (2): 74–82. doi:10.1145/1964897.1964918.

Li, Shuai, Nan Li, Burcin Becerik-gerber, and Gulben Calis. 2011. “Rfid-Based Occupancy Detection Solution for Optimizing Hvac Energy Consumption.” In *Proceedings of the 28th ISARC*, 587–92.

Li, Yingjiu, Peng Ning, X. Sean Wang, and Sushil Jajodia. 2001. “Discovering Calendar-Based Temporal Association Rules.” In *Proceedings of the International Workshop on Temporal Representation and Reasoning*, 111–18. doi:10.1109/TIME.2001.930706.

Lu, Hongjun, Ling Feng, and Jiawei Han. 2000. “Beyond Intratransaction Association Analysis: Mining Multidimensional Intertransaction Association Rules.” *ACM Transactions on Information Systems* 18 (4).

Manna, Carlo, Damien Fay, Kenneth N. Brown, and Nic Wilson. 2013. “Learning Occupancy in Single Person Offices with Mixtures of Multi-Lag Markov Chains.” *2013 IEEE 25th International Conference on Tools with Artificial Intelligence*, 151–58. doi:10.1109/ICTAI.2013.32.

Mathew, Wesley, Ruben Raposo, and Bruno Martins. 2012. “Predicting Future Locations with Hidden Markov Models.” *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, 911–18.

McNett, Marvin, and Geoffrey M. Voelker. 2003. “Access and Mobility of Wireless PDA Users.” *ACM SIGMOBILE Mobile Computing and Communications Review* 7 (4): 55. doi:10.1145/965732.965744.

Melfi, Ryan, Ben Rosenblum, Bruce Nordman, and Ken Christensen. 2011. “Measuring Building Occupancy Using Existing Network Infrastructure.” In *International Green Computing Conference*, 1–8.

Mozer, Michael C. 1998. “The Neural Network House: An Environment That Adapts to Its Inhabitants.” In *Proceedings of AAAI Spring Symp. Intelligent Environments*, 110–14.

Nguyen, Tuan Anh, and Marco Aiello. 2013. “Energy Intelligent Buildings Based on User Activity: A Survey.” *Energy and Buildings* 56. Elsevier B.V.: 244–57. doi:10.1016/j.enbuild.2012.09.005.

Noulas, Anastasios, Salvatore Scellato, Neal Lathia, and Cecilia Mascolo. 2012. “Mining User Mobility Features for next Place Prediction in Location-Based Services.” *Proceedings - IEEE International Conference on Data Mining, ICDM*, no. April: 1038–43. doi:10.1109/ICDM.2012.113.

Otsason, Veljo, Alex Varshavsky, Anthony Lamarca, and Eyal De Lara. 2007. “Accurate GSM Indoor Localization.” *Pervasive and Mobile Computing* 3 (6): 698–720. doi:10.1007/11551201_9.

Page, Jessen. 2007. “Simulating Occupant Presence and Behaviour in Buildings.” ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE. doi:10.5075/epfl-thesis-3900.

Pérez-Lombard, Luis, José Ortiz, and Christine Pout. 2008. “A Review on Buildings Energy Consumption Information.” *Energy and Buildings* 40 (3): 394–98. doi:10.1016/j.enbuild.2007.03.007.

Petzold, Jan. 2004. “Augsburg Indoor Location Tracking Benchmarks.” https://www.informatik.uni-augsburg.de/de/lehrstuehle/sik/publikationen/reports/2004_09_pet/2004_09_pet_pdf.pdf.

Petzold, Jan, Faruk Bagci, Wolfgang Trumler, and Theo Ungerer. 2006. “Comparison of Different Methods for next Location Prediction.” *Euro-Par Parallel Processing*, 909–18. doi:10.1007/11823285_96.

Petzold, Jan, Faruk Bagci, Wolfgang Trumler, Theo Ungerer, and Lucian Vintan. 2004. “Global State Context Prediction Techniques Applied to a Smart Office

Building.” In *The Communication Networks and Distributed Systems Modeling and Simulation Conference*.

Petzold, Jan, Andreas Pietzowski, Faruk Bagci, Wolfgang Trumler, and Theo Ungerer. 2005. “Prediction of Indoor Movements Using Bayesian Networks.” In *Location and Context Awareness 3479 LNCS*, 211–22. Munich, Germany.

Rainsford, C P, and John F Roddick. 1999. “Adding Temporal Semantics to Association Rules.” *3rd European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD’99)* 1704: 504–9.

Ryan, Conor. 2015. “Occupant Location Prediction in Smart Buildings Using Association Rule Mining - Dataset and Source Code.” http://cnoryan.net/thesis.

Ryan, Conor, and Kenneth N Brown. 2012. “Occupant Location Prediction Using Association Rule Mining.” In *Proceedings of Workshop on AI Problems and Approaches for Intelligent Environments*, 27–32.

———. 2013. “Predicting Occupant Locations Using Association Rule Mining.” In *Research and Development in Intelligent Systems XXX*, 63–77.

Scellato, Salvatore, Mirco Musolesi, Cecilia Mascolo, Vito Latora, and Andrew T. Campbell. 2011. “NextPlace: A Spatio-Temporal Prediction Framework for Pervasive Systems.” *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 6696 LNCS: 152–69. doi:10.1007/978-3-642-21726-5_10.

Scott, James, A.J. Bernheim Brush, John Krumm, Brian Meyers, Michael Hazas, Stephen Hodges, and Nicolas Villar. 2011. “PreHeat: Controlling Home Heating Using Occupancy Prediction.” In *UbiComp’11*. doi:10.1145/2030112.2030151.

Song, Libo, David Kotz, Ravi Jain, and Xiaoning He. 2006. “Evaluating next-Cell Predictors with Extensive Wi-Fi Mobility Data.” *IEEE Transactions on Mobile Computing* 5 (12): 1633–48. doi:10.1109/TMC.2006.185.

Srikant, Ramakrishnan, and Rakesh Agrawal. 1995. “Mining Generalized Association Rules.” In *21st VLDB Conference*, 407–19. doi:10.1016/S0167-739X(97)00019-8.

———. 1996. “Mining Quantitative Association Rules in Large Relational Tables.” *ACM SIGMOD Record* 25 (2): 1–12. doi:10.1145/235968.233311.

Tabak, V. 2008. “User Simulation of Space Utilisation: System for Office Building Usage Simulation.” Technische Universiteit Eindhoven. http://alexandria.tue.nl/extra2/200910371.pdf.

University of Waikato. 2015. “WEKA ARFF Format Specification.” https://weka.wikispaces.com/ARFF.

Vintan, Lucian, and Arpad Gellert. 2004. “Person Movement Prediction Using Neural Networks.” *First Workshop on Modeling and Retrieval of Context* 114 (4): 1–12. https://www.informatik.uni-augsburg.de/de/lehrstuehle/sik/publikationen/papers/2004_mrc_vin/2004_mrc_vin_pdf.pdf.

Voigtmann, Christian, Sian Lun Lau, and Klaus David. 2011. “A Collaborative Context Prediction Technique.” In *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, 1–5. Yokohama, Japan.

Vu, Long, Quang Do, and Klara Nahrstedt. 2010. “Exploiting Joint Wifi / Bluetooth Trace to Predict People Movement.” http://hdl.handle.net/2142/16944.

———. 2011. “Jyotish: A Novel Framework for Constructing Predictive Model of People Movement from Joint Wifi/Bluetooth Trace.” *2011 IEEE International Conference on Pervasive Computing and Communications, PerCom 2011*, 54–62. doi:10.1109/PERCOM.2011.5767595.

Wan, Jie, Michael J. O’Grady, and Gregory M P O’Hare. 2015. “Dynamic Sensor Event Segmentation for Real-Time Activity Recognition in a Smart Home Context.” *Personal and Ubiquitous Computing* 19 (2): 287–301. doi:10.1007/s00779-014-0824-x.

Yun, Jaeseok, and Sang-Shin Lee. 2014. “Human Movement Detection and Identification Using Pyroelectric Infrared Sensors.” *Sensors (Basel, Switzerland)* 14 (5): 8057–81. doi:10.3390/s140508057.

Zaki, Mohammed J. 2000. “Scalable Algorithms for Association Mining.” *IEEE Transactions on Knowledge and Data Engineering* 12 (3): 372–90. doi:10.1109/69.846291.

Zimmermann, Gerhard. 2007. “Modeling and Simulation of Individual User Behavior for Building Performance Predictions.” In *Summer Computer Simulation Conference*, 913–20.