Recognition of Emergent Human Behaviour in a Smart Home: A Data Mining Approach

Sebastian Lühr *, Geoff West, Svetha Venkatesh

*Department of Computing, Curtin University of Technology,
Kent Street, Bentley 6102, Western Australia*

Abstract

Motivated by a growing need for intelligent housing to accommodate aging populations, we propose a novel application of intertransaction association rule (IAR) mining to detect anomalous behaviour in smart home occupants. An efficient mining algorithm that avoids the candidate generation bottleneck limiting the application of current IAR mining algorithms on smart home data sets is detailed. An original visual interface for the exploration of new and changing behaviours distilled from discovered patterns using a new process for finding emergent rules is presented. Finally, we discuss our observations on the emergent behaviours detected in the homes of two real world subjects.

Key words: Intertransaction association rules, Emergent behaviour, Visual data mining, Smart homes

1 Introduction

Current trends suggest that the global population will consist predominately of older people, those aged sixty and over, in as little as fifty years [1]. This demographic shift is expected to lead to an increase in the need for smart homes; intelligent environments that are able to assist their occupants in maintaining independent lifestyles for as long as possible [2]. In our work, we seek to facilitate the creation of homes that are able to detect the presence of new,

---

* Corresponding author. Tel: +61 8 9266 7680. Fax: +61 8 9266 2819.
  Email addresses: luhrs@cs.curtin.edu.au (Sebastian Lühr),
geoff@cs.curtin.edu.au (Geoff West), svetha@cs.curtin.edu.au (Svetha
Venkatesh).

possibly abnormal, behaviour in their occupants and to take action accordingly. Appropriate action may be to query the occupant on the new behaviour, to jog their memory on a task that they were carrying out or even to alert an occupant's relative that their assistance is required. Detecting such deviations from normal behaviour requires models of a person's expected behaviour to compare with incoming data.

One of the more popular approaches favoured by researchers for modelling human behaviours has been the application of graphical models. Training and inferencing using these models is, however, computationally expensive and generally limited to applications in which it is reasonable to assume that human activity can be represented as sequences of asynchronous activities or events. The task of modelling human behaviour as precise sequences of events is made difficult, however, by our tendency to interleave our activities and to adjust our behaviour when we are interrupted. We propose a novel application of Intertransaction Association Rule (IAR) mining [3] as a means of tackling this issue. IARs are implication rules that allow us to capture the associative, non-sequential, relationship of events observed within a home while retaining some of the higher level temporal context in which these events occur.

The research contributions of this work are threefold. The first addresses the issue of mining the IARs present in the sensor event logs from a home. Such environments generate data sets that contain frequent occurrences of a large number of events over relatively short periods. This poses a problem for the current EH-Apriori [3] and FITI [4] algorithms for IAR mining as they rely on a computationally costly candidate-generation-then-test approach for rule discovery. This technique requires that k passes over a database are made to retrieve the set of frequent rules up to length k. Each pass over the data requires the generation of candidates – the set of all *possibly* frequent associations given those found in a previous pass. The scalability of such algorithms is hence limited due to the computational complexity of generating and testing the frequency of a combinatorial number of candidates. The number of candidates generated at each pass k of a worst case scenario given n database items and an intertransaction window of length w is $\sum_{r=0}^{k-1} \left[\binom{n}{k} \binom{nw}{r} \right]$. We propose the application of our Extended Frequent Pattern Tree (EFP-Tree) [5] as a means of tackling this issue. The EFP-Tree is an extension of the Frequent Pattern Tree (FP-Tree) [6] for IAR mining. It uses a divide and conquer approach to avoid candidate generation and requires only three passes over a database.

The second issue we face is how to gain insight into a person's behaviour so as to detect abnormality from an overwhelming number of rules that the mining process is likely to uncover. We introduce the use of emergent IARs as a novel means of finding patterns of behaviour that are of interest to us. Emergent IARs are those rules that display significant growth from one data set to another. Their presence may indicate abnormality evident as either

2

a previously unseen pattern of events or unusually frequent occurrences of behaviour that would otherwise be considered normal. Emergent IARs offer a convenient means of identifying changes that would otherwise be difficult to discern through manual inspection of the rule sets. For example, the real world event logs from a single week of data can produce around 7,200 patterns which can be distilled down to approximately 150 emergent rules that are likely to be of interest to us. The same data set mined with a slightly higher support threshold will retrieve 46 emergent rules from circa 2,700 discovered patterns.

We further build on this with the introduction of a novel graphical interface that maps the emergent rules back onto the original data. Visualisation on the original data space is important in our application as it enables users to see implications of patterns that are not immediately apparent from uncovered rules. We have used our system for the discovery and analysis of emergent IARs to investigate the emergent human behaviour from a real world smart home data set. Experimental results show that we are able to detect emergent IARs that indicate both previously unseen behaviours and sensor aberrations.

2 Related Work

Several dedicated smart home projects are currently underway elsewhere. The MavHome [7] project and the Adaptive House [8] both seek to develop home automation that does not require users to manually program the components in their homes. Researchers at the Aware Home [9] are exploring issues such as context aware computing, human-computer-interaction, and occupant tracking and identification with an application to caring for the elderly. A recently opened project is the PlaceLab [10], a live in laboratory facility for multidisciplinary research into areas such as human behaviour, and the use of technology to simplify home control and promote healthy living. Our work is similarly motivated in that we seek to enable the detection of new and changing behaviours in a smart home's occupant. IAR mining allows us to do this by finding the significant state-change sensor event associations and their temporal relationships from large amounts of data generated in a home.

Intertransaction association rule (IAR) mining is an extension of association rule mining, or "market basket" analysis, first introduced in [11]. Association mining is concerned with the discovery of sets of items that frequently occur together within the records of a transactional database. Association mining is, essentially, a counting exercise; the problem being tackled is how to efficiently find, possibly large, subsets of frequently occurring associations within a combinatorial search space.

IAR mining extends the discovery of association rules to include relationships

that span transactions in one or more domain specific dimensions. IAR mining is different to frequent episode and sequential pattern mining. Episode mining [12] seeks to find frequent partial ordering of items within a sliding window but does not consider the time interval relationships among those items. Sequential patterns [13] are concerned with finding frequent sequences where the ordering of items or events is important yet the temporal relationships are not considered. These two areas contrast to IAR mining where, assuming that the intertransaction dimensional attribute is temporal, the associative relationship among items is considered between time intervals but the ordering of items within the intervals is unimportant. IAR mining was selected for this work due to the interleaved nature of human behaviour as it allows us to capture the non-sequential relationships between observed activities while retaining some of the temporal aspect of such relationships. Opportunity exists for sequential pattern mining to be applied when the precise ordering of activities is desired.

IAR mining was first proposed in [14] with the Apriori [15] inspired EH-Apriori algorithm. EH-Apriori is a stepwise algorithm that makes numerous passes over a data set to test, in each pass, the frequency of the k-length candidate itemsets generated from frequent $(k-1)$-length associations discovered in the previous pass. Candidate generation may, however, result in a prohibitively large number of candidate itemsets being produced. This limits the scalability of such algorithms. The First Intra Then Inter (FITI) algorithm [4] is an improved candidate generation algorithm that first finds the complete set of intratransaction associations and transforms the data set into lookup structures that aid the subsequent mining of intertransaction itemsets. The advantage of FITI lies in its ability to discard unnecessary data early on in the mining process and to use the discovered intratransactions to more efficiently guide the generation of intertransaction candidates.

Our emergent behaviours work is similar to Emerging Patterns (EPs) [17] and Jumping Emerging Patterns [16] which seek to find sets of items that show significant growth from one data set to another. EPs are used in trend detection and have application to classification problems due to their ability to identify the discriminative features among classes of data. EPs differ from IARs in that they do not consider the associative relationship among items.

Visual data mining tools for association rule exploration aim to provide users with a convenient way by which to navigate through large sets of discovered associations and the relationships they entail. We limit our focus on previous work in visual data mining to the problem of understanding association rules; interested readers may wish to refer to [18] for a wider survey of research into the area of visual data mining.

Two dimensional matrices showing the one-to-one relationship of rules was one of the first techniques used to visualise associations. The antecedent and

4

consequent of the rules were used to label the rows and the columns of the matrix while the support and confidence of the rules were visually represented through icons on the grid. Commercial tools such as MineSet [19] and Quest [20] were the first to make use of this method.

A more versatile matrix representation that shows the relationships between items and itemsets was introduced in [21]. The matrix columns were used to represent associations and rows the individual items. Item membership in the rules was depicted by binary coloured bricks placed onto the grid, the brick colour distinguishing between a rule's antecedent and consequent items. The grid was displayed in three dimensions with the support and confidence of the associations charted alongside the x-axis. The two dimensional matrix theme was again used in [22]. Here, associations of the same items were grouped into cells and placed onto the grid in descending support and descending confidence order. Interesting changes in the rules were marked on the grid and on an accompanying hierarchical view of the associations.

None of these methods map discovered rules back onto the original data space.

3 Intertransaction Association Rules

Consider the set of all items $I = \left\{ a^1 a^2 \ldots a^i \ldots a^M \right\}$ occurring in a database $DB = \langle T_1 T_2 \ldots T_N \rangle$ of transactions T_i $(1 \leq i \leq N)$ such that $T_i(x) \in I$ $\forall$ items x in T_i. For any transaction T_i the items are said to form the set $S_{T_i} = \left\{ a^i_{T_i} \ldots a^k_{T_i} \right\}$. For the case of a single intertransaction dimension attribute, an intertransaction sliding window of size w transactions is passed over the transactions in DB to extract the extended transaction items such that the extended transaction at T_i is $E_{T_i} = \left\{ S_{T_i}, S_{T_i+1} \ldots S_{T_i+w} \right\}$ and the set of all possible extended transaction items is $E = \left\{ a^1_0 a^2_0 \ldots a^i_d \ldots a^M_w \right\}$. The mining problem reduces to the traditional intratransaction case when $w = 0$, that is, when only intratransaction items are included in an extended transaction itemset. In this work the extended transaction items $\left\{ a^1_0 \ldots a^M_0 \right\}$ are referred to as intraitems and the extended transaction items $\left\{ a^1_1 \ldots a^M_w \right\}$ are referred to as interitems. The superscript notation is dropped when the value of an item is known.

Intertransaction association rules are implication rules such that $X \Rightarrow Y$ with the properties $X \subseteq E, Y \subseteq E, \exists a^i_0 \in X, \exists a^i_d \in Y$ where $d > 0$ and $X \cap Y = \emptyset$ [4]. The support and confidence measures of an itemset are calculated as $\frac{|T_{xy}|}{N}$ and $\frac{|T_{xy}|}{|T_x|}$ respectively where $|T_{xy}|$ is the number of extended transactions containing all items in $X \cup Y$, $|T_x|$ is the number of extended transactions containing all items in X and N is the number of extended transactions. The

support of a rule is a simple measure of the rule frequency; the most frequent rules being likely to reflect common knowledge about a domain while rules with lower support may highlight insights that are little known and may even be unexpected. The confidence measure, in contrast, is a statistical measure of the accuracy of a rule's implication. That is, how confident the implication that a transaction containing the items in X will also contain the items in Y.

As an example, the rule A_0, B_0, C_1, $D_3 \Rightarrow E_3$ implies that if we encounter A and B in the current transaction interval, C in the next and D three transactions from now then E will also occur three transactions from now. If A, B, C, D and E map to the event descriptors "kitchen sink cold water on", "kitchen sink cold water off", "dishwasher open", "dishwasher closed" and "dishwasher on" respectively then one possible interpretation of the rule may be that it is normal to open the dishwasher, close it again and then turn the machine on shortly after having used the cold water faucet in the kitchen sink, presumably to rinse the dishes.

4 An Extended Frequent Pattern Tree for IAR Mining

The proposed Extended Frequent Pattern Tree (EFP-Tree) is a tree structure of descending frequency ordered intraitem nodes with zero or one interitem Frequent Pattern Tree (FP-Tree) subtrees where the frequency ordering of the interitems is conditioned on the intratransaction item parent [5]. Each node contains an item ID mapping to a codebook of item descriptors, a frequency counter, a link to its parent node, links to zero or more children and a link to the next node in the tree of the same item ID. Interitem nodes also carry the dimensional offset of the item in relation to its intratransaction parent.

Nodes are placed into the tree such that the entire set of frequent items for an arbitrary intertransaction can be restored by traversing the tree. The ordering of nodes into descending frequency increases the likelihood of items placed into the tree sharing common nodes, creating a compact representation of the database transactions that captures the associative relationship of the transaction items.

4.1 EFP-Tree Construction

Three passes over a database are required to build the tree structure. As in the FP-Tree, the frequency of single items is gathered in an initial pass over the database to build the set of frequent single intraitems and the set of frequent interitems given a minimum support threshold α. The intraitems are

Table 1
Extended transactions retrieved from a database using a window of size $w = 5$.

Time	Extended transaction items
1	B_0 C_0 A_0 B_1 B_2 C_2 A_2 C_4 E_4 B_5 E_5
2	B_0 B_1 C_1 A_1 C_3 E_3 B_4 E_4 B_5 A_5
3	B_0 C_0 A_0 C_2 E_2 B_3 E_3 B_4 A_4
5	C_0 E_0 B_1 E_1 B_2 A_2 C_4 B_5 C_5 D_5
6	B_0 E_0 B_1 A_1 C_3 B_4 C_4 D_4 B_5 C_5 A_5
7	B_0 A_0 C_2 B_3 C_3 D_3 B_4 C_4 A_4
9	C_0 B_1 C_1 D_1 B_2 C_2 A_2
10	B_0 C_0 D_0 B_1 C_1 A_1
11	B_0 C_0 A_0

ordered by descending frequency to become the item lookup header table for the intraitem tree. The frequent intraitems from Table 1 are B_0:7, C_0:6 and A_0:4 with frequency counts of 7, 6 and 4 respectively. The frequent interitems are A_1:3, A_2:3, B_1:6, B_2:3, B_4:4, B_5:4, C_1:3, C_2:4, C_3:3 and C_4:4.

The intraitem FP-Tree is built and the conditional frequencies of the interitems are found in the second pass. The intraitems for each transaction T_i are first filtered to remove items not present in the known frequent intraitem and interitem sets from the first pass and are sorted in order of descending frequency. The ordered item list is recursively inserted into the tree such that at each level l in the tree the child node with the ID of the lth item in the ordered array is traversed and its frequency count is incremented. Children nodes that do not exist will be created prior to traversal and kept in codebook ID order so that binary search can be used when traversing the tree. The linked list of nodes of same item ID that originates from the root node header table is updated whenever a new node is created. The frequency of the interitems relative to T_i are incremented in the final intraitem node that is traversed. The ordered lists of frequent intraitems are stored for use in the third pass. The root node is said to be at level $l = 0$.

The third and final pass over the database builds the interitem sub-trees in the EFP-Tree structure. At each transaction T_i the cached ordered list of frequent intraitems are used to traverse the intraitem tree and locate the intraitem node that will become the root node of the interitem subtree. The extended items within the intertransaction sliding window at T_i are filtered to remove known globally infrequent interitems and the remaining items are sorted in order of local descending frequency given the intraitem parent. The ordered interitems are then recursively inserted into the interitem subtree as before.

Given a minimum support threshold of $\alpha = 3$ and the example intertransactions in Table 1, the example tree in Figure 1 is built as follows. First, the intraitems B_0, C_0 and A_0 are added to the root node of an empty tree such that all nodes are recursively created. The frequency of item B_0 is incremented by the next transaction. The items B_0, C_0 and A_0 are added again, the existing nodes are traversed and their frequency counts are each incremented. The next two transactions see the node C_0 created as the second child of the root node and an increment to the count of B_0. B_0 and A_0 are then added such that A_0 becomes the second child of B_0 and the count of B_0 is incremented once more. The count of the nodes representing the intratransaction associations C_0 and B_0, C_0 are incremented by the next two transactions. Finally, the counts of the items in the path B_0, C_0, A_0 are once again incremented.

In our example the known frequent items found in the first pass of some database allow us to reduce the extended transaction items from Table 1 at time 1 to B_0, C_0, A_0, B_1, A_2, B_2, C_2 and C_4 and at time 3 to B_0, C_0, A_0, C_2 and B_4. The interitems for both extended transactions will be inserted at the node identified by following the path B_0, C_0, A_0 through the tree in Figure 1. The insertion of the ordered list of interitems from the first extended transaction will create the children interitem nodes C_2, B_1, A_2, B_2, C_4 and B_5. The child node C_2 will be incremented and a new node for item B_4 inserted as a child of C_2 when the extended transaction at time 3 is ordered and added. This process is repeated for the interitems in the remaining extended transactions.

The complete EFP-Tree of the example database is shown in Figure 1.

4.2   EFP-Tree Mining

As in the FP-Tree, retrieval of association rules from the EFP-Tree is made possible by the pattern growth property [6]. Pattern growth uses a divide and conquer approach that recursively builds the entire set of frequent associations by constructing trees conditioned on known frequent base rules and taking the dot product of the frequent items in the conditional tree and the conditional base itemset to produce new rules. These new rules then become the conditional base for the next set of conditional trees to be mined. The Frequent Pattern Growth (FP-Growth) algorithm from the FP-Tree differs to our Extended FP-Growth (EFP-Growth) algorithm in that the latter must consider intertransaction relationship inheritance along the intraitem nodes.

Starting with an EFP-Tree T and an empty conditional base, or rule suffix, EFP-Growth iterates over the set of intraitems I in T to build a conditional tree T_c conditioned on I for each frequent I. At each recursion, I is prepended to the conditional base to generate, or grow, a new association rule and build
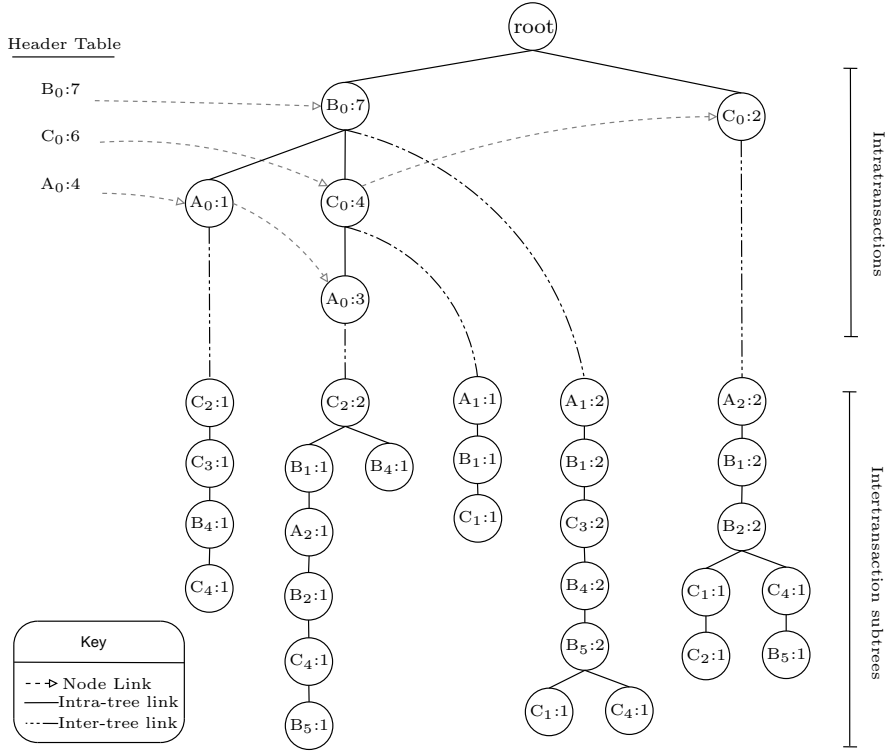
Fig. 1. The Extended FP-Tree for the example database with $w = 5$ and $\alpha = 3$. Subscript numbers represent the dimensional offset, in this case the item time, relative to the intratransaction items while colon delineated numbers depict the node frequency. The header tables of the intertransaction item subtrees have been omitted for clarity.

the conditional tree for the next recursive step. No candidate generation is necessary as the frequency of the items is stored in the tree structure and all generated rules are guaranteed to be frequent.

Two types of conditional trees are used in EFP-Growth; a conditional EFP-Tree T_c used for finding the related intraitems and interitems that can be used to extend the present intraitem rule suffix and a FP-Tree T_e of the interitems inherited by the conditional base. This latter tree is used to find the interitem associations for a given intraitem rule suffix and is required as not all interitems inherited by the conditional base may be included in T_c.

Given a tree T, the conditional tree T_c conditioned on some I is found by collecting the set of extended transactions formed through the union of the prefix path and the inherited interitems for each node in T whose item ID is I and whose immediate parent is an intraitem node. The prefix path for any given node is the set of its parent nodes and corresponding frequencies as stored in the EFP-Tree. All nodes in T of item ID I are found by following the linked list of same item ID nodes, the head of which is stored in the intraitem header table of T. The extended transactions are then used to build T_c as
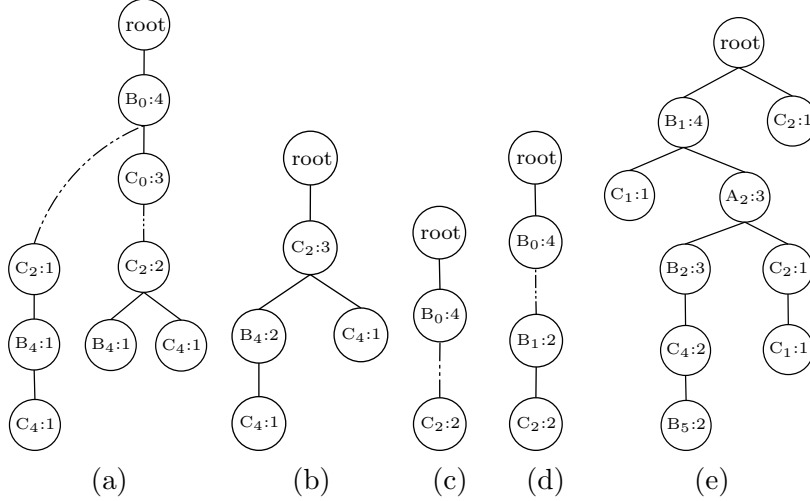
9

Fig. 2. The conditional EFP-Tree tree T_c and conditional interitem FP-Tree T_e for (a) $T_c|A_0$, (b) $T_e|A_0$ and $T_e|B_0A_0$, (c) $T_c|C_0A_0$, (d) $T_c|C_0$ and (e) $T_e|C_0$ when the EFP-Tree in Figure 1 is mined with a minimum support threshold of $\alpha = 2$.

described in Section 4.1.

The conditional interitem tree T_e for a given conditional base is found by constructing an FP-Tree of the interitem transactions inherited by the conditional base rule and using FP-Growth to mine the resulting tree. Taking the dot product of the conditional base and the set of interitem associations returned by FP-Growth produces the entire set of intertransaction associations related to the conditional base. This process continues recursively until no more conditional trees are built or until only a single intraitem path exists in T_c. If T_c contains a single intraitem path we can avoid recursion and find the complete set of rules given the conditional base by finding the dot product of the intraitem combinations in T_c and the interitem associations returned when calling FP-Growth on the interitem subtree.

The example EFP-Tree T in Figure 1 was built with a minimum support level of $\alpha = 3$. We will now set the support threshold to $\alpha = 2$ in order to demonstrate the mining process in finer detail than is possible at the original support setting.

Conditioning T on A_0, we find the conditional prefix paths $\langle B_0:1 \rangle$ and $\langle B_0:3\ C_0:3 \rangle$. The interitems $\langle C_2:1\ C_3:1\ B_4:1\ C_4:1 \rangle$ are related to $\langle B_0:1 \rangle$ and the interitems $\langle C_2:1\ B_1:1\ A_2:1\ B_2:1\ C_4:1\ C_5:1 \rangle$ and $\langle C_2:1\ B_4:1 \rangle$ are found for $\langle B_0:3\ C_0:3 \rangle$. The conditional tree $T_c|A_0$ with $\alpha = 2$ is shown as Figure 2(a) and the interitem FP-Tree $T_e|A_0$ as Figure 2(b). The set of IARs associated with the conditional base A_0 is found by taking the dot product of A_0 and the interitem associations returned by FP-Growth from the tree in Figure 2(b). The resulting rules are $A_0 \Rightarrow C_2:3$, $A_0 \Rightarrow B_4:2$, $A_0 \Rightarrow C_4:2$, $A_0C_2 \Rightarrow B_4:2$ and $A_0C_2 \Rightarrow C_4:2$.

10

Recursing into the conditional tree in Figure 2(a), EFP-Growth grows the conditional base by finding the least frequent intraitem whose support meets the support threshold for mining. This item, C_0, is prepended to A_0 to create the new conditional base C_0A_0 and the intratransaction rule $C_0 \Rightarrow A_0$:3. The single interitem prefix path $\langle B_0$:4$\rangle$ and its inherited interitems $\langle C_2$:1 B_4:1$\rangle$ and $\langle C_2$:1 C_4:1$\rangle$ form to create the single intraitem path conditional tree shown in Figure 2(c). The inherited items of C_0A_0 are used to build a single node FP-Tree T_e containing C_2:2 resulting in the generation of a single rule $C_0A_0 \Rightarrow C_2$:2 when the dot product of the interitem associations found in $T_e|C_0A_0$ and the conditional base is found. Recursively mining the tree in Figure 2(c) generates the rules $B_0C_0 \Rightarrow A_0$:2 and $B_0C_0A_0 \Rightarrow C_2$:2.

The mining now returns to $T_c|A_0$ to create the next conditional base B_0A_0 and generate its respective rule $B_0 \Rightarrow A_0$:4. No prefix path of B_0A_0 exists in Figure 2(a) so no conditional tree $T_c|B_0A_0$ needs to be built. The mining of the interitem FP-Tree $T_e|B_0A_0$, the same as for $T_e|A_0$ in Figure 2(b), generates the rules $B_0A_0 \Rightarrow C_2$:3, $B_0A_0 \Rightarrow B_4$:2, $B_0A_0 \Rightarrow C_4$:2, $B_0A_0C_2 \Rightarrow B_4$:2 and $B_0A_0C_2 \Rightarrow C_4$:2.

Upon return from a recursive call EFP-Growth will update the immediate parent of each node whose item ID is I such that the interitems are inherited and ready for conditioning on the next frequent item. It is for this reason that the mining algorithm grows rules by recursing into trees conditioned on the least frequent intraitems first.

Returning to the original tree T in Figure 1 the recursive mining technique will be applied in turn to the conditional bases C_0 and B_0. The conditional trees $T_c|C_0$ and $T_e|C_0$ are given in Figure 2(d) and Figure 2(e) respectively.

The EFP-Growth algorithm is detailed below.

For each item a_i in header table of N from least to most frequent such that support$(a_i) \geq \alpha$ **do**
    Find the conditional prefix path and the extended items for a_i, propagate
    the interitems of each occurrence of a_i to its parent and
    build the conditional EFP-Tree T_c

    If T_c contains a single intratransaction path P such that no non-leaf node
    contains an intertransaction subtree **then**
        $T_c \leftarrow T_c$ with P removed
        $singlePathRules \leftarrow$ all combinations of intratransaction nodes in P
        $singlePathRules \leftarrow singlePathRules \times$ rules returned by call to
                FP-Growth(leaf node of P, null) as in [6]
    end
    $returnedRules \leftarrow$ call EFP-Growth(T_c, α, $maxSpan$)
    Build the intertransaction item FP-Tree T_e using the extended items from a_i

    $interRules \leftarrow$ call FP-Growth(T_e, null) as in [6]
    $ruleSet \leftarrow returnedRules \cup interRules \cup singlePathRules$
    For each rule R in $ruleSet$ **do**
       add a_i to R with support(R) = min(support(R), support(a_i))
    end
    add a_i to $ruleSet$ with support = frequency of a_i in N
    $minedRules \leftarrow minedRules \cup ruleSet$
end
Return $minedRules$

4.3 Algorithm Performance

Experimentation on both synthetic and real world data has shown that the EFP-Growth algorithm is computationally an order of magnitude more efficient than the existing algorithms for IAR mining when dealing with dense data sets such as those generated by a smart home environment.

Both synthetic data, employed to model the best and worst case scenarios for association rule mining, and real world data sets, to indicate the practical application of the mining algorithms, were used to compare the computational performance and peak memory requirements of EFP-Growth with FITI. These performance measures are important as they empirically demonstrate the scalability of the algorithms on input data of varying characteristics. For each data set the ability of the algorithms to scale with respect to the length of the intertransaction window and a decreasing minimum support threshold is observed. Tung *et al.* [4] have previously shown FITI to be computationally more efficient than EH-Apriori and so the latter algorithm is not considered.

The real world data used [23] are event logs from an array of state-change sensors installed in the homes of two subjects, a thirty year old working professional and an eighty year old retiree, over a period of sixteen days. The sensors, 77 in the first subject's home and 84 in the second, were fitted to a variety of appliances, containers and furniture to log the times of use. These events were discretised for mining into transactions of five minute intervals to produce 658 transactions for the first subject and 748 transactions for the second. Unique sensor IDs were stripped from the event logs to reduce the sensor information to only include the sensor state and its room and object context. For example, multiple sensors installed on the doors of a cabinet are reduced to `Kitchen/Cabinet true` and `Kitchen/Cabinet false` events. The event codebooks contained 76 and 80 entries for the first and second subjects respectively.

Two synthetic data sets representing sparse and dense data were generated using the method described in [3,4], the same method used to compare the

Table 2
Parameters used in the generation of the synthetic data sets.

Parameter	Sparse	Dense
Number of intratransactions	500	200
Size of the intertransaction source pool	50	200
Average length of intratransactions	5	25
Maximum length of intratransactions	10	50
Average length of intertransactions	5	8
Maximum length of intertransactions	10	20
Maximum number of unique items in the data	500	100
Maximum interval span of intertransactions	4	6

EH-Apriori algorithm to FITI. Table 2 lists the parameters used to create the data sets used in the experimentation.

IAR mining in FITI occurs only after the set of frequent intratransaction associations have been found. Knowledge of these rules is then used to transform the database into a lookup structure that aids intertransaction mining. For this experiment, FITI was implemented using the FP-Tree and FP-Growth algorithm for the initial mining phase. This was necessary in order for a fair comparison of the algorithms to be made, it having previously been shown that FP-Growth performs an order of magnitude faster than the Apriori algorithm used in the original FITI implementation [24].

The algorithms were implemented in Ruby, an interpreted language, and benchmarked on a 3.2GHz Pentium 4 running FreeBSD.

4.3.1 Limitations of the Benchmark Environment

Before discussing results, execution time irregularities should be noted in the EFP-Growth curve in Figure 5(a) at $w = 2$ and at $w = 7$. Irregularities also appear for FITI in Figure 4(a) at the 1.1% support threshold, in Figure 4(b) at 1% support and in Figure 5(b) at $w = 4$. Profiling revealed that these irregularities are caused by an erratic garbage collector in the Ruby interpreter. When triggered, the garbage collector will spend a disproportionally long time seeking memory to free. This behaviour was consistently reproduced on the FreeBSD 5.3, Linux 2.6 and Windows XP platforms using the 1.6 and 1.8 Ruby interpreter series. This behaviour is independent of the algorithm being run and was found present in the implementations of the EH-Apriori, FITI, FP-Growth and EFP-Growth algorithms. The garbage collector behaved normally for all other points on the graphs and hence the irregularities found do not

invalidate the results obtained.

4.3.2 Minimum Support Threshold

For the first set of results, the support threshold was gradually lowered from
1.6% to 0.6% with a fixed intertransaction window size of 4 and from 13% to
8% with a fixed window size of 6 for the synthetic sparse and synthetic dense
data sets respectively.

The plot in Figure 3(a) shows FITI outperforming EFP-Growth until the 1%
support threshold is reached. FITI has an advantage at the higher support
thresholds as it is able to remove unnecessary data prior to counting. This
benefit is reduced as the number of candidates generated by FITI increases
when the support threshold is lowered. EFP-Growth outperforms FITI at the
lower support thresholds and especially at the 0.6% level where an explosion
in the number of rules results in an exponential increase in the number of
candidate itemsets generated and counted by FITI. We begin to see an order
of magnitude difference in the algorithm execution times on the dense data
in Figure 3(b). Although FITI marginally outperforms EFP-Growth at the
12.5% and 13% support threshold, FITI is overwhelmed by the number of
candidate itemsets generated at the lower thresholds.

The execution times in Figure 4(a) for the first real world data set compares
the algorithms' performance as the support threshold is lowered from 1.5% to
0.7%. An order of magnitude difference in the running times exists at the lower
support levels due to the large number of discovered rules and a high number
of FITI generated candidates. Figure 4(b) depicts the execution time of EFP-
Growth and FITI on the second real world data set over a support threshold
range of 0.4% to 1.3%. EFP-Growth is able to maintain its computational
advantage over FITI at all support levels.

4.3.3 Intertransaction Sliding Window Size

The intertransaction window size in Figure 5 is incremented from $w = 0$ to
$w = 10$ for the sparse data and $w = 8$ for the dense data with fixed minimum
support thresholds of 1% and 10% respectively. Figure 5(a) shows that EFP-
Growth has only a marginal computational advantage on the sparse data set,
the number of rules found and the number of candidates generated by FITI
remaining relatively low. FITI has similar execution times to EFP-Growth on
the dense data in Figure 5(b) until the intertransaction size $w = 5$. The curves
begin to diverge at this point, the FITI execution time eventually being an
order of magnitude greater than EFP-Growth at $w = 8$.

Performance on the real world data is compared by incrementing the sliding
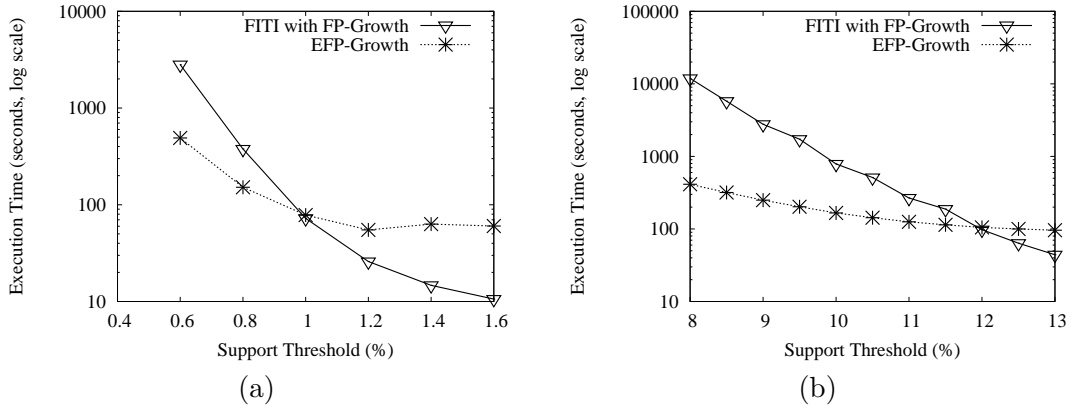
Fig. 3. The execution time (a) for the synthetic sparse data set with the intertransaction window size fixed at $w = 4$ and the execution time (b) for the synthetic dense data set with $w = 6$ as the minimum support threshold is adjusted.
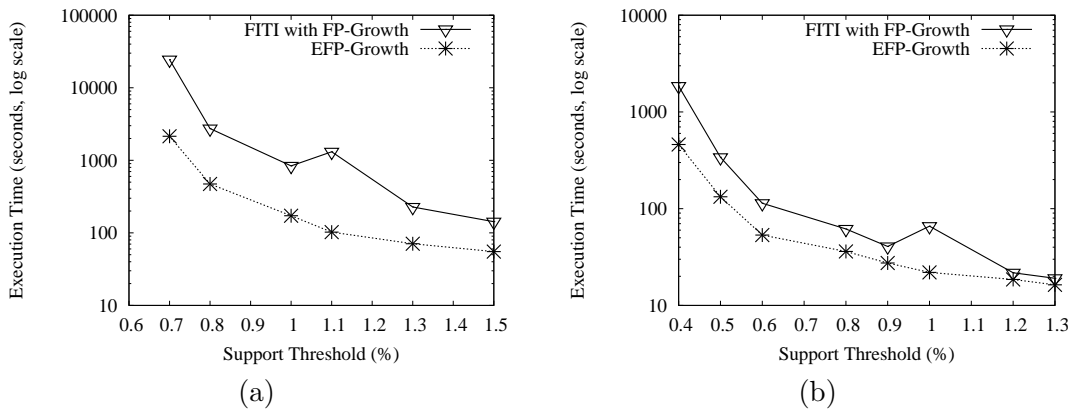

Fig. 4. The execution time (a) for the working professional subject and the execution time (b) for the retiree subject as the minimum support threshold is adjusted. The intertransaction window size is fixed at $w = 6$ with an interval size of 300 seconds for both data sets.

window size up to $w = 12$ to find associations spanning up to an hour. The support thresholds are fixed at 1% and 0.4% for the first and second data sets.

EFP-Growth outpaced FITI computationally in both real world data sets in Figure 6(a) and Figure 6(b). A sudden increase in the execution time of the FITI algorithm is seen in Figure 6(b) when the sliding window size is increased from $w = 2$ to $w = 3$. This increase is caused by a sudden large jump in the number of rules being discovered.

4.3.4   Conclusion

It has been shown that EFP-Growth is a more scalable algorithm for IAR mining than FITI. The candidate generation and testing approach of the latter
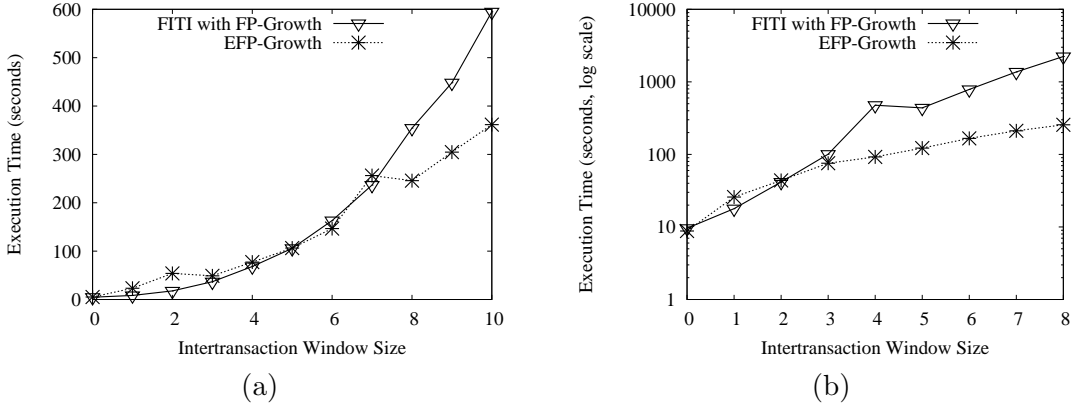
Fig. 5. The execution time (a) for the synthetic sparse data set and the execution time (b) for the synthetic dense data set as the intertransaction window size is increased. The minimum support level was fixed at 1% and 10% for the sparse and dense data sets respectively.
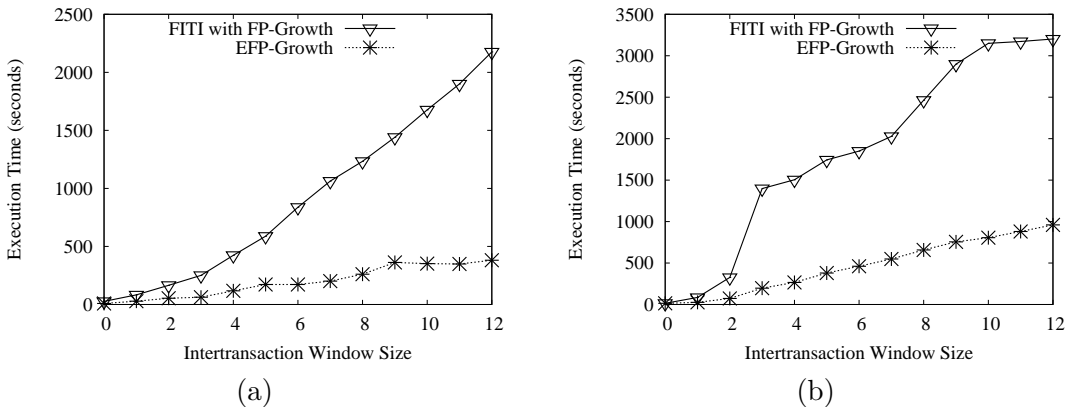

Fig. 6. The execution time (a) for the working professional subject with a minimum support threshold $\alpha = 1\%$ and the execution time (b) or the retiree subject with $\alpha = 0.4\%$ as the intertransaction window size is increased.

was found to limit its scalability when the number of discovered itemsets, and hence the number of candidates being generated and counted, become too large. In contrast, EFP-Growth was shown to be able to scale well in such cases, especially when applied to the synthetic dense data and at the lower support levels of the synthetic sparse data set. EFP-Growth was also shown to outperform FITI on the real world data sets, making it a more suitable algorithm for IAR mining on the sensor event logs from an intelligent environment than the EH-Apriori and FITI algorithms.

$$growth\left(r|DB_H, DB_N\right) = \begin{cases} 0, & \text{if } sup\left(r|DB_H\right) = 0 \text{ and } sup\left(r|DB_N\right) = 0 \\ \infty, & \text{if } sup\left(r|DB_H\right) = 0 \text{ and } sup\left(r|DB_N\right) \neq 0 \\ \frac{sup\left(r|DB_N\right)}{sup\left(r|DB_H\right)} & \text{otherwise} \end{cases}$$

$$(1)$$

5   Discovering Emergent Behaviour

The discovery of emergent intertransaction associations [25] seeks to find those rules that display significant growth in a database of new observations DB_N over a historical data set DB_H. We say that an association rule r is emergent when its growth, the ratio of its support in DB_N to its support in DB_H, is greater than or equal to some threshold δ and a minimum support threshold α on DB_N has been met. We have adopted Equation 1 [16] as the growth function for this work.

In our application, it is reasonable to expect that our historical data set will be larger and contain a much greater number of associations than will be present in the new data set. It is impractical, therefore, to find the complete set of association rules for both DB_H and DB_N and then compare these sets to find the emergent association rules when the rules in DB_N are expected to be a subset of those in DB_H. We instead employ the EFP-Tree as an intermediate representation of both DB_H and DB_N. Item constraints [26] are applied to the mining of DB_H so that only a desired subset of rules are extracted. This provides us with the ability to "query" DB_H on the historical frequency of only those associations found to be frequent in DB_N.

Only the set of minimal emergent rules are sought. This is to prevent the discovery process from returning an overwhelming number of rules, the majority of which would be unlikely to offer any valuable information not already present in the minimal set. Finding the set of minimal rules was achieved by ordering the mined associations by ascending rule length and discarding those rules known to contain emergent association subsets using previously detected emergent rules. An emergent association q is said to be non-minimal when $\exists\{r, t\}$ such that $r\left(t\right) \subset q\left(t = 0\right)$ where $r\left(t\right) = \left\{r_1^{d_1+t}, r_2^{d_2+t}, \ldots r_i^{d_i+t}, \ldots r_Z^{d_Z+t}\right\}$ is a known emergent rule whose intertransaction offsets $\{d_1, d_2, \ldots d_i, \ldots d_Z\}$ have been incremented t intervals. For example, if r is an emergent rule $C_0, D_2 \Rightarrow E_2$ and q is the rule $A_0, B_0, C_1, D_3 \Rightarrow E_3$ then q is also known to be emergent because $r\left(1\right) \subset q$ holds true.

Although we applied filtering as a post process, the EFP-Tree offers an opportunity to move the filtering process into the tree mining algorithm so as to guide the mining of both DB_H and DB_N.
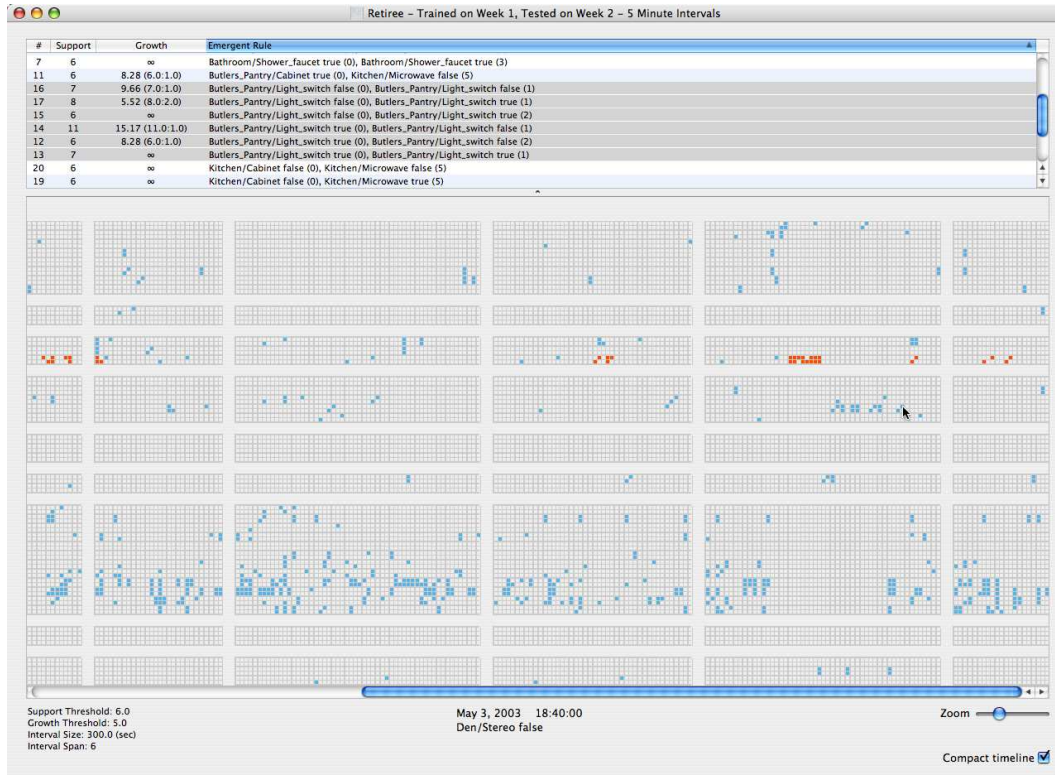
17

Fig. 7. The visual interface highlighting the presence of selected rules (top) on the event time line. The horizontal axis represents the date and time of the intervals while the vertical axis represents the sensor events. The date and time and the name of the sensor event that the mouse cursor is pointing at is being displayed.

6 Visual Analysis of Emergent Behaviour

Previous work [19–22] has attempted to tackle the issue of visual exploration of complete sets of mined association rules. In contrast, the visualisation approach we propose seeks to limit users to consider only those IARs found to be emergent. These rules are likely to be a minority of the rules discovered and hence they will be difficult to discern or gain meaning from using any form of visualisation that covers all rules. A way of visualising the effect of the emergent rules in isolation and in the context of the original data is necessary. This is done by mapping the emergent IARs back onto the original data in order to establish the original context in which the rules occur. Doing so allows users to see both the date and time that the emergent behaviour occurred and which other sensors were triggered around this time. Although it was not available for the data set used in the experimentation, a corresponding video of each instance of the unusual behaviour could also be retrieved and shown to the user.

A screen capture of the proposed interface with a sample data set loaded is shown in Figure 7. The emergent rules are displayed in a table in the top

18

portion of the screen. The rules are selectable and can be ordered by their support, growth measures or by rule similarity. The main display element is a grid that displays a compact view of the sensors triggered in each transaction interval. The horizontal axis represent the date and time of the transaction intervals while the vertical axis shows the sensor events grouped by their room location. Triggered events are indicated with blue coloured cells while cells that correspond to currently selected rules are highlighted in red.

Users are able to select whether they wish to view a compact representation of the time line, where only intervals in which events were recorded are shown, or the full grid. The thickness of the vertical lines between cells is used to indicate jumps in time in the compact view, the cells of contiguous intervals being delineated by hairlines while the cells of non-contiguous intervals are separated by thick lines. All cells in the full time line view are contiguous and hence are delineated using hairlines. Figure 8(a) presents a magnified view of the grid in which the effect of the compact view is more visible. In contrast, Figure 8(b) shows a magnified view of a small subset of the same time period displayed using the expanded view. The start of a new day is indicated with a gap in the time line in both the compact and full view modes. Users are able to view the exact date and time of an interval by positioning the mouse cursor over a cell, the time being indicated in a text box horizontally centered underneath the grid. If the cell being pointed at represents a triggered sensor event then the name of that event is also displayed in the text box. The screen capture in Figure 7, for example, shows the descriptive name of the sensor event, and the date and time of the interval in which it occurs, of the cell that the mouse cursor is presently pointed at. In this example, the event represents the switching off of the stereo in the retiree subject's den.

Cells are similarly partitioned on the vertical axis by the sensor events that they represent. Sensors are grouped by their location in the home with gaps on the grid delineating rooms. Sensor events are further grouped by their textual description such that all events related to one particular sensor, currently limited to "on" and "off" events in our application, appear in contiguous cells whose boundary is drawn using hairlines. Boundaries between the sensors in a room are delineated using thick horizontal lines. Partitioning the sensor events in this way provides users with a clear view of the time and location of events being triggered and their relation to other sensors and rooms.

The lower portion of the interface shows meta information about the data being displayed. The minimum support and growth thresholds, the interval size and the intertransaction window size used to mine the rules are shown here. A text box horizontally centered underneath the cell grid displays information about a cell whenever the user positions the mouse cursor over it. This area also contains a slider that allows the magnification of the grid to be adjusted and a check box with which to toggle the compact time line representation.
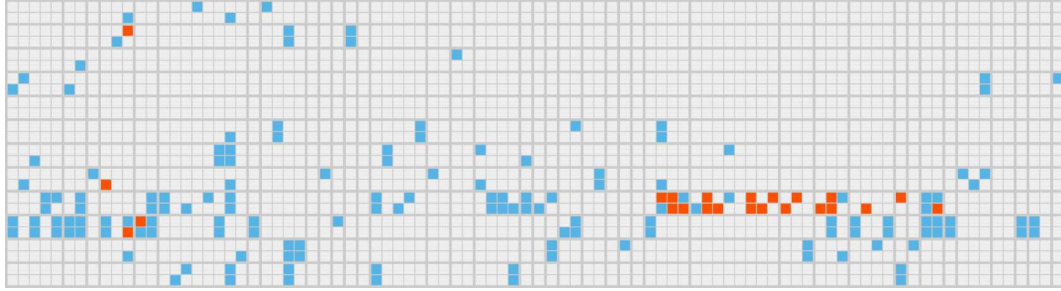
(a)
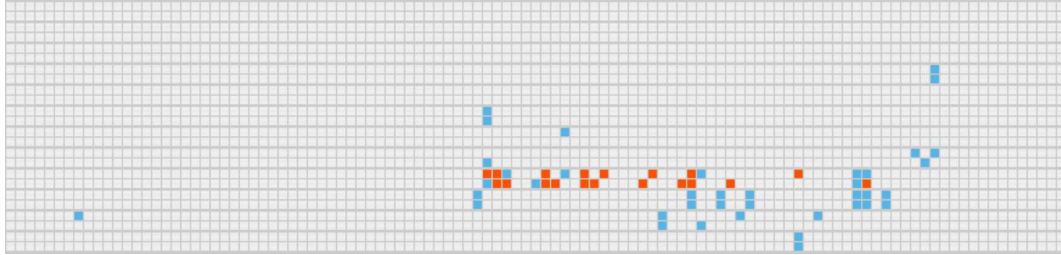
(b)

Fig. 8. Magnified view of (a) the compact time line showing the sensor events within a single room over a period of one day and (b) the expanded view showing approximately three hours from the same room and period.

7 Experimentation and Results

We have applied our method of emergent IAR discovery on the real world data described in Section 4.3. The event logs were mined and analysed with transactions of one, two, three, four and five minute intervals

The sixteen days worth of event logs from each subject were divided into two halves, each half containing eight days of events. Each half was in turn used as the historical database DB_{H} to find the emergent rules present in the other. We found that our visual data mining tool helped us attain greater insight into the emergent behaviours discovered and in a shorter period than we were able to obtain through manual inspection of the rules and the log files.

Our examination of the data sets revealed several instances of unusual behaviour that we have interpreted to be caused by sensor malfunctions. These conclusions are substantiated by Tapia *et al.* [23] who note that sensors were observed to fail or were dislodged during the data gathering period. Such analysis is further reinforced through inspection of the sensor event logs which revealed several instances of rapid and repetitive toggling of a sensor's state.

Mining using the four and five minute intervals generally provided more stable, noise free, results than the one, two and three minute intervals due to the limited amount of training data available. The benefit of reduced noise was

diminished somewhat with the increase in the granularity of the results. The cause of the vast majority of emergent behaviours, whether valid or the result of a lack of historical data, was apparent regardless of the interval size chosen.

7.1 Working Professional Data Set

IARs from the first subject were mined using a raw minimum support threshold of $\alpha = 8$ for both sets of data. This support level was chosen to balance the quality and the quantity of the discovered associations, providing us with an ample number of rules for analysis while remaining resistant to noise. Association rules discovered in DB_N were classified emergent if a minimum growth of $\delta = 5$ was measured.

7.1.1 First Half

A large group of emergent IARs related to a flurry of activity in the subject's kitchen were discovered in the third day from the first half of the data. The discovered rules describe the temporal relationships of the kitchen drawer, cabinet and refrigerator spanning transaction intervals over a thirty minute period. These patterns do not, however, appear to signify any anomalous behaviour given the overall context of kitchen related activity being carried out.

An abnormality does appear during this kitchen activity, however, in the form of a repeated opening and closing of the kitchen door. The regularity, frequency and the length of time over which this occurs is unusual. The lack of activity outside of the kitchen reinforces our suspicion that a glitch with the door sensor has been discovered here. The phenomenon is repeated again over a shorter period early the next day. A magnified view of the interface showing this behaviour using an interval length of two minutes is shown in Figure 9(a).

7.1.2 Second Half

A significant portion of the emergent rules found in the second half again relate to activity in the kitchen. Unlike the kitchen related rules from the first half, these rules describe emergent relationships that are new combinations of kitchen sensors whose temporal relationship span only a few intervals. This suggests that the kitchen activities in this half are more involved and that not enough historical data is available to account for these patterns.

A lack of historical data also explains the emergent IARs describing the opening and closing of the bathroom sink faucets one to two intervals after their previous use. This behaviour is rarely seen in the first half where we expect
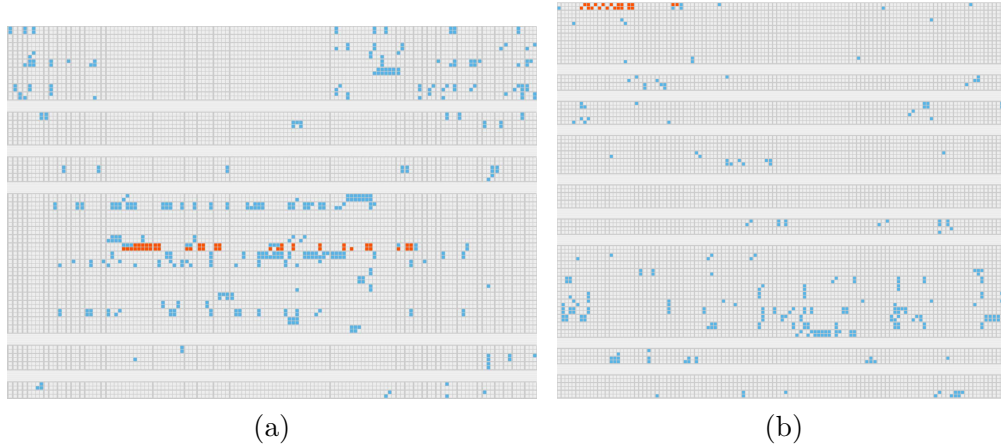
(a)   (b)

Fig. 9. Visualisation of (a) the abnormal kitchen door activity using a two minute transaction interval on the third day of the first half of the working professional data and (b) the abnormal bathroom activity using a two minute transaction interval on the first day of the first half of the retiree data. Triggered events are coloured in blue while events matching the emergent IARs are coloured in red

these events to occur within single transaction intervals and without being repeated for some time.

A mundane change in the subject's behaviour is apparent, however, with the use of the bathroom door being significantly more frequent in this half; the bathroom door is rarely used in the other half.

7.2   Retiree Data Set

The minimum support threshold for mining the first half of the data was reduced to $\alpha = 6$ in order to provide us with a sufficient number of rules for analysis not otherwise available at higher support levels. The support level for the second half was again set to $\alpha = 8$. A minimum growth measurement of $\delta = 5$ was again required for both.

7.2.1   First Half

A noticeable change in the retiree subject's use of the bathroom is evident on the first morning in the first half. Here we see a forty-five minute period in which the bathroom door is repeatedly being opened and closed. The television in the living room is operated and the microwave and the refrigerator in the kitchen are also accessed during this time. In this context, the emergent IARs suggest that the subject is making frequent short visits to the bathroom, a behaviour that we believe is unusual and would warrant further investigation. This emergent behaviour is visible in the event grid in Figure 9(b).

22

Emergent rules were discovered that describe the kitchen door being repeatedly opened and closed throughout a twenty minute period on the second day. The behaviour we expect to see is this door being left in one state for longer periods of time or to see it being both opened and closed within a single time interval. This behaviour appears to be innocuous given the activity seen in the rest of the home.

The subject appears to be carrying out more kitchen related activity in this half than in the next. This explains the discovery of a host of emergent IARs involving kitchen sensor events. These patterns do not indicate any unusual behaviour, however. Rather, they reinforce the notion that we do not possess enough historical data to account for the wide variety of patterns present in normal kitchen behaviour.

A behaviour that is unusual, however, is the frequent use of the television throughout this half. This behaviour is unusual given that the subject rarely turns on the television in the historical data set.

7.2.2 Second Half

A prominent example of abnormality is apparent on the first day in the second half of the data. Here, emergent IARs highlight a malfunction in the hot and cold shower faucet sensors as evident through the regular and repeated triggering of faucet events over a forty-five minute interval. The theory that this behaviour is due to a hardware glitch is further reinforced by the conspicuous absence of any further shower faucet events for the remainder of the week.

On the seventh day the subject seems to repeatedly enter and leave the butler's pantry over a forty minute period. The light in the pantry is switched on for a period of up to two minutes before being switched off again, a pattern that is repeated every one or two intervals. The use of the pantry in this way is unusual; normal behaviour for this person is to have the pantry light on for a few minutes at a time and then to not return again for several hours. This new behaviour may not be abnormal.

The last emergent behaviour worth noting is an increased use of the drawer in the subject's home office. The drawer is used several times in short succession on days one and three of this half yet it is only used once in the previous half.

8   Conclusion

This paper has introduced a novel data mining approach for the detection of new and changing behaviour in people living in a smart home. We presented the EFP-Tree, an extension of the FP-Tree for intertransaction mining, and the corresponding EFP-Growth algorithm for the efficient retrieval of frequent IAR patterns from the tree structure. The EFP-Tree was shown to be an order of magnitude computationally more efficient at mining IARs from both synthetic data and from event data collected in a smart home environment than the existing EH-Apriori [3] and FITI [4] algorithms.

A method for finding IAR patterns exhibiting significant growth from one data set to another was also introduced. These emergent IARs allow us to detect the presence of new, possibly anomalous, behaviour and unusually frequent occurrences of behaviour that would otherwise be considered normal from amongst the mined rules. Emergent IARs are important in our work as they provide a means of distilling many discovered rules down to those most likely to be of interest to us. A novel visual data mining tool that maps the discovered emergent IARs back onto the original data space was discussed. We found that the use of this interface allowed us to obtain clearer insight into the emergent rules than through manual inspection of the event logs.

Observations made on the emergent IARs discovered in the sensor event logs from the homes of a working professional and an elderly retiree volunteer were discussed. We demonstrated that emergent behaviours due to both sensor aberrations and real changes in behaviour could be detected. At least one behaviour, documented in Section 7.2.1, warranting concern for the well-being of the elderly subject was discovered.

References

[1] United Nations, World population ageing: 1950-2050 (2002).

[2] E. Mynatt, I. Essa, W. Rogers, Increasing the opportunities for aging in place, in: Proc. 2000 Conf. Universal Usability, 2000, pp. 65–71.

[3] H. Lu, L. Feng, J. Han, Beyond intra-transaction association analysis: Mining multi-dimensional inter-transaction association rules, ACM Trans. Information Systems 18 (4) (2000) 423–454.

[4] A. K. H. Tung, H. Lu, J. Han, L. Feng, Efficient mining of intertransaction association rules, IEEE Knowledge and Data Engineering 15 (1) (2003) 43–56.

[5] S. Lühr, G. West, S. Venkatesh, An extended frequent pattern tree for

intertransaction association rule mining, Tech. rep., Department of Computing, Curtin University of Technology (March 2005).

[6] J. Han, J. Pei, Y. Yin, R. Mao, Mining frequent patterns without candidate generation: A frequent-pattern tree approach, Data Mining and Knowledge Discovery 8 (1) (2004) 53–87.

[7] D. J. Cook, M. Youngblood, E. O. Heierman, III, K. Gopalratnam, S. Rao, A. Litvin, F. Khawaja, MavHome: An agent-based smart home, in: IEEE Int'l Conf. Pervasive Computing and Communications, 2003, pp. 521–524.

[8] M. C. Mozer, Lessons from an adaptive house, in: D. Cook, S. Das (Eds.), Smart Environments: Technology, Protocols and Applications, 2004, pp. 273–294.

[9] C. D. Kidd, R. Orr, G. D. Abowd, C. G. Atkeson, I. A. Essa, B. MacIntyre, E. D. Mynatt, T. Starner, W. Newstetter, The aware home: A living laboratory for ubiquitous computing research, in: Int'l Workshop Cooperative Buildings. Integrating Information, Organizations, and Architecture, 1999, pp. 191–198.

[10] S. S. Intille, K. Larson, J. S. Beaudin, J. Nawyn, E. M. Tapia, P. Kaushik, A living laboratory for the design and evaluation of ubiquitous computing technologies, in: Conf. Human Factors in Computing Systems, 2005, pp. 1941–1944.

[11] R. Agrawal, T. Imielinski, A. Swami, Mining association rules between sets of items in large databases, in: Proc. 1993 ACM SIGMOD Int'l Conf. Management of Data, 1993, pp. 207–216.

[12] H. Mannila, H. Toivonen, A. I. Verkamo, Discovering frequent episodes in sequences, in: Proc. Int'l Conf. Knowledge Discovery and Data Mining, Montréal, Canada, 1995, pp. 210–215.

[13] R. Agrawal, R. Srikant, Mining sequential patterns, in: Proc. Eleventh Int'l Conf. Data Engineering, 1995, pp. 3–14.

[14] H. Lu, J. Han, L. Feng, Stock movement prediction and n-dimensional inter-transaction association rules, in: Proc. 1998 SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, 1998, pp. 12:1–12:7.

[15] R. Agrawal, R. Srikant, Fast algorithms for mining association rules, in: Proc. 20th Int'l Conf. Very Large Data Bases, 1994, pp. 487–499.

[16] G. Dong, J. Li, Efficient mining of emerging patterns: Discovering trends and differences, in: ACM Int'l Conf. Knowledge Discovery and Data Mining, 1999, pp. 43–52.

[17] G. Dong, J. Li, X. Zhang, Discovering jumping emerging patterns and experiments on real datasets, in: Int'l Database Conf. Heterogeneous and Internet Databases, 1999, pp. 155–168.

[18] M. C. F. de Oliveira, H. Levkowitz, From visual data exploration to visual data mining: A survey, Visualization and Computer Graphics 9 (3) (2003) 378–394.

[19] C. Brunk, J. Kelly, R. Kohavi, MineSet: An integrated system for data mining, in: Proc. Int'l Conf. Knowledge Discovery and Data Mining, 1997, pp. 135–138.

[20] R. Agrawal, M. Mehia, J. Shafer, R. Srikant, A. Arning, T. Bollinger, The quest data mining system, in: Proc. Second Int'l Conf. Knowledge Discovery and Data Mining, 1996, pp. 244–249.

[21] P. C. Wong, P. Whitney, J. Thomas, Visualizing association rules for text mining, in: IEEE Sym. Information Visualization, 1999, pp. 120–123.

[22] K.-H. Ong, K.-L. Ong, W.-K. Ng, E.-P. Lim, CrystalClear: Active visualization of association rules, in: ICDM 2002 Workshop on Active Mining, 2002.

[23] E. M. Tapia, S. S. Intille, K. Larson, Activity recognition in the home using simple and ubiquitous sensors, in: Int'l Conf. Pervasive Computing, 2004, pp. 158–175.

[24] J. Han, J. Pei, Y. Yin, Mining frequent patterns without candidate generation, in: Proc. 2000 ACM SIGMOD Int'l Conf. Management of Data, 2000, pp. 1–12.

[25] S. Lühr, S. Venkatesh, G. West, Emergent intertransaction association rules for abnormality detection in intelligent environments, in: Int'l Conf. Intelligent Sensors, Sensor Networks and Information Processing, 2005, pp. 343–347.

[26] R. Srikant, Q. Vu, R. Agrawal, Mining association rules with item constraints, in: Proc. Int'l Conf. Knowledge Discovery and Data Mining, 1997, pp. 67–73.