# Mining Implicit Correlations Between Distributed IoT Devices from Embedded Databases

Márcio Alencar, *Master's Student, PPGI/UFAM,* Raimundo Barreto, *Associate Professor, PPGI/UFAM,* and Richard Pazzi, *Associate Professor, UOIT*

**Abstract**—Identifying user behavior patterns is an expected feature for Internet of Things devices. Finding these patterns and using them for decision-making can provide ease, comfort, practicality and autonomy when executing daily activities. Although knowledge extraction is common in centralized intelligent environments, performing it in a decentralized architecture is still a relevant computational challenge considering tight storage and processing constraints of IoT devices. This paper describes a method of mining implicit correlations among the actions available from distributed IoT devices through embedded associative analysis. Based on a variance of the support, confidence and lift metrics the method identifies the most relevant correlations between a pair of actions from different devices and suggests the integration between them through HTTP requests to the user. Experimental results show that, on average, the most relevant rules for both architectures are the same in 99.75% of cases. In addition, the proposed method identified relevant correlations that were not identified by the centralized architecture. The proposed method emphasizes that analyzing device state change patterns is an efficient approach for providing a highly integrated and intelligent IoT platform, while avoiding the single point of failure problem.

**Index Terms**—Internet of Things, Knowledge discovery, Embedded Software, Associative Analysis, Decentralized data mining.

## 1 Introduction

The ability to provide intelligence and autonomy to devices in Internet of Things (IoT) relies mainly on the need to identify patterns and implicit correlations from them. The architecture in which the intelligent environments are implemented is directly related to the pattern recognition technique adopted.

According to [1] and [2], the state-of-the-art in pattern recognition is Deep Neural Networks, which have a high computational cost. Such approach demands the use of devices with a considerable amount of memory and a processor that can handle a massive volume of operations, therefore, increasing the cost of these devices. Also, adopting devices that can handle an embedded deep learning process would be a waste of resources in a decentralized approach (e.g.: a raspberry pi for each object).

To circumvent these barriers, the proposed method performs an associative knowledge extraction in a decentralized analysis, where each device allows all mechanisms to store, process, analyze and share all data needed to identify implicit correlations between its own actions and the actions of other devices in the same environment.

Driven by the challenge to obtain globally interesting correlations based on local associative analysis, the proposed method, called MAKE (eMbedded Associative Knowledge Extraction), presents an approach that identifies the strongest correlations between a pair of devices and uses them to create an intelligent and integrated environment that models the user behavior. These correlations could be used as triggers to synchronize the states of multiple devices in a chain reaction and adapt the environment during the user's activities, providing ease and comfort in their daily activities. Furthermore, this approach introduces an embedded storage and data processing approach to save space without loosing the capability to extract knowledge using the constrained resources from distributed IoT devices.

This paper is organized as follows: First, the associative analysis theory is discussed. After this theory is exposed, some related works are reviewed to understand how associative analysis has been used to perform extractions in a decentralized environment with small datasets and to correlate distributed devices. Next, the proposed method is presented that describes the architecture and its components followed by experimental results. Finally, this paper concludes by exposing the achievements and limitations of the proposed method, as well as future works.

## 2 Associative Analysis

An associative analysis is the discovery of rules that exhibit attribute-value conditions that often occur together in a given dataset [1], [3], [4], [5]. This dataset is formed by transactions which are composed of multiple items. A rule could be expressed as $A \Rightarrow B$ $[metrics]$ where $A$ defines the premise of rule (antecedent), $B$ is the conclusion of rule (consequent) and *metrics* are values that quantify the quality of this inference.

A simple example of this associative analysis could be applied in a supermarket database where the rules correlate groups of items that are often shown together. An instance of this correlation (rule) could be expressed as an inference that suggests that, usually (metrics), the consumers who buy coffee (antecedent) also buy milk (consequent). The relevance of this inference is given by the metrics that

- *PPGI/UFAM: Graduate Program in Informatics at Federal University of Amazonas. 6200, General Rodrigo Octávio, Coroado I, 69080-900, Manaus-AM, BR. E-mails: {macalencar,rbarreto} at icomp.ufam.edu.br*
- *UOIT: University of Ontario Institute of Technology. 2000 Simcoe Street North, L1H 7K4, Oshawa-ON, CA. E-mail: richard.pazzi at uoit.ca*

*Manuscript received ****; revised ****

indicate how frequent (support), how dependent (lift) and how reliable (confidence) this inference is.

## 2.1 Metrics

Typically, association rules are considered interesting if they satisfy the minimum *support* thresholds, which is an interesting measure of a rule that reflects its usefulness. Also there is another metric, callled *confidence*, which defines the assurance of discovered rules. Assuming $support(A)$ as the probability ($P$) of an item $A$ appears in a given dataset ($D$) of $|D|$ registers and $support(B) = P(B)$ as well, then, a rule that correlates $A$ and $B$ has both metrics formally represented as:

$$support(A \Rightarrow B) = P(A \cup B) = \frac{freq(AB)}{|D|} \quad (1)$$

$$confidence(A \Rightarrow B) = \frac{P(A \cup B)}{P(A)} = \frac{freq(AB)}{freq(A)} \quad (2)$$

The Equation (1) represents the probability that both items ($A$ and $B$) appear together in $D$, and the Equation (2) represents how frequent $B$ is, given all transactions that $A$ is in. Beyond these, a third metric called *lift* is defined by:

$$lift(A \Rightarrow B) = \frac{P(A \cup B)}{P(A)P(B)} = \frac{freq(AB) \times |D|}{freq(A) \times freq(B)} \quad (3)$$

This metric states that the occurrences of itemsets $A$ and $B$ are independent if $P(A \cup B)$ is $P(A)P(B)$, otherwise $A$ and $B$ are dependent and correlated.

Different from *confidence* and *support*, which have their range of values from 0 to 1 (or 0% to 100%), *lift* defines that values less than 1 indicate that $A$ and $B$ are inversely correlated ($A \wedge \neg B$); if *lift* equals 1 then A and B are independent and are not correlated; or if the *lift* were greater than 1, A and B are directly correlated.

Typically, an associative analysis aims to identify the largest and most frequent itemset in dataset. For that, there are several associative analysis algorithms such as Apriori [6], FP-Growth [7], Constraints-Based Mining [8]. Different from those, the proposed method aims to identify the strongest correlations between two items, in other words, a rule that strongly correlates the antecedent and the consequent formed by 1-itemset each.

## 3 Related Work

This section presents studies that use associative analysis techniques to correlate distributed devices and/or extract interesting correlations from small datasets. It also considers studies that perform extractions by grouping registers to reduce the datasets as a mechanism to decrease the processing workload during the associative analysis.

A distributed approach called $DPmining$, proposed by [9], extracts knowledge from several devices in a wireless sensor network. This approach adopts a clusters of devices where the head (cluster's controller) manages the database partitions which are processed by one or more remote devices in the same cluster. Although such approach considers a decentralized analysis, the amount of data collected by the head demands a considerable amount of data storage to deal with all cluster's data.

The method proposed in [4], called TEREDA, identifies temporal features and relations from sensor activities. This approach is able to extract the activities' order, their usual start time and duration by clustering the activities' start time and correlating them to their duration. Further, using an associative mining technique (FP-Growth), it correlates the current activity to the next most probable activity based on its timing interval. This approach needs to centralize the dataset and previously identify the specific activities in such a way that it is possible to perform that operation.

The research conducted by [5] has lead them to a method that predicts a devices' states, based on associative analysis, to identify if there is a failure probability. It correlates the current status to the predicted status, using FP-Growth, by mining unusual behaviour based on the device's changes. This approach correlates the states of the same device rather than correlating different devices with each other.

The work presented by [10] assessed the sensitivity and reliability of rules obtained from a small database that correlate clinical diagnoses and laboratory evaluations from a mammography dataset. The rules were evaluated by Effron's Bootstrap method, which consists of creating several models from a database sample and comparing those models. The smaller the number of different models, the more reliable the rule. Even though it is an interesting technique, creating many models and comparing them demands a huge amount of processing and memory space.

The goal of the studies presented by [11] and [12] was to identify the usage patterns between devices in an intelligent environment. Through aggregation techniques and associative rule mining, it was possible to conclude that, although each device has a unique usage pattern, some devices have similar characteristics. It was possible to classify the users activity by clustering those similarities. However, this method cannot predict the number of clusters created and some interesting correlations may be lost if the clusters belong to slightly different time intervals.

The work by [13] explores the associative rule mining in a small database to identify correlations between unemployment rate and socioeconomic factors in southwestern Norway. To validate those correlations, the Formal Concept Analysis Technique was applied, which is a principle-based method of deriving a conceptual hierarchy, or formal ontology, from a collection of objects and their properties. When both techniques were combined it allowed for the identification of correlation between the attributes of unemployed citizens and the higher unemployment rate regions. This approach analyzed a small but dense dataset with a high computational cost, which is not recommended for constrained devices.

The proposal of [14] reduces the impacts on the searching cost in associative analysis by reducing the frequent itemsets and removing duplicate records. Moreover, another optimization proposed by the authors is data compression where, considering binary values to represent the absence (i.e: bit 0) and presence (i.e: bit 1) of a certain item, a transaction can be represented by a string containing a list of items present and its index in a checklist. Although this compression mechanism reduces the storage space needed to represent the data, this method considers a centralized transactions dataset, which could be a problem in decentralized IoT environments.

Seeking to explore the correlation between groups of sensors and actuators in a building, the method proposed by [15] proves their hypothesis that there is a correlation between the sensor's state changes from a specific environment in a given space of time. The proposed grouping of variables, called Weighted Transitive Clustering, is based on timing correlations which exist between the sensors' state changes that monitor the same environment. Thus, the grouped variables will have a strong correlation during their state changes, otherwise variables belonging to other environments will not be correlated. Therefore, it is possible to infer which variables are related to the same space and can be grouped together based on the rules of relationships. This method demands access to all dataset information. Besides that, considering a timing correlation, it may create an invalid set of correlations assuming devices from different environments have a similar pattern of activity.

The study conducted by [16] identified possible attacks at a water treatment plant through the generation of invariants. The experiment consisted of identifying correlated invariants among the 51 sensors with the Apriori Associative Rule algorithm. Such study provides evidence of associative rule analysis efficiency to correlate sensor states in a cyber-physical environment. The result shows that it was able to identify those invariant generations during attack to the system. Besides finding correlations between sensors from cyber-physical during an attack, it was necessary to analyze a great amount of data to identify them as a threat.

The evidences exposed by [4], [5], [11], [12], [15], [16] show that it is possible to extract interesting information from a device's change patterns instead of usage patterns. This slight difference reduces the amount of data that must be stored and analyzed, which satisfies an essential requirement in constrained environments. Also, the clustering data process, presented by [4], [9], [10], [15] and [5], certainly increases the computational cost due to the pre-processing, but optimizes the associative analysis reducing the number of candidates. The approaches presented by [13] and [14] expose more interesting evidence by extracting relevant correlations in a small but dense dataset.

This paper tries to combine the best features from the related works by performing an iterative and segmented analysis to reduce the amount of data handled during the associative analysis. It also registers all data in discrete embedded dataset, avoiding clustering operations in raw data. Reinforcing this storage method, a probabilistic analysis is used to determine which actions have a high frequency in each slot and, based on that, create a small transactions dataset (max itemset = 2) to perform the associative analysis, reducing the space and time to extract knowledge.

## 4 Proposed Method

The Embedded Associative Knowledge Extraction (MAKE) is a collaborative mechanism in which each device should compare its own pattern to the other devices' patterns to identify the strongest correlations between the device's actions and the remote devices' actions.

The device must perform those embedded comparisons periodically and the generated rules are only applied to this specific device, that is, each device will have its own set of correlation rules. These rules allow the synchronization of a remote device's state (rule's consequent) based on a single input action in the source device (rule's antecedent) if these actions satisfy both device's patterns at the current time slot.

To better understand the proposed method, it is necessary to provide a comprehension of devices expected behaviors (Section 4.1), data storage (Section 4.2) and the mining process (Section 4.3).

### 4.1 States and Actions

In the MAKE method each device controls a single object from the intelligent environment and has two well-defined sets: $S = \{s_1, s_2, \ldots, s_i\}$ as a finite set of $i$ items which represent the possible states that a device has, and $A = \{a_1, a_2, \ldots, a_i\}$ as set of $i$ actions available to transition between the $S$ states where each state represents a device's interaction with the environment (e.g.: "lamp on" and "lamp off") and each action represents the physical or logical stimulus to change the devices state (e.g.: "turn on" and "turn off").

TABLE 1
Illustration of a Embedded dataset ( $M_{ij}$ ), Transformed Dataset( $M_{ij}$ ') and the pattern of actions.

| | SLOTS (T)→ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\mathbf{M_{ij}}$ | **TURN ON** | 2 | 1 | 4 | 9 | 25 | 13 | 4 | 0 | 1 | 4 |
| | **TURN OFF** | 5 | 3 | 4 | 6 | 34 | 17 | 2 | 0 | 0 | 5 |
| $\mathbf{M_{ij}}$ ' | **TURN ON** | 1 | 0 | 2 | **3.16** | 4.64 | 3.7 | **2** | 0 | 0 | 2 |
| | **TURN OFF** | **2.32** | **1.58** | 2 | 2.58 | **5.08** | **4.08** | 1 | 0 | 0 | **2.32** |
| | $\mathbf{P}$ | TURN OFF | TURN OFF | - | TURN ON | TURN OFF | TURN OFF | TURN ON | - | - | TURN OFF |
| | $\mathbf{P_r}$ | OPEN | OPEN | CLOSE | - | - | OPEN | - | - | CLOSE | OPEN |
| | $\mathbf{D}$ | TURN OFF, OPEN | TURN OFF, OPEN | CLOSE | TURN ON | TURN OFF | TURN OFF, OPEN | TURN ON | - | CLOSE | TURN OFF, OPEN |

$\mathbf{M_{ij}}$ : Embedded Database
$\mathbf{M_{ij}}$ ' : Transformed Database
$\mathbf{P}$ : Pattern of actions (from $M'_{ij}$)
$\mathbf{P_r}$ ' : Pattern of actions (from a remote device)
$\mathbf{D}$ : Transaction Dataset ($P$ and $Pr$)

Considering that each device acts independently, they must provide all resources needed to handle the physical stimulus (signals/interruptions) and/or logical stimulus (HTTP requests) and manage the storage and data processing independently from other resources in network.

## 4.2 Embedded database and the pattern of changes

The proposed method also defines a specific way to store the raw data to mitigate the lack of resources in IoT devices. Unlike the usual data mining and machine learning approach, MAKE focuses on registering and analyzing the actions that stimulate a state change (pattern of actions), instead of creating multiple repeated registers for mapping the usage of devices along a period (pattern of usage). This difference reduces the amount of data that should be stored and analyzed during the knowledge extraction.

Assuming $T = \{t_1, \dots, t_j\}$ is a finite set of $j$ discrete time intervals, it is possible to define an embedded database as a matrix $M_{ij} = A \times T$ (see $M_{ij}$ in Table 1) where each element $c_{xy}$ is a counter for each action $a_x \in A$ at slot $t_y \in T$.

To extract the pattern of action from $M_{ij}$ it is necessary to perform a logarithmic transformation (see $M_{ij}{}'$ in Table 1) in all counters to reduce the impact of older information and to exclude unusual register (i.e: outliers). This transformation is defined by: $c_{xy} \leftarrow \log_{(|A|)} c_{xy} \mid 1 \leq x \leq |A| \;\wedge\; 1 \leq y \leq |T|, \forall c_{xy} \in M_{ij}$. Also, if the transformation values were lower than 1, the counter assumes a zero value, otherwise it assumes the value of logarithmic operation so that it is possible to avoid negative values in further transformations.

Once the transformation is performed, it is possible to extract a reliable pattern of actions as follow: let $C_y = \{c_{1y}, \dots, c_{iy}\}$ be a set containing all counters from column $y$ in $M_{ij}$, and a function $max\_action(C_y, A)$ which returns the action (from the set $A$) correspondent to the greatest value from $C_y$ (or *nil* in case non predominance of a single action), then it is possible to create a set $P = \{(p_1, \dots, p_j) \in A \mid \forall p_y \leftarrow max\_action(C_y, A)\}$, where $1 \leq y \leq |T|$ that represents the device's pattern of actions (see $P$ in Table 1).

This storage organization reduces the amount of data that should be pre-processed to perform the embedded associative analysis. Combining a pair of action patterns from different devices into a transactions dataset ($D$), we can extract interesting correlations between both devices.

It is important to clarify that all devices must assume the same number of slots ($|T|$) to make it possible to create a transaction dataset by grouping the actions from the same slot to form a single transaction, as presented in Table 2

TABLE 2
Transaction datasets

| SLOTS | 1 | 2 | ... | $\|T\| - 1$ | $\|T\|$ |
|---|---|---|---|---|---|
| $\mathbf{P_1}$ | ON | OFF | ... | - | OFF |
| $\mathbf{P_2}$ | - | OPEN | ... | - | OPEN |
| $\mathbf{P_3}$ | HIGH | - | ... | LOW | - |
| $\mathbf{D_{12}}$ | ON | OFF,OPEN | ... | - | OFF,OPEN |
| $\mathbf{D_{13}}$ | ON,HIGH | OFF | ... | LOW | OFF |

As this process combines the patterns in pairs ($D_{12}$=$P_1P_2$ and $D_{13}$=$P_1P_3$), it could create transactions datasets with different sizes if any empty slots exist in both patterns, as illustrated in $|T| - 1$ from $D_{12}$. During the associative analysis the extracted rules could not be fairly compared since the support (Equation (1)) and lift (Equation (3)) metrics were directly dependent on the dataset size.

To circumvent that, the metrics support and lift were slightly modified for the purpose of this paper. They assume that all transactions datasets will have the highest number of slots filled as possible in $P$, which is equals to $|T|$. This slight modification allows for the fair comparison of rules obtained from two transactions datasets with different sizes, without previously knowing the patterns of all devices.

## 4.3 Correlation Extraction

The process of correlation extraction occurs individually in each device in fixed time intervals (called checkpoints) so it is possible to adapt to the users' pattern of interactions. For that, a device must know all other devices (targets) that have a pattern to share. Therefore, each device must previously join a specific multicast group, shared by all devices, which allows a device to create a list of targets through a multicast echo request/response [17].

Once the device obtains the list of targets, an iterative process, illustrated in Figure 1, begins.

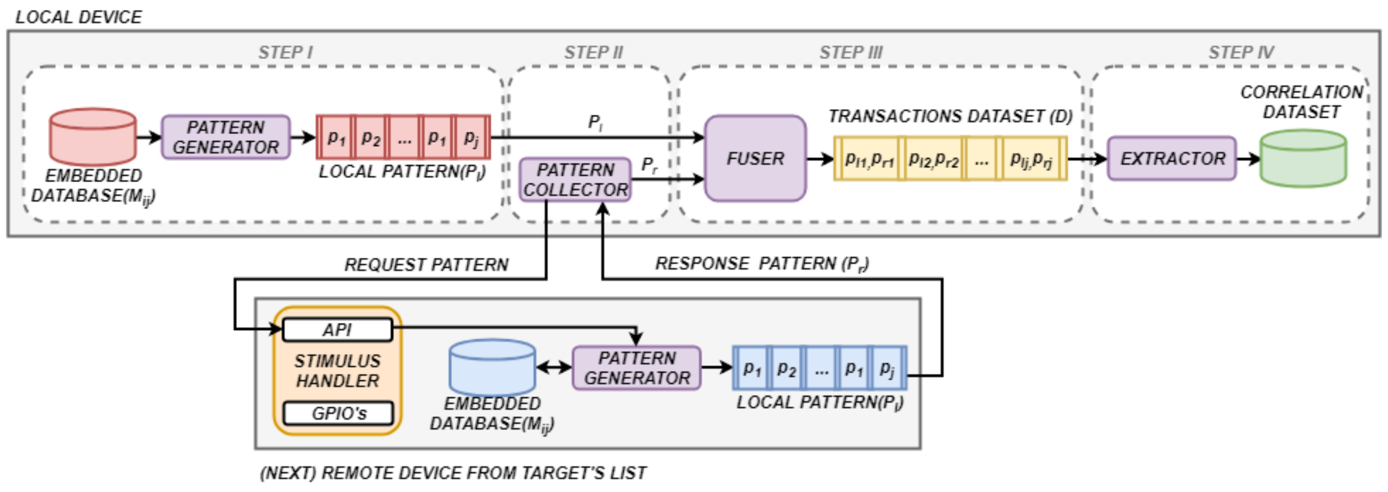

Fig. 1. Illustration of MAKE Steps

- **Step I:** the device identifies its own pattern of state changes ($P_l$: Local Pattern) from an embedded database ($M_{ij}$), as described in Section 4.2;
- **Step II:** the *Pattern Collector Component* requests the pattern of actions from the (next) target in the list. This target receives the request (through its own API), performs Step I on itself, and then replies its pattern, which is represented as $P_r$ (Remote Pattern).
- **Step III:** the *Fuser Component* receives both patterns ($P_l$ and $P_r$) and creates a set of transactions ($D$) by merging the action from both patterns in each slot as follows: $D = \{(p_{l1}, p_{r1}), \ldots, (p_{lj}, p_{rj})\}$ where $p_{ly} \in P_l$, $p_{ry} \in P_r$, and $1 \leq y \leq |T|$;
- **Step IV:** finally, the associative analysis extracts the strongest correlated pairs of actions from $D$ for each action from local set $A$. Then the rules from the current interaction are compared to the previously stored rules in *Correlation Dataset* which may replace, be appended or discarded according to their relevance by support, lift and confidence metrics (in that specific order).

This process repeats until all target devices are compared and the most relevant rules are stored in correlation dataset, if it is the case.

## 4.4 Device's architecture

This section describes a device architecture, all of its components and how they interact with the environment and with each other. The Figure 2 depicts the proposed architecture.

As specified in Section 4.1, each device is responsible for managing a single object from the environment (e.g: a lamp, a door, a window, an alarm, etc), therefore, it does not depend on any other device. The device must be connected in a local network, joined to a shared multicast group address and provide an API to make all its actions accessible to control the object states.

The correlated actions identified by the associative analysis act as a triggered reaction that synchronizes the states from a pair of devices if it satisfies their patterns of actions. For example, assume that an Air Conditioner (AC) identifies an interesting correlation between its action "Turn On" with the action "Close" from a Bedroom's Window (BW), then the rule would be $AC - TurnON \Rightarrow BW - Close$. Based on both devices' pattern of action, it is possible to use this rule to identify when the AC receives an input "Turn On" to send a request to BW perform the action "Close" if both were the most likely action to occurs at the current slot time.

To obtain such behavior each device must implement the components presented in Figure 2, which works as follows:

- **Stimulus Handler:** validates whether the input stimulus from the environment are logical, such as requests or messages, or whether they are physical, such as interruptions on the microcontroller GPIOs. If it is a valid input, it forwards a signal to the correspondent component to perform a change on device's state, sharing the device's information (patterns, states, id and others) or manage/fire a correlation rule. It may also implement additional features, such as, user interface, general configurations and custom functionality;
- **State Controller:** changes the representational device state and, in case of an actuator, sends a correspondent signal to GPIO's to change the module state (e.g: sends a HIGH signal to the pin where the LED is connected). Once the action is performed, this component consults the embedded database to check the most probable action in the current slot and increases the input action counter. If the input action most likely occurs at the current slot time, it forwards a signal to Stimulus Handle to send a logical stimulus due to an integration rule for this action (if one exists);
- **Recommender:** manages the rules stored in the Correlation Dataset. It forwards a HTTP request stimulating a correlation action (rule's consequent) to the correlated device, and allows the users to enable, disable or discard integration rules. In addition, after the extraction process, this component suggests the strongest correlations found to the user;
- **Correlations Dataset:** stores the rules that correlate a local action to the actions from others devices. It also stores the rule's metrics and the correlated device's address (e.g. ip/url);
- **Database:** is a matrix as specified in Section 4.2;
- **Pattern Generator:** performs a probabilistic analysis on the database to generate the device pattern of actions. This component can share these patterns through the API or send it to the *Fuser Component*;
- **Pattern Collector:** identifies others devices that belong to the same multicast group and performs an interactive process to retrieve their patterns of actions;

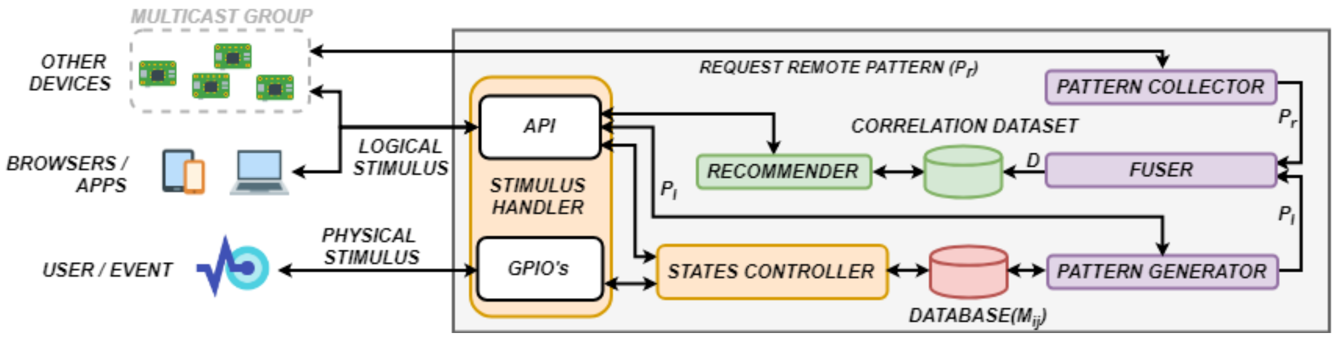

Fig. 2. MAKE architecture overview

- **Fuser:** performs data fusion of local and remote patterns (created by *Pattern Generator*) to create a transaction dataset. In other words, merges the actions slot by slot from both patterns;
- **Extractor:** performs the associative analysis on the transaction dataset to identify the strength of action correlations of both devices and, additionally, updates the rules in the *Correlation Dataset*.

This architecture must be the same for all devices and the implementation decisions must share the same parameters to perform a reliable knowledge extraction from the distributed dataset.

## 5 Experiments

The methodology employed to assess the performance of the proposed approach is based on simulation experiments of a smart environment with multiple devices that implement MAKE, followed by a comparison between the rules created by each device and the rules created by a centralized analysis. The goal is to evaluate how reliable MAKE is when dealing with identification of correlations based only on local decisions.

Multiple experiments were executed for different public datasets and considering different intervals between the checkpoints. All datasets were pre-processed to remove duplicated registers, discretize the continuous values and collect the registers that represented devices state changes.

To perform the centralized analysis, the statistical software R [18] was used with the help of the *aRules library* [19], which implements the Apriori Association Rule Algorithm [6]. For each checkpoint, all devices' patterns were gathered to create a unique transaction dataset, which was then analyzed by *aRules*. This allowed us to obtain all rules from all devices in a single Apriori's execution, and it was also possible to store those rules (and their metrics) in a file (*arules.log*). The correlations identified by MAKE were also stored together in a file (*mrules.log*) to facilitate the comparisons between the rules. Both files store the rules in a descending order by support, lift and finally confidence metrics.

The comparisons consisted of identifying whether the correlations in the *mrules.log* file were also the most relevant correlations in the *arules.log*, in that case a metric called *hit* was incremented, otherwise a second metric (*miss*) was incremented.

### 5.1 Database description

Five different datasets from WSU CASAS [20] were used to perform the simulation experiments. These datasets contain raw data from several sensors such as battery levels, magnetic doors sensor, light switches, lights sensors, infrared motions sensors and temperature sensors. Table 3 shows the number of devices, number of days and number of registers on each dataset.

These datasets were selected to make a stress test about the proposed method in different scenarios regarding the number of devices, number of registers and the number of days that should be analyzed.

TABLE 3
Dataset descriptions

| DATASET | DEVICES | DAYS | REGISTERS |
|---|---|---|---|
| hh107 | 110 | 371 | 3,369,689 |
| hh123 | 88 | 588 | 2,907,282 |
| hh129 | 86 | 668 | 12,303,984 |
| shib009 | 8 | 847 | 3,187,940 |
| tokyo | 67 | 115 | 802,534 |

### 5.2 Simulation parameters

To get as close as possible to a real intelligent environment, some parameters were pre-defined in the simulation.

- All devices created a database (counter matrix) for each day of the week (Monday to Sunday);
- Each slot comprehends a 15 minute interval, so each matrix has 96 columns ($|T|$);
- Minimum Support Threshold: 1% (1 slot filled);
- Minimum Confidence Threshold: 90%;
- Minimum Lift Threshold: 1.1 (only direct correlations: $A \to B$);

The experiment was executed for each dataset considering the following intervals:

- **Interval I**: the associative analysis was executed day by day;
- **Interval II**: the associative analysis was executed in alternate weeks, that is, one week yes and the other no;
- **Interval III**: the associative analysis was executed one week yes and three weeks no.

These parameters were the same for all datasets in all experiments.

## 6 Results

Table 3 shows the results of data cleaning before the simulation execution. This process ignores records that do not represent a devices state change, as specified in Section 4.2 and transforms continuous values into discrete intervals based on the range of recorded values.

This process resulted in a massive reduction in the data registers for hh129 and shib009 datasets, more precisely 99,54% and 97,16% respectively. In tokyo datasets, the reduction was 78,63% of its original content. The two other datasets, hh107 and hh123, had smaller reductions of 16,57% and 19,31%, respectively.

TABLE 4
Pre-processing results

| DATASET | USEFUL DATA | REDUCTION |
|---|---|---|
| hh107 | 2,811,279 | 16.57% |
| hh123 | 2,345,775 | 19.31% |
| hh129 | 56,523 | 99.54% |
| shib009 | 90,599 | 97.16% |
| tokyo | 171,483 | 78.63% |

Table 5 shows the results of the simulations for all datasets. It shows the number of rules identified as the most relevant in both analyses (*hits*) and the number of rules identified only by MAKE (*misses*). Moreover, there is the

Average Rates by each dataset and the Average Rates for the entire experiment.

All experiments for datasets hh129, shib009 and tokyo obtained 100% of agreement. In other words, all rules identified by MAKE were also the most relevant in a centralized analysis.

Exceptions occurred in two datasets (hh107 and hh123) for Interval III. The *aRules* disagrees with *MAKE* in 43 rules from hh107 dataset, and 15 rules from hh123. Those rules were not considered relevant by the centralized analysis. In the experiments for Interval I and Interval II, all rules matched as the most relevant.

TABLE 5
Experiments' Results

| DATASET | HITS/MISS BY INTERVAL I | II | III | AVG RATES (%) HITS | MISS |
|---|---|---|---|---|---|
| hh107 | 3,656/- | 3,493/- | 5,769/43 | 99.67 | 0.33 |
| hh127 | 1,952/- | 2,373/- | 4,040/15 | 99.82 | 0.18 |
| hh129 | 80/- | 54/- | 54/- | 100 | - |
| shib009 | 16/- | 135/- | 75/- | 100 | - |
| tokyo | 508/- | 337/- | 473/- | 100 | - |
| **TOTAL** | | **23,015 / 58** | | **99.75** | **0.25** |

The aRules identified 23,015 rules and MAKE identified 23,073 rules, namely, 58 more rules. This represents 0.25% of all correlations identified. The average accuracy of MAKE reaches 99.75% of similarity compared to the rules created by a centralized analysis.

## 7 Discussion

This section is organized in two parts: (i) pre-processing analysis; and (ii) comparison between the obtained rules. The first consists of exploring the results presented in Table 4, and the second discusses the results from Table 5.

### 7.1 Pre-Processing Analysis

By analyzing the pre-processing step, it was possible to identify a massive reduction in datasets, especially in hh129, shib009 and tokyo. This occurred due to the high number of devices that registered continuum values (e.g.: a temperature sensor). Considering that these values had to be discretized, many registers did not express a real change of discretized states. For example, the temperature sensor T105 from hh129 dataset, which has a range of registered values from 17.5° to 33°, and its average value is 25.25°. In this paper we assume that each value greater or equal to the average was labeled as "HIGH" and the lower values were labeled as "LOW". Analyzing the raw dataset, one can see that the first value lower than the average occurs after 26 registers. In other words, the first 26 registered values were greater than or equal to 25.25°. As they did not represent a real change ("HIGH" to "LOW"), all these registers were counted as a single "HIGH" in the embedded database ($M_{ij}$).

The same behaviour repeats for multiple devices in all datasets. In this case, the number of devices that register continuum values directly affect the reduction rate. On the other hand, the range from continuum registered values in hh107 and hh123 were smaller, which implies a smaller number of discretized label by devices. Additionally, there are also less devices that registered continuum values.

### 7.2 Rules Comparisons Analysis

A particularity was noted during the execution of the experiments with the datasets hh107 and hh123 considering extraction Interval III. In both cases, MAKE (decentralized) identified rules that could not be identified by *aRules* in R software (centralized).

Analyzing the experiment's report, it was possible to identify that some missing rules had metric values close to the minimum threshold. To check that particularity, in each experiment that contained missed rates, a new centralized analysis was performed considering more permissive metrics, which allowed the centralized analysis to identify the missing (non relevant) rules. A sample of these comparisons can be found in Table 6.

All three rules had the same consequence and their frequencies in the transaction dataset are expressed by column "count". Slight differences in support and lift metrics were noted for both analyses. The explanation for this difference is that in the centralized analysis all patterns are gathered (as in Step III during the mining process) into a single transaction dataset and its size ($|D|$) may assume any value from 0 to 96, since some slots may be empty in case there is no most probable action among all device's patterns. Furthermore, the centralized approach uses the Equations (1) and (3), which are directly dependent on $|D|$, leading the support and confidence metrics to assume values lower than obtained in the decentralized analysis, which always assume that $|D| = |T|$ in the same Equations (see in Section 4.3).

This modification allows MAKE to more sensitive to rules that may have values near the metrics thresholds, as exposed in Table 6. The rule $LS019-LOW \Rightarrow LS21-LOW$ is present in 71 of 96 registers and was considered an interesting correlation by the MAKE while it was ignored by the *aRules* since the transaction dataset size assumed a value lower than $|T|$ during the centralized analysis.

## 8 Conclusion

The main aim of MAKE is to provide an embedded mechanism to allow each device to extract knowledge from a intelligent environment that has limited resources for storage, management and processing. This mechanism correlates pairs of devices actions with the purpose of offering a set of intelligent integration suggestions via HTTP requests to users. These suggestions, when accepted, allow the devices to control each other based on their pattern of actions, which may stimulate automatic state changes between correlated devices.

The proposed method deals with pairs of devices and only analyzes the transactions that contain useful information, while a centralized environment unnecessarily considers correlations between actions from the same devices.

This work reproduces (with 99.75% of agreement) a well-known centralized data mining algorithm (Apriori) using a decentralized approach that processes an embedded associative analysis which correlates a pair of items. In addition, the use of a static size dataset allows for the identification of relevant correlations not found in centralized analysis.

This paper shows that MAKE is an alternative for implementing a decentralized and highly integrated environment

TABLE 6
Metrics' comparisons of MAKE(missing rules) to the metrics *aRules*(from permissive threshold analysis)

| Antecedent | Consequent | MAKE (Misses) | | | *aRules* (Permissive) | | | Count |
|---|---|---|---|---|---|---|---|---|
| | | Supp | Conf | Lift | Supp | Conf | Lift | |
| LS023 (HIGH) | LS021(LOW) | 0.5729 | 0.9821 | 1.1359 | 0.5978 | 0.9821 | 1.0886 | 55 |
| LS019 (LOW) | LS021(LOW) | 0.7395 | 0.9861 | 1.1405 | 0.7717 | 0.9861 | 1.0930 | 71 |
| LS004 (HIGH) | LS021(LOW) | 0.6250 | 0.9836 | 1.1376 | 0.6521 | 0.9836 | 1.0902 | 60 |

based on embedded knowledge extraction to correlate devices in a smart environment and to adapt itself to the users' behavior patterns. The proposed method avoids the single point of failure problem because it adopts a decentralized associative analysis.

Although it proves to be an efficient methodology, the MAKE presents some limitations that should be explored in future works, such as: (i) *dataset dimensionality*, since the embedded dataset grows proportionally to the number of actions/states available, which could cause a high storage consumption; (ii) *discretized time interval*, since the device has a high number of slots ($|T|$) the patterns process sharing may fail once the device cannot handle both pattern (local and remote) on memory to perform the associative analysis; and (iii) *predictions*, since the embedded database registers the action in discrete time slots (15 minutes for instance) this does not mean that the action is active in this whole interval, but it may be activated at any time in this interval.

Finally, for future research, we intend to further this work by creating prototypes to execute real-world experiments and evaluate the user experience related to suggestions for device integration.

## Acknowledgments

The authors acknowledge the support granted by FAPEAM, which provided funds to develop this research, the Institute of Computing of the Federal University of Amazonas (UFAM), and University of Ontario Institute of Technology (UOIT).

## References

[1] F. Chen, P. Deng, J. Wan, D. Zhang, A. Vasilakos, and X. Rong, "Data mining for the internet of things: Literature review and challenges," *International Journal of Distributed Sensor Networks*, vol. 2015, 2015.

[2] M. Verhelst and B. Moons, "Embedded deep neural network processing: Algorithmic and processor techniques bring deep learning to iot and edge devices," *IEEE Solid-State Circuits Magazine*, vol. 9, no. 4, pp. 55–65, 2017.

[3] L. Li, Q. Li, Y. Wu, Y. Ou, and D. Chen, "Mining association rules based on deep pruning strategies," *Wireless Personal Communications*, vol. 102, no. 3, pp. 2157–2181, Oct 2018. [Online]. Available: https://doi.org/10.1007/s11277-017-5169-0

[4] E. Nazerfard, "Temporal features and relations discovery of activities from sensor data," *Journal of Ambient Intelligence and Humanized Computing*, May 2018. [Online]. Available: https://doi.org/10.1007/s12652-018-0855-7

[5] V. S. Kireev, A. I. Guseva, P. V. Bochkaryov, I. A. Kuznetsov, and S. A. Filippov, "Association rules mining for predictive analytics in iot cloud system," in *Biologically Inspired Cognitive Architectures 2018*, A. V. Samsonovich, Ed. Cham: Springer International Publishing, 2019, pp. 107–112.

[6] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. of 20th Intl. Conf. on VLDB*, 1994, pp. 487–499.

[7] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," *Data Mining and Knowledge Discovery*, vol. 8, no. 1, pp. 53–87, Jan 2004. [Online]. Available: https://doi.org/10.1023/B:DAMI.0000005258.31418.83

[8] J.-F. Boulicaut and B. Jeudy, *Constraint-Based Data Mining*. Boston, MA: Springer US, 2005, pp. 399–416.

[9] M. Z. A. Bhuiyan and J. Wu, "Event detection through differential pattern mining in internet of things," in *2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, Oct 2016, pp. 109–117.

[10] M. Smith, X. Wang, and R. Rangayyan, "Evaluation of the sensitivity of a medical data-mining application to the number of elements in small databases," *Biomedical Signal Processing and Control*, vol. 4, no. 3, pp. 262–268, 2009.

[11] Y.-C. Chen, Y.-L. Ko, and W.-C. Peng, "An intelligent system for mining usage patterns from appliance data in smart home environment," 2012, pp. 319–322.

[12] E. Heierman and D. Cook, "Improving home automation by discovering regularly occurring device usage patterns," 2003, pp. 537–540.

[13] D. McArthur, S. Encheva, and I. Thorsen, "Exploring the determinants of regional unemployment disparities in small data sets," *International Regional Science Review*, vol. 35, no. 4, pp. 442–463, 2012.

[14] X. Wang, M. Chen, and L. Chen, "Research of the optimization of a data mining algorithm based on an embedded data mining system," *Cybernetics and Information Technologies*, vol. 13, no. SPECIALISSUE, pp. 5–17, 2013.

[15] L. Gonzalez and O. Amft, "Mining relations and physical grouping of building-embedded sensors and actuators," 2015, pp. 1–10.

[16] K. Pal, S. Adepu, and J. Goh, "Effectiveness of association rules mining for invariants generation in cyber-physical systems," 2017, pp. 124–127.

[17] S. Venaas, "Multicast Ping Protocol," RFC 6450, Dec. 2011. [Online]. Available: https://rfc-editor.org/rfc/rfc6450.txt

[18] R Development Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2008, ISBN 3-900051-07-0.

[19] M. Hahsler, S. Chelluboina, K. Hornik, and C. Buchta, "The arules r-package ecosystem: Analyzing interesting patterns from large transaction datasets," *Journal of Machine Learning Research*, vol. 12, pp. 1977–1981, 2011.

[20] D. J. Cook, A. S. Crandall, B. L. Thomas, and N. C. Krishnan, "Casas: A smart home in a box," *Computer (Long Beach Calif)*, vol. 46, no. 7, p. 10.1109/MC.2012.328, Jul 2013.

macalencar@icomp.ufam.edu.br.