Dissertation

Mining Functional and Structural Relationships of Context Variables in Smart-Buildings

Luis Ignacio Lopera Gonzalez

August 23, 2018

Supervisor:	Prof. Dr. Oliver Amft
External Examiner:	Prof. Dr. Michael Beigl

Copyright Information

Summary

The Internet of Things (IoT) is a network of computational services, devices, and people, which share information with each other. In IoT, inter-system communication is possible and human interaction is not required[1]. IoT devices are penetrating the home and office building environments. According to current estimates, about 35 billion IoT devices will be connected by the year 2021[2]. In the IoT business model, value comes from integrating devices into applications, e.g., home and office automation. In general, an IoT application associates different information sources with actions which can modify the environment, e.g., change the room's temperature, inform a person, e.g., send an e-mail, or activate other services, e.g., buy milk on-line.

In this thesis, we focus on the commissioning and verification processes of IoT devices used in building automation applications. Within a building's lifespan, new devices are added, interior spaces are refurbished, and faulty devices are replaced. All of these changes are currently made manually. Furthermore, consider that a context-aware Building Management System (BMS) is an IoT application, which measures direct-context from the building's sensors to characterize environmental conditions, user locations, and state. Additionally, a BMS combines sensor information to derive inferred-context, such as user activity. Similar to IoT devices, inferred-context instances have to be created manually. As the number of devices and inferred-context instances increases, keeping track of all associations becomes a time-consuming and error-prone task.

The hypothesis of the thesis is that users who interact with the building create use-patterns in the data, which describe functional relations between devices and inferred-context instances, e.g., which desk-movement sensor is used to infer desk-presence and controls which overhead light; additionally, use-patterns can also provide structural relations, e.g., the relative position of spatial sensors. To test the hypothesis, this thesis presents an extension to the new IoT class rule programming paradigm[3], which simplifies rule creation based on classes. The proposed extension uses a semantic compiler to simplify the device and inferred-context associations. Using direct-context information and template classes, the compiler creates all possible inferred-context instances. Buildings using context-aware BMSs will have a dynamic response to user behaviour, e.g., required illumination for computer-work is provided by adjusting blinds or increasing the dim setting of overhead ceiling lamps. We propose a rule mining framework to extract use-patterns and find the functional and structural relationships between devices. The rule mining framework uses three stages: (1) event extraction, (2) rule mining, (3) structure creation. The event extraction combines the building's data into a time-series of device events. Then, in the rule mining stage, rules are mined from the time series, where we use the established algorithm temporal interval tree association rule learner. Additionally, we proposed a rule extraction algorithm for spatial sensor's data. The algorithm is based on statistical analysis of user transition times between adjacent sensors. We also introduce a new rule extraction algorithm based on increasing belief. In the last stage, structure creation uses the extracted rules to produce device association groups, hierarchical representation of the building, or the relative location of spatial sensors. The proposed algorithms were tested using a year-long installation in a living-lab consisting of a four-person office, a 12-person open office, and a meeting room. For the spatial sensors, four locations within public buildings were used: a meeting room, a hallway, T-crossing, and a foyer. The recording times range from two weeks to two months depending on scenario complexity.

We found that user-generated patterns appear in building data. The rule mining framework produced structures that represent functional and spatial relationships of building's devices and provide sufficient information to automate maintenance tasks, e.g., automatic device naming. Furthermore, we found that environmental changes are also a source of device data patterns, which provide additional associations. For example, using the framework we found the façade group for exterior light sensors. The façade group can be used to automatically find an alternative signal source to replace broken outdoor light sensors. Finally, the rule mining framework successfully retrieved the relative location of spatial sensors in all locations but the foyer.

[1] R. Minerva, A. Biru, and D. Rotondi. "Towards a Definition of the Internet of Things (IoT)". in: *IEEE Internet Initiative* 1 (2015).

[2] L. Columbus. *Roundup Of Internet Of Things Forecasts And Market Estimates, 2016.* URL: https://www.forbes.com/sites/louiscolumbus/2016/11/27/roundup-of-internet-of-things-forecasts-and-market-estimates-2016/ (visited on 12/13/2017).

[3] F. Corno, L. D. Russis, and A. M. Roffarello. "A Semantic Web Approach to Simplifying Trigger-Action Programming in the IoT". in: 50.11 (2017), pp. 18–24. ISSN: 0018-9162.

Acknowledgements

I would like to thank my supervisor Prof. Dr. Oliver Amft for all the support, guidance and freedom to choose and follow the research path which culminated in this thesis. I would also like to thank Prof. Dr. Alexander Lazovik, Viktoriya Degeler, Marija Milenkovic and the rest of the GreenerBuildings consortium for all the inspiring work and fruitful discussions. The results of all that arguing led to some of the ideas presented in this thesis.

During my PhD, several avenues were evaluated, among which, a project in the library of economics at the University of Passau. The library project welcomed the collaboration from Daniel Shreckling and Juan David Parra from the chair of ITSecurity at Uni-Passau. I would like to thank Juan David and Daniel for the collaboration in the library project, and a special mention to Prof. Dr. Joachim Posegga for the financial support during the last months of this thesis.

I would like to thank the support of my friends, Adrian, Florian, Piero and Ulf. They endured stubborn discussions and wide birth tangents which allegedly lead nowhere. However, from every discussion, I took important lessons and ideas, some of which made it into the thesis. I would especially like to thank Rita, who had endured the hardest discussions of them all, and without whom writing this thesis would have been a harder task.

Finally, I would like to thank my family for allowing me to follow my own path, even though unorthodox, rejecting their own experiences and expectations wishing only that I would find the way to my own success.

Contents

Chapter 1

Introduction

1.1 The Internet of Things and Building Management Systems

The Internet of Things (IoT) is a network of computational services, devices, and people, which share information with each other. In IoT, inter-system communication is possible and human interaction is not required [75]. An IoT device is any physical object that can be connected to the Internet to provide or receive information, e.g., sensors and actuators. IoT devices are penetrating the home and office environment. By using the Internet, devices, computational services and users can be easily connected. Among all the foreseen IoT applications, building and home automation have found a quick entrance into the market. The IoT revolution brings low cost devices and infrastructure to the building and home environments, where users can control them through common interfaces like the mobile phone. Driven by energy saving and comfort, Building Management System (BMS) optimize the energy consumption in office buildings based on occupancy and other context information to provide the user with comfortable environmental settings while minimizing energy consumption [3, 82]. The balance between comfort and energy consumption is achieved by detecting context information derived from sensor data, e.g., activities which require energy [34] or ontology based recognition [83]. In office buildings, comfort is divided in two main topics: thermal [20] and lighting [47] comfort. In both cases context information is used to estimate the best comfort conditions. Similarly, BMS in home environments also use context to save energy and provide comfort [111]. Additionally, the BMS is used to facilitate assistance and care to people with disabilities [56, 84], while providing users the possibility of independent living. At the city level, a BMS connected to the internet is capable of reporting real-time feedback to the grid. Energy generators us the information provided by connected BMSs to balance energy's supply and demand, which in turn, reduces natural resource consumption such as gas, carbon, or water [2].

Adopting IoT systems and frameworks has commercial benefits for building owners. Technologies which reduce the energy footprint, while maintaining comfort, have a direct impact on the rent price and the effective rent. People will pay higher rent prices for apartments which are certified energy efficient, and the owner will spend less on operating the building, thus, increasing the owner's effective rent income. In 2010, the impact of building's energy efficiency certificates on the rent prices and the effective rent was measured at a 3% and 7% increase respectively for independently certified buildings [26]. However, the majority of new IoT devices and applications will have to adapt to current buildings and their infrastructure. A building's average service-life in Europe and the USA is expected to be between $\sim$ 50 to 70 years [50, 88]. Thus, most IoT installations will be made in existing buildings. New IoT devices will coexist with old technologies, e.g., old BMS, as the cost of a full replacement greatly exceeds the cost of an upgrade to support new technologies [97]. Furthermore, according to current estimates, about 35 billion IoT devices will be connected by the year 2021 [16]. Other than context measurement for energy and comfort in buildings, the added value of IoT devices depends on the applications that they support. From driving style recommendation based on environmental and traffic sensors [57], to smart appliances such as refrigerators [27, 30]. As more devices are installed, they need to be associated in meaningful relationships with the services that provide oversight and control, e.g., the refrigerator's and washing machine's state and power consumption can be associated to the household's appliance energy monitoring dashboard, but not to the house's outdoor light controller. The intended market for IoT devices is mostly composed of non-technical users. However, if the commissioning process and the user interaction cannot be simplified, IoT devices will have a difficult time being incorporated [59].

A simplified BMS model consist of three components: sensors, control logic and actuators. Therefore, BMS commissioning consist on assigning each sensor a corresponding actuator using rules that constitute the control logic. The installation procedure goes as follows: create groups based on locations with similar properties,

e.g., office rooms, assign sensors and actuators to the rooms, then create control rules, which associate sensors with actuators. In the basic commissioning scheme, location information is not technically necessary, as the rule associates the sensor and actuator. However, location information provides the user a hierarchical structure to manage all devices and system properties. Context-aware BMSs, in addition to the simplified model, use inferred context and control algorithms which calculate optimal actuator states based on current context. Nevertheless, the commissioning process follows the same logic: create groups based on locations, assign sensors to inferred context algorithms, map actuators and required context to the control algorithm. Although the commercialization of IoT devices for the public has made the trigger action paradigm the most popular due to its ease of use, the promised energy savings and comfort improvements will require carefully crafted rules, or the inclusion of advanced control strategies which only experts will be able to generate. Additionally, validating the installation and maintaining it during the life span of the building requires time and expert involvement. In particular, when managing old BMS infrastructure which are customized to the building's specifications, detailed information about the old BMS configuration and its current state is not easily attainable. As a result, when incorporating new IoT technologies into old buildings, the new devices rarely interact with the old infrastructure, creating unnecessary redundancies and sub-utilizing all possible sources of context information. Thus, commissioning, validating and maintaining a BMS is a manual, labour intensive, and error-prone task.

The hypothesis of the thesis states that users who interact with the building create use-patterns in the data, from where building structures can be inferred. The use-patterns can be used to validate the BMS installation, discover old BMS structures, or automatically associate sensor replacements. Use-patterns are extracted using rule mining techniques because, in particular, the simplified BMS model uses rules for the control logic, e.g., if *presence* then $light = on$, and in general, a rule is an approximation of the control logic, where a change in the input variables reflects a change in the output, e.g., a new set point for the room temperature causes the heating and ventilation system to modify the internal state, as consequence, a change in room temperature is observed. The hypothesis is illustrated in the following example: When users enter the building to go to their offices, they will activate motion sensors in their path. As users enter the office building they create a tree-like structure which starts with the main door in the root, branches out to the floors, then the offices, and finally the individual desks. Each tree node represents a motion sensor, and the edges represent the transition times between motion sensors, e.g., the time it takes to go from the office door to the desk. When tow nodes are connected due to a user transition, the association is called a **structural relationship**. Similarly, actuators that respond to the motion sensors are also assigned to the tree as nodes, e.g., a ceiling lamp that activates in response to a presence sensor. The association of nodes created by the BMS' control logic is defined as a **functional relationship**. The resulting tree can be mapped into the BMS' time-series as rules, e.g., if *hallway motion* is observed, then $room_1$ *motion* is also seen, within an expected time of $\bar{t} = 5 \pm 1$s. Each level of the tree-like structure represents a building's location, e.g., the root is the building as an entity, the next level represents the floors, followed by corridors, office rooms, areas, and finally, desks which are the leaves of the tree.

1.2 Related approaches to variable relation discovery

The thesis' hypothesis proposes that the structure of the building and the functional relationships of context variables can be extracted from BMS data. In this section, we discuss related methods where different types of relationships and structures are extracted. First, we examined methods to extract the exact location of wireless sensor network nodes; where the structure of the building can be reconstructed from the individual node positions. Then, we look into unsupervised learning of Bayesian networks algorithms, as users can be seen as a random process and their movement through the building creates a graph. Additionally, we thought of the user movement as creating activation patterns in the BMS's time-series, therefore, we considered motif search approaches to see if activation patterns could be extracted. Finally, we considered the methodologies used in complex system diagnostics, as the relationships of the system's components need to be modelled or approximated in order to find problems within the system.

Wireless sensor network localisation

Structural relationships are a well studied problem in wireless sensor networks. Savvides et al. [101] proposed the model for 3D localisation of wireless nodes, where at least three beacon nodes are required for calculating the absolute location of all nodes in the mesh. They use lasers rangers to measure the distance between nodes. In current work, Kirichek et al [52] replace the laser ranging mechanism with the sensor's wireless communication channel, and use the power propagation to distance relationship to estimate the mesh topology.

Absolute position of all sensor nodes will yield the structure of the building. However, since the approach is based on the power of the transmission signals, it means that all devices in the buildings have to be a part of the wireless sensor network, and all nodes have to use the same communication protocol. The rule mining approach proposed in this thesis uses the BMS' time-series, which is the data collected from all devices and context variables. Therefore, relative localisation of sensors can be achieved, regardless of the communications' media and protocols used to communicate with the BMS.

Learning graphical models and functional association

Users interacting with the building can be seen as a random process. Furthermore, the use-patterns can be modelled using graphical models, in particular, Bayesian networks. Hence, the use-pattern graph has the following structure: every context variable is a node and every edge represents the transition between variables, e.g., transitions between motion sensors, or a response created by the BMS control logic, e.g., motion sensor and ceiling lamp. Furthermore, the BMS' time-series can be seen as a sequence of building status snapshots, where each record contains the value of all context variables. Therefore, algorithms that learn graphical models in databases can be candidates to automatically extract the functional and structural relationships of the building's context variables. For example, Cheng et al. [13, 14] proposed the three-phase dependency analysis algorithm (TPDA) to automatically learn Bayesian networks from data. TPDA uses an information theory based approach, does not require specific expert knowledge, and is guaranteed to have polynomial complexity. However, using TPDA in time-series data to extract the building structure presents several challenges as three out of the four assumptions are not satisfied. First, the records are not independent and identically distributed, the information in a given record depends on the previous records, due to the time causality effect introduce by the BMS' control logic, and the temporal relation of the user generated events. In a real case scenario, values can be missing, e.g., due to equipment failure, thus, TPDA's no missing data assumption is not maintained. Finally, TPDA requires large quantities of data, although buildings constantly generate data, some context variable relations are scars, e.g., context variables' relationships associated to the usage of service rooms. Therefore, the conditional independence test for scars variables is not reliable.

Instead of learning a graphical model to model functional relationships, Ilyas et al. [44] proposed an algorithm called CORDS, a method to find correlation and soft functional dependencies in databases. Their proposed method finds pairwise relationships between database columns and builds the dependency graph from the pairs. TPDA and CORDS find record wise relationships, i.e., the dependencies are learned from variations within records. Whereas the functional and structural relationships are time dependent, i.e., the relationship between database columns is evidenced across multiple records. For example, given a pairwise relationship $a \rightarrow b$, and $\forall r \in D$ records in a database D, TPDA and CORDS will find $a, b \in r_i \forall r_i \in D' \subset D$. Therefore, CORDS and TPDA cannot be directly applied to the BMS time-series. The rule mining approach used in this work finds$a \rightarrow b$ such that $a \in r_i \wedge b \in r_{i+n}$ where n is the expected value of record separation, i.e., number of positions or time interval.

Motif search

The problem of functional and structural associations of context variables from a BMS' time-series can be approached from a pattern recognition perspective. In principle, as users move through the building, they create use-patterns, which are recorded in the BMS' time-series. Yeh et al. [115] and Zhu et al. [120] presented the STAMP and STOMP algorithms respectively. Fine-tuned for performance, STAMP and STOMP compute the distance between each subsequence of the time-series. Then, they find the nearest neighbour for each subsequence based on euclidean distance. A matrix profile is built, where for each segment, the index of its nearest neighbour is stored. Finally, the matrix profile and its index of minimum segment distances is used to assemble the motifs. STAMP and STOMP differ on how they search for the minimum distance, while STAMP does an aleatory search of the minimum distances, STOMP does an ordered search. While the matrix profile is robust against slight motif transformation in time, i.e., pattern stretch, and in amplitude, i.e., measurement error, in the case of the building's use-patterns, the differences between sequences does not follow the minimum distance criteria as use-patterns can be overlapping. Additionally, an instance of the use-pattern is expected to have random insertions of other symbols which surpass the measurement error expected by the matrix profile. For example, Eq. 1.1 illustrates three sample time-series segments, where the pattern 1 – 5 – 9 is present. Using the minimum euclidean distance criteria, all three sequences are considered different. In contrast to motif search, rule mining looks for the appearance of symbols in the segment rather than its shape. Therefore, rule mining can extract the pattern 1 – 5 – 9 regardless of other symbols appearing

in the sequence.

$$\begin{matrix}[1, & 5, & 9, & 0, & 0, & 0, & 0, & 0] \\ [1, & 5, & 9, & 10, & 10, & 10, & 10, & 10] \\ [1, & 4, & 8, & 7, & 5, & 3, & 2, & 9]\end{matrix} \tag{1.1}$$

System diagnostics

One application of functional and structural relationship mining is in system diagnostics. In general, a time-series is the recorded evolution of a generating process through time. The health, i.e., status, of the generating process can be assessed by studying the time-series. Kavulya et al. [48] summarised diagnostic techniques of complex system in six classes: rule-based, model-based, statistical modelling, machine learning, count and threshold, and visualization techniques. An intuitive approach to diagnostics is the rule-based diagnostic technique, which relies on expert's knowledge and is limited to the failure cases foreseen by the expert. As systems grow in complexity, e.g., increasing number of devices in a building., the knowledge base is difficult to maintain. Model-based techniques are based on physical, regression, or graphical models. Although model-based techniques for diagnostics are the most accurate, they require extensive knowledge of the system and of all factors affecting the system's behaviour. In contrast to model-based diagnostics, statistical techniques require less in-depth knowledge of the system. However, statistical diagnostics depends on the moment where the base line was calculated, as the system behaviour evolves, a malfunction or an expected change are equally considered as anomalies, e.g., full CPU use due to high throughput vs. a runway process. Machine learning is used to cluster faults by mapping measurable performance and behaviour metrics, i.e., features, into fault state clusters. The clustering approach works as long as the system's behaviour does not drastically change over time. Additionally, fault clusters can be over-fitted increasing the number of undetected fault instances. Count and threshold is used to determine whether the type of fault is transient, i.e., caused by external factors, or intermittent, i.e., caused by failing internal components. Depending on the application, a threshold on the rate of both fault types is set to determine when to replace or improve a component. Finally, visualization summarizes the time-series information into human readable diagrams and info-graphics to allow operators to identified anomalous behaviour patterns.

The rule mining based approach to functional and structural relationship modelling uses the intuitive idea of a rule-based model, but instead of expert's knowledge, the rules are learnt from the system. So, a large knowledge base is maintainable. Rule mining combines statistical and unsupervised machine learning methods. Therefore, the internal system behaviour can be modelled without in-depth system's knowledge. However, unlike statistical modelling, where performance evaluations are used to determine faulty states, rule mining finds the frequency of events in the system and the relative rate of finding the same events during an observation window. Thus, diagnostic models based on functional relationships, are not susceptible to the variation of performance metrics due to changes in behaviour. For example, a runway process does not interact with other processes in the same way a process with high load interacts with its pears, even though, both processes cause high CPU load.

1.3 Thesis goals

This thesis has two goals: (1) simplify the process of instantiating inferred context and control algorithms when using class rule programming, and (2) study use-patterns discovery to validate, maintain, and model the Building Management System (BMS).

While the class rule programming paradigm for BMS systems simplifies rule creation for non-technical users, additional work is required to simplify the instantiation of inferred context variables and control algorithms. The goal is that a non-technical user can program a context-aware BMS without the need of direct expert participation, while maintaining the benefits of an expert's system in terms of energy saving and comfort. The approach should leverage device's location information to create dynamic associations between sensors, actuators, inferred context variables and control algorithms. Additionally, the approach needs to be scalable in order to support installations from a single-room home to a skyscraper.

When users interact with the building, the resulting building's use-patterns should be discoverable through rule mining of the BMS' time-series. In general, propositional rules of the form $a \rightarrow b$ are an approximation of the BMS' control logic, as variations in the controller's input variables cause changes in the outputs. Additionally, building occupants' generate a correlation between context variables. For example, when a user enters and sits at a desk, the room's motion and desk presence will often appear within similar time intervals, and thus can be encoded as a rule, e.g., if $room_1$ *movement* then $desk_1$ *presence* in $\bar{t} = 3 \pm 1$s. The extracted rules reflect the variables' **functional relationship**, i.e, variables are involved in the control process, and

the **structural relationship** from a user's perspective, i.e., the user path through the different levels of the building's hierarchy. With the extracted rules, the goal is to discover functional and structural relationships of context variables and extract building structures and the association of each variable to the different levels of the structure.

1.4 Thesis outline and related publications

This thesis is divided in four parts: (1) the introduction to a new programming paradigm for building management systems, (2) functional and structural context variables' relationship mining from the Building Management System (BMS)' time-series data when using the new programming paradigm, (3) an extension of structural mining to relative localization for spatial sensors, and finally, (4) an atomic rule mining algorithm which does not use any thresholds on the confidence or support of the rules.

The preference based programming paradigm is introduced in chapter 2. Chapters 3–4 present two methods for functional and structural context variable relationship mining from BMS's time-series. Thermopile array sensors are introduced with an application in chapter 5, and an extension of structural relationship mining to relative sensor position is presented in chapter 6. Chapter 7 introduces the Bayesian Rule Extractor (BRE) algorithm, which is based on the belief of a rule always increasing. The algorithm does not use thresholds on the rule's support. Finally, the thesis conclusions are presented in chapter 8. Table 1.1 indicates which chapters from the thesis that have been independently published.

Chapter	Publication
3	L. I. Lopera Gonzalez and O. Amft. "Mining relations and physical grouping of building-embedded sensors and actuators". In: *PerCom 2015: Proceedings of the International Conference on Pervasive Computing and Communications*. IEEE, Mar. 2015, pp. 2–10. DOI: 10.1109/PERCOM.2015.7146503
4	L. I. Lopera Gonzalez and O. Amft. "Mining Hierarchical Relations in Building Management Variables". In: *Pervasive and Mobile Computing*. Pervasive and Mobile Computing 26 (Feb. 2016), pp. 91–101. ISSN: 1574-1192. DOI: 10.1016/j.pmcj.2015.10.009
5	L. I. Lopera Gonzalez, M. Troost, and O. Amft. "Using a thermopile matrix sensor to recognize energy-related activities in offices". In: *SEIT 2013: Proceedings of the 3rd International Conference on Sustainable Energy Information Technology*. Procedia Computer Science. Recipient of the Best Paper Award at SEIT 2013. Elsevier, 2013, pp. 678–685. DOI: 10.1016/j.procs.2013.06.090
6	L. I. Lopera Gonzalez, R. Stier, and O. Amft. "Data Mining-Based Localisation of Spatial Low-Resolution Sensors in Commercial Buildings". In: *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*. BuildSys '16. New York, NY, USA: ACM, Nov. 2016, pp. 187–196. ISBN: 978-1-4503-4264-3. DOI: 10.1145/2993422.2993428

Table 1.1: Published chapters.

1.5 Additional publications

The following publications were written in addition to the thesis. The scientific contributions of these publications are complementary to the thesis and served to introduce concepts, and empirically test ideas.

- V. Degeler, L. I. Lopera Gonzalez, M. Leva, P. Shrubsole, S. Bonomi, O. Amft, and A. Lazovik. "Service-Oriented Architecture for Smart Environments (Short Paper)". In: *SOCA 2013: Proceedings of the IEEE 6th International Conference on Service-Oriented Computing and Applications*. 6th IEEE International Conference on Service-Oriented Computing and Applications, SOCA 2013. IEEE, 2013, pp. 99–104. DOI: 10.1109/SOCA.2013.26
- P. Jaramillo-Garcia, L. I. Lopera Gonzalez, and O. Amft. "Using implicit user feedback to balance energy consumption and user comfort of proximity-controlled computer screens". In: *Journal of Ambient Intelligence and Humanized Computing* 6.2 (Feb. 2014), pp. 207–221. DOI: 10.1007/s12652-014-0222-2

- L. I. Lopera Gonzalez, U. Großekathöfer, and O. Amft. "Novel stochastic model for presence detection using ultrasound ranging sensors". In: *ACOMORE 2014: IEEE International Conference on Pervasive Computing and Communications Workshops*. PerCom Workshops. 1st Symposium on Activity and Context Modeling and Recognition. IEEE, 2014, pp. 55–60. DOI: 10.1109/PerComW.2014.6815164
- L. I. Lopera Gonzalez, U. Großekathöfer, and O. Amft. "An Intervention Study on Automated Lighting Control to Save Energy in Open Space Offices". In: *PerEnergy 2015: IEEE International Conference on Pervasive Computing and Communications Workshops*. IEEE, Mar. 2015, pp. 317–322. DOI: 10.1109/PERCOMW.2015.7134055
- L. I. Lopera Gonzalez and O. Amft. "Mining Device Data to Auto-Commission Buildings: Poster Abstract". In: *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*. BuildSys '16. New York, NY, USA: ACM, Nov. 2016, pp. 249–250. ISBN: 978-1-4503-4264-3. DOI: 10.1145/2993422.2996413

Chapter 2

Semantic Compiler for Smart-Building Ontologies

2.1 Introduction

With the Internet of Things (IoT), devices are becoming interconnected and people may achieve automatic control over their smart spaces by interconnecting sensors with actuators. Unlike previous building automation strategies, which were closed systems and required professional installers, IoT has propelled products that non-experts can install. However, individual device manufacturers are not aware of the entire IoT ecosystem. Therefore, information processing happens at a device level. For example, Nest smart thermostat uses on-board sensors and on-line services to learn and adapt to occupant's behaviour and preferences [114]. Although the thermostat can share its knowledge about user patterns with other appliances [112], it is not designed to take other thermostats into consideration. As a result, the number fo possible interactions between the thermostat and other devices is limited to a specific set of actions. The advantage of the actions-set is that an average person can create associations to other devices known to the thermostat.

Two of the driving forces for IoT integration in buildings is energy saving and occupant comfort. Research has shown that by closely monitoring the building's context, i.e., the occupant's state and activity, and the space's environmental conditions, as more devices are installed and interconnected, detail context information becomes available and fine grained actions can be taken to minimize energy consumption while preserving comfort [64, 73]. Detailed context information can be divided in two categories: direct-context and inferred-context. Information provided by devices or external service, e.g., light sensors, weather forecast, constitute direct-context. Smart-buildings use algorithms to compute additional context information called inferred-context [24]. For example, a sensor fusion algorithm to recognize office-desk activities [74]. Furthermore, a sequence of algorithms in combination with direct-context instances, required to compute inferred-context, is considered a context-aware framework. For example, Cook's et al. [17] introduced the middle ware to have context-aware home automation.Once inferred- and direct-context are evaluated, smart-buildings use context information to modify the environment using actuators. Algorithms, called composers, compute new actuator states based on context, preferences, and system operation goals. For example, a light composer uses a room's available lamps, blinds, exterior light sensors, and activity monitoring to set the room's illumination to an occupant-specified level while minimizing energy consumption. Degeler et al. [23] proposed a service oriented, context-aware building management system that used composers to provide comfortable and energy efficient environment for the building's occupants. The problem of context-aware frameworks and composers is that the configuration is unsuitable for non-experts, and is labour-intensive for experts.

Smart-building behaviour is programmed with the creation of control rules which associate context events with composers, e.g., if an occupant is present in the room, activate the room's light composer. This approach to control rule creation is called a Triguer-Action Paradigm (TAP). TAP provides a logical programming mechanism that people with no technical background can use, albeit limited to associating a few trigger sources to specific actions[11, 107]. Furthermore, IoT motivates the integration of devices from different manufacturers, and TAP has been widely adopted by device management services such as IFTTT, smart assistants such as Amazon® Alexa™ or Google® home, which provide the functionality for people to conveniently operate their smart environments from one application. For TAP to work, all composers, direct- and inferred-context instances must be pre-defined in the system for non-experts to effectively program the building.

The challenge of commissioning a smart-building with IoT devices, context-aware frameworks, and TAP is that there is considerable effort in creating and configuring all required instance for each building location.

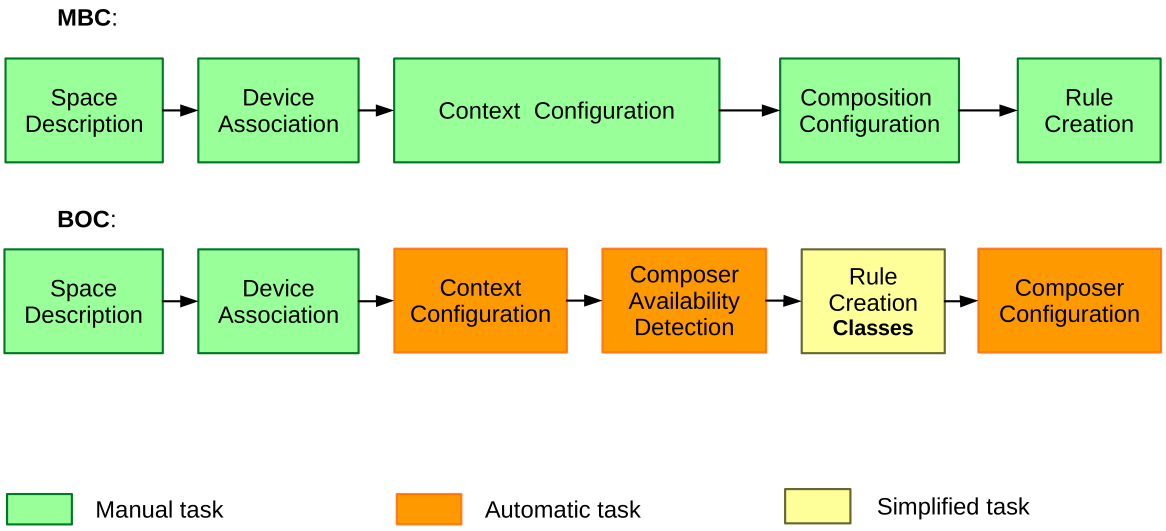


Figure 2.1: Comparison between the Manual Building Commissioning (MBC) process and our proposed approach the Building's Ontology Compiler (BOC). BOC automates the creation of inferred-context instances. In addition, BOC presents composer alternatives to the building-programmer for the rule creation step using the Class Rule programming Paradigm (CRP). Finally, BOC implements the selected composer classes and convert the class rules to instance rules.

Therefore, commissioning the smart-building is a labour intensive task that requires expertise. In general, the commissioning process for each building location goes as follows:

1. Describe the space.
2. Associate devices to locations in the space description.
3. Create and configure inferred-context framework instances.
4. Create and configure composer instances.
5. Specify control rules using TAP.

In this work we present the Building's Ontology Compiler (BOC), a methodology for simplifying the smart-building commissioning process. Our goal is to minimize complexity to expert building-programmers such that non-exeprt can also commission the smart-building. BOC works as follows, after the building-programmer associates devices to building locations, BOC instantiates all inferred-context instances. Then, BOC creates a list with all composers that it can instantiate per location. Context instances and composers are organized using an ontology into classes. BOC uses the Class Rule programming Paradigm (CRP) paradigm introduced by Corno et al. [18]. The rule creation process is simplified using CRP, as rules can be expressed using context, composer, and location classes instead of the specific instances. For example, in CRP, the rule "if Presence in Office activate Office's Light Composer" applies to all locations of type Office, and for each office activates the office's light composer based on its occupant presence condition. After rules are created using CRP, BOC converts the context and composer classes used in the rule to instances. Only the composers that match the class and location specified in the rules are instantiated from the composer list. The location class is used to determine the correct instances to associate, e.g., lamps, sensors, and composer from the same office. Fig. 2.1 illustrates the comparison between Manual Building Commissioning (MBC) and BOC.

We evaluate BOC with the configuration data for a four person office with a total of 17 sensors, 12 inferred context instances, and 8 actuators. We created three scenario descriptions of the same room and simulated the room lighting system for each description to evaluate energy saving using 337 days of recordings, which used the recorded presence and outdoor light data to simulate the lighting system response under each scenario description.

2.2 An Ontology for Smart-buildings

Building's Ontology Compiler (BOC) depends on an ontology in order to create inferred-context and composer instances. The ontology specifies seven main classes: location, context, composers, topic, algorithms, mapping and cost functions. Location class defines all the different types of spaces found in a building. The building's hierarchical structure is created by associating instances through properties. In particular, using the property

hasLocation, context and composers instances are assigned to the appropriate location instance, which in turn, can be assigned to a containing location instance. For example, *presence*1, *ceiling_lamp*1, *composer*1 *haveLocation desk*1, and *desk*1 *hasLocation room*1.

The context class is divided into three subclasses: object state/activity, environmental state and actuators. Object state/activity refers to context information which describes objects in the space, e.g., occupants. For example, occupant states like presence, fallen, or sitting, as well as activities such as computer work, desk work, talking. Environmental state refers to the physical variables which describe the space that contains objects. For example, temperature, humidity, and CO_2. Finally, following the definition from the Sensor-Observation-Sampling-Actuator ontology (SOSA) [19], actuators represent devices/services which can execute actions. For example, ceiling lamps, radiators, and notification services. A subsequent class hierarchy is proposed as follows: function, modality, device model. For example, Presence as function, Passive Infra-Red motion detectors (PIR) for modality, Wireless-self-powered-PIR sensors as device model. For object and environmental classes an inferred modality is added. As a result, inferred-context instances are classified in the ontology. For example, when presence in room one (*presence_1*) is derived from the room's people counter, then, it can be classified as follows: $presence\_1 \in [Context, Object\ state/activity, Presence, Inferred]$. Fig 2.2 illustrates other context hierarchy examples.

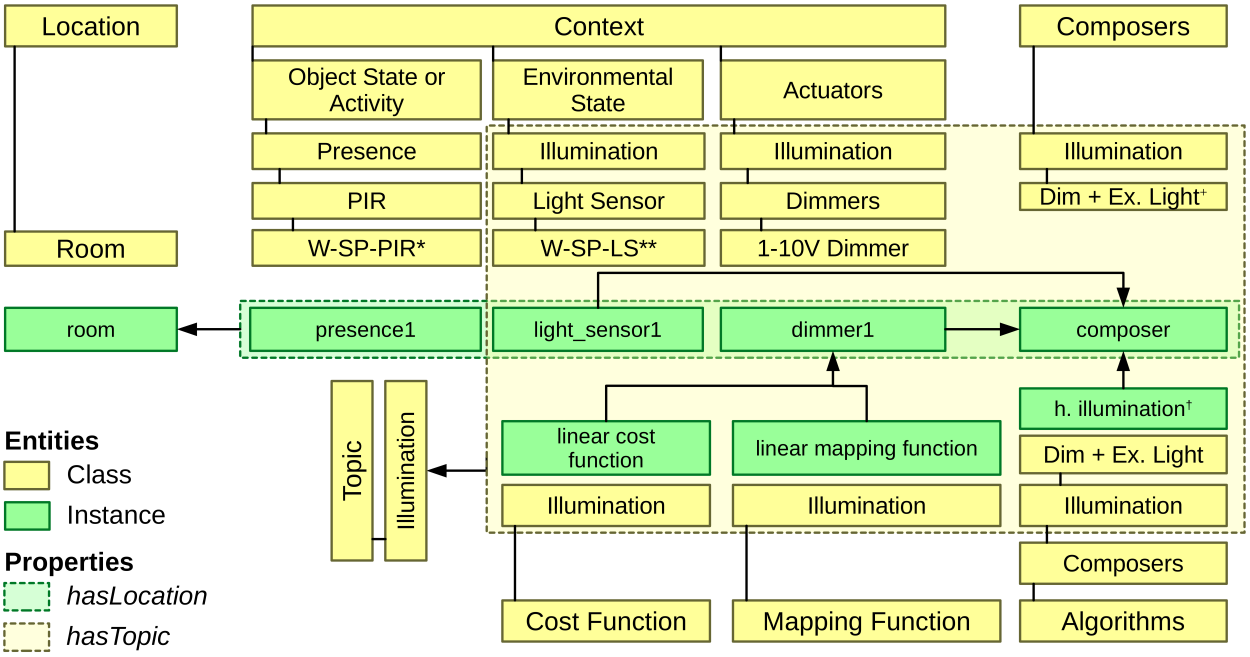


Figure 2.2: Ontology illustration of the asserted facts about a presence, a light sensors, an actuator, and a composer associated to a room instance. The arrows represent associations by properties as listed in Table 2.1.

* W-SP-PIT: Wireless self powered passive infra-red.
** W-SP-LS: Wireless self powered light sensor.
† h. illumination: Homogeneous illumination
+ Dim +Ex. Light: Dimmable lamps and exterior light sensor.

In Internet of Things (IoT) applications, specifically building automation, an alternative context categorization by function is called a Topic. For example, the illumination topic groups light sensors, ceiling lamps, and composers for illumination. Topic are better suited to group instances around environmental state classes, such as illumination, which can be mapped to a physical phenomenon or quantity. Then, the topic uses the environmental state's variable space as common ground to exchange information between the instances of a topic. Therefore, all devices under that topic provide interfaces to operate in the topic's value space. For example, the illumination topic uses an unsigned real variable space with a Lux unit. The common light sensor provides data already in Lux, the composer estimates lighting requirements also based in Lux and dimmable ceiling lamp actuators provide an interface to convert Lux to the 1|10 Volts format used by some commercial dimmer controllers. Topic interfaces might not be directly provided by the actuator, therefore we propose the Mapping and Cost functions classes to overcome the missing information. The mapping function class defines functions which have as input the topic's variable space and converts the value into an actuator understandable value, e.g., $f(200Lux) == 2V$. A cost function instance also takes as input the topic's variable space value and convert it into consumed resources, e.g., $f(200Lux) == 30W$. Energy aware composers can

Property	Inverse	Transitive	Domain	Range	Description
hasLocation	Contains		Location, Context, Composers	Location	Associates context, composer and location instances to their containing location.
hasLocationT	ContainsT	•	Location, Context, Composers	Location	Transitive version of hasLocation, used to traverse the location hierarchy.
hasTopic			Location, Context, Composers, Algorithms, Cost and mapping funcitons.	Topic	Associates a topic to classes or instances.
hasAlgorithm			Context, Composer	Algorithms	Associates algorithms with composers and inferred-context.
hasCostFunction			Actuators	Cost Function	Associates cost functions with actuators.
hasMappingFunction			Actuators	Mapping Function	Associates mapping functions with actuators.

Table 2.1: Object properties defined in the smart-building ontology.

then use the information to compute optimal solutions.

Context class granularity minimizes information input and promotes re-usability. For example, a building has one hundred dimmable ceiling lamp actuators, which come in tow models: one or two fluorescent tubes. Thus, the actuators are categorized as follows: context, actuators, illumination, dimmable lamps, model 1|model 2. If the difference between models is the power consumption and the maximum illumination provided, then, an energy saving composer can be instantiated with instances of either model, the required information to produce the correct illumination is retrieved from the ontology by consulting the actuator's associated mapping and cost functions.

A composer instance is classified in the ontology under the Composer class, which is an extension to Class Rule programming Paradigm (CRP)'s Action class. Corno et al. [18] defined the Action class in CRP as the set of commands each individual IoT devices or service can execute. We found that the definition limits the scope of the actions to a conjunction of commands per device. Instead, a composer instance represents an algorithm/procedure which can combine current context information and generate commands to multiple devices. For example, a homogeneous illumination composer can maintain a room's illumination by compensating changes in the exterior light with the room's dimmable lamps. The composer uses the position of each lamp with respect to the windows to provide homogeneous illumination distribution throughout the room. Alternatively, a composer can turn on lamps. Thus, a CRP command is redefined as the procedure to set a specific actuator to a given state. Therefore, commands are considered composers.

The Algorithms class is divided into inferred-context and composer subclasses, then, each subclass has the same hierarchy as the respective instance class which uses the algorithms. Fig 2.2 illustrates an example of the hierarchy used to classify the algorithm "homogeneous illumination". A composer or inferred-context instance is the parametrization of an algorithm for a specific location. Hence, the same algorithm can be used in multiple locations. Fig. 2.2 illustrates the ontology with asserted facts about a room, a light sensor, a presence sensor and a composer. The associated mapping and cost functions, and the composer's algorithm, and relationships created by properties are also shown in Fig. 2.2

The ontology defines the object properties listed and described in Table 2.1. For the building-programmer the most important property is *hasLocation*, which is used by the building-programmer to associate devices to locations, and to create the building hierarchy. The properties *hasAlgorithm* and the transitive version of *hasLocation* are used in the creation of templates which are explained in the next section. The remaining properties are intended for experts, and device manufacturers.

CRP proposes a device manufacturer/technology independent abstraction for creating rules, which is achieved by using an ontology to define context and command classes. In CRP, rules are formulated by using context as event sources, e.g., presence, and commands as actions, e.g., turn on lamps. With the rule "when presence, turn on lamps", using the ontology's reasoner, which lamps to turn on is determined by which rooms have presence. We continue the manufacturer/technology independent abstraction by proposing context and composer hierarchies, and the associated property classes. Further more, the mapping and cost functions compliment information which might not be provided by the device manufacturer. Therefore, the composers can operate in an abstraction level independent of the manufacturer/technology.

From Fig. 2.2 it is clear that the ontology requires many definitions and assertions. However, using device discovery and self registration technologies, such as universal plug-n-play or Kim et al.'s [51] seamless integration of heterogeneous devices, most of the registration burden can be passed to the devices' manufacturers. In other words, when a device is added to the network, it will register its information regarding context class hierarchy, mapping and cost functions if applicable. Although, we are aware that there is no standard for IoT device ontologies, there are techniques to incorporate and translate between the building's and the device ontology [1]. Additionally, algorithms can be independently developed and downloaded from a central repository, in the style of the IFTTT[43] platform, because the ontology is designed to operate independently of hardware manufacturers/technologies. As a result, the ontology can be maintained by experts, and for the specific smart-building installation only the building description, device associations and CRP rules are required.

In a manual commissioning process, once a complete description of the available devices, inferred-context and composers is created, frameworks like GreenerBuidlings [23] and Han et al. [38] can take over the smart-building runtime.

2.3 Ontology extension for the compiler: The Templates Class

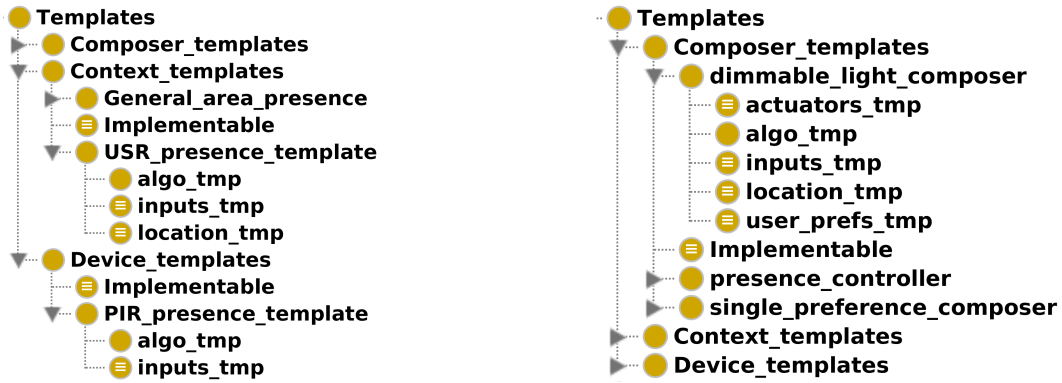


Figure 2.3: (a) Context templates defined in the smart-building ontology. The USR_presence_template defines presence, computer- and desk-work context classes based on two ultrasound ranging sensors installed in separate desk areas. The PIR_presence_template defines the presence context class for each passive infra red sensor available. The Implementable class groups all sibling templates which have instances in all requirement subclasses. (b) Composer templates defined in CCP's Ontology. The Implementable class groups all sibling templates which have instances in all requirement subclasses. The dimmable_light_composer_template uses desk and room light sensors and controls ceiling lamps independently.

Before Building's Ontology Compiler (BOC) can create inferred-context and composer instances, information has to be provided to guide the instantiation process. Therefore, we created three template classes used for controlling the instantiation process: Device, Context, and Composer templates. In general, a template class is designed to classify, by assertion or by inference, context, algorithms and location instances in grouped in respective requirement subclasses. A device template class classifies sensors as inputs, and an algorithm to derive information from each input individually. The inferred-context instance inherits the location from the input. The template has an assertion of the *hasOutput* property which links the template to the context class function. For example, Fig. 2.3a illustrates the taxonomy of the Device template PIR_presence, which looks for Passive Infra-Red motion detectors (PIR) motion sensors as inputs, and an algorithm that converts the motion sensor into presence. The template asserts that *hasOutput* some Presence. Context templates are used to specify inferred-context which is derived from multiple inputs. An additional location subclass is used to determine the locations where inputs will be merged using the algorithms. Fig. 2.3a illustrates the taxonomy of the Context template USR_presence, which searches for Ultra Sound Ranging sensors (USR) sensors installed in left and right areas of desks and the algorithm that converts USR sensors mounted on desk areas into presence, computer- and desk-work. The template asserts three *hasOutput* properties one for each algorithm's output class. In contrast to the other templates, the Composer template uses an actuator subclass to classify the actuators which will receive commands from the composer. The *hasOuput* property assertion is used to indicate the class of composer that will be instantiated. In addition, a user preference

subclass is used to set the composer's control reference when applicable. Fig. 2.3b illustrates the taxonomy of composer templates.

2.4 Building Ontology Compiler

Building's Ontology Compiler (BOC) uses the templates to automatically create inferred-context and composer instances. After the building-programmer has created the space description and assigned devices to their respective locations, BOC executes five steps: (1) execute the ontology reasoner to classifies the context instances into the template requirement classes, (2) add a data property assertion to each template for each requirement class that has instances, (3) run the ontology reasoner again to classify templates under the implementable class of each template type. (4) using the templates from the implementable class, create instances using the information contained in the template. (5) repeat from (1) until no new instances are created. For composer templates, when the compiler finds more than one implementable template, it does not run step (4). Instead, the composer availability list is presented to the building-programmer when creating the system rules. The list is populated by all composers templates that are classified as implementable. Finally, once the rule are specified using Class Rule programming Paradigm (CRP), the composers used in the rules are implemented according to the templates.

Goal: When there is presence in the room, dim the ceiling lamps as out door light increases.

Paradigm	Premise		Conclusion
CRP:	Presence **and** Room	→	DimmerController
CRP + BOC:	presence_room == True	→	dimmercontroller_room(ceiling_lamp, outdoor_light)
Degeler et al.:	presence_room == True	→	ceiling_lamp_room + outdoor_light_room > 1000lux
TAP:	presence_room == True	→	dimmercontroller_room(ceiling_lamp, outdoor_light)

Legend: Class, Instance, value

Figure 2.4: Comparison between Triguer-Action Paradigm (TAP), Degeler et al. [23], and Class Rule programming Paradigm (CRP) + the Building's Ontology Compiler (BOC) paradigms. We consider that Class Rule programming Paradigm (CRP) is a Triguer-Action Paradigm (TAP) superset, because CRP's compiled form is equivalent to a TAP rule.

Fig. 2.4 illustrates an example of a rule created using CRP, and BOC's compilation result. Additionally, we included for comparison the rules expressions defined by Degeler et al. [23] which could describe the composer functionality directly in the rule formulation, and the rule in TAP, assuming the composer is considered an available service. Fig. 2.4 shows that the compiled version of CRP rule is equivalent to TAP therefore, CRP is a superset of TAP.

2.5 Evaluation scenarios

Using the four person office from GreenerBuidlings dataset [23], we made a Building's Ontology Compiler (BOC)'s proof of concept. The room has four desks with computer screens, two screen-mounted Ultra Sound Ranging sensors (USR), one ceiling-mounted Passive Infra-Red motion detectors (PIR), and a light sensor on each desk's work surface. Additionally, the room is equipped with an outdoor-light sensor, and 15 dimmable fluorescent ceiling lamps controlled in pairs by seven 1-10V dimmer controllers and an extra single tube controller for the ceiling lamp over desk two. The room's layout is illustrated in Fig. 2.5. The room had the most utilization from the living-lab with 337 days of data recordings. For all three scenarios the evaluation goal is to configure all necessary inferred-context and composer instances such that energy is saved based on room or desk presence, based on a Class Rule programming Paradigm (CRP) rule of the form "if Presence then activate Energy-efficient-composer".

We tested three room description scenarios, where we simulated different sensors and actuators availability and location assignments: (1) room-with- presence-only, (2) room-with-outdoor-light, and (3) room-with-desks. Fig. 2.5 illustrates the room's layout and the device associations for the **room-with-desk** description, where a ceiling lamp controller u_i, a light λ_i and PIR sensor, were assigned to each desk d_i. Additionally, two USR sensors were mounted on the top corners of each desk's screen, which created a left and right desk areas. The remaining four ceiling lamp controllers, and the outdoor light sensor were assigned to the room. All

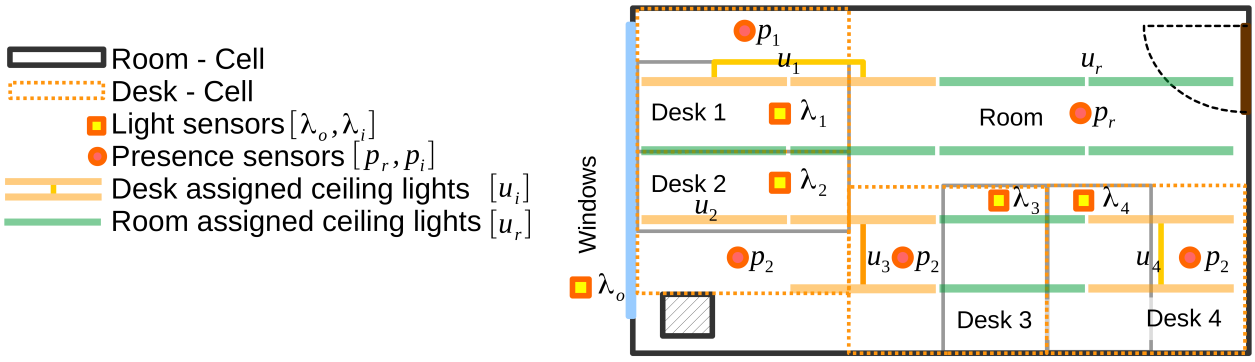


Figure 2.5: Room layout. Illustration of the positions of the desks, dimmable ceiling lamps, and sensors. Presence in the room P_r is an inferred context variable thus its location does correspond to a logical rather than a physical location. The layout reflects the room with ceiling lamps associated to the desks. In all other descriptions the ceiling lamps are associated to the room, and only the outdoor light sensor λ_o is used.

descriptions use the same location associations for the USR and PIR sensors. The next description, **room-with-presence-only**, assigns all ceiling lamps to the room and light sensors were not included. Finally, the **room-with-outdoor-light** description also assigned all ceiling lamps to the room , but it includes the outdoor light sensor λ_o. We used the GreenerBuidlings data recording of the individual desk presence and outdoor light to simulate the system's response to each description. We used the dimmers cost functions to convert the specific illumination requirements during the simulation to power consumption. Eq. 2.1 illustrates the cost function for the dimmer with one (c_1) and two (c_2) fluorescent light tubes. The functions use the change of illumination $\triangle x_l$. The constants were calculated using the recorded data.

$$\begin{aligned} c_1(\triangle x_l) &= 0.017 \triangle x_l + 21.25 \quad [\mathrm{W}] \\ c_2(\triangle x_l) &= 0.069 \triangle x_l + 9.75 \quad [\mathrm{W}] \end{aligned} \tag{2.1}$$

We tested two algorithms to compute the three USR based context classes: presence, computer- and desk-work. The first algorithm used windowed statistical analysis of the distance measurement and an estimation of the occupant's path on the field of view of both sensors to determine all three classes simultaneously [46]. The second method first detects presence on each individual sensor, by applying stochastic models of the presence and away conditions, then the individual outcomes were used to compute all three classes at the desk level [65]. Additionally, the USR's presence was aggregated with the PIR motion output with a sensor fusion algorithm [46]. Similarly, the room's presence is inferred from the disjunction of all four desk's presences. In general, a location's presence is the disjunction of all presence sources in that location including its sub-loactions. Hence, we created three templates: USR, with two algorithmic choices, PIR, and location presence. Fig. 2.6 illustrates the flow of compilation process to generate the inferred-context instances. Presence-based USR and PIR templates are instantiated in the fist compiler pass and the location-based presence template is instantiated in the second pass. By using the ontology reasoning, the compiler skips one pass and associates all sensor-based presence into the room's presence, which would be equivalent to associate desk presences in a third compiler pass.

We defined two composer templates for the illumination topic. The first template has an On/Off controller behaviour, when an instance of this composer template is enabled, it calls the dimmer's turn on interface, which sets the lights the maximum, e.g., 10V. The second template uses a light sensor in the location to control available dimmable lamps which compensate for diminishing exterior light. Specifically, Eq.2.2 describes the implementation of the composer algorithm for the optimal lamp control, where x_l is the illumination level of each lamp controller in location$l$, and $|\lambda_l$ is the light sensor. The algorithm then uses the dimmers mapping function, shown in Eq. 2.3, to convert from x_l to the corresponding 1|10 Volts value used by the dimmer's controller. Both dimmer models use the same mapping function.

$$\sum_{i \in l} (x_l + \lambda_l) > 1000\mathrm{Lux} \tag{2.2}$$

$$m(x_l) = 0.009\, x_l [\mathrm{V}] \tag{2.3}$$

BOC is evaluated in two aspects: feasibility and impact on energy consumption. Using the protégé ontology editor [79] we verified feasibility by creating a simple building description, where only the room, desks and

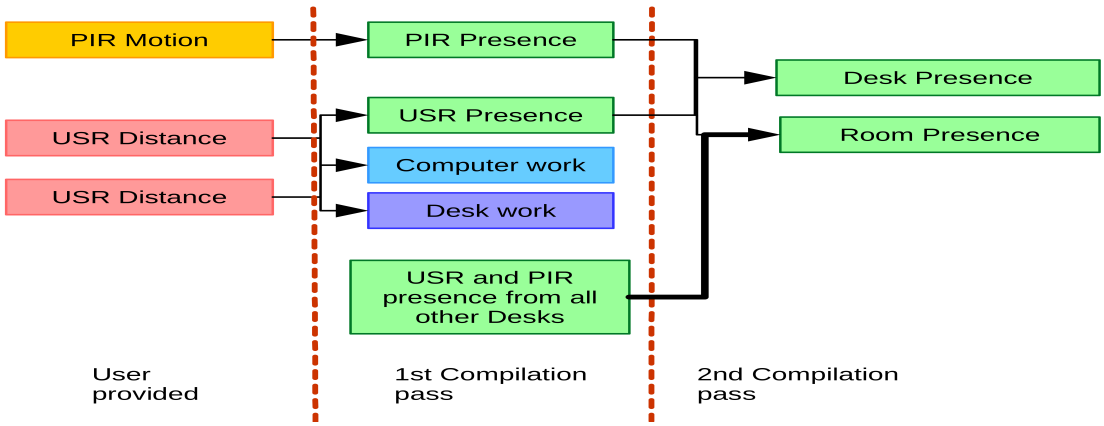


Figure 2.6: Illustration of Building's Ontology Compiler (BOC)'s inferred-context instance creation based on the device and space descriptions. Desk presence and room presence are implemented by the same template. Therefore, BOC saves a compilation pass.

desk areas were described. Our goal was to verify if the correct instances are created in the ontology, because a service oriented architecture, such as [23, 38], can use the composer instances from the ontology to configure their composition services. We created instances which represent the sensors and actuators, and associated them into the corresponding locations according to the different scenario descriptions. Finally, we implemented BOC using owlready2 [58] python library.

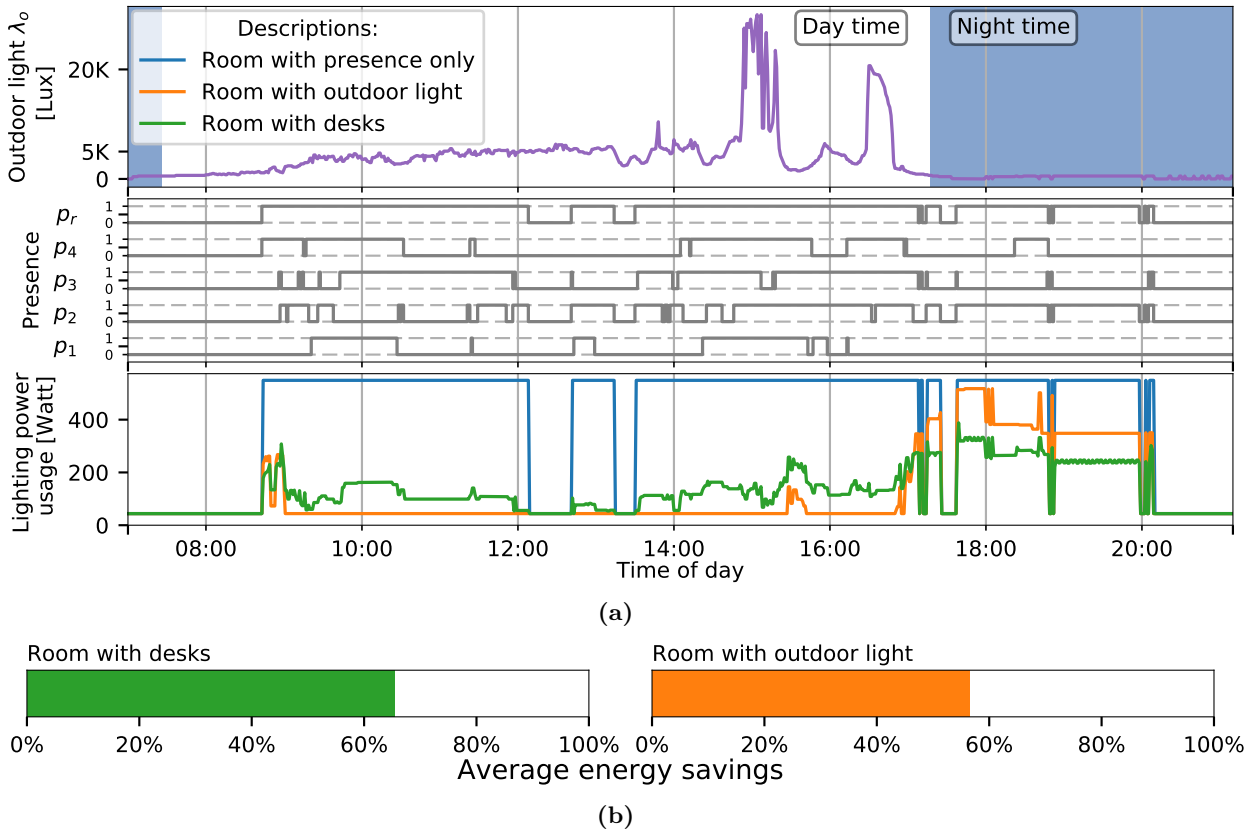
2.6 BOC's performance

Building's Ontology Compiler (BOC) successfully created inferred-context instances and associated the composers templates into their implementable class for each room description. Tab. 2.2 indicates which composer templates were implementable for each description •, we indicate with ⋆ which templates were selected for the energy saving evaluation.

	Optimal light control	On/Off control
Room-with-presence-only		•⋆
Room-with-outdoor-light	•⋆	•
Room-with-desks	•⋆	•

Table 2.2: Association between different room description and the implementable composers (•). Additionally, the table shows the room description – composer combination used for the energy saving (⋆) analysis.

We used the 337 days of desk-presence and light-sensor data to evaluate energy consumption performance of each description scenario. We used the *room-with-presence-only* description as power usage baseline. Fig. 2.7a illustrates a typical day at the office and the energy consumption as a result of the provided room description and sensor association. For the day illustrated in the example, the room and outdoor light description achieved the most energy saving with 78%. When compared to the room and desk description, we can see that the composer is completely shutting down the lights as a response to the high exterior illumination provided by the sensor. Thus, achieving the best energy savings. However, the desk light sensors from the room and the composer implemented as a result of the room-with-desks description maintains a level of illumination for the users present, which can bee seen as a comfort improvement. Further more, Fig. 2.7b illustrates that on average the room and desk description saves 65% and the room and outdoor light sensor only saves 56% of energy. Night hours and the unbalanced occupancy throughout the day explain why, on average, the room-with-desks description has better energy savings than the room-with-outdoor-light description.



(a)

(b)

Figure 2.7: Results for energy saving analysis for the four-person office. (a) Illustration of Outdoor light, presence, and illumination power needed for each composer, for a normal office day. *Room presence and outdoor light* composer is the most energy efficient, at 78% savings with respect to the room presence baseline. The *Context aware* composer gives all users an adequate illumination according to their preferences and still scores 68% of energy saving. However, (b) shows that over 337 days of recording, the *room and desk* description with the single user preference composer saves on average 65% of energy with respect to the *Room presence* baseline. In contrast, the *room presence and outdoor light* composer only saves 56% when compared to the *Room presence* baseline.

2.7 Future directions

In Building's Ontology Compiler (BOC)'s proof of concept we have showed that the templates can provide the information to automatically generate context and composer instances. However, we need to elaborate on how the compiler treats properties assertions in order to have better control over the generation of instances. For example, room presence was expected to be computed from desk presences only, and not from the individual sensor presence. Although, equivalent for the case of presence, we foresee applications where the specific inputs require more selectivity on the instances. For example, a situation where there are multiple inputs of the same class, but the algorithm require pairwise distinct associations.

We showed that Class Rule programming Paradigm (CRP)'s manufacturer's independent class abstraction also applies to composer algorithms regardless of sensor and lamp manufacturers. Furthermore, we showed how the optimal light composer used the mapping and cost functions to maintain control over the room's illumination. Additionally, similar to the ON/Off control composer, the turn on command also applies to other devices. Therefore, in our ontology, devices provide the necessary interfaces to be used. We illustrated the use of a procedural algorithm instance to implement the optimal light control composer. However, based on schemas design for the description of buildings and their equipment, e.g., BRICK [6], we envision that our approach can be extended to use description schemas as algorithms, or incorporate them into the context classes accordingly.

The ontology presented in this chapter can be consider a building runtime ontology. Furthermore, there are building information models which part of the design and manufacturing process, e.g., buildingSmart IFC [10] or gbXML[33]. We envision that both ontologies will be merged following a procedure similar to what Niknam et al. [85] proposed for unifying the design, cost and scheduling ontologies. When considering IoT devices from the design process, the individual device information is known in the cost ontology, therefore, the information can be merged with the building's description. Once the building is constructed, the devices only need the physical installation, and a set of factory rules following CRP's approach, and by using the compilation approach proposed in this chapter , the building will considerably minimize the commissioning time. Furthermore, with a detailed information model, the absolute position of ceiling lamps can be described and used to implement a composer that uses the "homogeneous lighting" algorithm.

Acknowledgements

The authors gratefully acknowledge the GreenerBuildings consortium www . greenerbuildings . eu, which developed an earlier open-source version of the runtime engine presented here, and the supported by the "Bavarian State Ministry of Education, Science and the Arts" as part of the FORSEC www . bayforsec . de research association.

Chapter 3

Mining relations and physical grouping of building-embedded sensors and actuators

Chapter originally published as: L. I. Lopera Gonzalez and O. Amft. "Mining relations and physical grouping of building-embedded sensors and actuators". In: *PerCom 2015: Proceedings of the International Conference on Pervasive Computing and Communications*. IEEE, Mar. 2015, pp. 2–10. DOI: 10.1109/PERCOM.2015.7146503

3.1 Introduction

In modern buildings, various sensors and actuators are used to control lighting, blinds, Heating, Ventilation, and Air-Conditioning (HVAC), as well as appliances. Frequently, motion detectors and temperature sensors are deployed per desk to adjust lighting and HVAC system [3, 28, 69]. Milenkovic et al. [73] showed that several office activities can be recognised from sensors commonly found in office buildings, e.g. to save energy. Furthermore, adding sensor modalities, such as proximity sensors at office desks helps to control computer screens according to user activity and increase comfort [46]. Overall, offices may contain some 5-10 embedded sensors and actuators around a desk, resulting in up to 50 measurement and status variables in a 5-user office space. For an average 10-floor office building with some 500 desks, ∼5000 variables can be expected, not considering variables originating from corridors, meeting rooms, and common areas. Measurement and status information is typically managed via dedicated building networks and processed in a building management system (Building Management System (BMS)). With the continuous increase in building functionality, the amount of variables that need to be processed in a BMS grows and creates substantial challenges for classical manual commissioning and continuous management of a building.

During building commissioning, sensors and actuators need to be linked into BMS rules that determine functionality, including automation functions, such as moving blinds up and down depending on desk lighting state. Further context information can be derived, such as counting the number of users occupying an office space at any moment. User count information is relevant for a BMS to adaptively control the HVAC system of an office space [74]. For example, presence in an office room could be determined as a logic combination of individual desk presence variables and used to control overhead lighting. An example BMS presence rule then is:

$$\textbf{If } \textit{presence} \textbf{ in} : \textit{desk}1|\textit{desk}2|\textit{desk}3 == \textbf{True}$$
$$\Rightarrow \textbf{Then} : \textit{OverHeadLight}1 == 50\%$$

The exemplary presence rule shown still ignores relevant information, such as the outside lighting conditions. As a consequence, rules are typically complex and are constructed hierarchically, which further increases overall variable count in a BMS. While attempts have been made to structure information and apply service oriented approaches for BMS [23], the challenge of verifying correct configuration and continuously updating rules persist. Rule updates are necessary following revisions in installed devices, refurbishment, and rearrangement of building spaces, which renders any manual configuration management labour-intensive and error-prone. An automatic inference of relationships and location groups among building variables could help building managers to extract and optimise rules and identify configuration errors, for example when a presence detector of one desk was erroneously matched to a light of another desk.

In this chapter , we introduce a framework to automatically group building variables by their physical location. As building variables, we consider data provided by a BMS system, which represents a sensor measurement, actuator state, or values derived by processing sensor data. Building variables are multi-modal and represents different physical and virtual quantities.

In our approach, we exploit the fact that occupants use and interact with a building during everyday activity, producing streams of sensor and actuator events, i.e. building variable changes. Building variables that change state in recurring temporal relation could indicate that they share the same observation space, hence location. We infer rules that represent variable relationship using association rule mining. We hypothesise that spatial relation of building variables is reflected in the mined rules and therefore variable groups can be extracted reflecting physical location. For example, a room presence detector and any lights activated by that presence detector will appear in the same location group. Additional variables related to the HVAC system of the same room, may be located in a nearby technical closet. Hence, our approach aims at extracting all variables, including lights, presence detectors, HVAC-related, etc. that belong to the same location group. In particular, the contributions of this work are the following:

1. We present a novel grouping framework to (1) consistently derive events from various sensor and actuator modalities to obtain building variables, (2) extract temporal association rule relations among variables, and (3) derives variable groups using transitive grouping. Our grouping framework can be considered as fundamental for various applications. The derived groups can be used to validate BMS operation or to provide new rules for additional building automation functions.
2. We evaluate variable grouping performance of our approach and compare to a manually configured BMS and a baseline using standard hierarchical agglomorative clustering. For our evaluation, we consider operational data gathered over several months from three living-lab office rooms of different type and purpose that are typical for office buildings: a ∼20 people meeting room, a 4-people office room, and a 12-people open desk space.
3. We illustrate potential applications of our grouping framework in several situations. First, we derive new rules for counting people in a room, which is relevant for advanced HVAC operations. For example, more air circulation is necessary when more people use a building space. The corresponding BMS rules could be automatically derived by our grouping framework and suggested to the building manager. We then show how BMS rule errors could be identified based on the derived variable grouping.

In this work, we focus on the grouping framework and its performance in realistic office conditions with actual occupants who regularly work or use the rooms and thus show application examples. We consider that our approach is relevant to office buildings, but similarly for public buildings such as schools, universities, hotels, and potentially for modern private houses that use networked ubiquitous sensors and actuators for automation.

3.2 Related Work

The use of tools and applications to aid in building management has been an active field of study. Jianguo et al [25] used statistical models to estimate if a fault would occur in the future. Modelling building complexity and different modalities found in the building is challenging. Practical approaches aim at modelling partial behaviour, and in general benefit from location information to reduce model complexity. A building is a perfect example of an expert system, governed by prepositional rules. One situation, illustrating the uncertainty in building variable relations, is the control of interior lights and blinds based on outside light information, where location affects sensors and actuators coupling.

Data generated by all the variables in the building can be seen as a time-series of events. Furthermore, a sequence triggered by a single user looks like a very specific event pattern. The following event sequence illustrates one such pattern: (1) Going into the office (P), (2) turning on the lights (L), (3) working at the computer (W). The resulting pattern in the stream is PLW. Bayardo [7] proposes a method to extract such patterns from large data streams. However, the reality is that the events P,L,W are shuffled with events coming from other sources. i.g. other offices. The method relies on the definition of transaction, which translate to use an observation window to extract patterns of fixed length. Selecting and dynamically adjusting the observation window length is still an open problem.

The approach proposed in this work uses variable associations to overcome modelling any internal processes. Agarwal et al [3] look for associations rules in the data and deals with big datasets, unfortunately, time is not included. Yu et al [117] introduces the concept of adding time as feature although applied in the context of web searches. Guillame-Bert and Crowley [35, 36] presents an algorithm with an application to mine rules extracted from a time series of events.

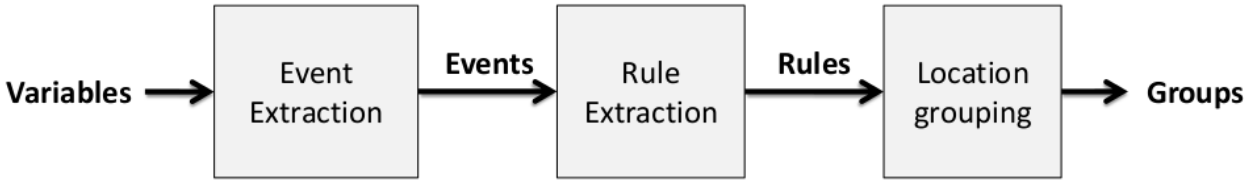


Figure 3.1: Schematic overview of our variable grouping framework. Based on building variables of a Building Management System (BMS) an event sequence was derived, providing event label and time-stamp information. From the event sequence logic rules were discovered using the Temporal Interval Tree Association rule learning (TITArl) algorithm. Finally, variables were grouped according to physical location using Weighted Transitive Clustering (WTC) method.

In the field of pervasive computing other approaches have been applied to extract rules with different purposes: Rollins et al [98] extract rules to detect energy usage patterns in homes and [22] propose a novel method to learn the rules of the system in an automated way. Rule mining is not the only method to find associations of variables. Krishnan et al [55] propose to discover the taxonomy in activity patterns. The resulting hierarchy could indeed convey information about activity location. Unfortunately the activity patterns seem to be unique for an entire home. In the building environment, the activity patterns are repeated through out the building.

Our approach extracts rules from the event stream of building variables to mine sensors and actuators that share a location. Rules and their corresponding confidence values create an unordered space, where distances can not be defined. To our knowledge, no machine learning algorithms exist that operate seamlessly in this kind of feature space.

3.3 Approach Overview

Our building variables grouping approach relies on the following two ideas: (1) Variables that belong to the same physical space will change value in some temporal relation. In other words, there will be a strong correlation between *when* the variables that are grouped together change value. In contrast, variables from different areas will not show recurrent co-articulation of their events. (2) Based on the rules relating variables, we can infer which variables relate to the same physical space and thus should be grouped together.

Implicit interaction with the environment, e.g. opening a door, walking in a room, flipping a light switch, etc. causes events across different building variables to correlate. Variable correlation is larger when the variables are located close to each other. As an example, correlation between a hallway movement sensor and a desk computer screen state is lower than the correlation between the computer screen state and a presence at the same desk. We use rule discovery to measure the variable correlation and group variables that belong to a specific location.

We propose a three step architecture to mine relevant location groups of building variables without supervision, in particular: (1) Derive an event sequence from all variables in a BMS by quantising their state or value. Here we abstract from the particular variable meaning and modality, as any information other than state or value require specific information from the particular BMS used, or supervised information, e.g. from an expert. (2) Discover prepositional logic rules from the event time series that relate the variables based on event occurrence correlation. Here we used the TITArl algorithm, designed for learning temporal association rules from symbolic sequences [35]. (3) Derive variable groups from the mined rules. Here we propose WTC approach that - as we show in this work - outperformed standard hierarchical clustering on the mined rules. Fig. 3.1 illustrates the framework components.

3.4 Variable grouping framework

In this section, we explain the functionality of all framework components and detail algorithms and implementation.

Event extraction

Often variable relations in Building Management System (BMS) are made using direct ascending mapping. Hence, if variable A is related to variable B ($A \Rightarrow B$), then an increase in value of A will produce an increase in value of B. For example, if $presence == true$ in a room then $light = On$. Similar mappings also apply for more complex variable relations, which are not joined by a logic relationship. For example, when a user sets the thermostat at a higher temperature, the HVAC system will react by starting the heating unit, reflected in higher energy consumption.

To derive an event sequence, we consider ascending changes in all building variables. The sequence is formed by event label (according to variable name) and timestamp. Table 3.1 shows an example of event sequence.

Table 3.1: Example of event sequence derived using the event generator component.

Event timestamp	Event label
1353935850.335012	powerconsumption11
1353935868.344362	powerconsumption2
1353935898.467623	presence3
1353935900.453503	luxlevel3
1353935934.376489	powerconsumption1
1353935936.701680	powerconsumption4
1353935946.382288	powerconsumption3

Implementation: To construct the event sequence, we searched in all available variables v_i at any given moment t for value changes using the condition $\frac{\Delta v_i}{\delta t} > \Theta_i$. Some measurement variables contain noise that does not qualify as suitable value change. We applied threshold filters with $\Theta_i > 0$, as to only derive events when the variable change is larger than threshold Θ_i. The event stream is formed by storing label and timestamp of events from all building variables in order of occurrence.

Rules extraction

Rule extraction aims to discover temporal association rules from the event sequence. Temporal association rules are logic relations of the form $A \Rightarrow B$. When A is observed, B is expected to occur before a time laps τ. In addition, we obtain a rule confidence and denote the support of a rule. Here we consider rule confidence as: given all events of label A, how many times $A \Rightarrow B$ occurred within τ. The concept of support can be defined as: given all events with label B, how many B events can be produced by the rule $A \Rightarrow B$ within τ. Rule confidence and support are further detailed in [4, 35]. Table 3.2 shows an example of mined rules.

Implementation: We used the Temporal Interval Tree Association rule learning (TITArl) algorithm by Guillame-Bert and Crowley [35]. TITArl discovers rules in the event sequence by tracking the temporal distribution of candidate event occurrences and their correlation over time. TITArl provides a validity period, indicating the rule lifetime in a event sequence. TITArl requires a series of parameters to effectively process event sequences. In this work, we chose the following settings: *inputEvent* predicate ".∗", *outputEvent* predicate ".∗", time window future $15s$, time window past $15s$, minimum rule confidence 0.05, minimum support 0.05, minimum number of use 2, all other settings were left at their default configuration. We used very low settings for confidence, support, and number of uses to obtain a sensitive rule mining behaviour. Subsequently, we studied performance in the group creator module.

Location grouping

We define a group as set of variables that are related to each other through a rule or a set of rules. The smallest possible group could be constructed from a single two-element rule, i.e. $A \Rightarrow B$ produces group $\{A, B\}$. Groups do not have order, hence $A \Rightarrow B$ and $B \Rightarrow A$ produce the same group $\{A, B\}$. Larger groups can be constructed from a union of smaller groups that have common variables. For example, $A \Rightarrow B$ and $C \Rightarrow B$ yield groups $\{A, B\}$ and $\{C, B\}$. Since both rules contain B, they are united to $\{A, B, C\}$. Equation 3.1 states the general grouping algorithm in recursive form.

Table 3.2: Example of mined rules with corresponding rule confidence and support.

Rule	Confidence	Support
$blindsangle3 \Rightarrow dimmer2$	0.05	0.05
$dimmer1 \Rightarrow blindsheight2$	0.05	0.06
$blindsangle2 \Rightarrow blindsheight2$	0.05	0.11
$blindsangle1 \Rightarrow dimmer2$	0.05	0.25
$luxlevel1 \Rightarrow luxlevelout2$	0.05	0.10
$deskwork9 \Rightarrow presence3$	0.05	0.06
$door1 \Rightarrow statuslights2$	1.00	1.00
$powerconsumption1 \Rightarrow statuscomputer1$	1.00	1.00
$statusbeamer1 \Rightarrow presentation1$	1.00	1.00

$$G_{k+1} = \left\{ \bigcup_{i=1}^{N} \left\{ \bigcup_{j=1}^{N} g_i \cap g_j \quad \textbf{if} \quad g_i \cap g_j \neq \emptyset \right\} \right\} \tag{3.1}$$

The grouping is resolved iteratively. G_k is the set of all groups at iteration k, g_i and g_j are any single group from G_k and N is the number of groups in G_k. If G_{k+1} contains two or more groups with the same elements, these groups will be merged into one group. The grouping procedure is terminated if $G_{k+1} == G_k$. We refer to this technique as transitive clustering, as it resembles the transitive property of mathematical operations ($a = b \wedge b = c \Rightarrow a = c$).

When error-less rules were mined or expert-defined rules are considered, Eq. 3.1 can be applied and will produce correct groups. However, rules mined from actual event sequences contain errors that must be avoided. We utilise confidence and support of rules to deal with errors. When starting the grouping, variables may appear in multiple groups with different confidence values. We define *group confidence* (GC) as the average over the maximum confidence at which each variable joined a group. The first groups are derived from the mined rules, thus variables join a group with the confidence of the rule which contained it. In order to merge two groups, we compute the *group threshold*(GT) as shown in Equation. 3.2.

$$GT = GC - bw \times \text{var}(MCV) \tag{3.2}$$

Where bw is the acceptance bandwidth and MCV is the array of the maximum confidence at which each variable joined a group. Equation 3.3 denotes the new concept.

$$G_{k+1} = \left\{ \bigcup_{i=1}^{N} \left\{ \bigcup_{j=1}^{N} g_i \cap g_j \ \textbf{if} \ \begin{array}{l} g_i \cap g_j \neq \emptyset \\ GC(g_j) > GT(g_i) \end{array} \right\} \right\} \tag{3.3}$$

Implementation: The location grouping module was implemented in two modes: as Trivial Transitive Clustering (TTC) and Weighted Transitive Clustering (WTC). TTC is the implementation of the algorithm described in Eq. 3.1. TTC is used to process rules that come from sources that do not provide confidence or support information, i.e. expert-defined rules. WTC implements the algorithm described in Eq. 3.3 and is used to process rules mined by the rule extraction module. WTC uses a rule acceptance threshold (RT) to filter out rules that have weak confidence. WTC parameters RT, bw, GC, and GT are updated on every iteration. The WTC algorithm can run continuously as new rules become available. For WTC all incoming rules that pass the rule confidence threshold are permanently stored. With new incoming rules, existing variable grouping is revised. WTC differs from TTC in two main aspects: For WTC, (1) new groups have to pass the RT and (2) confidence information is used to decide whether or not to merge groups.

3.5 Evaluation Methodology

We evaluated our framework for grouping building variables with real-life data from three office spaces. In this section, we describe the dataset, evaluation metrics, and the comparative analysis.

(a) Meeting room (R1)

(b) 12-people office (R3).

Figure 3.2: View into the different room types in the GB living-lab.

Evaluation Dataset

The GreenerBuildings (GB) living-lab dataset was collected over the span of 14 months, with recordings in three office rooms on the TU Eindhoven campus; R1: a ∼20-people meeting room (13 months), R2: a 4-people office (14 months), and R3: a 12-people open desk space (2 months). Table 3.4 shows the available variables types per room. In total, the living-lab recorded 243 variables. A Building Management System (BMS) was managing each living-lab room. Fig. 3.2 shows examples of the room types.

During recordings the living-lab was constantly updated and extended, devices were installed and broken ones replaced, as well as sampling frequencies revised, sensor range and resolution modified, all according to individual analysis goals. With the device changes, variable count changed and processing algorithms to derive variables changed too. For example, presence at the desk was computed from a motion detector or ultrasound ranging sensors, or by analysing desk power consumption. BMS rules where added and modified to take advantage of additional information and optimize energy consumption and user comfort. We consider that for our mining and grouping analysis in this work, the modifications reflect part of an office building life-cycle, where often changes and adjustments are made.

Table 3.3 shows the typical sensors, actuator and context variables associated to a desk cell in our living-lab. In a hierarchical aggregation, desk cells are associated to a room cell. Rooms provide common services to underlying cells. When building rules in a BMS are formulated, room variables usually appear in tandem with desk variables. For example, the room has one outdoor light sensor, thus, each desk uses that outdoor sensor to estimate dimming level for the corresponding overhead lamp. The exact behaviour is described by the building rules. Finally rooms are associated to floors, and floors to the buildings. In addition to desk cells, we use areas to denote spaces in the meeting room, where each area contains a presence sensor, temperature sensor, and light sensor.

Table 3.3: Variables associated to a typical office desk. In a hierarchical aggregation, desk cells are associated to a room cell.

Sensors and actuators			
appliance power-meters	3	appliance switches	3
overhead lamp	1	desk lamp	1
ultrasound sensors	2	movement sensor	1
light sensor	1	override switch	2
Context variables			
presence	1	deskwork	1
computer-work	1		

Performance metrics

Our framework aims at creating variable groups that represent spatial entities, e.g. a desk or a room. We use the clustering metrics homogeneity and completeness to describe the variable grouping performance in office rooms [99]. Homogeneity measures if all result clusters contain only data points which are members of a single class. Completeness measures if all the data points that are members of a given class are elements of the same cluster. Both metrics are in the range $[0, 1]$, where 1 indicates a perfect clustering result.

	Rooms				Rooms		
Variable type	R1	R2	R3	Variable type	R1	R2	R3
blindsangle	3	0	0	outdoortemperature	1	0	0
blindsheight	3	0	0	piroverhead	1	5	3
brainstorming	1	0	0	powerconsumption	3	12	12
co2level	1	0	0	presence	3	4	12
computerwork	0	4	12	presenceroom	1	1	0
deskwork	0	4	12	presentation	1	0	0
dimmer	4	6	0	screenswitch	0	5	0
distance	1	0	0	statusbeamer	1	0	0
door	1	1	0	statuscomputer	0	4	0
humidity	4	0	0	statushvac	2	0	0
hvac	2	0	0	statuslamp	0	4	0
lamp	0	4	0	statuslights	2	2	0
luxlevel	1	4	0	statusscreen	0	4	0
luxlevelout	2	1	0	switch	0	0	13
motionin	1	1	0	temperature	4	0	0
motionout	1	1	0	user	0	0	24
numberofpeople	1	1	0	usr	0	0	24
outdoorco2level	1	0	0	window	2	2	0
outdoorhumidity	1	0	0				

Table 3.4: Variables available per living-lab office room considered in our evaluation study. R1: ∼20-people meeting room, R2: 4-people office, R3: 12-people open desk space.

Homogeneity and completeness measure grouping quality. In addition, we proposed two new metrics to measure the capacity of the framework to group all variables (relative coverage) and understand how many groups are created (cluster factor). Relative coverage is derived as the ratio between variables that appear in the groups and variables available in considered evaluation period. Due to the changes in the living-lab during the recordings, we only consider variables available in a given period as a fair way to evaluate our system. Cluster factor is derived as ratio between identified group count and expected groups count. Cluster factor is thus a relative performance metric to evaluate group count against ground truth for any time period in the dataset.

Ground truth

The living-lab variables were defined including location information in their names. Specifically, all variable names contained a reference to the room that they belong to, e.g. $deskwork1\_R1$, indicating room R1. The ground truth groups, i.e. the rooms, were derived from the names of all variables available for a given period.

Comparative analysis

Initially, we extracted expert rules from the BMS and applied the Trivial Transitive Clustering (TTC) algorithm to derive groups. Further, to quantify the benefit of our algorithms over the performance that random guessing yields, we generated rules by randomly selecting variables from the total variable set. The generated rules were subsequently grouped using TTC. To obtain a random performance baseline, the process was repeated 10000 times and performance metrics averaged.

We selected an evaluation period length of one month, aligned with calendar months. One month was chosen, as it increased chances of gathering multiple events for all variables. In our evaluation, rules were parsed from the Temporal Interval Tree Association rule learning (TITArl) xml files. Each rule was encoded into a group containing all variables mentioned in the head or body of the rule. Each group is accompanied by a confidence vector containing rule confidence, support, number of events predicted, number of events to predict, and a counter on how many times a rule containing those variables was observed. As a result, GC,

Table 3.5: Comparison of average performance across different evaluated methods. Random association was used as a baseline. HAC was included to assess the benefit of our proposed WTC algorithm. The low homogeneity scores of HAC indicate that it is not suitable for variable grouping into office spaces. WTC homogeneity and completeness high scores indicate that the algorithm is creating relevant associations, however at lower coverage and higher cluster count.

	Rand	HAC	WTC
Homogeneity	0.0067	0.0729	0.9635
Completeness	0.0682	0.9310	0.9697
Relative coverage	0.8651	1.0000	0.2189
Cluster factor	4.0230	1.0000	3.1905

GT, RT become vectors. Based on the parsed rules, parameter sweeps were performed to obtain RT and bw. The best parameters were selected by maximising the sum over all metrics (except for cluster factor) after running Weighted Transitive Clustering (WTC) in each step.

As there are no established algorithms for rule grouping and besides the random performance baseline, we used Hierarchical Agglomerative Clustering (HAC) for comparison to our approach. Our HAC implementation relies on a linkage matrix built by assigning arbitrary indexes to the variable names and using the rule confidence as linkage value. We used the same RT as in the WTC evaluation to obtain the same set of initial rules. To implement HAC, we used the scikit library [94] and set the targeted clusters to the number of rooms expected for that evaluation period, which results in a cluster factor of 1 (cf. Sec. 3.5).

Finally, a noise analysis was performed, to confirm WTC robustness. For this purpose, we randomly deleted up to 25% of the rules and inserted up to 25% of erroneous rules constructed by randomly selecting variables assigned with the highest confidence vector. We then simulated deletions and insertions in variable events.

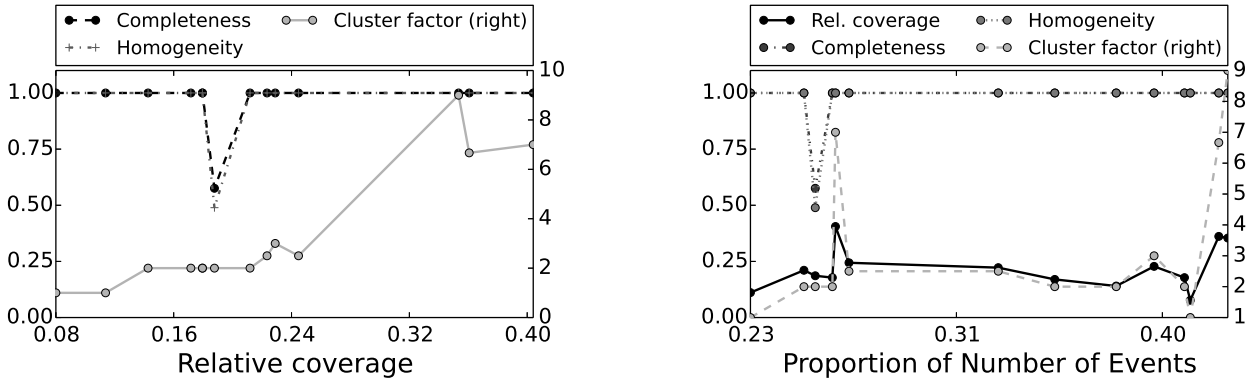
3.6 Results

The random performance baseline and HAC were considered to assess benefits of our proposed Weighted Transitive Clustering (WTC) algorithm, as described in Sec. 3.5. Table 3.5 shows a comparison of all three methods for all four performance metrics individually averaged. While random association cannot obtain reasonable clustering performance, HAC can obtain completeness above 0.9, similar to WTC. However, HAC fails to obtain homogeneity, hence many variables of one space are confused into different clusters. WTC can perform above 0.9 on both, homogeneity and completeness. HAC obtained perfect cluster factor because the expected number of variable groups was configured as a parameter. Full relative coverage could be achieved, since HAC is designed to assign all instances to clusters. While WTC obtained a relative coverage of ~ 0.22, manually grouping all rules contained in the living-lab Building Management System (BMS) showed a relative coverage of 0.14 only. Hence, WTC still mined a larger set of rules and building variables than those used in the operational BMS. The relatively large cluster factor confirms that WTC provides sub-room resolution in building variable grouping, i.e. variable groups that are subsets of the targeted room group.

Fig. 3.3a shows monthly averages of the WTC performance sorted according to relative coverage. In contrast to the average metrics show in Tab. 3.5 above, we analyse here monthly performances individually. As the figure shows, WTC provides high homogeneity and completeness independent of relative coverage. Cluster factor increased with relative coverage, indicating that when more variables are grouped, a sub-room clustering is obtained.

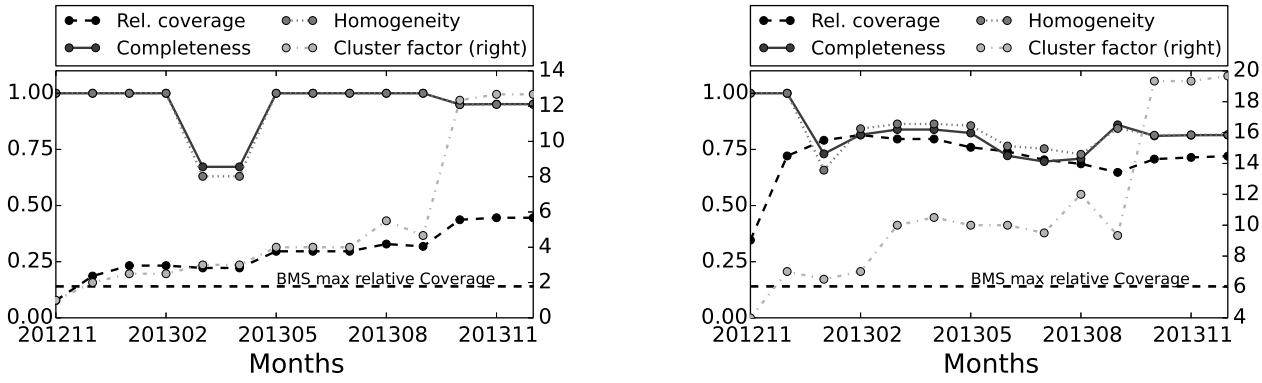
Fig. 3.3b shows the performance metrics over the number of events used to create groups, as more events suggest that the mining and grouping becomes more challenging. As the figure indicates, completeness and homogeneity metrics were not affected by proportion of events used to create groups.

We subsequently applied WTC continuously to the data, hence not for individual months only. Fig. 3.4a shows performances for the rule confidence threshold optimized to achieve best completeness and homogeneity performance. Fig. 3.4a also shows that the framework is capable of fixing grouping mistakes, as seen by the drops and subsequent recovery of completeness and homogeneity. The standard deviation of available variables per month is of 35.27 variables this shows that adding and removing devices has no effect in the performance of the framework. When optimising for clustering performance, relative coverage reached 30% only and it would take about 5 months to reach this coverage. Moreover, the sub-room clustering effect was observed here too. Fig. 3.4b shows performances for optimising the rule confidence threshold for best relative coverage.



(a) **(b)**

Figure 3.3: (a) Weighted Transitive Clustering (WTC) performance metrics vs. relative coverage. Each data point corresponds to one evaluation month. WTC provides high homogeneity and completeness independent of relative coverage. (b) WTC performance metrics vs. number of events used to create groups. Each data point corresponds to one evaluation month. WTC completeness and homogeneity metrics were not affected by proportion of events used.



(a) Clustering performance **(b)** Relative Coverage

Figure 3.4: Continuous mode performance over the study months with rule confidence threshold optimised for (a) clustering performance and (b) relative coverage.

In this configuration, WTC acquired higher relative coverage ($\sim 75\%$) and converged within two observation months. In contrast, using BMS rules only resulted in a relative coverage of only 14%. Temporal Interval Tree Association rule learning (TITArl) parameters for achieving maximum relative coverage were 0.1 for rule confidence, 0.01 for rule support, and 0.4 for group bandwidth.

We furthermore analysed the effect of basic noise as deletion and insertion on the rule mining and grouping performance. Removing rules (deletions) up to 25% reduced relative coverage by less then 2% as shown in Fig. 3.5b. Inserting wrong rules directly impacted homogeneity and completeness performance. Fig. 3.5a shows the insertion scaling. The rapid performance drop suggests that rule mining errors do critically affect operation. It is clear that various other forms and distributions of noise could occur under real-life conditions.

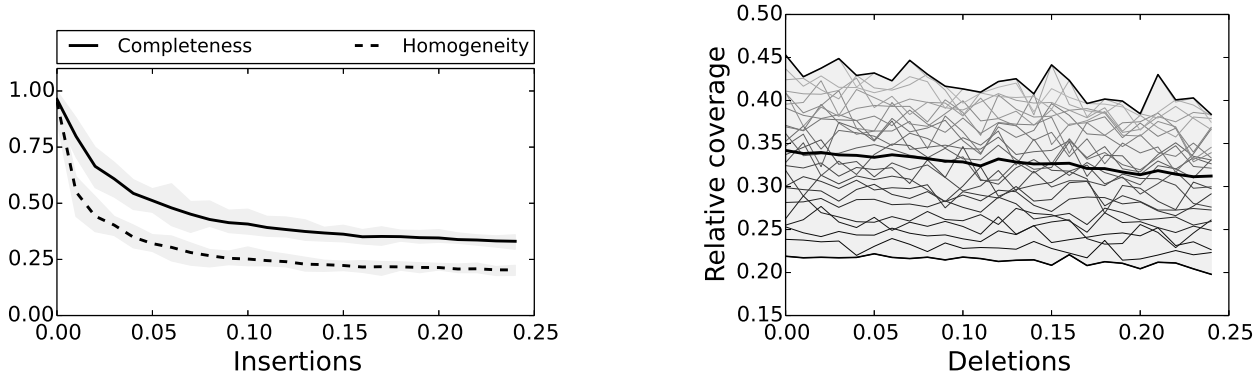


Figure 3.5: (a) Insertion noise analysis showing homogeneity and coverage vs. percentage of inserted erroneous rule rules. Lines indicate the average behaviour and shaded areas shows a sweep over different rule deletion settings (0% to 24%, in 1% increments). Insertions were obtained using erroneous rules set to highest confidence. (b) Deletion noise analysis showing relative coverage vs. deleted rules. Shaded region shows a sweep over different rule insertion settings (0% to 24%, in 1% increments). Deletions causes the relative coverage to drop as each deleted rule will remove variables.

3.7 Example Applications

In the context of large building installations, the following example applications illustrate how our variables grouping by location can be used.

People counting estimation

In modern buildings, people counting is relevant for safety, comfort, and energy saving. However, accurately measuring the number of people in the room is still a work in progress. In this application we estimate people count in monitored areas using presence sensors. For simplicity, we assume that presence sensors map directly to people count, which was adequate in all office desk rooms of our living-lab. We consider here a BMS application that would work by (1) a building manager writes a global rule of the form $numberOfPeople$ in space = count of all *presence* sensors that are activated. (2) The application uses variable groups found by our WTC and applies the global rule. (3) Finally, the BMS obtains new people count variables and can perform counting, given the rules. We applied the global rule idea to our living-lab rooms considered in this work. Fig. 3.6a shows the average normalised accuracy with respect to available presence variables during each evaluation period. Results indicate that the estimation performance is above 75% across all rooms.

Detecting/correcting variable naming error

BMS variables are commonly named including a location label. For example, BMS variables could contain type, unique counter within the group, and location label, such as $deskwork1\_R1$, where variable type = $deskwork$, unique counter = 1, location label: $R1$. Our framework can detect location reference errors in variable names following these steps: (1) Generate a group location label by majority voting among group variables. (2a) Variables that have a different individual location label than group location label are tagged for revision by a building manager. (2b) Optionally: automatically correct the database entry and propagate accordingly. Fig. 3.6b shows an example of the process for detecting a wrong variable name.

Detecting missing/broken variables

Modern office buildings use repeating patterns for installing devices and instrumentation, i.e. each desk has the same sensor and actuator features. The regular pattern can be converted to a template and used in our framework to check if detected groups that fulfil the template condition follow the expected specifications. By mining location, devices tampering or misconfiguration can be detected as user activity will not correlate with active sensors. Thus, the tampered sensor can be excluded from the group. Fig. 3.6c illustrates the application in an example. Assuming the previous naming convention, the grouping can be used to propose a missing element's name.

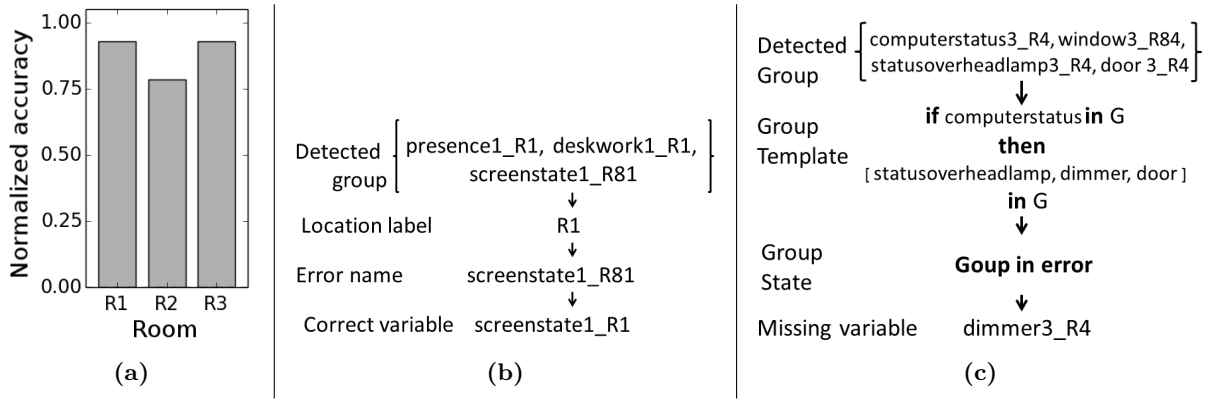


Figure 3.6: BMS application examples. (a) People counting estimation. For this application, the considered living-lab rooms were analysed and averaged normalised accuracy for the people count estimation is shown. (b) Detecting variable naming error example. (c) Detecting missing variable example. See main text for details (Sec. 3.7).

3.8 Discussion

Density of ubiquitous sensors and actuators in office buildings is rapidly raising, as fine grained activity and context recognition is required for comfort and energy saving [73]. Although, existing networked sensor and actuator technologies use one-of-many self-registration protocols, a team of experts is usually required to virtually interconnect sensors and actuators, and fill databases with information with specifics of the installation, e.g. location. Our work introduced a foundation for tools and applications to assist in different building commissioning/management tasks.

In addition to the example applications presented, our framework could assist in autonomous error management, e.g. by working around a broken sensor and using measurements of a similar device to estimate the missing value. For example, a desk has a light sensor and uses that information to dim the overhead light. Also, there is a light sensor measuring the outdoor light. If the desk light sensor breaks, a Building Management System (BMS) could work around the issue by using an outdoor light sensor that provides related measurements, to estimate the overhead light dimming. Our framework retrieves groupings that not necessarily reflect a naive structural interpretation. For example, we observed situations where all outdoor light sensors got grouped, thus discovering a façade group. A façade group could nevertheless help the BMS to easily find replacement for a broken outdoor light sensor.

We estimate that a permissive confidence thresholds our Weighted Transitive Clustering (WTC) algorithm will pull all building variables into one group. Hence a commissioning step is required for our approach too. Nevertheless, we expect that the parameter fitting could be implemented as an optimisation problem, thus minimising manual labour. Further, the parameter fitting essentially permits building managers to match the WTC algorithm with a particular task, e.g. grouping all variables at floor level.

We chose Hierarchical Agglomerative Clustering (HAC) for comparison to our WTC algorithm as HAC is frequently used in clustering, e.g. for words in text. In contrast to typical clustering problems, no feature space and corresponding similarity measures were available between the building variables. Consequently, for HAC, we considered the rule confidence as similarity indicator between building variables. However, rule confidence alone is clearly insufficient to group variables. WTC addresses the variable grouping by employing an iterative merging considering transitivity and group confidence. WTC thus provides appropriate results even if variable naming or ordering errors prevent an naive grouping using variable names only.

3.9 Conclusions

The Weighted Transitive Clustering (WTC) algorithm proved to be effective approach for grouping variables according to their physical location. Our approach could outperform the hierarchical clustering method and is robust against lost events or deleted rules. Although WTC created more groups than expected, these groups reflected the smallest granularity of association, e.g. desks, blinds, temperature sensors. Our analyses further showed that rule mining quality is critically important. When the WTC algorithm is used continuously over time, it showed robustness to errors in the data and could adjust to new relationships found in the data, such as installation and Building Management System (BMS) rule changes. The WTC confidence threshold could be used to balance discovery speed and overall accuracy depending on the particular setting.

The performance analysis indicates that more events are not necessary better, rather, valid distinguishable events from where correct rules can be extracted, will produce good groupings in short time. Expert rules are incomplete by nature, mostly because they are written in a minimum effort basis. We showed that automatic grouping of variables is a crucial step towards a self-commissioning building. Several applications can be designed on top of our framework to enrich building capabilities and support building management/commissioning in rapidly detecting errors.

Acknowledgement

This work was kindly supported by the EU FP7 project GreenerBuildings, contract no. 258888.

Chapter 4

Mining hierarchical relations in building management variables

Chapter originally published as: L. I. Lopera Gonzalez and O. Amft. "Mining Hierarchical Relations in Building Management Variables". In: *Pervasive and Mobile Computing*. Pervasive and Mobile Computing 26 (Feb. 2016), pp. 91–101. ISSN: 1574-1192. DOI: 10.1016/j.pmcj.2015.10.009

4.1 Introduction

Effective operation of modern buildings relies on interconnected ubiquitous devices. Examples of ubiquitous devices installed in many large buildings include movement detectors and light sensors, besides actuators, e.g., ceiling lights and air ventilation. The devices are represented as measurement and control variables in a building management system (Building Management System (BMS)). The variables are then combined in rules or more advanced interpretation algorithms to infer building context states. For example, Milenkovic et al. [74] showed how several office user activities could be recognised from sensors already available in many buildings. Detected user activities are again variables. A BMS thus links measurements from sensors and operations of actuators. For example, occupancy measurements in a meeting room, indoor temperature, and outdoor weather conditions are used to control optimal operation of a Heating, Ventilation, and Air-Conditioning (HVAC) system in this meeting room.

The motivation of building owners and operators to further increase measurement and automation functions are manifold, including saving energy [21], and improve occupant comfort under energy efficiency constraints [46, 68]. Buildings, such as office towers may utilise about ten measurement and control variables per desk, resulting in 100 variables for a 10-occupant open office space. In a typical office building with ten floors and 500 desks, as many as 50.000 variables can be expected, not considering corridors, elevators, meeting rooms, etc. Priyadarshini et al. [95] reported an application example of wireless sensor nodes, where ubiquitous integration and rapid increase in devices and variables is well illustrated. For scalability, BMS development is thus migrating to a service-oriented architecture, as illustrated by Degeler et al. [23]. Nevertheless, a great deal of manual labour is required during commissioning to correctly relate sensors and actuators, derive context variables, and maintain consistency during a building's lifetime.

Whether newly build or refurbished, commissioning of a BMS is today performed by expert technicians that work together with device and system suppliers to correctly interlink variables. During maintenance, further effort is spent to correct variable naming errors and updating configurations upon device exchanges, failures, or upgrades. It is technically conceivable that any modern building-installed device would automatically identify itself, revealing its location and interrelation of variables with other devices. However, several constraints prohibit self-configuring features from being implemented in individual building-installed devices, including device cost, size, power consumption, and robustness for continuous operation over many years. Due to the cost of large BMS installations, sensors and actuators need to be affordable. For example, EnOcean (www.enocean.com) develops wireless self-powered sensors and actuators, including motion detectors and wall switches, which are even used in new building constructions to minimise physical network infrastructure. Therefore, methods that could mine variable relations from a central BMS based on the variables' data streams are sought. In earlier work, we showed that mining relations among BMS variables and grouping variables has many applications, including detecting missing or broken devices, detecting and correcting variable naming errors, and deriving new variables such as people count in office spaces [63].

We present a novel approach to derive groups and group hierarchy from variable association rules that are mined in event time series of variable states. Our hypothesis is that variables that relate to the same physical space in a building will provide events with temporal relation. Variables that have a causal relation, i.e., a variable changes in response to a change in another variable, should have a temporal dependency in states that could be extracted from their event time series. For example, occupants walking through a building space towards a desk will create changes in presence and activity variables and potentially in light actuator states, first for the building space, i.e. room, and then at the desk. For our approach, the variables could have different origin, including sensor measurements, actuator states, and derived states. Our approach considers state changes across all variable as one unified, temporal ordered event stream and derives variable grouping without supervision. As rules are mined, temporal relation of events provides the basis for variable groupings. Subsequently, variable hierarchy is determined to describe dependencies.

In particular, the chapter provides the following contributions:

1. We present a group mining framework to extract variable association rules from the unified variable event stream and then extract variable groups from the rules. A novel parameter-free Hierarchical tree clustering (HTC) algorithm is introduced to group variables into hierarchical structures. HTC works in two steps, where initially a rooted variable tree is constructed based on the extracted variable association rules and subsequently the tree is clustered and trimmed to represent variable group relations.
2. We evaluate our mining framework using actual BMS data of several months and derive variable grouping performance. The grouping performance is compared to expert generated groups and to groups derived from BMS configurations. The evaluation confirmed that HTC provides relevant groupings that are robust even if only subsets of the BMS data was considered as source.

In our previous work, we devised a Weighted Transitive Clustering (WTC) algorithm to discover variable grouping [63]. However, the groups derived by WTC provided no hierarchy information. As a result, when an environmental variable is associated with more than one room group, the linked room groups are merged into one. Additionally, to distinguish different hierarchy levels, e.g. room and desks, WTC needs to be run with different parameter settings. In contrast, the HTC algorithm presented in this work produces hierarchical groups that describe the relationship between rooms and desks. The hierarchical groups can also describe interconnections between groups created by environmental variables. In this work, we demonstrate that HTC can outperform the WTC algorithm. HTC finds groups that associate a majority of variables and identifies alternative relations between variables. Moreover, due to the hierarchical representation, variable relations could be inspected graphically.

4.2 Related work

In wireless sensors networks, device localisation is an active field of research. One common approach to localisation is device-based, i.e. using sensor node communication capabilities to estimate a node's relative location with respect to its neighbours [91, 93]. Aspens et al. [5] provide a detailed theory for automatically constructing graphs that represent the sensor location in a bi-dimensional or three-dimensional space. They considered the distance between neighbouring nodes known and assume fixed positions of beacon nodes. In practice, the location of beacons is manually configured in order to map a node's location to a physical location. Furthermore, beacons become critical path points in the system, as broken beacons may prevent correct node localisation.

Patwari et al. [92] illustrate how to obtain the distance between neighbouring nodes using time of arrival and received signal strength. The localisation techniques used in wireless sensor networks require that a node spends some energy in the task of localisation. In a highly ubiquitous and distributed environment, sensor nodes will have very restricted energy budget and on-sensor localisation might not be feasible. In addition, alternative communication media and protocols may require separate device-based localisation solutions. Since buildings are used for decades, different communication media may get installed as technology evolves. In this regard, modern Building Management System (BMS)s are already design to be extensible and handle different protocols or media [23]. Instead of device-based localisation, it is therefore interesting to utilise the data available in a BMS.

While modelling energy consumption patterns of a building, Moreno et al. [77] investigated which building variables had the largest impact on individual energy-consuming appliances, e.g. HVAC. They proposed a clustering approach to infer a group of variables related to the energy measuring device in each appliance. To implement the clustering, snapshots of the building variables' values were used to compute features. The features would cluster variables for each appliance. In the work of Moreno et al., energy measuring devices represent the clusters to which other variables get assigned. In contrast, our variable grouping does not require a particular energy measuring device or appliance to be present.

Rule mining approaches are able to extract association of data from multiple modalities [76, 109]. For example, Koperski et al. [54] showed how useful rules can be extracted by mining a geographical database, e.g., house prices rise for houses closer to the beach. Furthermore, Ma et al. [70] showed that their rule mining approach was able to correctly handle many different datasets. Yin et al. [116] illustrated the difficulty of mining relevant rules, and the importance of balancing the support of the rules in order to get meaningful results. In our framework [63], we used the temporal rule association method proposed by Guillame-Bert and Crowley [35]. The framework receives a labelled stream of events and extracts rules of the form $A \Rightarrow B$, where B happens within an observation window after A. Temporal association rules can represent the behaviour of real systems that have delays between the time a sensor changes value and the time an actuator responds to that change. In addition to sensor actuator relationships, we consider relations created by a user moving from one context state to another, e.g., presence at the desk to computer work. The delayed actuator reaction and the time of a context transition are not possible to detect in a snapshot of the data. Thus temporal rule mining methods are better suited to extract relations in a building environment than variable clustering using data snapshots.

After obtaining variable relations, groups among all variables are extracted. When using Weighted Transitive Clustering (WTC) as group extraction method, we observed that it was not possible to differentiate structures within variable groups [63]. WTC uses thresholds to control grouping. We observed that the thresholds also controlled the hierarchical level of groups, e.g., room level or desk level. Grouping is thus challenging in building structures that share common elements from external environmental variables, e.g., sun light, or from joint building variables, e.g., presence in hallways. Consequently, WTC required a parametric search over the thresholds to extract each grouping hierarchy level, i.e., distinguish rooms or desks in a room.

In this work, we replace WTC with a novel group extraction algorithm that yields hierarchical group structures. These structures describe individual variable relations, thus solving the grouping in shared building structures. Unlike WTC that relies on thresholds, our new approach is non-parametric. We evaluate the resulting groups to compare with WTC, and discover additional variable relations that might be of interest.

4.3 Group Mining Framework

For our mining approach, we assume that sensors and actuators that are located, or related to the same physical space will produce temporally related events. Relations will appear in particular then, when variables have a causal relation. The event relations among variables may be caused by the user interacting with the space, but could as well be coming from weather changes and other sources of dynamics. For example, changes in outdoor lighting conditions may cause a Building Management System (BMS) to respond by changing blind or lighting states. A change in outdoor light should be observable in most outdoor light sensors. Therefore, all outdoor light sensors will present a high temporal correlation of their change events. Moreover, the events of associated actuators will show correlations to the outdoor light events.[1]

Our group mining framework comprises three steps: (1) event extraction, (2) rule extraction, and (3) group extraction. In this work, we focus on the group extraction to obtain hierarchical variable groups that could be used in building management. Fig. 4.1 illustrates the group mining framework. In this section, we describe the processing steps of our framework.

Event extraction

Event extraction is used to convert the data time series of building management variables into a unified time series of events. Events are extracted by detecting positive value changes in the variable data time series. Then, all events are labelled with the variable name and inserted into a single, temporal ordered event time series. The actual data values or other information about the detected changes are not used in the subsequent processing. Table 4.1 shows an example sequence of the resulting unified event time series.

Every variable v_i was processed for value changes at every sample time t using the condition $\frac{\Delta v_i}{\Delta t} > \Theta_i$. The threshold filter $\Theta_i > 0$ was used to filter out variable-specific noise.

Rule extraction

Rule extraction converts the unified event time series into a set of prepositional rules with a confidence assigned to each extracted rule. Rules were derived as variable associations based on event occurrences over an observation window. The resulting variable association rules are denoted as logic relations of the form

[1] There may be different relations observable among the variables relating to individual building facades, depending on the building orientation.

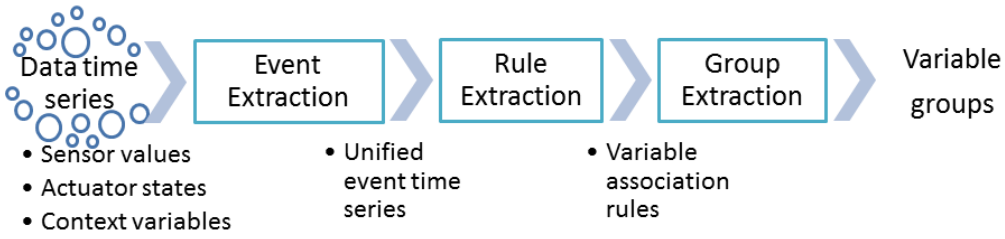


Figure 4.1: Group mining framework to derive variable group relations from the data time series of building management variables. The groups are derived by mining a unified time series of events for variable association rules. In this work, we focus on extracting variable groups using a novel hierarchical tree clustering (Hierarchical tree clustering (HTC)) algorithm.

Table 4.1: Example of unified event time series derived from building management variables by the event extraction step of our group mining framework.

Event timestamp	Event label
1353935850.335012	powerconsumption11
1353935868.344362	powerconsumption2
1353935898.467623	presence3
1353935900.453503	luxlevel3
1353935934.376489	powerconsumption1
1353935936.701680	powerconsumption4
1353935946.382288	powerconsumption3

Table 4.2: Example of variable association rules with confidence and support derived from a unified event time series by the rule extraction step of our group mining framework.

Rule	Confidence	Support
$blindsangle3 \Rightarrow dimmer2$	0.05	0.05
$dimmer1 \Rightarrow blindsheight2$	0.05	0.06
$blindsangle2 \Rightarrow blindsheight2$	0.05	0.11
$blindsangle1 \Rightarrow dimmer2$	0.05	0.25
$luxlevel1 \Rightarrow luxlevelout2$	0.05	0.10
$deskwork9 \Rightarrow presence3$	0.05	0.06
$door1 \Rightarrow statuslights2$	1.00	1.00
$powerconsumption1 \Rightarrow statuscomputer1$	1.00	1.00
$statusbeamer1 \Rightarrow presentation1$	1.00	1.00

$A \Rightarrow B$. When A is observed, B is expected to occur before a time laps τ. In addition, we obtain a rule confidence and denote the support of a rule. Here we consider rule confidence as follows: Given all events of label A, how many times $A \Rightarrow B$ occurred within τ. The concept of support can be defined as: Given all events with label B, how many B events can be produced by the rule $A \Rightarrow B$ within τ. Table 4.2 shows an example of the variable association rules.

To implement the rule extraction, we used the Temporal Interval Tree Association rule learning (TITArl) algorithm by Guillame-Bert and Crowley [35]. TITArl discovers rules in the event sequence by tracking the temporal distribution of candidate event occurrences and their correlation over time. TITArl provides a validity period, indicating the rule lifetime across an event sequence. TITArl parameters were set identical as in our previous work [63]: *inputEvent* predicate `".*"`, *outputEvent* predicate `".*"`, time window future 15s, time window past 15s, minimum rule confidence 0.05, minimum support 0.05, minimum number of use 2. All other settings were left at their default configuration. We set confidence, support, and number of uses parameters to obtain a sensitive rule mining behaviour.

Group extraction

Group extraction intends to convert the varying variable association rules into variable groups according to their relationship. In this work, we consider that organising the variables in a hierarchical representation

according to their relationship benefits variable grouping as the groups retain information about the variables' causal relation. We propose here the HTC algorithm that is detailed in the following section. The Weighted Transitive Clustering (WTC) algorithm, used previously, considered all elements in a rule as belonging to one group [63]. WTC also considered that a newly extracted rule would be part of an existing group, if the new rule shared elements with the existing group. WTC used a threshold on a new rule's confidence to decide whether to include the new elements from the rule in the existing group.

4.4 Hierarchical tree clustering (HTC)

To derive hierarchical variable groups our Hierarchical tree clustering (HTC) algorithm operates in two steps. In the first step, a rooted tree is constructed from the variable association rules. In the second step, the tree is clustered into hierarchical groups.

Constructing a rooted tree

In the first stage, we construct a rooted tree $G(N, \xi)$, where N is the set of nodes consisting of the variables in the rules plus a node *root* from where all branches grow. Set ξ is the set of edges constructed from pairing the premise of the rule to the conclusion. Every edge ξ is assigned a weight that is equal to the value of the rule confidence c. A *main branch* is defined as the sub-tree structure in which the top node is a child of the *root* node. Fig. 4.3 (b) depicts the main branch for a four-people office room. The resulting tree is also called a rooted directional graph and is similar to an arborescense as defined by Mesbahi et al. [72], where there should exist no simple cycles, every node should have only one path leading to it, and all paths direct outward from the *root* node. For our tree construction, one exception to the definition was needed: Leaf nodes are allowed to have alternative paths. For the building variable tree, there may be shared elements at the leafs of the hierarchical structure. For example, two independent users presence variables can be related to one common ceiling light.

Subsequently, the tree is rearranged in two forms: (1) Nodes that are children of *root* and several other parents are removed from the *root* node. (2) Nodes that have multiple parents and are not a leaf node are promoted to be the parent of their parents. Promoting nodes enforces the arborescense. The rearrangement process is repeated until every node has a single parent, with the exception for leaf nodes. Fig. 4.3 (a) shows a tree constructed from the Building Management System (BMS) configurations, Fig. 4.3 (b) illustrates the result of applying node promotion.

In building management data, node promotion has logical implications. Variables that are used to compute an additional variable are often promoted. For example, presence in the room is computed from the presence at the desks. Therefore, promoting room presence over the presence at the desks creates a correct logical hierarchy. In the rule mining step, rules will reflect the order of computation. Thus, node promotion is needed to reflect the logical hierarchy.

After the tree rearrangement is concluded, trimming is done by removing unwanted edges. We iterate over the main branches and do a one-to-rest comparison. We defined a normalised branch length $\overline{L}_B$, as shown in Equation 4.1, where $\overline{L}_p$ is the normalised length of path p, e_p is number of edges in the path p, c_i is the edge weight in path p, and β is the set of paths of the main branch that have common nodes with other main branches. All paths in β are trimmed by enforcing Equation 4.2. Finally, if the trimming process yields loose branches, these loose branches become main branches. Orphaned nodes are discarded. Fig. 4.3 (b) illustrates the trimming process by removing the connection between the door sensor, dimmers, and outdoor light sensor.

$$\begin{aligned} \overline{L}_p &= \frac{\sum_{i=1}^{e_p}(1-c_i)}{e_p} \\ \overline{L}_B &= \max\left(\overline{L}_p \middle| \forall p \notin \beta\right) \end{aligned} \tag{4.1}$$

$$\overline{L}_p \leq \overline{L}_B | \forall p \in \beta \tag{4.2}$$

Tree clustering into hierarchical groups

The second and final step of the HTC algorithm is to convert the main branches into hierarchical groups. A hierarchical group consists of variables arranged in a head set, an elements set, and an array of children. The children are groups with the same properties. We define a *main group* as the hierarchical group derived from a main branch. In addition,we considered *group members* as set of variables contained in head and elements sets of a group, as well as the variables of all child groups.

To convert the tree into groups, we analysed the main branches. Each node at the top of a main branch becomes the group head. The following nodes that have only one child also become part of the head set. The node elements are the leaf nodes which directly descend from the last head node. Any node derived from the last head node which contains more than one child, will be in the head set for the subgroup. The process is repeated recursively until all nodes are covered.

Groups can be of five different types: basic, alternative, shared, equivalent and overlapping. An *alternative group* comes from a branch that has all its paths ending in the leafs of another branch. When all paths of a branch end in different branches' leafs, then the resulting type is *shared group*. An *equivalent group* comes from branch that shares all leafs with another branch. An *overlapping group* derive from a branch that has some paths ending in their own leafs, and some paths that end in some other branches' leafs. Finally the *basic group* is a group that has no overlaps, but can have some leafs shared, or leafs appearing as part of an alternative group. Fig. 4.2 illustrates all group types. Stand-alone trees, e.g. Fig. 4.3 (b), are also considered basic groups in our work.

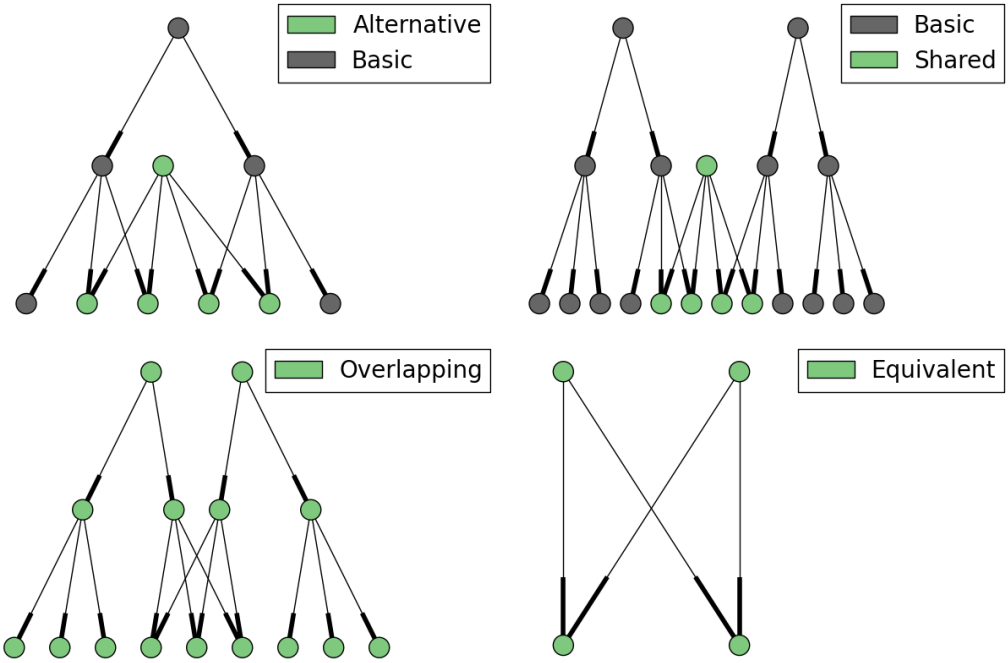
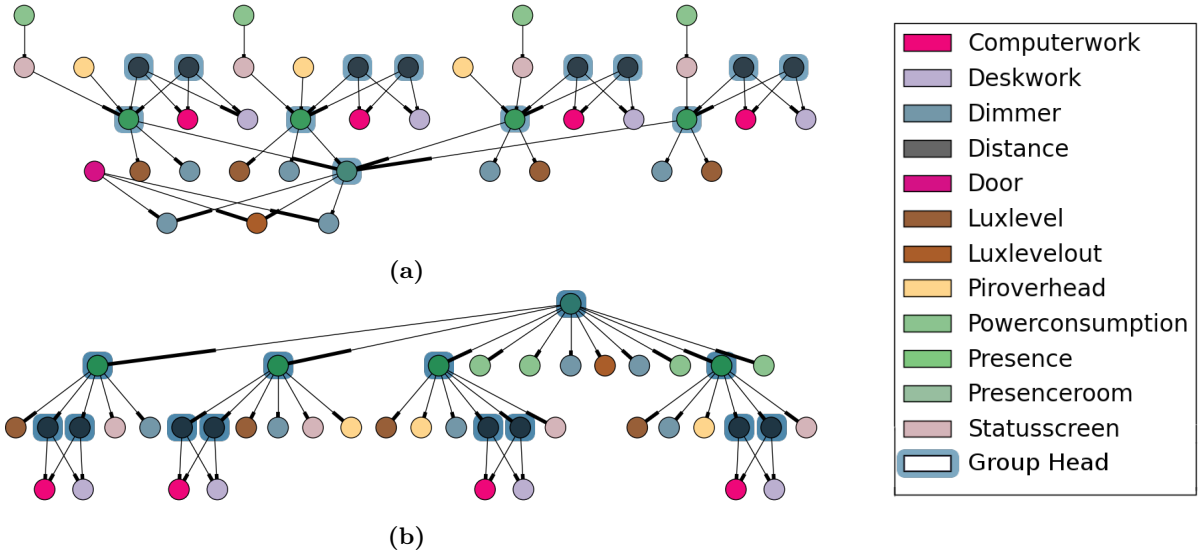


Figure 4.2: Illustrations of branches that generate different group types in our Hierarchical tree clustering (HTC) algorithm. Please refer to Section 4.4 for more details.

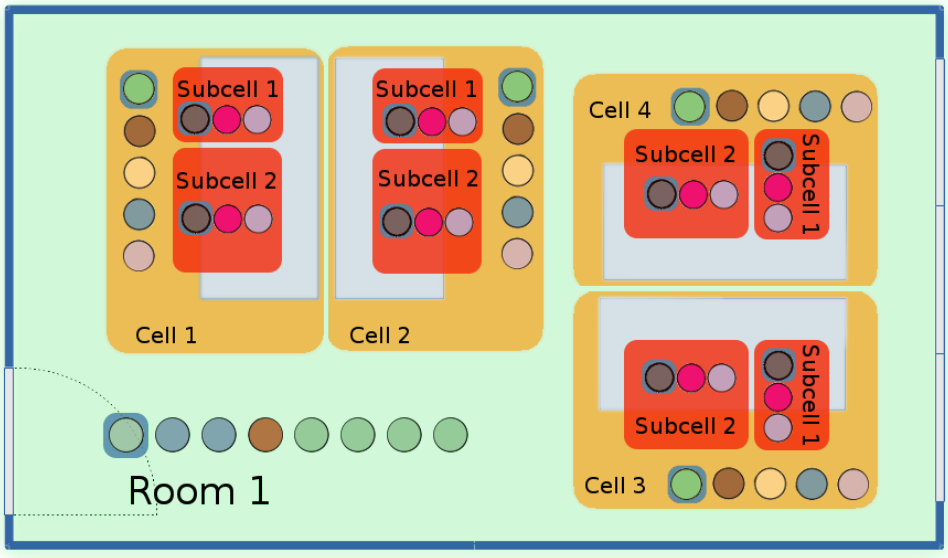
Fig. 4.3 (c) shows an example of a main group corresponding to a room, cells corresponding to actual office desks in the room, and subcells corresponding to desk areas in front and at the sides of a computer screen. The hierarchical groups produced by HTC resemble the variable locality.

From the rules that produce the tree in Fig. 4.3 (a), a group is extracted that represents a room with cells and subcells subgroups. In contrast, when the same rules were used with Weighted Transitive Clustering (WTC), the result is single flat group, e.g., Room 1, without information on cell or subcell association.



(a)

(b)



(c)

Figure 4.3: Illustration of the Hierarchical tree clustering (HTC) algorithm steps. (a) Rooted directional graph constructed from system configuration rules of a four-people office room. (b) Processing result after arranging and trimming the tree. (c) Respective conversion of the tree into hierarchical variable groups. The groups are depicted in correspondence to the physical architecture of the room.

4.5 Evaluation methodology

For the evaluation of the Hierarchical tree clustering (HTC) algorithm, we used the same set of rules mined for our previous analysis of the Weighted Transitive Clustering (WTC) algorithm [63]. However, we created a new ground truth and composed a complete hierarchical description of the living-lab data to validate the framework. Additionally to the new ground truth, we used configuration information from the Building Management System (BMS) to validate the grouping stage of the framework. Finally, we took external environmental variables into consideration as they represent parallel hierarchies to the room organisation created from user-driven events.

Evaluation Dataset

The GreenerBuildings (GB) living-lab dataset was recorded at the TU Eindhoven campus during 14 months. The living-lab consisted of 3 rooms; R1: a 4 people office room (14 months), R2: a ∼20 people meeting room (13 months), and R3: a 12 people open office space (3 months). In total 277 variables were recorded. The

dataset contains sensors (outdoor light, room CO_2, humidity, temperature, etc.), actuators, and contextual variables. Context variables included room presence, desk presence, computer work, presentation, brain storming, etc. During the 14 months of recording, sensors were added, broken ones replaced, rules were updated, and configurations fine-tuned. We consider that the modifications are part of a real building life-cycle, and thus presents a rich set of test cases to evaluate our approach. The dataset is described in detailed in [63].

Ground truth and BMS configurations

We derived Ground Truth Groups (GTG) manually considering user-driven and environmental dynamics to create a rooted tree. The tree was subsequently grouped into hierarchical groups. After grouping, GTG contained ten main groups in total. Two of the main groups represented R1 and R2 respectively. R3 had no variable that joined the available cells, i.e., no door or presence in the room variables. Thus, the other eight main groups represented cells from R3.

In addition, we created BMS configuration groups (BCG) automatically parsing the BMS configurations and extracting rooted tree. Similarly to GTG, we grouped the tree into hierarchical groups using the group creation step of HTC. The configurations were selected from the final month of recording, where all system variables were present. The resulting groups were then manually verified to ensure correctness. The BCG represents a baseline for comparison to our HTC algorithm where all the edges' weights were set to one.

Evaluation metrics

Metrics of graph similarity could be applied to compare the ground truth with the mined groups [86]. However, graph similarity requires strict agreement in variable associations. The ground truth used in our evaluation contained all the variables ever recorded by the system, whereas individual periods would contain a subset of the variables only. In the BMS application domain, different objectives should be assessed: It is desirable that all available variables are used, and that each variable is associated correctly with its corresponding variables. Thus, relative coverage and correctness were chosen as metrics to evaluate performance.

Relative coverage measures how many variables are associated from those variables available in the test period. By measuring relative coverage, we can keep track of the effectiveness of the algorithm. Accurate grouping of a fraction of variables only could hinder applications of the algorithm.

Correctness assesses the quality of the groups rather than the similarity to the GTG. To establish the group quality, we define room level correctness and variable neighbour correctness. The room to which a variable belongs to is encoded in the variable's name, e.g., *dimmer1_R1*. The naming convention facilitates checking correctness at the room level. We defined three approaches to determine the label of a group: majority voting (CMV), head majority voting (CHMV), and sensor class (CSC).

Correctness CMV computes the label of the group by parsing the name of all variables in the group, including subgroups. The room label with the most occurrences becomes the group label. *Correctness CHMV* only applies majority voting to the head of the main group. Correctness CHMV is thus a stricter comparison than correctness CMV and emphasises performance of the hierarchical approach. *Correctness CSC* separates sensors in two classes: outdoor, and the rest, we want to measure how well is the grouping when we consider that external environmental sensors might link variables from different rooms. Therefore, correctness CSC applies correctness CMV for the groups that do not contain external environmental sensors, and otherwise considers groups to be correct. In each case, the respective correctness score is computed by the ratio between the number of variables that have the chosen group label and the total amount of variables in the group.

The room analysis is insufficient to evaluate the information gained from the hierarchical grouping. Therefore, we consider correctness from a variable's neighbourhood point of view too. Equation 4.3 defines the neighbourhood of x with respect to a set of groups Γ, where $gm(g)$ are the group members of g.

$$\mathrm{Ne}(x, \Gamma) = \{\mathrm{gm}(g) \forall g \in \Gamma | x \in g.head \vee g.elements\} \tag{4.3}$$

Correctness at the group level (Correctness at the group level (CG)) computes for a given variable x, the ratio of the number of elements in the intersection between $\mathrm{Ne}(x, G_d)$ and $\mathrm{Ne}(x, GTG)$ over number of elements in $\mathrm{Ne}(x, G_d)$, where $\mathrm{Ne}(x, G_d)$ is the neighbourhood set of x with respect to the groups extracted from the data , and $\mathrm{Ne}(x, GTG)$ is the neighbourhood set of x with respect to the GTG. Equation 4.4 shows how the total score of $G_d$is computed.

$$\mathrm{CG} = \mathrm{mean}_{\forall g \in G_d} \left(\mathrm{mean}_{\forall x \in g} \left(\frac{|\mathrm{Ne}(x, G_d) \cap \mathrm{Ne}(x, GTG)|}{|\mathrm{Ne}(x, G_d)|} \right) \right) \tag{4.4}$$

Moreover, we want to measure the effect of allowing the external environmental sensors to mix different variables together. We therefore measure correctness at the group level with sensor class selection (CGSC). Similarly to correctness CSC, correctness CGSC considers a subgroup being correct, which contains at least one external environmental sensor and otherwise applies correctness CG. In either case, to compute the total score of a group, the score for all variables is averaged. Finally, to compute the score of a grouping result the score of all groups are averaged.

Applications in the BMS domain depend greatly on the correctness of the groups [63]. For example, in autonomous configuration applications mistakes in the group extraction will create user discomfort, e.g. when light sensors and switches may be linked to wrong overhead lights. Similarly, in error detection application erroneous variable grouping would require additional effort from technicians in reviewing the application's error reports.

Group quality analysis

In addition to correctness and relative coverage, we measured group quality by counting how many groups are created according to the group types defined in Section 3, e.g., basic, alternative, etc.

Groups containing external environmental sensors may interact with other groups. By using a naming convention that differentiates groups with and without external environmental sensors, we measured their occurrence as a ratio. Any group containing external environmental sensors were named *Out_variable-type_id*, e.g., *Out_light_001*, while other groups were named *Space_label_id*, e.g, *Space_R1_0001*.

Comparative analysis

We compared the performance of HTC against the WTC method and therefore used the same rules as mined in our previous work [63]. Rules are derived in periods of one month. We evaluated the performance on individual months and compared results against GTG and BCG. Against GTG, an absolute benchmark on the grouping capabilities is obtained. BCG serves as a baseline of a basic functional hierarchical grouping.

Subsequently, we evaluated performance at the end of the recordings by accumulating period rules. The rules of periods were added sequentially to the processing.

4.6 Results

BCG achieved a relative coverage of only 33% and produced 15 additional groups compared to the three main groups in Ground Truth Groups (GTG). While correctness at room level was perfect for BMS Configuration Groups (BCG), Correctness at the group level (CG) was 0.96.

The average correctness results at the room level were: Correctness by head majority vote (CHMV) = 0.91± 0.07, Correctness by majority vote (CMV) = 0.93±0.06, and Correctness with sensor class selection (CSC) = 0.96± 0.04. Fig. 4.4 shows the correctness results for each evaluation period. An almost identical correctness CHMV and CMV suggest that errors are related to the group members or their children, rather than to the head set. In other words, the head set is of the same label as the majority of the group members. The evaluation of the correctness grouping from a variable perspective yielded: CG = 0.66 ± 0.11 and Correctness at the group level with sensor class selection (CGSC) = 0.72 ± 0.1. Hierarchical tree clustering (HTC) achieved an average relative coverage of $75\% \pm 6$ p.p.. The correctness CSC metric shows that the system can correctly separate user generated hierarchies from external environmental variable hierarchies. Fig. 4.4 shows all four metrics for the individual periods.

The per-period group quality analysis is summarised in Table 4.3, where the distribution of group names with respect to group types is shown. There were overlaps between *Out light* and *Space* groups. Thus, outdoor light sensors were associated with variables from different rooms, which is consistent with previous observations [63]. The average number of basic groups is similar to the number of groups obtained with GTG. Equivalent and alternative groups may occur as a result of incomplete information from the rule extraction stage.

Fig. 4.5 illustrates the cumulative system performance, i.e. when data from all preceding monthly periods was accumulated. Due to the installation changes (device additions, replacements) during the living-lab recordings, system performance varies. At the end of the last period, i.e. month 14, relative coverage was 96%, and correctness metrics were: CHMV=0.95, CMV=0.95, CSC=0.97, CG=0.78, and CGSC=0.81.

Finally, we analysed group quality in the cumulative setting. A total of 46 basic *Space* groups were obtained at the end of the last period. In conjunction with the high correctness scores, we interpret that the hierarchical groups contain correct variables, but that there was missing information in the mined rules to assemble the variables in the expected ten main groups.

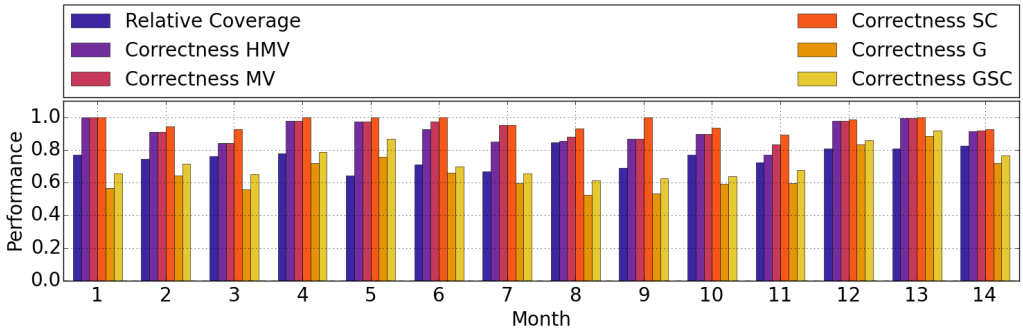


Figure 4.4: Performance results for relative coverage and correctness derived on individual monthly periods of the dataset. The correctness metrics are described in Sec. 4.5

	Basic	Alternative	Equivalent	Overlapping
Out CO_2	0.07	–	–	–
Out humidity	0.07	–	–	–
Out light	1.57	–	–	0.80
Out temperature	0.07	–	–	–
Space	11.93	1.40	2.33	3.00

Table 4.3: Per-period group quality analysis showing average number of group name per period assigned to each group type. As expected, *Out light* groups overlap with *Space* groups. The average number of basic groups is similar to the number of groups obtained by the GTG.

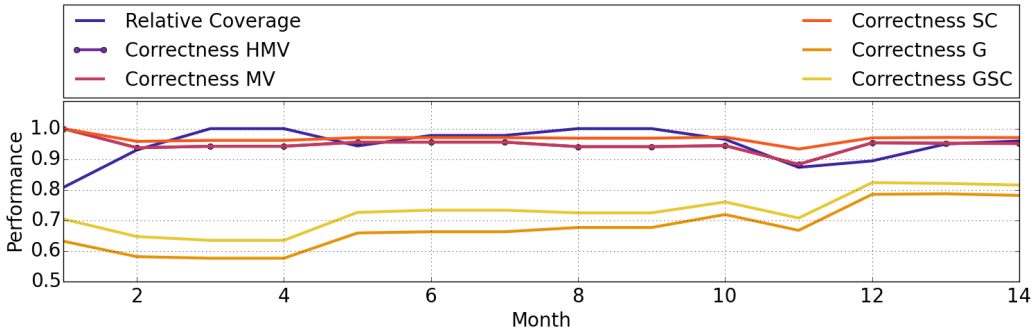


Figure 4.5: Performance results of the cumulative analysis for relative coverage and completeness. Performance variations could be explained by installation changes (device additions, replacements) during the living-lab recordings.

WTC achieved an average relative coverage of $\sim 22\%$, whereas HTC reached 75% for the same data. For the cumulative analysis, WTC finished with a relative coverage of $\sim 70\%$, whereas HTC scored 96%. HTC was able to use more variables per grouping than WTC. At the room level, correctness metrics were 0.91 for HTC, and ~ 0.94 for WTC. However, correctness CSC yielded an average of 0.96 for HTC, indicating that in the overall grouping of room level variables both algorithms have similar performance. HTC finished with a score of 0.95 in the cumulative analysis for correctness at the room level. In comparison, WTC reached only 0.75 for correctness, indicating that HTC was able to improve its performance as new information became available.

4.7 Discussion

The framework performance is closely related to the quality of mined rules and in turn, the stability of the monitored system is key to obtain good rules. Fig. 4.5 shows a performance drop at month 11, which is due to adding office room R3 to the Building Management System (BMS). R3 inserted noisy sensor data, resulting in random actuation. The noisy sensors produced irrelevant variable changes and events, which subsequently yielded false associations and rules. In Hierarchical tree clustering (HTC), we did not apply filters on the incoming rules. The quality of group structures was therefore dependent on the quality of the extracted rules.

Nevertheless, subsequent analysis months show that the performance recovered as variable errors declined. We used several variants of the performance metrics correctness and relative coverage to evaluate the framework and HTC algorithm. With these performance metrics we could ensure that the framework is suitable for different applications within a BMS.

In addition to the framework evaluation, we performed a group quality analysis, The group quality analysis provided insight into the internal structure of the resulting group sets. In the per-period analysis we found that the external environmental sensors related to air quality (humidity, CO_2, temperature) formed basic groups. The basic groups consisted of variables of the same type, e.g. all temperature sensors, or a combination of CO_2, temperature and humidity. The tendency of external environmental variables of producing basic groups could be explained with event delays. While user-driven changes in building context occur within a second, a change in air quality due to user activity could take several minutes. However, changes in external environmental variables are reflected without delay. For example, outside temperature events are almost simultaneous with indoor temperature events. Similarly, *Out light* basic groups contain light related sensors and actuators. The specialisation of the group is explained by a higher number of outdoor light events with respect to presence changes, i.e., during working hours only a handful of presence changes will occur, whereas the outdoor light could change many times during the same period.

Alternative and equivalent group types were not considered in the GTG, however, they are present in the BCG. BCG has only 33% relative coverage, thus, the resulting groups are incomplete. Similarly, 46 basic *Space* groups were found at the end of the cumulative period compared to ten in GTG and 25 in BCG. The results can be explained by incomplete information in the rules. Nevertheless, the relative coverage and correctness results indicate that correct variables are grouped. We can interpret the result as follows: Applications that depend on correct hierarchical information, i.e., people counting estimation, would obtain reduced performance due to the accumulating basic groups. However, applications that depend in the general association of the variables, i.e., cell variable finder, will function properly.

During the recording months, many device modifications were performed, e.g. adding new sensors and replacing broken devices. In addition, rules were updated, and configurations fine-tuned, resulting in changes to variable behaviour. The changes may have negatively affected the rule mining and group extraction. However, such modifications are part of the building life-cycle and thus correspond to an even longer evaluation in a real building. In the living-lab dataset, three office rooms with different properties were included: meeting room, four-person office, open space office. We consider these room types representative for office buildings in general.

A threshold filter was applied to the unified event time series with the goal to prevent continuous variables, e.g, desk power consumption, from triggering new events at every change in measured value. In this work, thresholds were set manually for selected variable types, e.g. for desk power consumption to distinguish computer on and off events, for light intensity sensor to detect on/off states in overhead lighting, etc. The threshold filter was not used for variables that may have several states, including external light intensity and temperature measurements. If the threshold filter would be omitted for any variable, additional events may be created that are uncorrelated with actual building state changes. As a result, the rule extraction may produce irrelevant rules. The tree trimming step, described in Section 4.4, was designed to eliminate connections created by any sources of noise and thus would eliminate parts of the tree that were created due to the noise events. A combination of threshold filter and tree trimming in our framework however still yielded trees that better related to actual context changes than the tree trimming step alone. Based on our results, we consider the thresholds robust enough to be applied for any variable of the same type. Thus, to implement our framework, filter thresholds could be specified in the BMS for each variable type and applied accordingly. Another option is to estimate filter thresholds from variable observations over a short operation time of the building, e.g. one week. As building states change will be reflected in the variables, robust estimation of boundaries between two variable states will become feasible.

In building environments, one of the strongest motivators to install or retrofit high sensor densities is energy saving due to more fine-grained automated control. An energy saving study conducted in our living-lab [64] showed that even the energy consumed by the sensors and actuators affects energy saving. Therefore, adding additional functionality to devices may not be feasible under energy saving restrictions. HTC deals with energy-constrained devices as the method does not impose additional functions on sensors and actuators.

Applications such as the variable failure management [63], benefit from the hierarchical groups extracted by HTC. In several cases, the extracted groups contained multiple outside light sensors that could be used interchangeably to control ceiling lights and blinds states if the original sensor fails. The group mining framework provides the basis to handle failure exchanges automatically.

In general, the mining framework could be extended to other processes, which can be described by a temporal sequence of events, and where the resulting associative temporal rules qualitatively describe a property of the process. The resulting hierarchical groups will indicate the intrinsic hierarchy of the variables.

For example, if the framework is used in an automated production line, the resulting groups will indicate the association of sensors to each step of the production line.

For example, we envision that the mining framework could be applied in crowd sourcing applications. Participating users will no longer have to provide location using energy-expensive methods like GPS. Rather, the location could be estimated from the relation of low-power sensor measurements in smartphones.

4.8 Conclusions

The HTC algorithm successfully created hierarchical variable groups that resembled those of GTG and the BCG baseline. We showed how the grouping at low hierarchy levels, i.e. at cell and subcell levels, resembles corresponding groupings of GTG, which is a substantial improvement over the WTC algorithm. Moreover, HTC does not require tuning parameters, whereas WTC required a parametric search to find each hierarchy level. HTC extracts all hierarchy levels simultaneously from the variable association rules. When compared with WTC, HTC performed similarly or even better in all performance metrics used. The additional information stored within the hierarchical groups and the ability to describe variable interactions, led us to conclude that HTC is a better choice for the variable group extraction stage of the framework.

HTC yielded variable relations that were not considered in GTG or BCG baseline. These relations consisted mainly of external environmental sensors that created overlapping groups with room-related variables. These unexpected relationships can become useful for optimising system performance. For example, we found instances that associated external light sensors to user-operated windows. The association can be understood as an indirect measure of room temperature and user comfort. Consequently, the BMS could adjust the HVAC to compensate accordingly.

Acknowledgement

This work was kindly supported by the EU FP7 project GreenerBuildings, contract no. 258888.

Chapter 5

Using a thermopile matrix sensor to recognize energy-related activities in offices

Chapter originally published as: L. I. Lopera Gonzalez, M. Troost, and O. Amft. "Using a thermopile matrix sensor to recognize energy-related activities in offices". In: *SEIT 2013: Proceedings of the 3rd International Conference on Sustainable Energy Information Technology.* Procedia Computer Science. Recipient of the Best Paper Award at SEIT 2013. Elsevier, 2013, pp. 678–685. DOI: 10.1016/j.procs.2013.06.090

5.1 Introduction

Energy conservation while maintaining occupant comfort is a critical optimisation tradeoff in commercial and residential buildings. Although modern building energy management systems (BEMS) can control lighting and heating/ventilation systems, various installations and appliances used by occupants during the day are manually operated, including office devices, kitchen appliance, washing basins, etc. By providing feedback on appliances usage and thus energy consumption, occupant awareness on energy needs can be improved. To provide accurate feedback, usage patterns and occupant activities could be recognised from ambient sensors.

Ambient sensor modalities that have been successfully used for activity recognition include video cameras and microphones, e.g. [49, 89]. However, these modalities are often perceived by occupants as privacy intrusive. Moreover, cameras may require regular maintenance to ensure their robust operation. Previous investigations on activity recognition in buildings considered Passive Infra-Red motion detectors (PIR) too, e.g. [113]. While PIR sensors are optimal for motion detection, are low cost, and require minimal maintenance, the obtained motion information is often too limited for activity recognition. In particular, PIR sensors can detect an initial motion in their field of view, but cannot detect constant presence of a heat source, such as a person. This effect is a result of the PIR's operation principle that works by detecting heat-differences. In contrast, thermopile sensors that exploit the Seeback effect to detect temperature differences could continuously detect heat differences. Thus, occupants in the sensor's field of view could be continuously recognised, not only while moving. Moreover, objects that show a temperature difference compared to its surroundings could be identified, such as a coffee pot or sink used with warm or cold water. Activities recognised with a single thermopile sensor could be attributed to energy consumption, e.g. to provide feedback on actual consumption due to using warm or cold water, or using the kettle.

In this work, we investigate a novel generation of thermopile sensors constructed in a 2D-matrix for recognising objects and occupant-object interactions from a single sensor installed in an office pantry area. We used a ceiling mounted sensor matrix to detect heat distributions captured by the matrix' elements and subsequently recognise objects and occupants. With this approach we can show that a single sensor installation can provide information on various activities, rather than instrumenting many devices and appliances with individual sensors.

In particular, this paper provides the following contributions:

1. We introduce the thermopile sensing concept and our processing framework to process the sensor matrix data. The framework detects and tracks objects in the sensors field of view, and classifies the detected objects according to state and interaction categories. The framework provides concurrent responses for all configured and detected objects, thus can process multi-user scenarios.
2. We present our evaluation study comprising (1) a scripted set of 21 activities used as training dataset, and (2) a uncontrolled, real-life dataset used for testing. During the training analysis, optimal features

and parameter sets for the classifiers were determined.
3. We evaluate our approach and processing framework using the real-life study dataset and determine classification performances for all concurrent state and interaction classifiers.

5.2 Related Work

PIR sensors have been used to recognize activities, in [108] PIR sensors where used to keep track of how many people where in a room, and in [78] PIR's where used to detect activity as a series of activations of certain areas in the home. Although both efforts presented promising results, their approaches depended on a gateway and assume that activities are performed when entering or exiting a coverage area. If applied to a constrained area i.e. a bathroom, sensor need to be placed on all areas or objects of interest, as presented in [106]. These works required to place sensors in locations where they could interfere with the activity being performed. Multiple devices also means that maintenance requires more effort, even if the sensors are *"tape and forget"*, as argued in [106].

Infra-red cameras provide thermal images that can be conveniently used to tracking people, as they usually shine against cooler backgrounds. In [96] it was shown that thermal images provide advantages for problems like identifying pose and thus, inferring the activity a person. Although well suited for activity recognition, the cost of thermal cameras is higher than that of a thermopile sensor grid. Furthermore, since infra-red cameras look like standard visual light cameras may yield similar privacy concerns, from the perspective of end users.

5.3 Approach

This section details the thermopile sensor concept and particular device choice. Moreover, the processing architecture used to recognise object state and interaction classes are described.

Thermopile sensor

Thermopile sensors are capable of measuring the thermal radiation absorbed on their active area. They belong to the category of thermal detectors, which generate a small thermoelectric voltage proportional to the detected radiation. Their operation principle is based on the Seebeck effect. The Seebeck effect describes the electric current in a closed circuit composed of two dissimilar materials when their junctions are maintained at different temperatures [31, 53]. When several thermopile sensors are arranged in a matrix, a scene image can be constructed from the heat radiation. These temperature differences between the sensing elements (pixels), can be interpreted as objects by measuring how the pixel's values differ from the ambient temperature. In this work, we considered the Panasonic GridEye sensor [90], which is an array of 64 thermopile sensors arranged in a 8 x 8 matrix. Example images and processing steps performed based on the thermal images obtained from the sensor are detailed in the following section.

Processing architecture

The proposed architecture for detecting fine-grained interactions between people and objects of interest in a scene consists of three main modules: sensor layer, object detection layer, and classification layer. See Fig. 5.1 for a diagram of the architecture.

Sensor layer The sensor layer uses raw data from the thermopile sensor matrix and applies a Brown's lens correction (see Eq. 5.1) to fix the barrel distortion due to the sensor's construction. Here, r_c and r_u are corrected and uncorrected distances of pixels with respect to the optical axis. K_n are radial distortion coefficients, here $K_1 = 7.4 \times 10^{-3}$ and $K_2 = 0.17 \times 10^{-3}$ are used.

$$r_c = r_u + K_1 {r_u}^3 + K_2 {r_u}^5 \tag{5.1}$$

$$T_c = \alpha \cdot A_u (T_u - T_{amb}) + T_{amb} \quad \text{and} \quad \alpha = \frac{1}{4d^2 \tan\theta^2} \tag{5.2}$$

To compensate for the sensor mounting angle, an area correction was applied as described by Eq. 5.2. Here, T_c and T_u are corrected and uncorrected temperature of a pixel respectively, T_{amb} is the ambient temperature, A_u is the uncorrected projected area of a pixel and α is the area normalization constant, parametrized by

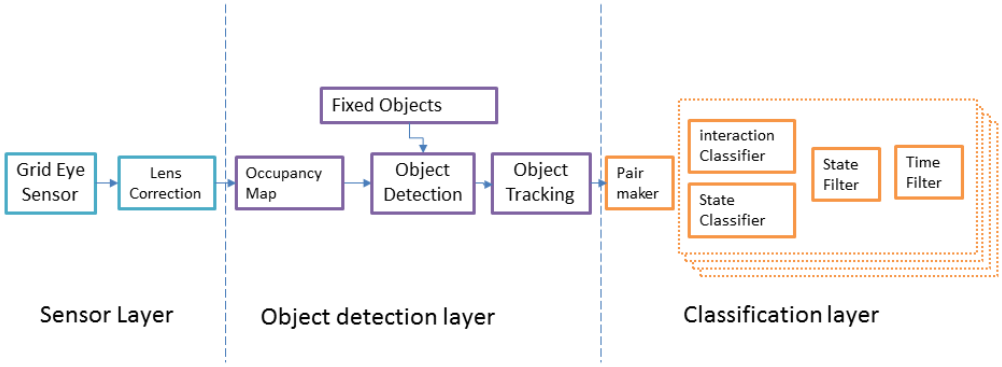


Figure 5.1: Detailed diagram of the processing architecture to process thermopile sensor images in this work. Please refer to the main text for more details regarding the functionalities inside each block.

d and θ, which are the sensor's distance to the projected area and the average pixel view angle (8°). These corrections provide a grid where features appearances are independent of their location in the matrix. The parameters K_n and A_u where fitted according to [90].

Object detection layer By taking the corrected matrix from the sensor layer, an occupancy map was derived by assigning the most probable state given the pixel's temperature difference with the ambient temperature. The states can be one of empty, cold or hot occupant. The resulting occupied pixels are then grouped by searching the surrounding pixels for the ones in the same state. This approach does not allow for an object to be partially hot and partially cold with respect to ambient temperature. If adjacent pixels present hot-cold behaviour, separate objects will be detected.

Subsequently, information about the scene context was added. We used here prior knowledge on the stationary objects located in the sensors field-of-view. Such objects might not be visible by the sensor, e.g. objects that are overshadowed or at room temperature. In Fig. 5.2 (c) it can be seen how the inclusion of the prior object location knowledge allows to split object blobs into two separate objects. Also, this process allowed us to make a first classification: objects created from intersecting blobs and considering prior object location knowledge are considered static objects, while the remaining ones are considered dynamic objects. The final step in the object detection layer was to keep track of each object across frames. This is needed to keep a consistent history of each object as required by the following steps.

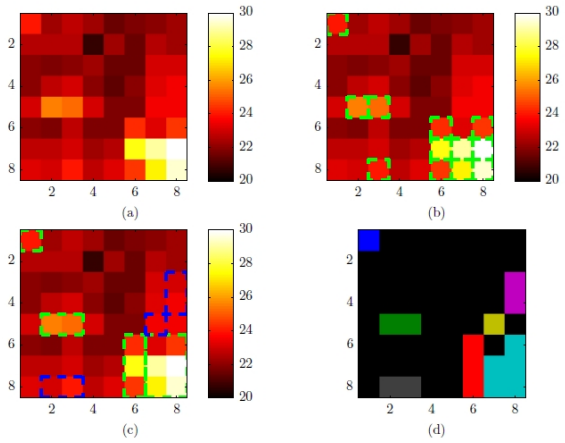


Figure 5.2: Output object detection layer for a scene with one dynamic and four static objects; (a) lens corrected sensor output, (b) resulting occupancy map with occupied pixels outlined in green, (c) detected objects (green) and fixed objects (blue), (d) resulting labelled objects.

Classification layer We arranged all objects identified in an image scene into possible pairs as follows: (1) static object paired with dynamic objects, and (2) pairs of two dynamic objects. For each pair, a reference object was selected. For static-dynamic pairs, the static object was always used as reference. Object processing

queues were then created per object pair and results grouped according to the reference object. As a result, the classification layer will provide the current activity for each reference object.

An activity is defined as the state an object or the interaction this object with another one in the scene. We classified the reference object state and its interaction with the other object in the couple. State and interaction results were then fed into a state filter to remove unlikely or impossible states given the sequence in which activities have occurred.

At this point an activity interest list was employed. This list contained all possible interactions with the reference object ordered by their relevance. The current activity for an reference object was determined by selecting from all the detected interactions the one ranking higher in the list.

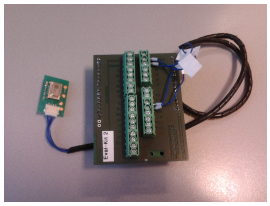
As a final step, a temporal filtering was applied, equivalent to a low pass filter, to remove short transient states that last very short time periods only.

5.4 Study Methodology and Implementation Details

This section describes the study methodology and implementation details of the recordings with the thermopile sensor.

Test installation and data recording The sensor was placed in an office building pantry area, overseeing several static objects and dynamic objects. In particular, the pantry area contained a faucet with hot and cold water, a coffee pot, and a microwave. In this space several activity scenarios are regularly performed, where static objects interact with dynamic objects. As an example, a person (dynamic object) uses the faucet (static object), or two persons are talking (dynamic objects). Table 5.1 shows the states and interactions considered for each static object, and the interaction considered between dynamic objects. It also lists the activity occurrences in both scripted and real-life dataset.

After classifier training using the scripted dataset, a validation was performed using recordings of $(4.9 Hours)$ on a regular working day. All recordings were performed in the pantry area. Ground truth for the activity dataset was obtained using manual annotations from a video recorded at 1 fps. The annotations were up-sampled after the recordings to match the sensor data rate.



(a) Thermopile sensor used: Panasonic Grid-Eye.

(b) Office pantry area used for evaluation.

Figure 5.3: Illustrations of the thermopile sensor and placement in the studies. The sensor was placed at the celling, capturing the table corners, microwave, and counter with the faucet, refrigerator, and the coffee pot.

Object and interaction classifiers Classifiers were used to determine state and interaction classes. Table 5.3 presents the features selected and used for classifications per pair of objects. For state classifiers, only features for the reference object were considered. For interaction classifier, the complete set was used. Since multiple dynamic objects could exist in a scene at any given time, the interaction classifiers ran for all object pairs containing the same static object as reference. In the pantry area and corresponding to the number of objects, a total of eight classifiers were used. We used Support Vector Machines (SVM), where parameters had been tuned with a grid search method as suggested by [42] on training data.

The state filtering was implemented using Hidden Markov Model (HMM)s and fitted on the training dataset using hmm-estimate from Matlab. The function calculates the maximum likelihood estimate for transition and emission probabilities given the sequence and known states extracted from the training set. Subsequently the activity interest list refinement was applied. Table 5.2 shows the activity interest lists defined per reference object. For example, if there are four dynamic objects, and the *Coffee Pot* as static object, then the classified interactions are: $[Away, Away, Serving, Present]$. After applying the activity interest list, the recognised result is *Serving*.

Object	Activity		Scripted dataset	Real-life dataset
Coffee Pot	State	Off	1	0
		On	3	1
	Interaction	Away	15	|
		Present	17	|
		Serving	7	27
Faucet	State	Off	8	39
		Hot	3	38
		Cold	3	|
	Interaction	Away	8	913
		Present	6	305
Microwave	State	Off	6	10
		On	3	9
	Interaction	Away	9	|
		Present	6	|
Refrigerator	State	Closed	8	7
		Open	6	6
	Interaction	Away	8	|
		Present	6	|
		Interacting	12	12
People	Interaction	Single	34	24
		Meeting	8	23

Table 5.1: Overview on objects, interactions, and occurrence instances in scripted and real-life datasets.

Reference object	Coffee pot		Faucet		Microwave		Refrigerator		People	
	Index	Activity	Index	Activity	Index	Activity	Index	Activity	Index	Activity
	3	Away	2	Away	2	Away	3	Away	2	Single
	2	Present	1	Present	1	Present	2	Present	1	Meeting
	1	Serving					1	Interacting		

Table 5.2: Activity interest list per reference object as used in our evaluation. The lower the index, the higher the interest (relevance) for the recognition. The interest can be adjusted according to application needs.

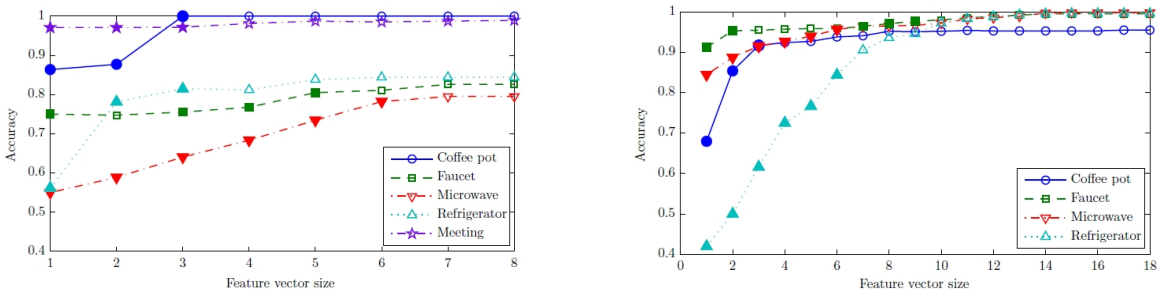
Reference objects		Paired objects	
Index	Feature	Index	Feature
1.	Area temperature product	2.	Area temperature product
3.	Temperature variance	4.	Temperature variance
5.	Area	6.	Area
|		7.	Distance to object 1
8.	Gradient X direction	12.	Gradient X direction
9.	Gradient Y direction	13.	Gradient Y direction
10.	Gradient magnitude	14.	Gradient magnitude
11.	Gradient phase	15.	Gradient phase
16.	Temperature	17.	Temperature
|		18.	Position variance

Table 5.3: Feature set considered for the object and interaction classification. The list is structured into features for state classifiers (Reference objects) and for interaction classifiers (reference & paired objects).

Evaluation procedure To evaluate our approach, we initially determined the relevance of features presented in Tab. 5.3 for both classifiers. We used a variation of the approach presented in [39] to determine relevance. Instead of using the complete feature set jointly, each feature was evaluated individually since a high degree of correlation could be expected. In result, this yielded a much smaller feature vector size. Subsequently, accuracy performance measurements were obtained using the real-life dataset.

5.5 Results

The feature relevance analysis results are shown in Fig. 5.4 for state and interaction classifiers. As the diagrams indicate, the six best features were sufficient to achieve high accuracy for state classifiers. For interaction classifiers, eight features were needed to obtain high accuracy. Choosing additional features did not improve performance for any of the two classifier groups.



(a) **(b)**

Figure 5.4: Accuracy vs. feature vector size for (a) state classifiers, (b) interaction classifiers. The analysis was performed in a 5-fold cross validation using the training dataset.

Fig. 5.5 shows an example of the grid search results obtained for Support Vector Machines (SVM) parameters C and σ. Here, a smooth surface near the Radial Basis Function (RBF) kernel centre with small σ could be observed. This indicates that the SVM should perform well with the test data, which is confirmed by the results shown in Table 5.4. Similar plots where obtained for the other seven classifiers.

(a) **(b)**

Figure 5.5: Support Vector Machines (SVM) parameter grid search for *Coffee Pot* classifiers: (a) state classifier, (b) interaction classifier.

Table 5.4 summarizes classifier modelling performances on the training data. As the result shows, the classifiers can sufficiently model most cases, except for the microwave state classifier. We observed that the microwave states did not result in sufficient temperature difference between *On* and *Off*. Rather, a sequence of interaction events involving the microwave could provide sufficient discriminatory power. This issue becomes more pronounced for the real-life dataset, shown in Table 5.5. As people will simply stand in front of the microwave, e.g. to read the billboard, the object state cannot be reliably determined.

Classifier	Coffee pot	Faucet	Microwave	Refrigerator	Meeting
State	1.00	0.70	0.56	1.00	
Interaction	0.82	0.97	0.86	0.73	0.85

Table 5.4: Normalized average training set accuracies per classifier. The result shows sufficient modelling capability of the approach for most state and interaction classifiers.

Object	Activity		Accuracy
Coffee Pot	State	Off	NaN
		On	100,00%
	Interaction	Away / Present	96,55%
		Serving	96,43%
Faucet	State	Off	80,00%
		Cold / Hot	45,45%
Microwave	State	Off	34,48%
		On	32,14%
Refrigerator	State	Closed	66,67%
		Open	50,00%
	Interaction	Away / Present	24,44%
		Interacting	11,36%
People	Interaction	Single	65,52%
		Meeting	60,71%

Table 5.5: Overview on recognition performance using the real-life dataset for testing classifiers. In the real-life dataset, not all activities were performed, shown here as combined states for some classes.

5.6 Discussion and Conclusion

The 2D-matrix thermopile sensor provides information for detecting complex activities, like serving coffee in a pantry area. Our test scenario showed that the matrix configuration simplified monitoring relatively complex areas. While our approach required to obtain a map of static objects, there was no need to carefully measure overlap of each thermal device with the sensor's field of view. This issue was mentioned by Wren and Tapia [113] as limiting factor for classifying activities using ambient sensors. Although not all interactions described in [113] where tested in this work, e.g. for meetings (corresponding to split and join activities) similar performances were achieved in our approach. Nevertheless, our method required a simpler sensor installation.

The hight at which a sensor is placed determines the tradeoff between coverage and resolution of the scene image. The tradeoff can be observed in Tabs. 5.5 and 5.4, where some good performance ratings obtained during training did not hold for validation. We consider that the performance reduction was due to proximity and size of the areas of interest. For example, the normal use of the faucet, makes people invade the refrigerator area, resulting in erroneous class responses.

Thermopile sensors allow us to recognize multiple activities with one sensing device in places where multiple sensor modalities would have been required. It can be expected that every recognized activity can be mapped to an energy cost, which in turn, can be fed back to the user to guide energy consumption awareness. This energy consumption feedback can be given instantaneously when the activity is being performed. Although our test showed that the sensor can be used to identify complex activities in a scene, the processing was done offline. In further work, the processing architecture could be implemented in the thermopile sensor device.

5.7 Acknowledgments

This work was kindly supported by the EU FP7 project GreenerBuildings, contract no. 258888, and the Netherlands Organisation for Scientific Research (NWO) project EnSO, contract no. 647.000.004.

Chapter 6

Data mining-based localisation of spatial low-resolution sensors in commercial buildings

Chapter originally published as: L. I. Lopera Gonzalez, R. Stier, and O. Amft. "Data Mining-Based Localisation of Spatial Low-Resolution Sensors in Commercial Buildings". In: *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*. BuildSys '16. New York, NY, USA: ACM, Nov. 2016, pp. 187–196. ISBN: 978-1-4503-4264-3. DOI: 10.1145/2993422.2993428

6.1 Introduction

Large numbers of ubiquitous sensors are being deployed in buildings to balance comfort and energy saving, maintain safety, or enforce security. As sensor technology evolves, new device generations with improved performance become available, typically at much faster rates than buildings are renewed. As a consequence, future buildings are likely to contain a 'zoo' of devices, all communicating to the Building Management System (BMS). At the BMS, sensor information is aggregated, processed, and used to provide the desired building features. During building commissioning, sensors need to be correctly associated, e.g., the sensors in a corridor must be correctly identified to activate corresponding ceiling lights. Furthermore, security, remote management, and maintenance services depend on the correct device representation in the BMS. Any newly added sensors must be integrated with the existing BMS configuration. Research on emerging sensor design and modalities has shown that a high sensor density, with multiple devices per building space is required to optimise comfort and energy consumption [64]. As the number of devices increases and building layout evolves, maintenance becomes time-intensive and error-prone, even for regular activities, as finding and fixing configuration errors, locating damaged sensors, or reconfiguring refurbished spaces. Novel methods to automate sensor commissioning and maintenance are needed to support continuous installation and management.

Classic building-installed sensors provide a single point of measurement, i.e., movement detector in a corridor. However, spatial sensors,e.g. radar, thermopiles, camars, etc, which can track objects in the field of view, are becoming common in buildings scenarios. Specially, the low-resolution spatial sensors which can detect but not identify objects. Thus, protecting user privacy. A typical low-resolution ubiquitous spatial sensors provides coarse spatial data only, i.e. less than 100 pixels. Typical applications, involve the use of multiple spatial sensors to monitor an area, their combined fields of view provides more coverage, e.g., to track users across a building space and detect possible safety hazards, such as blockages in emergency paths. For such applications, the sensor's relative position and rotation to its neighbouring sensors is a key information to properly construct a virtual representation of the combined fields of view. Typically, the process of configuration and verification of spatial sensors is manually done, which hampers their widespread use in buildings.

In this chapter , we investigate an approach to determining relative position and orientation of multiple spatial sensors. We envision that the proposed approach can be used at building commissioning time, where the technicians will trigger the correct configuration simply by walking in the field of view of the sensors as they install the network or other facilities. Additionally, once the building is open to the public, our approach can be used to detect faulty sensors or changes in configuration from user-generated movement tracks. Our approach

exploits each individual sensor's object tracking capabilities to extract transition events, then mines link rules to determine each sensor's likeliest neighbours, and finally uses link rules to create a spatial representation of the relative position and orientation of all sensors. The sensors considered in this work provide low spatial resolution only, which renders most vision-based and multi-camera tracking techniques infeasible. We validate our approach using thermopile array sensors. As network topology depends on the building space, we investigate four different building spaces in our analyses. While we focus in this work on thermopile array sensors being a novel type of spatial sensor, we expect that the proposed approach could be used with other spatial sensors too. In particular, the following contributions are made:

1. We propose an data-driven approach, independent of sensor modality and communication media or protocol, that uses data of a spatial sensor to detect and track object's in its field of view. Based on extracted transition events, link rules are mined that describe the probability of any two sensors being neighbours. Finally, a sorting algorithm is developed to determine sensors arrangement.
2. We evaluate our position and orientation estimation approach in four typical installation scenarios of increasing sensor number and complexity, involving corridor, corridor crossing, meeting room, and an open hall setting. For each scenario, we performed a study to obtain realistic movement patterns over at least two weeks. Here we compare performance of the method's variants.

6.2 Related work

A growing number of sensor types and control applications are proposed, addressing all building facilities, including Heating, Ventilation, and Air-Conditioning (HVAC), office appliances, and others. Presence detectors, light sensors [74], power meters [46], are considered a single point of measurement sensors, i.e., they provide a value which can not be mapped to a specific location within the field of view of the sensor. In contrast, microphones [29], high frequency radar [118], and novel thermopile array sensors [40, 67], among others, are considered spatial measurement sensors, because objects can be mapped to a region and tracked as the move through the sensor's field of view.

Thermopile array sensors have been successfully used to monitor activity in different environments. For example, Hevesi et al. [40] used the Panasonic GridEye thermopile array to track household activities. Earlier, Lopera et al. [67] used the same sensor to detect different activities in a office kitchen area. Both investigations mapped the sensor's spatial regions to activities. Mashiyama et al. [71] used the GridEye sensor for fall detection in an elderly people monitoring system. All three approaches used user and object detection using temperature thresholds across multiple sampling frames. We use a similar method in our object detection step. However on top, we track objects within the field of view of individual sensors. Specifically, we were interested in the exit and entry regions and the related time stamp of exit and entry events.

A common drawback of low resolution spatial sensors, i.e., less than 100 pixels, is their limited field-of-view. When mapping sensor regions to activities, as described above, sensor distance is indirectly related to information resolution. For example, for far-away sensors, e.g., more than three meters with a thermopile array sensor, multiple activities will share the same pixel and thus activities become indistinguishable. Similarly, fall detection performance is affected if the sensor is mounted on high ceilings as arms and legs become harder to identify. Typically, the view constraint is overcome by (1) using sensors with higher resolution, e.g. larger pixels count where available, and (2) by installing networks of multiple sensors at suitable target distances. When using sensor networks, the relative positions of the sensors are required. Our method automatically finds the relative position of the sensors so that a combined image of the sensor network can be derived.

While cameras are often considered obtrusive in buildings, the algorithms for tracking across multiple image feeds and mining camera arrangement have parallels to the ubiquitous spatial sensors considered here. In image processing, tracking objects and activities is used to compute exact location of cameras. For instance, Bodor et al. [9] proposed a method to compute the optimal positions of cameras and camera parameters based on observed activities, thus maximising the observability of a scene. In multiple camera scenarios, Zhao et al. [119] proposed a method to track objects over multiple cameras. Nistér [87] show how by tracking objects in multiple video frames, the relative position of the cameras to the object can be reconstructed. Similarly, thermal cameras have been used to recognise activity, i.e., walking, jugging, and running, as presented by Han et al. [37]. Thermal cameras have been used to identify people in biometric applications too, as described by Jain et al. [45].

Camera position inference methods, in both visible light and thermal imaging, utilise image resolutions of typically 64k pixels and above to capture features. The high resolution allows to build statistically robust object models, which can positively identify objects across multiple cameras. In the case of low resolution spatial sensors, object identification across sensors is not feasible. For example, the Panasonic GridEye sensor used in this work provides 64 pixels. People in the sensor's field-of-view may be represented by three to nine

pixels only, and the pixel with an object will only take up to 12 possible values (variation of $\pm 3C^{\circ}$ with $.25C^{\circ}$ resolution). Consequently, none of the typical image processing features can be found in the thermopile sensor image, such as arm position, stride, texture, or colour, making unequivocal object identification between sensors infeasible. Our method uses the tracking information from each sensor and builds a statistical model of possible transitions between sensors.

A foreseeable deployment mode of ubiquitous sensors in buildings is a Wireless Sensor Network (WSN), where localisation is a widely studied problem. For example, Payal et al. [93] and Patwari et al. [91, 92] proposed using the node's own radio signal to estimate location. Furthermore, Aspens et al [5] presented a theory on how to construct graphs that represent sensor location in a given node and proposed how to map location to a real space if the coordinates of a beacon are known. For the proposed WSN methods to work, all nodes need to implement the exact same location estimation method. A change of operating frequency, transmitting power, or communication protocol will render the approaches inaccurate. As new manufacturers, devices, and communication means appear in the building market at a much faster pace than entire buildings are renewed, the typical WSN assumptions imposed for localisation hamper their applicability in the building context. Our approach aims at mining the relative location of sensor nodes from their data, thus our approach is independent of the underlying sensing and communication technologies and infrastructure. Our approach allows the sensor node to be simpler, as the node does not require the hardware/software to deal with the localization problem.

A methodology of finding associations between sensors and organizing them into groups has been proposed by Lopera et al. [62, 63]. Hierarchical and flat groups associate sensors and actuators, which can then be associated to a general part of a building. However, the groups are unable to provide relative location or rotation information of the individual sensors. To our knowledge there are no purely data-driven techniques to find relative or absolute sensor positions in a network.

6.3 Sorting approach

Building occupants moving in a sensor-monitored space create correlated data in the sensors data streams. For example, a person moving across a corridor can be observed as a temporal sequence of objects appearing in the nodes of the sensor network. People moving across the sensor's field-of-view create tracks that can be interpreted as local object tracks. Our sorting approach is based on the idea that orientation and relative location of individual sensors can be estimated when matching the local object tracks between sensors. Depending on the particular sensor application, the installation could be sparse, i.e., leaving gaps between sensors, or dense, i.e., having sensor overlaps. Our proposed approach and subsequent evaluation consider both installation types.

Our sorting approach can be summarised in three processing stages: (1) transition event extraction, (2) link rule estimation, and finally (3) sensor matrix arrangement. Fig. 6.1 illustrates the complete sorting approach. The resulting sensor matrix can be conveniently used to determine interaction and automation controls associated to individual sensors or the complete sensor network. Below, we describe the general approach. The implementations details of the sorting method are described in the subsequent section of this paper.

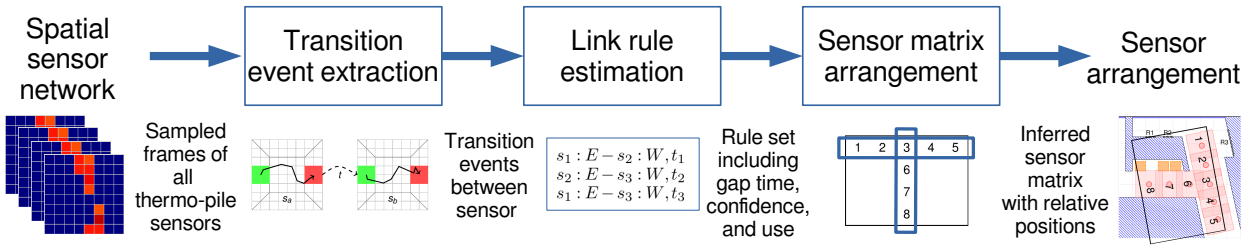


Figure 6.1: Overview on our sorting approach to infer arrangement of spatial sensors and our implementation using thermopile array sensor nodes. Three processing states are used: (1) transition event extraction, to derive relations of sensors based on real-life movement data; (2) link rule estimation, to describe probabilistic relations of sensor nodes; and finally (3) sensor matrix arrangement, to represent the relative position and orientation of a sensor network.

Transition event extraction

We consider that every spatial sensor type has its specific methods to detect and track objects within their field of view. For example, high frequency radar [118] can detect and track people through walls; and speakers can be localized and track using audio based methods [100]. Therefore, we specify the information required from the specialized tracking system and formalize the event extraction as follows: A track in a sensor s is described by the moment, location and region a detected object enters and exits the field of view. A region is a division of the sensor field of view to define general orientation and overcome misalignments which result from the installation process. The length in time of the track is the lifetime. Eq. 6.1 shows the complete description of a track in sensor s, where the entry and exit tuples:$(\dots)_e$, $(\dots)_x$, contain the respective region r, position $\mathbf{X}$, and timestamp t. The life time lt of the track is computed as $lt = t_x - t_e$.

$$T_s = \{(r, \mathbf{X}, t_e, (r, \mathbf{X}, t_x, lt\} \tag{6.1}$$

A transition event e_{sf} is defined as the union of two tracks in sensors s and f respectively, where exit time stamp of the track in s is within and observation window ow from the exit time of the track in f. The gap time d is calculated as shown in Eq. 6.2, where t_{ef} is the track entry time in sensor f and t_{xs} is the track exit time in sensor s.

$$d = t_{ef} - t_{xs} \tag{6.2}$$

The gap time d is effectively the separation between the borders of the field of view of the sensors. Therefore, a negative d implies that an object entered f before leaving s and the sensors are overlapping. The complete definition of a transition event is shown in Eq. 6.3.

$$e_{sf} = \{T_s, T_f, d\} \big| t_{xf} - t_{xs} < ow \wedge s \neq f \tag{6.3}$$

A key benefit of ubiquitous sensors placement is the reduction of meticulous installation, i.e., set-and-forget. A problem which affects the proposed approach is large rotation misalignment, e.g., a sensor is rotated 90 °C with respect to the expected general orientation. Rotations are estimated and corrected as follows: Given regions r_a and r_b, define r_a matching r_b if for two correctly positioned sensors s and f, $r_a = r_{xs}$ and $r_b = r_{ef}$. The rotation of a sensor s is estimated by selecting the region r_n which appears the most in the transition events which contain s, then selecting the most common corresponding region r'_n. If r_n does not match r'_n, define $\delta = r'_n - r_n$, then in all e_{sj} and e_{js} update the regions r_e, r_x of all tracks in s as $r_e + \delta$ and $r_x + \delta$.

Finally, a set of filters is applied to harness the sensors' installation specific knowledge, for example: source of noisy local tracks, details regarding sensor modality and installation procedures. In general, a filter determines whether or not an transition event e_{ij} remains in set of all transitions to be considered for the next steps. Since the filters are application specific, the details are explained in Section 6.4.

Link rule estimation

Extracted transitions events are then aggregated into a link rule. Rules describe the link between originating and receiving sensors (s_i,s_j), regions of origin and destination (r_a,r_b), expected gap time of observed transitions events ($\overline{d}$), confidence (c), and the number of times the transition event was observed (q). Throughout this paper we will use the complete or simplified notations to refer to link rules as shown in Eq. 6.4.

$$\begin{aligned} &\text{Complete notation:} && \{(s_i : r_a - s_j : r_b), c, \overline{d}, q\} \\ &\text{Simplified notation:} && s_i : r_a - s_j : r_b \end{aligned} \tag{6.4}$$

Although transitions are directional, link rules are not. Thus, for sensors s_i and s_j, and connecting regions r_a and r_b, the link rules in Eq. 6.5 are considered identical. As transitions describe the same relative position in both directions, only one rule is extracted. In contrast, link rules are considered different when they link the same sensors, but connect through different regions, as shown in Eq 6.6.

$$s_i : r_a - s_j : r_b \equiv s_j : r_b - s_i : r_a \tag{6.5}$$

$$s_i : r_a - s_j : r_b \neq s_i : r_a - s_j : r_c \tag{6.6}$$

The link rule extraction process works as follows: Define the set of all transition events as $\hat{E}$, then, let $\epsilon \subset \hat{E}$, contain the transition events which have sensors s_i, s_j connecting through regions r_a, r_b respectively.

Parameter q is defined as the size of ϵ. To compute c and $\overline{d}$, a Gaussian kernel is fitted to the gap times d of all $e_{ij} \in \epsilon$ to estimate the gap time distribution $P(d)$. Eq. 6.7 shows how c, $\overline{d}$ and q are computed.

$$\begin{aligned} c &= \max(P(d)) \\ \overline{d} &= E(d) = \operatorname{argmax}(P(d)) \\ q &= |\epsilon| \end{aligned} \tag{6.7}$$

Fig. 6.2 shows examples of different fitted distributions obtained for one day of recordings. Sensor which are neighbours to each other will produce link rules with high c on the neighbouring regions. In contrast, link rules created from spurious transition events will have low c.

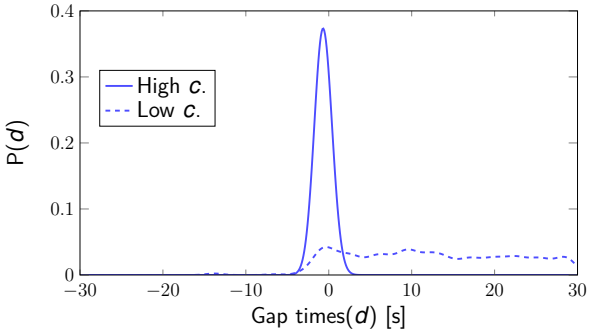


Figure 6.2: Example of probability distributions of gap times for highest and lowest confidence c from a day of recordings. Sensor which are neighbours to each other will produce link rules with high c on the neighbouring regions. In contrast, link rules created from spurious transition events will have low c.

In practice, when noise is present in the data or the use of an area is very low, the following situations will affect the outcome of the link rule estimation process: (1) although only one transition event is necessary to build a rule with high confidence, the rule might be incorrect, especially in busy and noisy environments. True link rules will have more samples but inherently less confidence than the one sample rule. (2) Noisy sensors will trigger many false events, thus some link rules will have an excessive number of events and relatively high confidence. (3) Gap times generated by spurious noisy transition events can be smaller than the real events. (4) Two sensors can have more than one link rule, which implies that the two sensor will connect through different regions. In conventional rule mining techniques, rules are sorted by confidence, support and finally number of samples. None of these measures can be sequentially sorted to guarantee that the correct link rules are used first.

To overcome the situations mentioned above, two processing steps are added to link rule extraction: (1) computing a weight $w_{s:r}$ for sensor s and region r, which modifies the confidence c, and (2) aggregate all link rules for each pair of sensors. The new weight serves as sorting proxy instead of the confidence, and rule aggregation ensures that any two sensors are only connected through one respective region.

The weight $w_{s:r}$ aims at assigning a higher confidence to those sensor-regions pairs that have the expected behaviour, while punishing those which come from noisy transition events. The weight is computed as shown in Eq. 6.8, where $\sigma_{s:r}$ is the standard deviation of the number of link rules containing $s : r$, cm_s is the maximum confidence observed in any link rule which involves sensor s, and finally, $dw_{s:r}$ is a gap time weight as defined in Eq. 6.9 where $\overline{d}_s$ represents all the estimated gap times from all link rules containing sensor s.

$$w_{s:r} = dw_{s:r} \cdot \frac{1}{\sigma_{s:r}} \cdot cm_s \tag{6.8}$$

$$dw_{s:r} = 1 - \frac{\text{mean}(\overline{d}_s)}{\max(\overline{d}_s) - \min(\overline{d}_s)} \tag{6.9}$$

The next step is to convert all link rules which have the same pair of sensors connecting on different regions into a single link rule. The process of unification is called **link rule averaging** and it works as follows: First map each region r_n to its normalized relative position vector $\mathbf{rp_n}$ as shown in Eq. 6.10, where $\mathbf{I}$ is the matrix with the unit vectors of the space where $\mathbf{X}$ is defined, and $\mathbf{rpw}_n$ is the unique weight vector for region r_n.

$$\mathbf{rp_n} = \mathbf{I} \times \mathbf{rpw}_n \Big| \parallel \mathbf{rp_n} \parallel = 1 \tag{6.10}$$

Then, compute the link rule orientation using the relative position vectors as shown in Eq. 6.11, where a link rule for sensors s_i and s_j is shown; the link rule orientation $\mathbf{o}_l$ is computed using the relative position vectors $\mathbf{rp_a}$ and $\mathbf{rp_b}$ for the respective connecting regions r_a and r_b.

$$l = s_i : r_a - s_j : r_b \rightarrow \mathbf{o}_l = \mathbf{rp_a} - \mathbf{rp_b} \tag{6.11}$$

The final orientation $\overline{\mathbf{o}}$ is the result of the weighted sum of rule orientations as shown in Eq. 6.12, where λ is the set of all link rules which join sensors s_i, and s_j.

$$\overline{\mathbf{o}} = \sum_{l \in \lambda} \mathbf{o}_l \cdot c_l \cdot q_l \tag{6.12}$$

Finally, convert the orientation $\overline{\mathbf{o}}$ into regions $\overline{r_a}$ and $\overline{r_b}$ by selecting the region where the distance between the normalized orientation and the respective relative position vector is minimized as shown in Eq.6.13.

$$\begin{aligned} \overline{r_a} &= r_n \left| \min \left(\frac{\overline{\mathbf{o}}}{\| \overline{\mathbf{o}} \|} - \mathbf{rp_n} \right) \right. \\ \overline{r_b} &= r_n \left| \min \left(\frac{-\overline{\mathbf{o}}}{\| \overline{\mathbf{o}} \|} - \mathbf{rp_n} \right) \right. \end{aligned} \tag{6.13}$$

A new link rule $\bar{l}$ for sensors s_i and s_j is written as shown in Eq. 6.14, were c', $\overline{d}\,'$ and q' are the respective maximums observed for all link rules in λ.

$$\bar{l} = \{ s_i : \overline{r_a} - s_j : \overline{r_b}, c', \overline{d}\,', q' \} \tag{6.14}$$

Next, for every sensor-region $s : r$ the best available neighbour is selected as the link rule $s : r - s' : r'$ which has the highest weighted confidence wc_l. Eq. 6.15 shows how to compute wc_l.

$$wc_l = c_l \cdot d_l \cdot w_{s':r'} \tag{6.15}$$

Once the neighbours are decided, the final set of link rules is constructed by selecting first the rules which have confirmation, i.e., $s_i : r_a$ and $s_j : r_b$ were selected as best neighbours respectively; only the link rule with best confidence is selected from the confirmed pair. Finally, the link rules which add information but do not contradict confirmed link rules are added to the list. For example, let link rules $l_1 = s_1 : r_b - s_2 : r_a$ and $l_2 = s_3 : r_c - s_4 : r_d$ be confirmed link rules, and let $l_3 = s_1 : r_b - s_3 : r_c$ and $l_4 = s_5 : r_c - s_4 : r_d$ be unconfirmed link rules. Thus, l_4 adds information about sensor s_5 and it is included in the final set. l_3 states that sensor s_1 connects to s_3 on region r_c, contradicting l_2 which states that s_3 on region r_c has a confirmed neighbour s_4. Thus l_3 is omitted from the final list of link rules.

Sensor matrix arrangement

In the final processing stage, link rules are used to place sensors into the matrix according to the directional information encoded in the link rules. The basic principle of the sorting algorithm is to sort the mined link rules in descending order according weighted confidence wc_l computed as shown in Eq. 6.15.

The sensor matrix $\mathbf{M}$ is a spatial representation of the relative position of all sensors. $\mathbf{M}$ has the same number of dimensions as $\mathbf{X}$ but the units are sensors. The general *sensor-placement* method works as follows: Given a rule $l = s_i : r_a - s_j : r_b$, and $\mathbf{p}_i$ the position of sensor s_i in $\mathbf{M}$. The relative position $\mathbf{p}_j$ of s_j with respect to s_i is calculated as shown in Eq. 6.16, $\mathbf{o}_l$ is the orientation of the link rule as defined in Eq. 6.11.

$$\mathbf{p}_j = \mathbf{p}_i + \mathbf{o}_l \tag{6.16}$$

If $\mathbf{M}$ is empty s_i is selected first and placed in the origin of the space and s_j is placed according to the link rule. A more general interpretation of l is as the direction of travel from s_i to find s_j. Therefore, if $\mathbf{p}_j$ is occupied by another sensor, then s_j is located in the subspace of $\mathbf{M}$ centred in s_i delimited by the infinite extension of $\mathbf{rp}_{a+1}$ and $\mathbf{rp}_{a-1}$. The subspace is termed $\mathbf{M_S}$. s_j is placed in an available position within $\mathbf{M_S}$ minimizing the distance between s_i and s_j. The final position of s_j is considered to be in an inferred position when s_j is not placed adjacent to s_i. For example, assume that l indicates s_j is to the left of s_i, and $\mathbf{p}_j$ is occupied by s_k then *sensor-placement* would place s_j to the left of s_k; s_j is in an inferred position, and link rule l is still satisfied. Inferred positions can be used for sensor placement if the minimization criteria is satisfied. When a new sensor is placed in an inferred location, the existing sensors are moved one position in direction $\mathbf{o}_l$. When s_j is an inferred sensor, and new link rule l_n contains s_j and s_i, such that s_i is already in

$\mathbf{M}$, then, after applying l_n, if s_j does not change position, s_j is considered verified and the inferred status is removed.

As the sorting algorithm progresses, new link rules can add information regarding already existing sensors. Occasionally, the directional information encoded in a new link rule will contradict the current state of $\mathbf{M}$ and is discarded. A contradiction is define as follows: Given a new link rule $l = s_i : r_a - s_j : r_b$, when applying the general interpretation of the rule, s_j is not in the subspace $\mathbf{M_S}$. When the rule is not discarded, the new information is incorporated as follows: Let l_1 be a link rule which positioned sensors s_i and s_j, let l_2 be a link rule which positioned sensors s_i and s_k, finally, let l_n be the new link rule which relates sensors s_j and s_k. From the trio of link rules select the pair which has the minimum sum of gap distances. Discard l_n if it is not in the selected pair. Otherwise, remove the unselected rule from the list of used rules, append l_n, and finally clear $\mathbf{M}$ and start over using the updated list of used link rules.

When both sensors contained in a new link rule l_n are not present in $\mathbf{M}$ and $\mathbf{M} \neq \emptyset$, then l_n is placed in a pending list. The *process-pending-list* procedure is executed every time a new link rule is successfully added using the *sensor-placement*. With *process-pending-list* procedure, the pending list is checked for link rules which contain sensors already in $\mathbf{M}$. Every matching rule is removed of the pending list and added using the *sensor-placement* procedure. The *process-pending-list* procedure is recursive, every successful placement will immediately trigger a call to *process-pending-list* again.

The sorting algorithm ends when all sensors are placed in $\mathbf{M}$, or when there are no more rules available. Algorithm 1 summarizes the sorting algorithm.

Algorithm 1: Algorithm steps to derive the sensor matrix arrangement.

Start with an empty $\mathbf{M}$.;
Select the first link rule l from the wc_l ranked list.;
Place the first sensor pair starting at the origin and following the relative direction.;
repeat
  Select the next link rule l from the list. ;
  Run: *sensor-placement()*;
  Run: *process-pending-list()*;
until *Rules exhausted, or all sensors have been placed.*;

6.4 Implementation

To validate the proposed sorting algorithm, we chose to use networks of thermopile array sensors. Specifically, we used the Panasonic GridEye; a thermopile array sensor which comes in an 8×8 pixel arrangement as shown in Fig. 6.3. Here we describe how the propose method is applied and we give details about the implementation using the selected sensor.

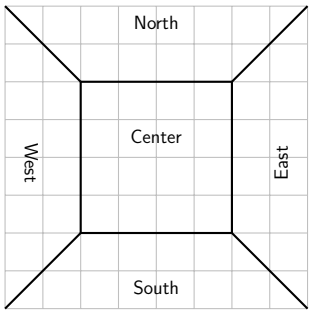


Figure 6.3: Example of the regions definition used with thermopile array sensor in this work. Regions are further used to support the transition event extraction stage. The particular thermopile array is a Panasonic GridEye with an 8×8 matrix.

Object detection and tracking

We based our approach in the object tracking used by Lopera et al. [67], where the 8×8 frame was analysed as follows: First a temperature threshold was applied to the entire frame to select candidate pixels for objects. The threshold was empirically set to the median of the frame plus 1 °C. Objects are started by selecting all local maxima from the candidate pixels. Then, each object is grown by adding neighbouring pixels. The

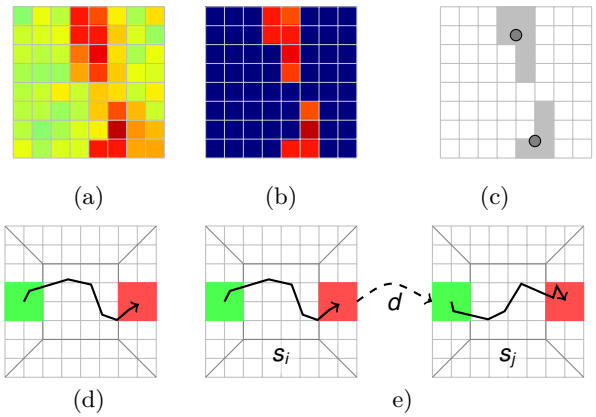


Figure 6.4: Illustration of the transition event extraction for an 8x8 thermopile array sensor. (a) Raw sensor data. (b) Pixel selection based on median and threshold. (c) Detected object with their respective centre of mass. (d) Local object track starting in region W (green) and ending in region E (red). (e) Matching of tracks into a event, beginning (green) and ending (red) regions in the respective sensors. The gap time d is the difference between object entry time in sensor s_j and exit time of sensor s_i.

complete object is extracted by recursively selecting neighbours of neighbours. When two objects in a frame overlap, pixels are assigned following a minimum distance to local maxima criteria. Finally, the object's estimated position is defined as the centre of mass.

A track is calculated by analysing consecutive frames. A track starts when an object o first appears in a frame, the starting point for the track is the position where o was first detected. On each consecutive frame, a new position is added as o changes location. Once o is no longer detected, the track is finished, and the last know position is recorded. In the case of multiple objects, tracks are created by minimizing the distance of the new positions to the last position of the track. If objects get too close and become indistinguishable, i.e., objects merge, all tracks are kept and updated with the position of the merged object. After a split, objects will be assigned to tracks using the minimal distance criteria. There are no guaranties that the object which started the track is the same object that finished the track after a merge or close passing condition. In the case multiple users cross the sensor in close formation, it is possible that they are detected as on single large user; then, only one track is created following the trajectory of the estimated centre of mass of the large object.

Framework implementation

Transition and event extraction: Fig. 6.4 illustrates different steps of the transition event extraction procedure. We defined $\mathbf{X}$ as a plane with origin in the bottom-left corner of the sensor frame, i.e., south west corner; starting at 0, each pixel occupies one unit. Also, the regions are defined as: north, south, east and west (N, S, E, W), as illustrated in Fig. 6.3. The centre region does not make part of the regions set. Tracks which end or start in the centre region are discarded. The observation window ow was chosen to be 30 s.

Filters are applied in sequence and the order of application does not change the final result; except for the relabelling filters and the directional filter. The latter will remove all the transition events on which the relabelling is applicable. We design the following filters to remove unwanted transitions. *Gap time filter:* Removes transition events whose $|d|$ are less than a gap time threshold. *Gap time inconsistency filter:* For transition events on overlapping sensors, i.e., negative gap time, removes transitions whose $|d|$ are larger than lt in either sensor. *Lifetime filter:* The transition event's lt must be greater than a life time threshold. *Relabel orthogonal transitions:* When an object moves on the edges of the sensor, it is likely that the exit and entry regions do not correspond to the direction of movement. For example, an object moving on the north edges of two sensors might have entry and exit positions $\mathbf{X}_e$, $\mathbf{X}_x$ in region N, but the direction of movement reflects a E to W transition. Therefore, both tracks entry and exit regions are updated to match the direction of movement. *Relabel parallel transitions:* When multiple objects are crossing the sensors, objects moving in opposite directions might cross the same region of adjacent sensors within ow. For example, let o_1 be an object crossing sensor s from W to E; a time t later, an object o_2 crosses sensor f in the opposite direction E to W; then, a transition event is triggered with $r_{xs} = r_{ef} = W$. Following the principle of the transition event extraction, the direction is determined by s and thus the destination track is inverted by swapping the entry and exit positions and regions, the times remain unmodified. *Directional filter:* Removes transition events where the direction, determined by the regions r_{xs} and r_{ef}, does not match the local trajectory direction ltd

in sensor s. ltd is defined in Eq. 6.17, where arg max returns the axis of $\mathbf{X}$ which has the dominant position change, and the sign provides the trajectory direction.

$$ltd = \text{sign}(\mathbf{X}_{xs} - \mathbf{X}_{es}) \arg\max(|\mathbf{X}_{xs} - \mathbf{X}_{es}|) \tag{6.17}$$

If the relabel parallel and orthogonal transitions filters are not applied, all rules which would have been relabelled are discarded by transition direction filter. *Use first transition filter:* Only accepts the first transition event created by every track in the destination sensor. *Stationary objects filter:* Removes transition events where the trajectory within either sensor starts and ends in the same position. Implies $(\mathbf{X}_e = \mathbf{X}_x)_s$ or $(\mathbf{X}_e = \mathbf{X}_x)_f$. *Chain of three filter:* Given two transition events e_{ij} and e_{jk} it will allow transitions where the time stamps of sensor j are within ow, i.e., in the case where an object transitioned from sensor i to sensor j to sensor k, the filter searches in the transition event set for the corresponding transition events e_{ij} and e_{jk}, allowing some discrepancy between the time stamps in sensor j of at most ow. *Chain of four filter:* Given two transition events e_{ij} and e_{kf} the same logic applied to building the transition events, is used between sensors j and k. The filter searches for transition events which build a chains that could be created by an object going from sensor i to f passing through sensors j and k.

Link rule estimation: For the link rule aggregation the weight vectors $\mathbf{rpw}_n$ were defined as ([1, 0], [-1, 0], [0, 1], [0,-1]) for N, S, E, W respectively.

Sensor matrix arrangement: Using the selected regions all relative position can be found in a plane. Thus, $\mathbf{M}$ can be represented by a matrix with the sensor id as seen in Fig. 6.6. The starting $\mathbf{M}$ is a 3×3 matrix and the origin is the centre position. The matrix $\mathbf{M}$ is expanded as needed.

6.5 Real-life evaluation

We validated our mining approach using four different real-life scenarios: (a) office corridor, (b) meeting room, (c) corridor with a T intersection, and (d) open space "foyer" area. The scenarios were chosen to provide different geometries with varying sensor arrangement complexity. Specifically, we aimed at progressively removing restrictions to the various ways in which the sensors could be positioned.

Evaluation scenario setup

All scenarios were located in different buildings on a university campus. Permission was obtained from the building management to install the sensors and record data. In all scenarios sensors covered public building areas. Sensors were small devices fixed to the ceiling, thus did not affect building users in their activities. Information plates were installed before and during the recordings of each scenario with study manager contact details, study description and an illustration of the data collected.

We used the 8×8 Panasonic GridEye sensors connected to an Arduino UNO board using I^2C protocol. Each Arduino UNO sampled up to two GridEye sensors at 10 Hz and sent measurements to a computer via USB connection. A cabinet rack was used to store the computer during the experiments.

Fig. 6.5 shows scenarios during the recordings and summarises the technical specifications and recording duration for each scenario.

Office corridor scenario

Sensors were mounted in a straight line leaving gaps between each sensor's field-of-view. Fig. 6.6 illustrates the sensor arrangement. Sensors recorded users coming in and out of offices. The corridor area could otherwise be accessed at both ends. The corridor was recorded between the months of January and February 2015. At peak hour, an estimated average of 36 users used the corridor.

Meeting room scenario

The meeting room consist of a square of eight sensors positioned on top of the meeting room table. Unlike the corridor, there were overlaps in the sensors field-of-view as shown in Fig. 6.7b. The meeting room is a stationary scenario, as users stay confined to their sitting places for most of the time. The meeting room was recorded between the months of September and November in 2015. At peak hour, an estimated average of 7 users were present.

(a) Corridor: 15 sensors, 48 days, height 2.40m, and area 124m^2

(b) Meeting room: 8 sensors, 54 days, height 2.65m, and area 26m^2

(c) T-crossing: 8 sensors, 23 days, height 2.80m, and area 31m^2

(d) Foyer: 20 sensors, 51 days, height 2.80m, and area 78m^2

Figure 6.5: Pictures of the four evaluation scenarios used in this work. For the corridor and meeting room, sensors were mounted on the ceiling. A recording rack is visible for the corridor scenario (Fig. 6.5a). For T-crossing and foyer scenarios, sensors were attached to grey pipes that were then installed hanging from the ceiling. The pipes contained the installation wiring.

T-crossing scenario

The corridor and the meeting room are restricted to two neighbouring sensors. The T-crossing scenario, extends options to at least three neighbours. The space includes vending machines. The intersection gave access to classrooms, hence at specific times many people were moving simultaneously in the monitored area. The T has an angle of 13°, which is different from other scenarios having strict orthogonal arrangements. Sensor installation height was of 2.80m, which is at the upper range limit of the sensors, lowering temperature differences between background and observed people. Fig. 6.7a depicts the area and sensor placement. The T-

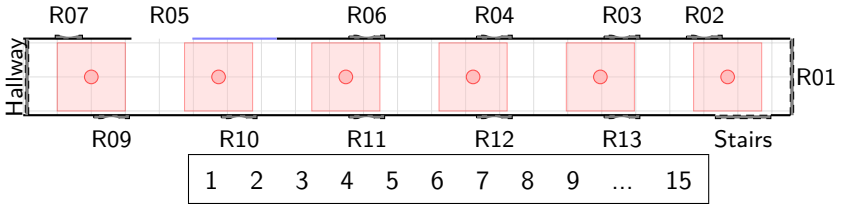
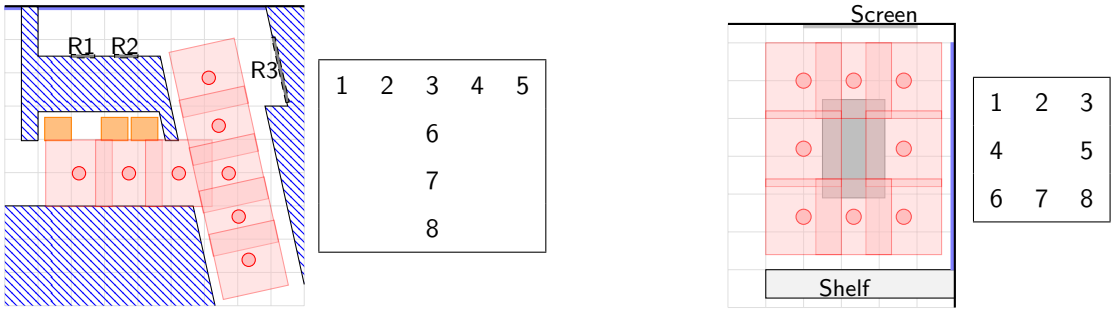


Figure 6.6: Floor maps of the corridor scenario illustrating sensor placement and corresponding sensor matrix representation **M**. Only 6 out of 15 sensors are shown.

crossing was recorded in November 2015. At peak hour, an estimated average of 50 users used the T-crossing.



(a) T-crossing **(b)** Meeting room

Figure 6.7: Floor maps of the (a) T-crossing and (b) meeting room scenarios illustrating sensor placement and corresponding sensor matrix representation **M**.

Foyer scenario

The foyer was the most complex scenario considered, involving walking and sitting areas and multiple access points. The sitting area was covered with a dense sensor placement, while gaps in the field-of-view were left in the walking area. The foyer gave access to class rooms, thus people moved in flocks at specific times. In addition, luggage lockers were available in the covered area, where people would remain stationary for brief period of time. Sensor installation height was also $2.80m$. Fig. 6.8 illustrates the sensor placement, and space characteristics. The foyer was recorded between December 2015, and January 2016. At peak hour, an estimated average of 80 users used the foyer.

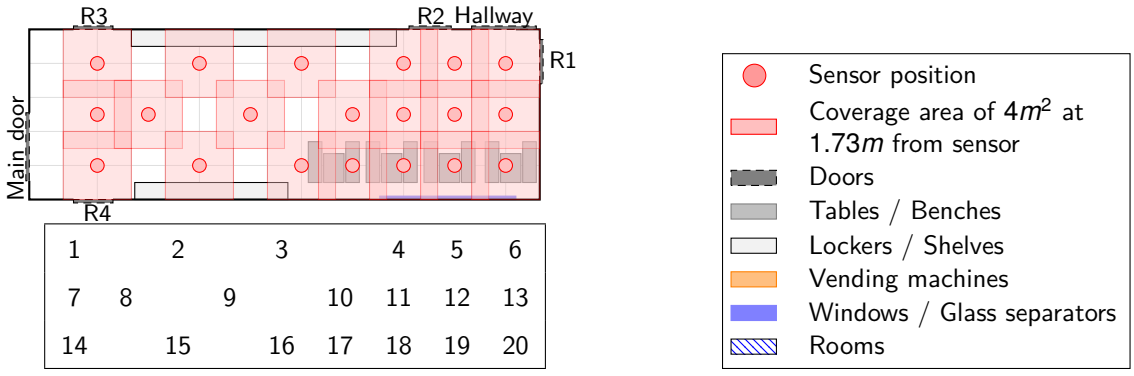


Figure 6.8: Floor maps of the foyer scenario illustrating sensor placement and corresponding sensor matrix **M** representation.

Evaluation metrics

To evaluate the mining performance of our approach, we measured the accuracy of the sensor arrangement with respect to the reference shown in Figures 6.6, 6.7a, 6.7b and 6.8. Accuracy was calculated as follows:

Given the inferred sensor matrix $\mathbf{M}$ with the reference matrix $\mathbf{M_{GT}}$, let $\hat{M}_i$ and $\hat{M}_{iGT}$ be the set of columns or rows from $\mathbf{M}$ and $\mathbf{M_{GT}}$ respectively. Partial accuracy pa_i is computed for the columns and rows sets independently as shown in Eq. 6.18. Total accuracy is given by $(pa_c + pa_r)/2$.

$$pa_i = \max_{\forall i \in \hat{M}_i,\ \forall i' \in \hat{M}_{iGT}} (\parallel i \cap i' \parallel) \cdot \frac{1}{\parallel \hat{S} \parallel} \tag{6.18}$$

We performed a parametric search on the observation window $ow = (30\,\text{s},\ 15\,\text{s},\ 10\,\text{s},\ 5\,\text{s},\ 2.5\,\text{s})$ and the minimum object life time $lt = (0\,\text{s},\ 0.5\,\text{s},\ 1\,\text{s}\ ,2\,\text{s})$. For each choice of parameters we tested different combinations of the filters defined in Section 6.4. In all test the *Basic* combination was used which consists of the gap time, life time, gap time inconsistency, relabel parallel and orthogonal transitions, and directional filters. Additional combinations were tested in conjunction with the basic combination. The filter combinations where chosen as follows: *Chain of 3* uses the chain of three filter with *ow* gap time. *Chain of 3 - 0* uses the chain of three filter with $ow = 0$. Special case where the time stamps of the common sensor j are equal. *Chain of 4* uses the chain of four filter with *ow* as gap time. *Combined* uses the first transition and stationary filters. *Full Chain of 3* uses the stationary objects and chain of three filters with *ow* as gap time. *Full Chain of 4* uses the stationary objects and chain of four filters with *ow* as gap time. *Full Chain. 3 - 0* uses the stationary objects and chain of three, with $ow = 0$. *Stationary* uses the stationary objects filter. *1^{st} Transition* uses the first transition filter. In the case of the full chain combinations, the first transition was not used, as it would considerably diminish the possibility of the chain filters to find matching transition events.

Additionally, we analysed the sorting performance when the link rules from several days were combined. We used a sliding window of size *ws* days to collect and combine link rules. We used values for *ws* in the range 1–16 days.

Finally, we tested the rotation detection and correction performance by randomly selecting a sensor s and a rotation $\delta \in \{\pm 90°, 180°\}$, then all tracks of sensor s used in transition events were modified by updating r_e and r_x with δ. Then we measured if the sensor were set back to the original state. Due to the large amount of possible sensor rotation combinations between all sensors, we opted for only simulating rotations of up to five sensors simultaneously.

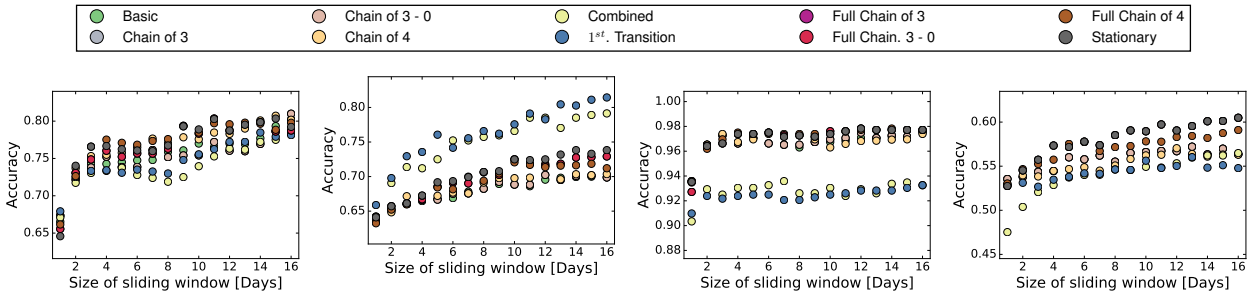
6.6 Results

We made a sweep of the filter parameters and combinations used to extract the transition events and measure the average daily accuracy. The best minimum life time lt was 0 s for all scenarios. For the observation window *ow*, a value of 30 s was the best for the T-crossing and the foyer, 10 s for the meeting room and 5 s for the corridor. The filter type with best performance was the 1^{st}. Transition for the corridor and meeting room, where as for the foyer and T-crossing scenarios, the Chain of 3 - 0 was the best filter configuration. Table 6.1 shows the maximum and average daily performances for all filter combinations. The T-crossing is the most dynamic of the scenarios and it benefits from having the users generally traversing the entire scenario. In contrast, the corridor suffers from the spurious correlation of users entering and exiting the rooms on ether side. Specially at lunch times where users leave the rooms at the same time other users are traversing the corridor. Therefore, a small *ow* focuses on tracks created by people actively moving through the corridor. The meeting room is a stationary scenario, users are sitting around the table most of the time, and the only usable tracks are made when getting to the chairs or when leaving. Therefore, a smaller *ow* is needed to differentiate dynamic from stationary tracks. However, in comparison to the corridor scenario, the larger *ow* points to slower tracks, which are expected as users crossing the corridor are much faster than when approaching to a table. In the meeting room scenario the $\approx 65\%$ accuracy is explained by the lack of use of the scenario, and the false transition events created by spurious user movements while sitting. The foyer scenario had predominantly a dynamic behaviour which is reflected in the larger *ow*. However, users followed preferred paths to move in and out of the rooms, thus the number of generated tracks over all sensors was not equally distributed. Additionally, the sitting area presented the same restriction as the meeting room. The performance is not related to the number of sensors in the scenario, rather to the space geometry and sensor distribution.

We studied the performance of the system when using sliding windows to aggregate the rules of multiple days, and found that performance improves and stabilizes with a window size of 5-6 days for all filter configurations. All scenarios showed and additional increase in performance after 10-12 days. The use frequency explains the performance stabilisation, e.g., five days corresponds to the work week, where most transition events can be found. The meeting room was less frequently used, thus showing a rather linear performance trend. Fig. 6.9 illustrates accuracy convergence for all scenarios.

	Corridor		Foyer		Meeting room		T-crossing	
	max	mean	max	mean	max	mean	max	mean
1^{st}. Transition	1.00	0.679	0.648	0.529	1.0	0.659	1.0	0.910
Basic	1.00	0.665	0.705	0.536	1.0	0.635	1.0	0.936
Chain of 3	1.00	0.674	0.705	0.536	1.0	0.642	1.0	0.936
Chain of 3 - 0	1.00	0.674	0.705	0.536	1.0	0.642	1.0	0.936
Chain of 4	1.00	0.672	0.634	0.529	1.0	0.638	1.0	0.935
Combined	1.00	0.671	0.672	0.475	1.0	0.642	1.0	0.903
F.C. of 3	0.93	0.656	0.648	0.528	1.0	0.638	1.0	0.927
F.C. of 3 - 0	0.93	0.656	0.648	0.528	1.0	0.638	1.0	0.927
F.C. of 4	1.00	0.662	0.656	0.530	1.0	0.632	1.0	0.935
Stationary	0.90	0.646	0.648	0.528	1.0	0.641	1.0	0.935

Table 6.1: Accuracy results for the different filter combinations. F.C. denotes full chain combinations.



(a) Corridor $ow = 5.0$, $lt = 2.0$

(b) Meeting room $ow = 2.5$, $lt = 2.0$

(c) T-crossing $ow = 5.0$, $lt = 0.0$

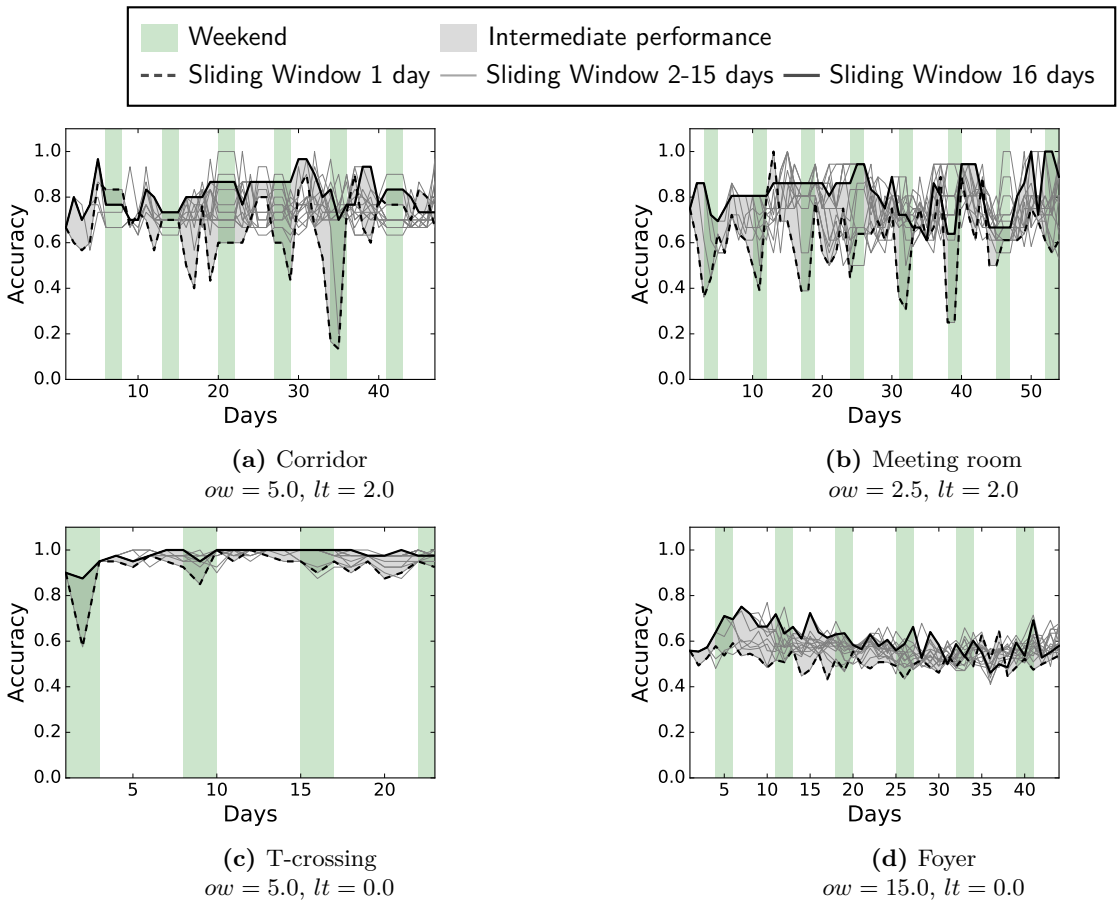
(d) Foyer $ow = 15.0$, $lt = 0.0$

Figure 6.9: Average performance of the different filter configurations as a function of the window size. The performance increases rapidly with window size ws, but then stabilizes at window sizes of 5-6 days. Use patterns explains the stabilisation periods, e.g., five days corresponds to the work week, where most transition events can be found. The meeting room was less frequently used, thus showing a rather linear performance trend.

Although, maximum daily performance for most scenarios was already achieved with daily evaluations, i.e., $ws = 1$, for the foyer scenario, the maximum accuracy reached in any given day was 0.78 with $ws = 12$, $ow = 2.5$, $lt = 2.0$ and the combined filter configuration. A small ow and a large minimum lt, benefit slow moving targets, the combined filter configuration will focus on people entering/exiting the sitting area, as well as in the correct transitions. The large ws implies that the parameters, in combination with the filter, require time to gather the enough correct link rules. Unfortunately, when comparing with the best average performance in Fig. 6.9d, it is clear that the conditions for reaching the best order are not commonly occurring in the normal use of the space.

Fig. 6.10 shows the daily performance of the best parameters and filter configuration for each scenario. Corridor and T-crossing scenarios evidence the importance of activity in the area, the low peaks in performance of the one-day sliding window clearly match with weekend periods. In the meeting room and the foyer, the one-day sliding window is usually insufficient to reconstruct the order, whereas an increase in ws substantially improves performance as more rules can be aggregated together. However, the performance of the 16 day window was occasionally surpassed by smaller size windows.

We tested the ability of the rotation detection and correction to fix sensors that were rotated using only the basic filter configuration. In all scenarios the sensor rotations were correctly identified most of the time. On average, the meeting room was able to correctly identify rotations in a 71% of the cases, while the rest of the scenarios yielded a daily average of 94%. The low performance of the meeting room was due to the large number of false transition events created by the spurious user movements while sitting.



(a) Corridor
$ow = 5.0$, $lt = 2.0$

(b) Meeting room
$ow = 2.5$, $lt = 2.0$

(c) T-crossing
$ow = 5.0$, $lt = 0.0$

(d) Foyer
$ow = 15.0$, $lt = 0.0$

Figure 6.10: Daily performance of the stationary filter configuration using sliding windows of different sizes and the best parameters ow and lt for overall average performance. The weekends have reduced activity, thus the one-day sliding window shows performance drops. As the sliding window size increases, performance is maintained over the weekends.

6.7 Conclusion and further work

We developed and evaluated a mining method to infer relative positions and orientations of spatial sensors within a network. We tested our approach using thermopile arrays which are a typical example of a spatial sensor. The sensor arrangement was inferred based on people movement patterns only, thus involving no supervised information. Evaluations in four different real-life scenarios were performed across recording duration of 23 to 54 days, to derive insight on realistic performance and identify open challenges. Overall, the evaluation results show that our mining method works excellent in scenarios where the sensors are restricted to linear shapes, i.e., any hallway, meeting room and corridors. Here, we successfully retrieved the correct order at least once. Relative position of open spaces like the foyer are harder to identify due to the combination of sedentary and moving areas. More observation days improves robustness against missing sensors due to lack of area use, e.g, the 16 day window size yielded constant performance even over weekends when less people were present. To reduce the performance variance when using different window sizes further refinements of the aggregation method and link rule ordering could be made. In the current state, we envision that our approach can be used during building commissioning and maintenance. Our mining method could be used to detect broken sensors, reorganise BMS information after device replacement, or after building space rearrangements.

6.8 Acknowledgements

The authors would like to thank the building management team from University of Passau for their help with sensor installation and allowing us to do the experiments.

Chapter 7

Bayesian Rule Extraction

7.1 Introduction

Mining time-series association rules is an interesting field of study with multiple applications. For example, rule-mining has been used to find the automatic arrangement of spatial sensors in corridors, meeting rooms and foyers [66]. In activities of daily living, rule-mining is used, as part of a framework, to extract complex activities such as house cleaning, which is composed of other activities such as washing dishes and mopping the floors [60]. In the emerging field of smart building maintenance, rule mining is used to find commissioning errors, automatic integration of Internet of Things (IoT) devices into the Building Management System (BMS) and update of sensors' locations [62, 63].

A time-series is the evidence of a process evolution in time. Rule mining is one of the many strategies which uses the time-series to model the underlying generating process. Specifically, rule mining extracts the temporal relationships of symbols in the time series. In a deterministic process, rules reflect the cause and effect relationship between the inputs and outputs of the model, e.g., when the switch is pressed then light turns on. In stochastic models, a rule reflects the likelihood of a process, e.g., when the office door is opened, the most likely event to occur next is desk presence. Real life time-series rule mining is faced by the challenge of extracting the relationship between symbols when the time-series is generated by the combination of multiple stochastic and deterministic processes. An example of a real-life application is people moving inside a building and the corresponding BMS responses. As variables are measured and converted into symbols, both the stochastic processes and the control systems will generate sequences of symbols which are related. For example, a building occupant enters the main door, and walks along the corridor to get to his/her office, as the occupant enters the corridor, the ceiling lights activate to maintain illumination. Additionally, several other occupants will follow similar paths and generate corresponding building responses. In the BMS time-series, patterns emerge where events of the main door are followed by corridor sensors and actuators and finally the office of each individual occupant. However, the entrance pattern is interlaced with the other patterns caused by users moving about in the building, e.g., users going from one office to another. Current data mining approaches use minimum support or minimum confidence as thresholds to find the rules and reduce the search space. However, thresholds on support and confidence are calculated by parametric searches and they become application specific. Furthermore, there is no intuitive way of knowing what is an adequate value for the thresholds nor can they be computed analytically. Additionally, support-based methods suffer from false associations. A rule which associates a rare and a frequent symbol will take precedence over a rule which associates two rare symbols. Similarly, high frequency symbols generate enough support and confidence of random symbol associations, which can bypass the thresholds.

Our proposed solution assumes that the time-series' generating deterministic and stochastic processes can be broken down into a set of atomic rules, $premise \rightarrow conclusion$, which form the constituents of the model, e.g., nodes and edges in a graphical model. Given the strenuous work required to clean up rules and find appropriate thresholds, we break the problem in two parts: (1) extract atomic rules from the time-series, (2) assemble the extracted rules into complex models with predictive capabilities. This chapter focuses on the first step: extraction of atomic rules. In this work, we present the Bayesian Rule Extractor (BRE), where we use the idea of increasing belief as criteria for rule selection. Formally, we use Bayesian inference to determine the likelihood of a rule. If the posterior is less than the prior, the rule is discarded. BRE focuses on the premise, which is the first symbol of the observation window ow, i.e., BRE determines what events are generated due to the premise. Then, BRE creates candidate associations with all remaining symbols in ow. BRE traverses the time-series only once, and has only one parameter: observation window size $|ow|$.

We propose three filters, which also work on probability measures alone and do not require parameters nor thresholds. BRE and the proposed filters present the following features:

1. Bayesian Rule Extractor (BRE) and its filters do not use thresholds on support or confidence.
2. Rules between rare symbols are always extracted.
3. High frequency noise rules are removed, as well as false rules composed of a rare and a frequent symbol.

We present the evaluation of the algorithm in a synthetic dataset as a well as two real-life datasets: a routine and activity diary and a smart building's living-lab recording. We present properties of the algorithm and give comments regarding how to select the observation window size $|ow|$.

7.2 Related work

Hipp et al. [41] describe the general formulation of the rule mining problem as follows: let $V = s_1, \ldots, s_n$ be the vocabulary of distinct symbols. A rule $A \to B$ is constructed of item sets such that $A \subseteq V$ and $B \subseteq V$. To determine that A and B are related, a partition of the dataset T must include A and B. When $A \cup B \subseteq T$ is said that T supports the rule $A \to B$. In general, support is defined as the number of partitions T which support the rule with respect to the size of database partitions. For example, a shopping receipt is a partition of the database of sold items, after analysing multiple receipts a rule can be extracted where if milk, sugar, flour and eggs are bought, then candles and paper plates are also on the receipt. Bayesian Rule Extractor (BRE) is designed for time series analysis, where the partition T is created by the observation window ow and the item set can only contain one symbol, i.e, $|A| = |B| = 1$. BRE follows an atomic rule approach, where instead of looking for the largest possible subsets of T, it looks for the most common association between individual symbols, e.g., $s_1, \to s_2$. The atomic rule approach reduces the complexity of the search space from growing exponentially with $|V|$, i.e., $|V|^{|V|}$ to $|V|^2$. More complex associations, i.e., rules of the form $A \to B$ can be constructed in post-processing, but the effectiveness and complexity of posterior methods is application dependent.

Rule mining algorithms focus on how to find frequent rules based on two thresholds: minimum support and confidence, as stated by Hipp et al. [41]. The difference between algorithms is the way the dataset is traversed to find the rules. For example, Muyeba et al. [80] proposed a method, based on fuzzy sets and weighted support, to ensure that rules with low support and high confidence are not discarded. Recent work such as Chen et al. [12], Guillame-Bert and Crowley [35], and Liu et al. [60], also used thresholds on support and confidence to determine which rules to keep. Najafabadi et al. [81] combines collaborative filtering, associative rule mining and clustering to improve the performance of a music recommender system. However, the rule mining module is still based on the principle of minimum support. Shokoohi-Yekta et al. [104] proposed a framework which combined motif search techniques with rule mining. Instead of using a threshold on support, Shokoohi-Yekta et al. [104] used a threshold on the probability of finding better rules. BRE is the first algorithm for rule mining without any threshold on the rule's support, confidence or the probability of finding a better rule. Instead, BRE uses the idea of update belief in the rule after each observation to determine which rules to keep. Additionally, We propose three threshold-free filters to control which types of rules to select.

One of the most advanced time series rule mining algorithms is the Temporal Interval Tree Association Rule Learner (Titarl), which was introduced by Guillame-Bert and Crowley [35]. The goal of the algorithm is to produce complex rules involving multiple symbols and with a precise tree of temporal associations. In general, for each rule, Titarl adds conditions to increase the rule complexity, separates rules and refines the timing histograms to better predict the conclusion of the rule. Titarl has two drawbacks: first, it requires different thresholds and conditions to perform correctly and to know when to stop; second, the price to pay for the added rule complexity, makes Titarl unsuitable for large datasets. In comparison, BRE is a lightweight rule extraction algorithm which uses no thresholds. The only parameter is the observation window size $|ow|$, which can be estimated with some basic knowledge of the application. However, the direct output of the algorithm is atomic rules, which require further processing.

A common problem of rule mining algorithm which rely in the minimum support threshold is the problem of rarity as defined by Weiss [110]. Since BRE does not use the minimum support threshold, rare rules can be captured without suffering the combinatorial explosion.

The Bayes theorem is commonly used in machine learning algorithms. The classic example is the Naive Bayes classifier used to filter spam e-mail. However, Bayesian theory, in general, has been applied to graphical and other classification models to overcome the naive assumption of conditional independence between features [8]. When applied to classification, Bayesian models use posterior maximization to assign a class to the sample. In contrast, BRE tracks the rule $a \to b$ belief as the posterior probability of finding a symbol b

given that symbol a was observed. BRE recursively updates the posterior probability after each rule observation, using as prior the posterior of previous rule observations. BRE's rule selection criteria is the monotonic increase of the belief.

7.3 Algorithm

Bayesian Rule Extractor (BRE) algorithm mines atomic rules of the form $a \rightarrow b$, which represent the relationship between symbols a and b, where, if the premise a is observed, then, the conclusion b will also be found within an expected time interval, i.e., within an observation window ow. BRE is a data driven approach and assumes that the observed data is a representative sample of the underlying processes. Therefore, BRE uses frequency statistics to calculate probabilities. BRE uses confidence and belief to extract pair-wise relationships between symbols from the time-series. Confidence is calculated using the probability of the rule $P(a \rightarrow b)$, which is equivalent to the conditional probability $P(b|a)$. Eq. 7.1 shows how, using the Bayes theorem, the rule's confidence can be simplified to the number of times the rule $a \rightarrow b$ has been observed, over the number of premises a observations. Furthermore, $P(a|b)$ is the number of times that $a \rightarrow b$ is observer over the total number of conclusion b observations, $P(a)$ and $P(b)$ are the respective number of symbol observations relative to the total amount of observed symbols s_t.

$$\begin{aligned} P(a) &= \#a/s \\ P(b) &= \#b/s \\ P(a|b) &= \frac{\#(a \rightarrow b)}{\#b} \\ P(a \rightarrow b) = P(b|a) &= \frac{P(a|b) \cdot P(b)}{P(a)} = \frac{\#(a \rightarrow b)}{\#a} \end{aligned} \tag{7.1}$$

The belief in a rule $a \rightarrow b$ is a recursive process. When the rule is first observed, the rule's confidence $P(b|a)$ is the initial belief $B(b|a)_1$. As rules are observed, the rule's belief $B(b|a)_k$ is updated using the Bayes theorem, where the prior probability p is the rule's belief in the previous observation $B(b|a)_{k-1}$ as shown in Eq. 7.2.

$$B(b|a)_k = \frac{P(a|b) \cdot p}{P(a|b) \cdot p + (1-p) \cdot (1 - P(a|b))} \tag{7.2}$$

We considered two types of time-series for BRE's design: ed-sampled and sampled. When symbols appear at equidistant time intervals, the series is called an ed-sampled time-series, and the relative position of the symbols in the array measures the rule's timing. As shown in the ed-sampled time-series example of Eq. 7.3, the timing between symbols a and d is one, and between a and g is three. The time unit depends on the sample frequency of the equidistant samples. On the other hand, in a sampled time-series symbols appear at arbitrary times, and the difference between the symbols' time-stamps measures the rule's timing. In the sampled time-series example of Eq. 7.3, the time between symbols a and d is $t_2 - t_1$.

$$\begin{aligned} &\text{time-serires examples} \\ \text{ed} - \text{sampled} &= [\dots, a, d, f, g, d, \dots] \\ \text{sampled} &= [\dots, t_1 : a, t_2 : d, t_3 : g, \dots] \end{aligned} \tag{7.3}$$

We designed BRE with only one parameter: the observation window size $|ow|$, which defines the maximum wait time for the conclusion b to appear, i.e., defines the maximum rule timing. With $|ow|$ set, BRE traverses the time-series one sample at the time. Starting with the oldest symbol, BRE uses the following five steps to mine rules: (1) retrieve observation window ow, (2) create candidate rules cr, (3) update related rules, (4) time candidate rules cr, (5) apply belief criteria. Additionally, filters can be applied to improve the algorithm's output. Given a symbol a at time t, BRE *retrieves the observation window ow* by selecting symbols from the time series which fulfil Eq. 7.4, where t_b is the time stamp of the conclusion symbol b, and $|ow|$ is the size of the observation window ow. For ed-sampled time-series, $|ow|$ represents the number of symbols to consider including symbol a.

$$t_b < t + |ow| \tag{7.4}$$

Once the observation window ow is retrieved, *candidate rules* (cr) are constructed by using the first symbol from the ow as the premise and all other symbols in ow as conclusions in the form: *premise* $\rightarrow$ *conclusion*. If a symbol appears multiple times in ow, only the first appearance of the symbol is considered. When the premise has been seen already, *related rules are updated*. A related rule contains the same premise as the

candidate rules cr but is not in the candidate rules cr. The belief in a related rule is updated by using the rule's confidence as the prior, effectively resetting the belief in the related rule. The rational for the update follows the idea that the belief in a rule has to change when the premise is observed but the rule is not present in cr. Additionally, the update is necessary to deal with the absolute belief property (see Prop. 1), which states that if the belief on a rule is one, the belief will always be one. Thus, if a related rule has a belief of one, it will be updated to match the fact that the premise appeared and not the conclusion, so the belief in the rule should not be one.

The next step in the algorithm is to time all rules in cr. *Rule timing* is defined as the symbol distance between the conclusion and the premise; in ed-sampled time-series the timing is simplified to the difference in symbol's positions in ow, while for sampled time-series, rule timing is the difference between the conclusion and the premise's time-stamps. Eq. 7.5 shows an ed-sampled time-series example of an observation window ow of $|ow| = 5$. In addition, Eq. 7.5 shows the candidate rules cr with their respective timing. For every rule in cr, a histogram is created of the sampled times. In ed-sampled time-series, the histogram is made using $|ow|$ as the number of bins, and the bin size equals to one. For sampled time series, automatic histogram creation is used. We introduced a new method for dynamically computing histograms (see Section 7.3). Our method ensures bin resolution in areas where the samples are clustered together. The proposed method is used to improve the approximation of the rule timing's expected value.

$$\begin{aligned} ow &= [a, b, c, d, b] \\ cr &= [(a \rightarrow b : 1), (a \rightarrow c : 2), (a \rightarrow d : 3)] \end{aligned} \tag{7.5}$$

When symbols have no self dependence, e.g. $a \rightarrow a$ does not exist in the generating process, BRE can use an alternative timing mode, which creates individual observation window sizes $|ow^*|$ for each symbol in the time-series. In the alternative timing mode, when the premise of ow appears also as a conclusion, e.g., $a \rightarrow a$, then the timing between the symbols becomes the new $|ow|$ for the premise. Eq. 7.6 shows an example of the change in observation window size from ow to ow^* for premise a. Every time $|ow^*|$ changes, all rules related to the premise are updated as follows: each related rule instance with timing greater than the new $|ow|$ are removed; then, the timing histogram is recreated; finally, if the rule is left without samples, the rule is discarded. BRE's alternative timing mode was created to maintain performance in situations where an arbitrary large $|ow|$ was set. For example in the building environment, when the heating, ventilation, and lighting variables are overlapped in the Building Management System (BMS)' time-series, BRE needs a sufficiently large $|ow|$ to capture the heating and ventilation relationships, whereas the lighting associations will benefit from a shorter $|ow|$.

$$\begin{aligned} ow &= [a, b, c, d, e, f, g, a, v, b, t, p] \\ ow^* &= [a, b, c, d, e, f, g] \end{aligned} \tag{7.6}$$

The next step in the algorithm is to *apply the belief criteria*. For every rule in cr, the confidence and the belief are computed. Using k as the number of observed symbols up to and including the premise, the rule's confidence is calculated using $k + 1$ as the total number of observed symbols s_t. As a result, the position of the rule's conclusion in ow does not affect the computation of probabilities, which makes BRE robust against symbol insertions between the premise and the conclusion. BRE uses the following criteria to consider a rule as true: the belief of the rule should not decrease over time. Rules which do not fulfil the criteria are discarded. If a rule reappears in the time-series after being previously discarded, the belief of the rule is computed using the current rule's confidence as a prior, which increases the likelihood that the rule will be kept. Finally, after the observation window ow is processed, a new ow is selected by shifting the premise of the window by one symbol. BRE's general procedure is shown in Alg. 2

Algorithm 2: Bayesian rule extraction algorithm.

```
Input: Time-series (ts), |ow|, set of filters
Result: Set of atomic rules.
for premise a in ts do
    ow = ts [a.index : a.index + |ow|];
    cr = candidate_rules(ow);
    update_related_rules(a, cr);
    time_rules (ow, cr);
    for rule r in cr do
        if r in atomic_rules then
            prior = atomic_rules [r ].belief
        else
            if r in history then
                prior = confidence (r)
            else
                b = r.conclusion;
                prior = P(b);                          /* Probability of the conclusion b */
            end
        end
        if belief (r, prior) ≥ prior then
            atomic_rules [r ] = [confidence (r), belief (r, prior), disjunction (r)]
        else
            atomic_rules.remove(r)
        end
        history.append({r:[confidence (r), belief (r, prior), disjunction (r)]})
    end
end
for rule r in atomic_rules do
    /* Remove r from atomic_rules if r does not pass all the filters.          */
    if not all (map (r, filters)) then
        atomic_rules.remove(r)
    end
end
```

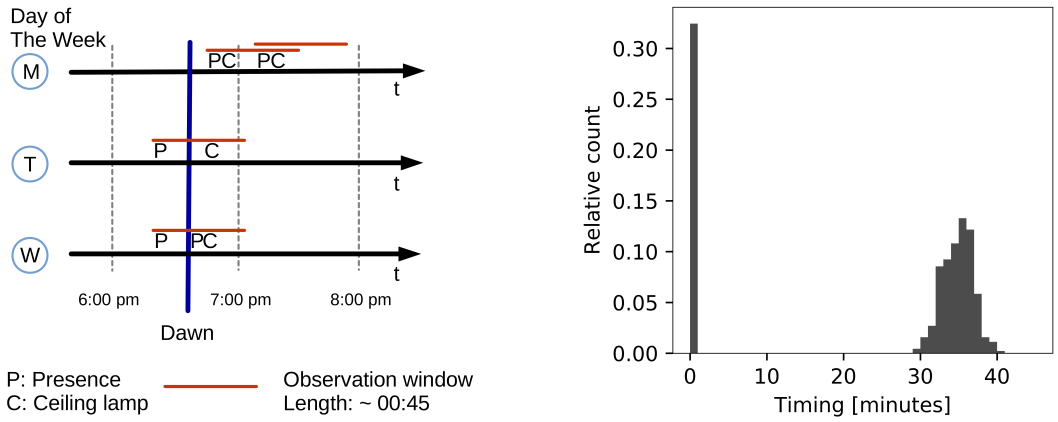
Once the entire time-series has been processed, filters are applied to further remove rules. We propose three filters: disjunction, belief , and rule timing filters. The *disjunctive filter* tests if the disjunction probability is less or equal than the confidence of the rule and the uniform symbol distribution. The filter's passing criteria is defined in Eq. 7.7, where $|V|$ is the size of the vocabulary, i.e., all distinct symbols in the time-series. By design, the disjunctive filter targets rules which associate common and rare symbols, since a rare premise should have a rare consequence.

$$(P(a \cup b) <= P(b|a)) \wedge (P(a \cup b) <= \frac{1}{|V|}) \tag{7.7}$$

The *belief filter* rejects any rule where the rule's confidence is greater than the belief in the rule. The passing criteria is shown in Eq. 7.8. A typical case of a rule that has a lower belief than confidence is a rule that has as conclusion a symbol b which has been observed at least twice as many times as the rule. In this case, the confidence will be larger than the belief because the conditional probability $P(a|b)$ is already below 0.5 (see Prop. 3). Typically, additional observations of the rule will remove it. However, if there is no more data, the rule remains as a candidate and is removed by the belief filter.

$$B(b|a) >= P(b|a) \tag{7.8}$$

The *timing filter* tests that the probability of the rule and its timing's expected value is greater than the uniformly distributed timing. Eq. 7.9 shows the filter's passing criteria, where μ_t is the expected value of rule r's timing, h is the bin size for sampled time-series and $|ow|$ is the size of the observation window ow. Since the rule appearing and the timing of the rule are independent, $P(\mu_t \cap \mu_t)$ can be computed as the product of the probabilities as shown in Eq. 7.9. The timing filter targets rules which are the association of independent symbols. The timing of a rule is expected follow a modal distribution, e.g., normal, whereas independent symbols will create rules with uniformly distributed timing.



(a) (b)

Figure 7.1: Time-series example of three working days, Monday (M) through Wednesday (W). (a) Illustration of how a ceiling lamp, which reacts to presence and outdoor light, generates a multimodal timing relationship between presence (P) and ceiling light (C) events. (b) Illustration of the resulting multimodal histogram for the situations illustrated in (a) after several observations.

$$P(r \cap \mu_t) > P(U(\mu_t))$$
$$P(\mu_t)P(\mu_t) > \begin{cases} \frac{1}{|ow|} & \text{ed-sampled} \\ \frac{h}{|ow|} & \text{sampled} \end{cases} \tag{7.9}$$

BRE's works on the principle of increasing belief in a rule. Therefore, we analyse BRE under three conditions: (1) when a rule is rejected on the first observation, (2) when belief in a rule decreases, and (3) when belief in a rule saturates. The analysis' results are the following three properties: Property 1, absolute belief, states that when a rule has a belief of one, the belief will never change. Property 2, rejection boundary, defines that unless a rule has absolute belief because both the premise and the conclusion have not been seen before, the rule will not be accepted in the first observation. Finally, Property 3, diminishing belief, states that if the conditional probability $P(a|b) < 0.5$, the rule's belief will always diminish. The formalization and proof of BRE's properties are found in the appendix.

Dynamic histogram creation

BRE's timing filter uses the probability of the expected value of the timing distribution to differentiate random symbol associations from rules. Therefore, BRE requires a good estimation of the expected value and its associated probability. However, in sampled time-series, symbols appear at arbitrary times. In general, given a rule $a \rightarrow b$, considered correct, the rule timing distribution is typically normal distributed around a value μ_t, i.e., $N(\mu_t, \sigma^2)$. However, depending on the underlying event generation process, the distribution could also be multimodal or become skewed. For example, suppose a scenario where a ceiling lamp responds to presence and outdoor light. Thus, the timing between ceiling lamp and presence events will have a multimodal distribution. Fig. 7.1 illustrates three cases which can cause the scenario's multimodal distribution and the resulting histogram after multiple observations. In Fig. 7.1(a) an $|ow|$ of 45 minutes is used to illustrate the presence and ceiling lamp activations of three days of the week. Monday's (M) time-series illustrates how the first mode of the distribution is created: after dawn, the ceiling lamp responds immediately, e.g., $\mu_t < 1$s. Tuesday's (T) time series illustrates the case for the second modality: when a presence event occurs before dawn, and the ceiling lights react once the outdoor light is gone. Finally, Wednesday's (W) time-series illustrates a condition which reinforces both modes. The activation of the ceiling lamp happened within the 45 minute *ow* of the first presence event, thus, two rules are created, one for each presence event using the same ceiling lamp event as a conclusion.

The challenge of dynamically constructing a histogram is to determine an appropriate number of bins or a bin size. For normal distributions, there are two commonly used methods to estimate the number of bins for a histogram: Freedman–Diaconis' choice [32] and Sturges' formula [105]. However, neither method works well for multimodal distributions. Therefore, our proposed empirical methodology dynamically calculates the bin number and size and extends to multimodal and skewed distributions.

We expect that most of atomic rules will have a normally distributed timing. Therefore, we set the minimum number bins to 18, which is the result of adding one to the average between applying Sturges and the Freedman–Diaconis rules to a normal distribution with 1000 samples. The Freedman–Diaconis rule was approximated as shown in Eq. 7.10, where R is the data's range, IQR is the interquartile range, σ is the standard deviation, h is the bin size, and n_h is the number of bins in the histogram.

$$\begin{aligned} h &= 2\frac{IQR}{n^{1/3}} \approx 2\frac{1.349\sigma}{n^{1/3}} \\ n_h &= \left\lceil \frac{R}{h} \right\rceil \approx \left\lceil \frac{6\sigma}{h} \right\rceil = \frac{3 \cdot n^{1/3}}{1.349} \end{aligned} \tag{7.10}$$

The next step in our method is to maintain the number of bins in the regions where samples cluster. The histogram is created progressively, and the bins are evenly distributed between the maximum and minimum of the samples. When a new sample falls outside the current range, two actions can be taken: (1) if the new sample is less than a bin width away from the closest range edge, use the sample as new boundary and redistribute all bins evenly between boundaries; otherwise, (2) increase the bin number until the new sample is covered. By increasing the number of bins, our method is robust to outliers and maintains a high number of bins in the regions where samples cluster. As a result, the error of the expected value is reduced. Fig. 7.2 illustrates the benefits of our proposed methodology when compared to Sturges and Freedman–Diaconis. The data distribution in Fig. 7.2 is the same as in the multimodal example in Fig. 7.1(b).

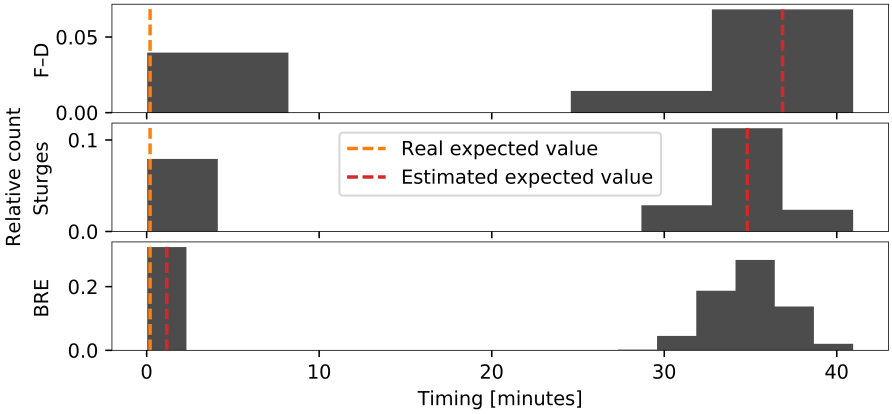


Figure 7.2: Resulting histograms by applying Freedman–Diaconis (F-D), Sturges, and Bayesian Rule Extractor (BRE)'s histogram creation method. Data distribution is the same as in Fig. 7.1(b). BRE's method for binning copes with multimodal distributions and produces an expected value which is closer to the real one.

7.4 Evaluation Methodology

The Bayesian Rule Extractor (BRE) algorithm is evaluated in three different scenarios: (1) a synthetic time-series, (2) a building's Building Management System (BMS) data stream, and (3) rehabilitation users' routine diary. The synthetic time-series is designed to measure and verify the rule extraction capabilities under controlled circumstances and quantify the limitations of the algorithm.

For the BMS data stream, we used the GreenerBuidlings 's dataset introduced by Lopera et al. [62, 63]. The dataset is a real world example of building management system data. We test 14 months of data, where variables are added and control rules are constantly updated. The goal of the test is to evaluate BRE's performance on a real world, causal system, with multiple underlying generator processes.

The final test uses a diary collected in a rehabilitation study conducted by Seiter et al. [102, 103]. The diary has two time-series: one has labels for activities of daily living, e.g., making coffee or talking, and the second time-series has labels of user's routines, e.g., socialising. Routine labels group together activities, e.g., the socialising routine groups making coffee and talking activities. In the second test, we mine the activity time-series and extract rules which are mapped into routines. The test uses a continuous time-series and provides a scenario where the order of the symbols is not always maintained, e.g., for a given routine, activities a and b appear as $a \rightarrow b$ or $b \rightarrow a$. Furthermore, the symbols are shared across routines.

Synthetic Time-series Evaluation

A synthetic-time series provides a controlled situation where the algorithm's strengths and limitations can be evaluated. Therefore, we analysed BRE's behaviour when extracting one rule from a noise background. For simplicity, we use a numeric vocabulary and the rule is constructed by a rule generator process using symbols 10 and 11, e.g., $10 \rightarrow 11$. To emulate real data conditions, each rule was created with timing sampled from a uniform distribution in the range $[1, \ldots, 6]$. Background noise is generated by the random-generator process by uniformly sampling the following vocabulary: $V_u = 0, 1, 2, 3$. There are two parameters used to generate the synthetic time-series: number of samples n_s, and rule instances n_r. The synthetic time-series is constructed by combining the rule and random generator processes in four steps: (1) for every rule instance n_r randomly choose a timing between one and six (t_{max}) samples, and fill the gap with symbols from the random generator. (2) Randomly select positions p_r for the rules, such that $p_r < n_s$, also ensuring that rule instances do not overlap. (3) Build the time series by filling spaces between rules with symbols from the random process generator. (4) Finally, pad the beginning and the end of the synthetic time-series until the number of samples n_s is reached. Eq. 7.11 illustrates a synthetic time-series example segment.

$$[\ldots, 10, 4, 1, 0, 2, 11, 0, 1, 4, 3, 10, 11, \ldots] \tag{7.11}$$

We designed four tests to evaluate BRE's performance using the synthetic time-series. The first test (1) is the filter test, using 1000 random samples, i.e., $n_s = 1000$ and 20 rule instances, i.e., $n_r = 20$, we tested all possible combinations of applying each of the three filters: disjunctive, timing, and belief. 200 Synthetic time-series were generated for each BRE operation mode. We used $|ow| = 6$ to match the rule's maximum timing. The best filter combination was used for the rest of the tests. The second test (2) is a sweep of the observation window size $|ow|$, from 2 to 100 symbols. A discrete time-series was generated with 20 rule instance, i.e, $n_r = 20$, and 1000 symbols sampled from the random generator process. The test was repeated 100 times with a new synthetic time-series each time. We use the results from the $|ow|$ sweep test to choose an $|ow|$ for the remaining evaluation tests. The third test (3) measures BRE's rule extraction performance by progressively adding rules to a fixed size time-series. We started with one rule and finished with 500 rules with steps of size one; each step was repeated 100 times. For each step, we used the random generator process to create a discrete time-series 500 samples long. The fourth and final test (4) measures the performance impact of inserting additional premises or conclusions. The time-series was generated using 20 rule instances, i.e., $n_r = 20$, and 1000 symbols sampled from the random generator process. Additionally, we progressively inserted one rule symbol up to 250 additional symbols. The test was repeated 100 times for premise and 100 times for conclusion and insertions.

Metrics:

To evaluate each test we used the following metrics: recall, insertions, and error rate. Each metric is defined to measure different aspects of the rule extraction process. Recall measures if the algorithm is able to extract the rule. Insertions measures how many rules were extracted that use the rule generator's vocabulary V_r but are in the wrong order. Finally, errors account for rules which contain symbols from the random generator process' vocabulary V_u. For completeness, we present the general form of the metrics under the assumption that the rule is a concatenation of all symbols of the rule's vocabulary and is separated in pairwise atomic rules. For example, if the vocabulary is $V_r = 10, 11, 12, 13$, then, the rule is $r = 10 \rightarrow 11 \rightarrow 12 \rightarrow 13$ and the correct atomic rules are: $10 \rightarrow 11$, $11 \rightarrow 12$, $12 \rightarrow 13$.

Formally, **recall** is defined in Eq. 7.12 as the ratio of extracted rules over the total number of atomic rules. Where $\#e$ is the count of correctly extracted rules and $|V_r|$ is the size of the rule generator's vocabulary. We tested with only one rule, e.g.,$10 \rightarrow 11$, thus, $|V_r| = 2$ and $\#e$ indicates if the rule was extracted or not.

$$\text{Recall} = \frac{\#e}{|V_r| - 1} = \#e \tag{7.12}$$

An **insertion** is formally defined as a rule which uses the rule's vocabulary but are not in the expected order, e.g., $10 \rightarrow 10$ or $11 \rightarrow 10$. We measure insertions rate as a ratio between the count of extracted insertion rules $\#e_i$ over the total number of possible insertions as shown in Eq. 7.13, where $|V_r|$ is the size of the vocabulary used to build the rules.

$$\text{Insertions} = \frac{\#e_i}{|V_r|^2 - |V_r| + 1} = \frac{\#e_i}{3} \tag{7.13}$$

Finally, an **error** is defined as a rule which contains symbols from the random process' vocabulary, e.g., $1 \rightarrow 2$ or $11 \rightarrow 2$. The error rate is calculated as shown in Eq. 7.14, where $|V_r|$ and $|V_u|$ are the respective

lengths of the rule's and random generator processes vocabularies; $\#e_e$ is the number of extracted rules which match the error criteria.

$$\text{Error} = \frac{\#e_e}{|V_u|^2 + 2 \cdot |V_r| \cdot |V_u|} = \frac{\#e_e}{32} \tag{7.14}$$

GreenerBuidlings 's Dataset Evaluation

A smart building is a good example of a causal system that can be modelled using rules. The basic principle of the BMS is that given a change in environmental conditions or user activity, the building reacts according to the control strategy, which creates a cause and effect association. A common control strategy in smart buildings is to use prepositional rules to link context information to actuator status, e.g., presence (cause) $\rightarrow$ ceiling light= On (effect). Heating, Ventilation, and Air-Conditioning (HVAC) useregulators, e.g., PID controllers, to maintain the temperature at a constant level. In an HVAC, a reference adjustment, e.g., increase temperature, will imply a change on energy consumption and room temperature, similarly, a change in the external conditions, will trigger a reaction from the controller. BRE can directly extract prepositional logic rules, and for regulator type controllers, BRE extracts the relation between changes in references, energy consumption and environmental states.

To build the continuous time-series events are extracted from each GreenerBuidlings variable stream. An event is composed of a timestamp and a symbol. The timestamp is when a variable experiences a positive change in its value, and as a symbol, the name of the variable is used. The final continuous time-series is created by sorting all events by their timestamps. We use the same process as Lopera et al [62, 63]. Eq. 7.15 shows a rule example that BRE will extract from the BMS time-series.

$$presence\_r1 \rightarrow ceiling\_light\_r1 \tag{7.15}$$

The dataset was recorded over 14 months, with a total of 277 variables which were added over time. The dataset covers a four people office room (14 months), a $\sim$ 20 people meeting room (13 months), and 12 people open office space (3 months). Environmental sensors, such as CO_2, temperature and humidity, as well as context variables and actuators are used in the dataset. The effects of system failures, software and hardware changes, are included in the dataset, as we believe they are part of the life-cycle of the building.

To evaluate BRE's rule extraction performance in the GreenerBuidlings dataset two test where performed. The first test is an observation window size $|ow|$ sweep. Due to the size of the dataset and the processing times, only the following observation window sizes $|ow|$ were tested: (in seconds) $[1, 5, 10, 120, 600]$. For the second test all possible filter combinations were tested. The best value for ow obtained in the first test was used in the second test.

Metrics:

To measure performance we use the same hierarchical tree baseline used by Lopera et al. [62] with the addition of the edges created by the façade hierarchy, which consists on the relationship of the outdoor and indoor light sensors regardless of the room where they are installed. A rule is considered correct and is assigned a grade of 1.0 when one of the following cases occurs: (1) the premise and the conclusion form an edge on the tree E, (2) the premise and the conclusion are in the same branch Br, i.e., one is an ancestor of the other. Additionally, a rule which links variables in the same sub-tree S, i.e., in the same room, but does not fall in one of the previous cases, gets assigned a grade G_r as defined in Eq. 7.16, where w is the number of edges between the premise, the nearest ancestor, and the conclusion, w_m is the number of edges between the premise, the root of the sub-tree, and the conclusion. A grade of 0.0 is assigned to all other cases.

$$G_r(a \rightarrow b) = \begin{cases} 1. & a, b \in E \vee a, b \in Br \\ 0.5 + 0.5\left(1 - \frac{w}{w_m}\right) & a, b \in S \\ 0. & \text{Otherwise} \end{cases} \tag{7.16}$$

Accuracy is defined as the average grade of all extracted rules. The grading scheme is introduced to unify all the metrics proposed by Lopera et al. [62]. We also measure **coverage** as the ratio between the variables used in the rules, and the total number of variables available in the test period.

Activity Diary Dataset Evaluation

Routines are defined as periods in time where a set of activities are performed. A routine instance is characterized by a subset of the activities which belong to the routine. For example, "gym workout" is a routine

Routines	Num. Activites
Balance	4
Basic motor training	50
Hygiene	17
Intense training	37
Kitchen work	19
Lunch	15
Mental training	12
Music	4
Other	11
Rest/sleep	4
Socialising	30
Table games	20
Using hand-tools	4

Table 7.1: Number of activities per routine. Although some activities are shared across multiple routines, each routine has a unique set of activities.

made of activities such as rest, stretch, cardio and weight lifting, and a "gym workout" instance could be only stretch and weight lifting. Activities can repeat during a routine instance and their order can change across multiple routine instances. Additionally, different routines can share activities, e.g., resting is also part of the "outdoor exercising" routine. Non-routine activities are those which do not belong to any routine.

A continuous time-series is constructed from the routines and activity diary created from patient schedules and by manually annotating the activities. The annotation were made on a best-effort-bases as researchers followed patients during their rehabilitation sessions in the clinic. The diary is part of the study conducted by Seiter et al. [102, 103]. From the diary, the activity starting time and label are used as timestamp and symbol in the continuous time-series. The diaries from 11 patients were used. Each diary contained between six and eight days of recordings and each day lasted about eight hours. A time-series was built for each patient. The base line is created by assigning an activity set to each routine. The set is built by extracting the activities from each routine instance for all users. A total of 13 routines and 57 distinct activities were used. The size of each routine set is shown in Tab 7.1.

Two possible models which will produce a time-series that reflects the patient's session in the rehabilitation clinic are the Markov models as proposed by Chung et al. [15] and the logic rule model as proposed by Stier et al. [103]. BRE extracts the atomic transitions between activities, which are the building blocks of Hidden Markov Model (HMM). Additionally, atomic rules can be interpreted to construct the logic equations used in the rule model. However, converting from atomic rules to HMMs or logic equations is beyond the scope of this work. Eq. 7.17 illustrates a rule example that BRE will extract from the activity diary time-series. The rule is found in the Lunch routine.

$$eating \rightarrow drinking \tag{7.17}$$

To evaluate BRE's rule extraction performance in the activity diary time-series two test where performed. The first test is an observation window size $|ow|$ sweep. The size $|ow|$ was varied starting with 10 minutes until two hours in 10 minutes intervals. After two hours, $|ow|$ was increased by an hour until 16 hours was reached. The test was applied to both BRE's running modes. The best values for ow were selected and used in the following test. Finally, all possible filter combinations were tested. For both tests BRE and the respective performance metrics were calculated per user and the results averaged.

Metrics

To measure performance we use three metrics: accuracy, insertion rate, and coverage rate. A rule is considered correct when both the premise and the conclusion belong to at least one routine. **Accuracy** is defined as the rate of correct rules over the total number of retrieved rules. An **insertion** occurs when the rule's activities, i.e., symbols, belong to different routines, or one of them is a non-routine activity. Thus, the insertion rate is the number of insertions over the total number of extracted rules. The **coverage** rate measures how many routine activities were correctly associated in rules, following the accuracy criteria, with respect to the total number of routine activities available on each patient's time-series.

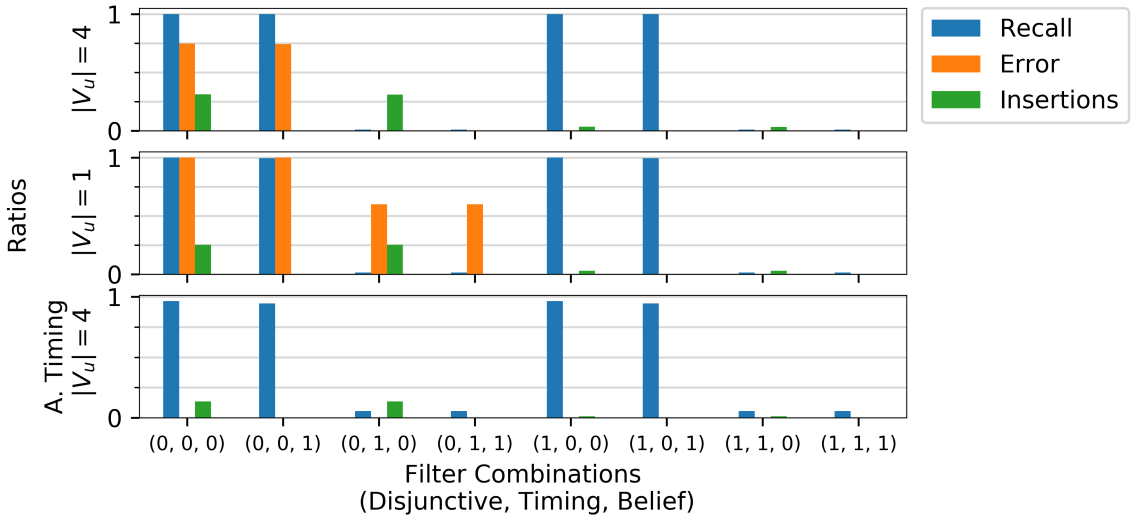


Figure 7.3: Results of applying different filter combinations to the synthetic time-series. The results are the average of 200 simulations of 20 rule instance inserted to a ed-sampled time-series of 1000 symbols. For the synthetic time-series, errors and insertions are removed by the disjunction and belief filters respectively. The timing has an effect on the errors, but it has the draw back that it requires a larger minimum $|ow|$. Based on the other test presented in this chapter, the best results are found with the alternative timing mode and the belief and disjunction filters.

7.5 Results

We present the result section in three parts: first, we present the analysis of Bayesian Rule Extractor (BRE)'s performance when processing synthetic time-series; then, we present the test results of the GREENER-BUIDLINGS dataset, and finally, we present the BRE's evaluation when using the activity diary dataset.

Synthetic Time-series Evaluation

In the synthetic time-series evaluation, we first examined the effect of the different filter combinations. We chose $|ow| = 6$ to match the maximum rule timing from the rule generating process of the synthetic time-series. We tested the following three filter configurations: disjunction, timer, and belief. Fig. 7.3 illustrates the filter configuration results, the tuples in the x axes indicate whether the filter was applied or not, e.g., (0, 1, 0) means that the timing filter is used and the disjunctive and belief filters were not applied. We found that, in normal mode, the rule was always extracted, i.e., recall= 1., when the disjunctive and belief filters were applied. The choice of $|ow| = 6$ caused the timing filter to mostly reject the rule because the generator process sampled from the range $[1, \ldots, 6]$ using a uniform distribution to create each rule instance. Thus, the rule's timing distribution becomes indistinguishable for the timing filter. When BRE runs in alternative timing mode, recall drops to 0.96 without filters and to 0.94 after applying the belief filter. The drop in recall is the result of the shrinking $|ow|$ which might cause the conclusion to be seen as a noise symbol, and the confidence to be higher than the belief depending on how the rules are distributed throughout the time-series.

The error results in Fig. 7.3 show that, when $|V_u| = 4$ and no filters are applied, the effect of Prop. 2 is observed for rules where the premise belongs to V_r and the conclusion to V_u. Thus, only 75% of the possible errors are extracted. However, for $|V_u| = 1$, Prop. 2 has no effect and all possible errors are always extracted. To remove erroneous rules the disjunctive filter is used. The random generation process symbol's probability is ≈ 0.25 when $|V_u| = 4$, and ≈ 0.96 when $|V_u| = 1$. Therefore, the disjunction probability will be dominated by the random generation process symbols and the disjunctive filter will remove erroneous rules, because, when the premise or the conclusion belong to V_u, then, the disjunctive probability is higher than the uniform distribution of all symbols, e.g., $P(1 \cup 2) \approx 0.25 > 1/6$ or $P(0 \cup 0) \approx 0.96 > 1/3$, and thus the erroneous rule is rejected. Furthermore, rules which include symbols from the rule generator's vocabulary V_r are rejected because the confidence in the rule is less than the disjunction probability. Errors are removed by the timing filter due to the uniform distribution of all erroneous rules. However, when $|V_u| = 1$, the timing distribution of rules starting with V_r symbols, e.g., $10 \rightarrow 0$, and the rule $0 \rightarrow 0$ becomes skewed with a timing expected value of 1 sample. Hence, the errors are 0.6 when the timing filter is applied and $|V_u| = 1$. In alternative timing mode, no filters are required because the shrinking $|ow|$ causes the erroneous rules to have a conditional probability $P(a/b) < 0.5$, as there will be more symbols than rule observations. Therefore, using Prop. 3,

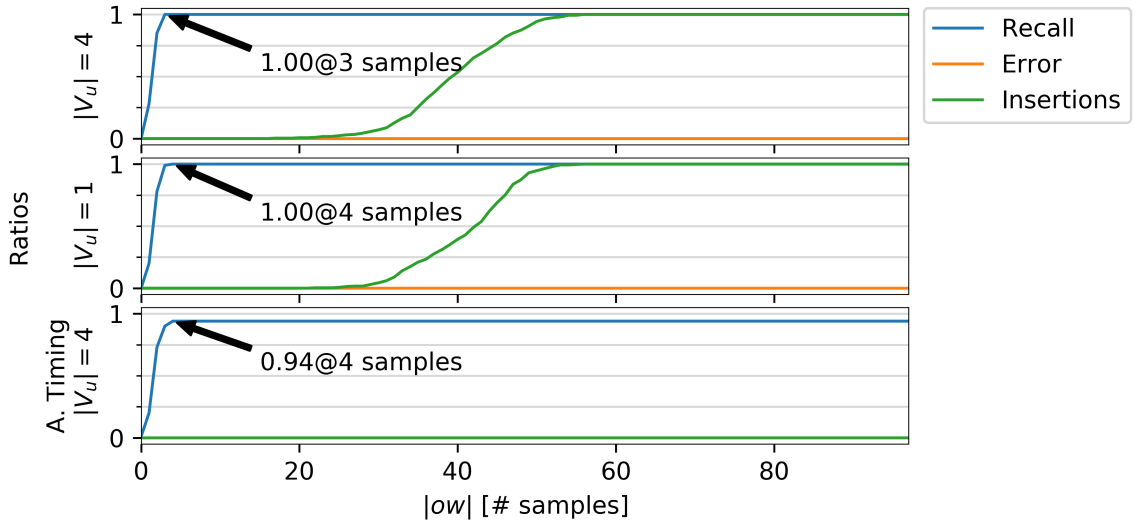


Figure 7.4: Observation window size $|ow|$ sweep for 20 rule instances inserted into a time-series made of 1000 randomly generated samples. Rule instances were created with variable rule timing sampled from the range $[1, 6]$ symbols. Bayesian Rule Extractor (BRE) always finds the rule once $|ow|$ reaches at least the average rule size. BRE's behaviour is unaltered by the size of the random generator process vocabulary $|V_u|$. Since BRE's alternative timing mode chooses the smallest distance between rule symbols as $|ow|$, in some instances of the synthetic time-series, a rule will have a timing μ_t smaller than the average $|ow|$ causing the rule to not be retrieved, thus, reaching an average recall of 0.94.

future rule observations will reduce belief and thus the rule will be rejected.

Insertion results in Fig. 7.3 illustrate that insertions are extracted, because, for an insertion rule to form, two rule instances have to be within $|ow|$ symbols of each other. Since rule's positions are uniformly distributed throughout the time-series, insertion rules appear, but there are not many samples of the rule. Therefore, the rule timing distribution for insertions will look skewed and the timing filter will not have any effect on the insertions metric. The V_u size has no impact on the algorithm's insertion handling behaviour. In general, BRE rejects insertions mostly without filters. Additionally, BRE's alternative timing mode removes all insertion rules which have the same symbol as premise and conclusion. However, the insertion rules which do pass BRE's belief criteria or alternative-running mode must have multiple observations, otherwise, Prop. 2 will reject them. Furthermore, the insertion rules have to be observed an odd number of times because, based on Prop. 3, a subsequent observation will yield a lower belief value and then the insertion rule is rejected. Finally, an insertion rule is accepted when it triggers Prop. 1, which happens when the first two rules are within the ow for the first rule's premise. However, due to the *related rule update* step, all insertion rules will be updated to have a belief based on the rule's confidence, which will follow Prop. 3, thus the belief filter will reject all insertion rules. According to Fig. 7.3, the disjunctive filter removes most of the remaining insertion rules. Those insertions which remain are generated towards the end of the time series where the insertion's disjunctive probability is less than 1/6. Based on the results of the filter combination test we chose the disjunctive and belief filters for all other tests of the synthetic time series.

Once an appropriate filter configuration was selected, we proceeded to evaluate the effect of the observation window size $|ow|$ on BRE's performance. Fig. 7.4 shows that an arbitrary large observation window size$|ow|$ does not affect the extraction of true rules. The disjunctive filter eliminates all error instances independently of $|ow|$ and random generator process vocabulary $|V_u|$. However, as $|ow|$ grows, insertions become possible. Specifically, once $|ow|$ reaches the expected distance between rules ($\approx\sim$ $1000/20 = 50$ samples), there is a high probability of finding two or more rules in the same observation window ow. Thus, rules and insertions will be similarly frequent and, as a result, BRE will extract insertion rules as the belief filter will stop working because their belief will be higher than their confidence. BRE requires $|ow|$ to be at least as large as the expected distance between the premise and the conclusion in order for BRE to always extract the rule. Additionally, BRE's alternative timing mode and the belief filter remove all insertions regardless of $|ow|$ since BRE's alternative mode automatically adjusts $|ow|$ for each symbol, which limits the possible insertions. A limitation of BRE's alternative timing mode is that it uses as $|ow|$ the minimum rule timing from a rule with the same symbol for premise and conclusion, e.g., $10 \rightarrow 10$. In the case of the synthetic time-series, the rule timing will drop bellow the expected value $|ow| = 3$ occasionally preventing the rule from being extracted. Consequently, BRE's alternative timing mode only reaches 0.94 recall.

The absolute belief property (Prop. 1) implies that BRE is selective of rare rules, i.e., only one observation

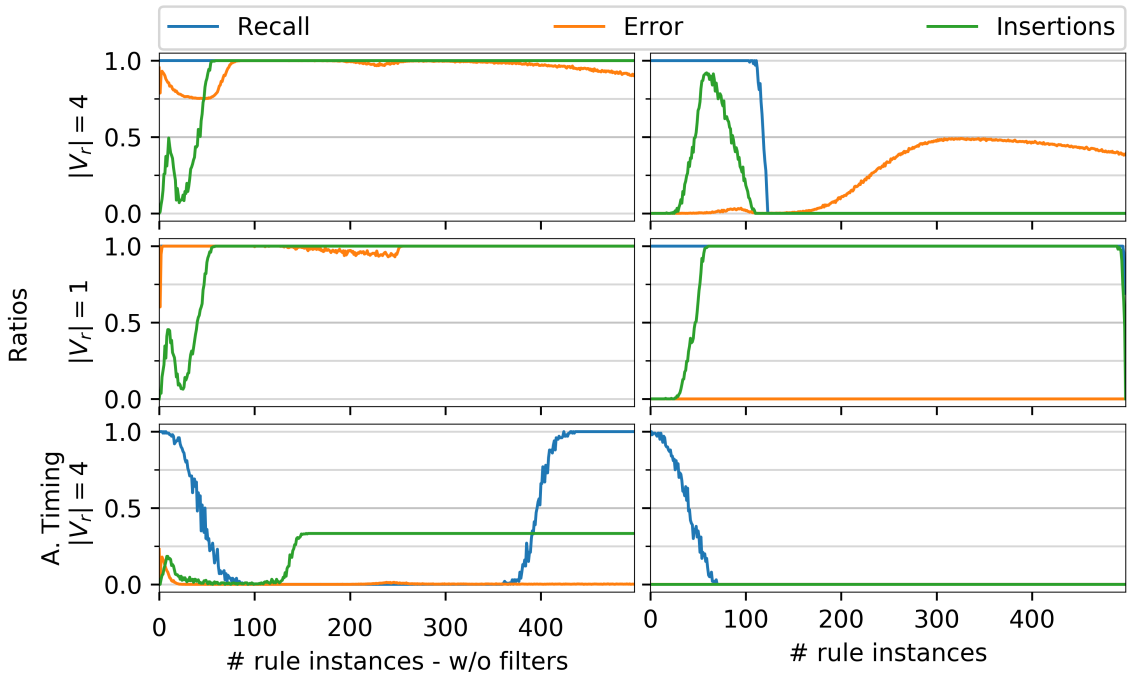


Figure 7.5: Performance results of adding rule instances to a 500 random samples in size, randomly generated, ed-sampled time-series. Bayesian Rule Extractor (BRE)'s normal mode retrieves the rule until the frequency of the rule and the random process symbols are the same. After 167 rule instances, the error rules start to appear. Insertions are expected after 16 rule instances.

is needed of previously unseen symbols within ow and the rule is accepted. We test BRE's selectivity by progressively adding rule instances to a time-series with 500 symbols created using the random-generation process. Fig. 7.5 illustrates the test results under two conditions: with and without filters. BRE's normal mode achieves perfect recall when no filters are used. However, when the disjunctive and belief filters are applied, the disjunctive filter discards the rule when the relative frequency of the rule's symbols is the same as the random-generator process. For $|V_u = 4|$ and $|V_u = 1|$, the rule is discarded at approximately 125 and 500 rule injections respectively. The rejection boundary corresponds to $1/|V_u|$. In BRE's alternative mode, as more rules are added, the distance between rules is reduced and BRE stops accepting the rule, as not all instances are seen due to the small $|ow|$. Additionally, a side effect of how the time-series is generated, after several rules are added, there are not enough symbols from the random-generator process to pad all rules, thus the rules appear back-to-back and with a timing of one sample. As a result, BRE's alternative timing mode, without filters, accepts the rule again. However, when the filters are applied, the disjunctive filter discards the rule after 100 samples because the relative frequencies of the symbols match. In addition, the belief filter speeds up recall descent rate because, with fewer observations, it is more likely that the confidence is higher than the belief as a side effect of diminishing belief property (Prop. 3).

Fig. 7.5 illustrates how, without filters, insertions are independent of $|V_u|$. As more rules are added to the time series, the likelihood of observing insertions increases. Initially, it is unlikely that the number of observed insertions triggers Prop. 3. However, after 10 rules, the likelihood increases also for multiple insertion observations. Thus, Prop. 3 removes the insertions between 10 and 20 rule instances. Finally, as the number of rules increases, the rules appear closer in the time-series and insertions are observed frequently enough to be considered a valid rule. When Filter are applied, Fig. 7.5 illustrates how the first two insertions trends are removed by the belief filter. However, the increasing number of insertion observations causes the belief to be higher than its confidence and the insertion remains. Additionally, the disjunctive filter also removes insertions once the relative frequencies of the insertion symbols overcome the insertion's confidence, which for $|V_u| = 1$ happens just before the rule is discarded. With BRE's alternative timing mode only one type of insertion is possible, e.g., $11 \rightarrow 10$. As a result, without filters, the initial insertion trend only reaches a third of the original value. The dynamic $|ow|$ also eliminates insertion observations, which delays the point where insertions are accepted.

BRE's error behaviour while inserting rules is illustrated in Fig. 7.5. When $|V_u| = 1$, the disjunction probability will always be higher than $1/|V| = 1/3$ for the evaluated range, e.g., 500 rules and zeros. Thus, the disjunctive filter removes all errors. Similarly, BRE's alternative timing mode, in combination with the

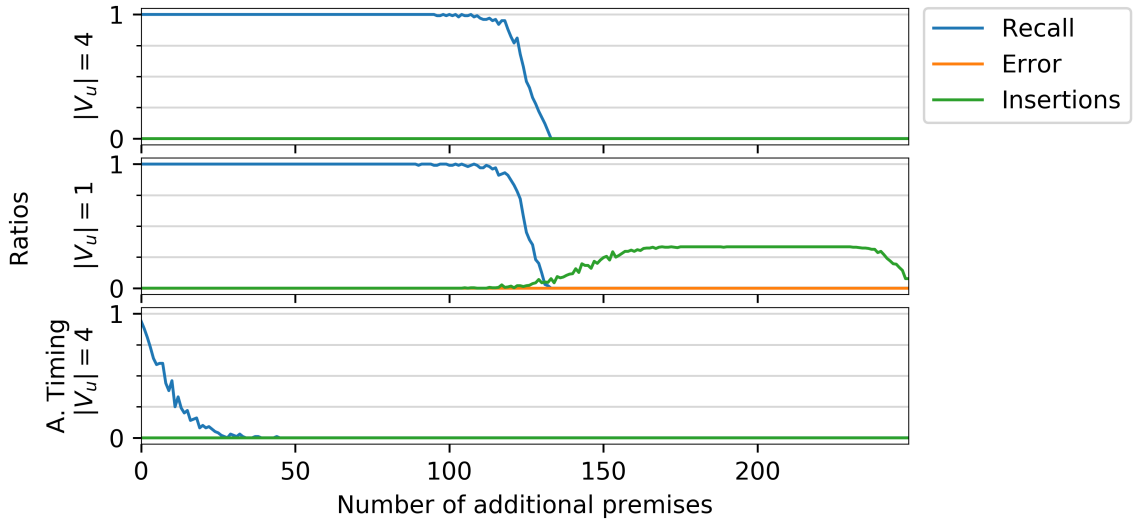


Figure 7.6: Performance results of progressively adding premises into a ed-sampled time-series of 1000 randomly generated symbols with 20 rule instances. The disjunctive filter causes the rule to be discarded after 132 additional premise. As more premises are added, insertions are expected to appear because the rule $premise \rightarrow premise$ becomes frequent enough to be considered a real rule. Eventually the premise becomes so frequent, that is discarded by the disjunctive filter. In Bayesian Rule Extractor (BRE)'s alternative timing mode, additional premises reduce $|ow|$ and rule instances are missed.

filters, also rejects all errors because the confidence on the error is less than the disjunction probability. The error rejection is the result of fewer observations caused by the small dynamic $|ow|$. For $|V_u| = 4$ with filters, when the relative symbol frequencies from the random process and the rule match, not only the rule is rejected but also errors start top appear. As more rules are added, the relation between rule symbols and background noise is inverted and the disjunctive filter starts accepting errors. The error rate reaches a maximum of 0.5, which are all the errors which involve a rule's symbol. Furthermore, After approximately 300 rules the error starts to decrease, as a result of BRE's Prop 3 affecting rules with a rule's symbol as conclusion. The overall error rate change is driven by the small irregularities in the random generator symbol distribution, otherwise a sharper reduction would be expected as seen with the recall change. Finally, BRE's effectiveness depends on the relative frequencies between the two generating processes. Furthermore, BRE's alternative timing mode is able to remove errors under most conditions, except when there are only a few rules, where the likelihood of seeing the error is low, and BRE's belief criteria does not apply.

In the following test, we examined the behaviour when premises and conclusion were progressively added to the generated time-series. Under the test conditions and in combination with the disjunctive and belief filters, BRE is robust against errors in the tested range, as shown in Figs. 7.6 and 7.7. According to previous results, the test range, i.e., 250 additional rule symbols, is close to the number of symbols which are required to match the time-series uniform symbol probability $1/|V|$, i.e., 215 symbols. Thus, the disjunctive filter removes all error rules.

BRE's performance is shown in Fig. 7.6 while premises are progressively added. The number of rules and conclusions is the same. Therefore, the rule's belief is one following Prop. 1, but the confidence depends on the number of added premises. As a result, BRE's normal mode extracts the rule until the disjunctive filter removes the rule because the premise relative rate, which drives the disjunction probability, surpasses the time-series' uniform symbol probability $1/|V|$. With additional premises and BRE's normal mode, recall is independent of the random generator process' vocabulary size, as the rule is rejected at the same point for both $|V_u|$ values. BRE's alternative timing mode is sensible to added premises because they can artificially shorten $|ow|$. As a result, BRE will miss most rule instances. The disjunctive filter is not affected by the inserted premises in the tested range. Therefore, no errors are seen. However, adding more premises, e.g., more than 250, will cause the error rules' disjunction probabilities to be less than their confidence and errors are expected to appear. Adding premises to the time-series only cause insertions to be retrieved when $|V_u| = 1$. Similar to the added rules test, after enough premises have been added, it is likely for the confidence of the insertion rule to be greater than the diminishing belief property threshold. However, the condition only applies to rules of the form $10 \rightarrow 10$, since rules involving the conclusion have their probabilities unchanged and will be removed independent of the number of additional premises. Thus, insertions only reaches 0.33. As more premises are added, the insertions' disjunction probability increases with every addition. Eventually, the insertion rule's

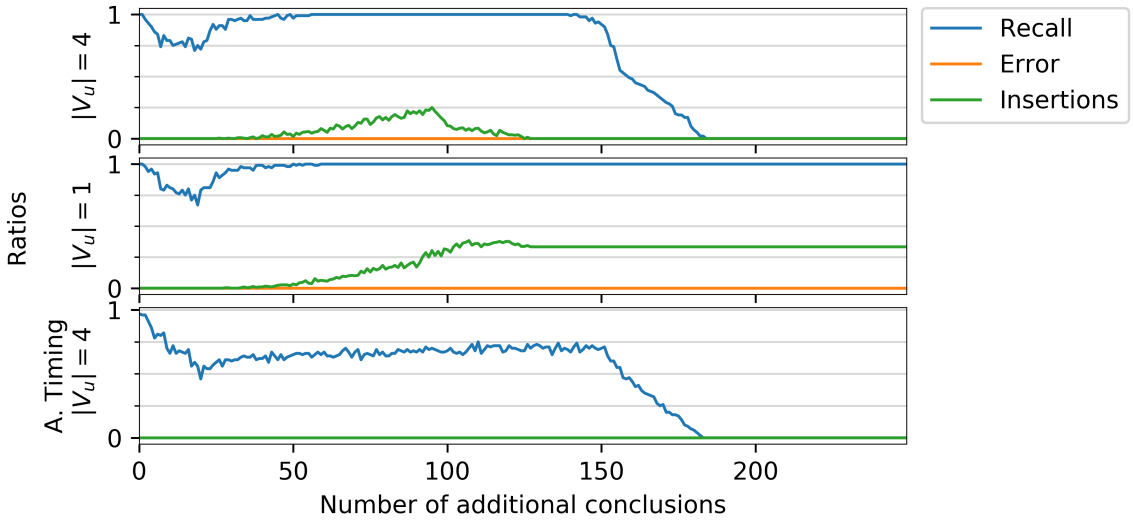


Figure 7.7: Performance results of progressively inserting conclusions into ed-sampled time-series of 1000 randomly generated symbols with 20 rule instances. When conclusions are added, their position has an impact on Bayesian Rule Extractor (BRE)'s performance. Specially when conclusions are observed before the rule. Although, BRE can tolerate more conclusion additions than premises, a small $|ow|$ does restricts sampling the rules, and thus the belief in the rule is lost.

confidence is less than the disjunction probability and the insertion rule is discarded by the disjunctive filter.

We also tested BRE's behaviour when adding conclusion symbols and the results are shown in Fig. 7.7. The confidence of the rule is determined by the number of premises. Therefore, the rule confidence remains unchanged despite the additional conclusions. Nevertheless, as conclusions are added, a conclusion can be located before the first rule instance, which prevents the rule's belief from saturating with the first observation (Prop. 1); thus, the rule is object of the rejection boundary property (Prop. 2) and after more instances the rule is accepted. Hence, the synthetic time-series generation can represent one of the following three cases: (1) the rule's belief increases, because the added conclusions do not trigger the diminishing belief property (Prop. 3); at the end, the belief filter removes the rule as the rule's confidence is one but the belief is not. (2) the rule is rejected as the conclusion to rate ratio satisfy Prop. 3; if the rule has an additional observation, the rule will have a confidence of one. BRE computes belief by using the rule's confidence as prior and, by Prop. 1, the rule will reach absolute belief and will not be discarded. Finally, (3) the rule might not be accepted at all if the rule to confidence ratio is below 0.5, i.e., $P(premise|conclusion) < 0.5$. In BRE's normal mode and regardless of random generator process' vocabulary size $|V_u|$, recall experiences a drop in performance at the beginning of the test due to cases (1) and (3). Then, recall recovers due to scenario (2). Finally, the rule is eliminated by the disjunctive filter as the conclusions relative rate approaches the uniform symbol probability $1/|V|$. BRE's alternative timing mode experiences the same problem with drops in performance as in the normal mode. However, recall drop is exacerbated because the dynamic $|ow|$ causes fewer rule instances' observations.

Adding conclusions to the time-series increase the chances for creating insertions. However, only two out of the three possible insertions are created: $11 \rightarrow 10$ and $11 \rightarrow 11$. As conclusions are added, the relative rates between the individual insertion rules and the conclusions surpasses 0.5 and belief in the insertion increases. Similar to the rule's recall behaviour, the disjunctive filter removes the insertions when their confidence is less than its disjunction probability. When $|V_u| = 1$, as confidences density increases, the insertion $11 \rightarrow 10$ is likely to be observed before the first rule instance. Therefore, the insertion will reach absolute belief in the first observation and will not be discarded during the evaluated range. Finally, the insertion $11 \rightarrow 11$ is removed by the disjunctive filters because its confidence becomes less than its disjunction probability. All insertions are removed by BRE's alternative timing mode in combination with the filters.

GreenerBuidlings 's Dataset Evaluation

The GreenerBuidlings dataset is created by multiple stochastic processes, i.e., the users, and deterministic control processes, i.e., the Building Management System (BMS) responses. Therefore, as $|ow|$ increases, the opportunity for creating connections between multiple generation processes increases. Hence, as evidenced in Fig. 7.8, accuracy decreases and coverage increases, as more symbols are associated in the larger ow. Fig. 7.8 also shows how $|ow|$ BRE's normal mode reflects the user's transition times between sensors, e.g.,

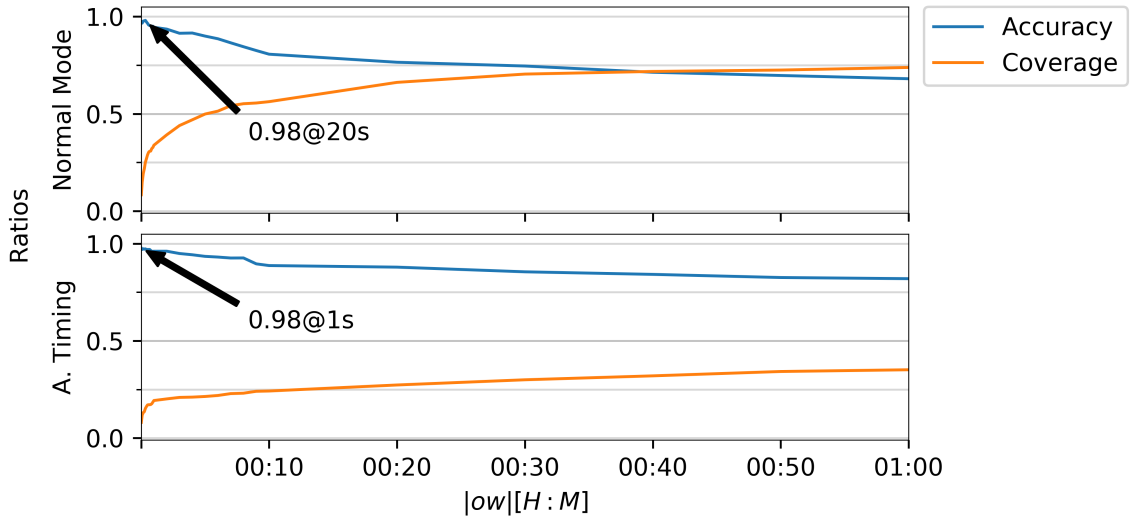


Figure 7.8: Observation window size $|ow|$ sweep for the GreenerBuidlings dataset evaluation. Results are the average performance of 14 month of recording. The 20 seconds $|ow|$ in Bayesian Rule Extractor (BRE)'s normal mode reflects the user transition times between sensors. In contrast, a one second $|ow|$ in BRE's alternative timing mode reflects the system response to sensor input.

entering room $\rightarrow$ *desk presence*. In contrast, $|ow|$ in BRE's alternative timing mode reflects the BMS response to sensor input, e.g., *desk presence* $\rightarrow$ *ceiling lamp*. However, the dynamic $|ow|$ causes accuracy to diminish at a slower rate. But, BRE's alternative timing mode restriction on rule observations causes coverage to grow at a much slower rate than in BRE's normal mode. As most of the expected connections are generated by user's transitions, the system response rules generate less coverage than the user driven rules. For example, at the best accuracy points of both BRE running modes, an $|ow|$ in the order of seconds does not benefit extraction of rules of air quality related variables, e.g., temperature, CO_2, since their response time is in the order of minutes. Additionally, the relationship of variables with external conditions, puts a strain on the opportunity to observe events. For example, when the ceiling lights depend on the exterior illumination level, user presence will only be associated with ceiling lights when the events occur after dawn.

The performance results for the different filter combinations test are shown in Fig. 7.9. We used the observation window size $|ow|$ computed in the $|ow|$ sweep test. Using BRE's normal mode, the highest coverage is reached without filters and with the timing filter. The achieved accuracy is of 52% and 50% respectively. The performance is due to the respective rarity of events such as changes in air quality variables and the correlation with other events, for instance, unrelated user movements. In general, the timing filter proved to have a negative effect on the accuracy because there were not enough samples of false rare rules to have an adequate comparison with a uniform timing distribution. Instead, the timing filter removed correct rules with an accuracy reduction in the ranging from 0.2 to 2 percentage points every time it was used. For the GreenerBuidlings dataset the belief filter is the most effective, improving coverage to 79% when used alone, to 99% when used in conjunction with the disjunctive filter. The increase in accuracy comes with the expected decrease in coverage. At the end, only 27% of variables are related when both filters are applied. The disjunctive filter removes rules where the confidence is less than disjunction probability, typically coming from the association of two frequent but unrelated processes, i.e., presence at different desks. Similarly, the belief filter removes rules which associate events that are frequent, but they are rarely observed together. Thus, the confidence is greater than the belief as a consequence of the diminishing belief property (Prop. 3).

In BRE's alternative timing mode, the timing filter has the same effect on rule removal as in BRE's normal mode. However, the individual application of the belief and disjunctive filters achieve a higher accuracy of 96% and 67% respectively. The 17 percentage points performance increase for the belief filter comes at the cost of 44 percentage points in coverage. Similarly, the disjunctive filter has a 25 percentage points drop in coverage. When combined, accuracy reaches 98% and coverage drops to 8%. BRE's alternative timing mode restricts the association of variables in a smaller $|ow|$. As a result, rule instance samples are discarded and coverage is affected. Specially, rules with multimodal timing distributions. For example, Fig. 7.1(a) illustrates a case where BRE's alternative timing mode will discard the sample $P \rightarrow C$ generated on Wednesday from the P and C events before and after dawn respectively; if the rule $P \rightarrow C$ is only generated around dawn, and there are many P samples throughout the day, it is likely the rule does not get enough samples to be accepted by the belief filter.

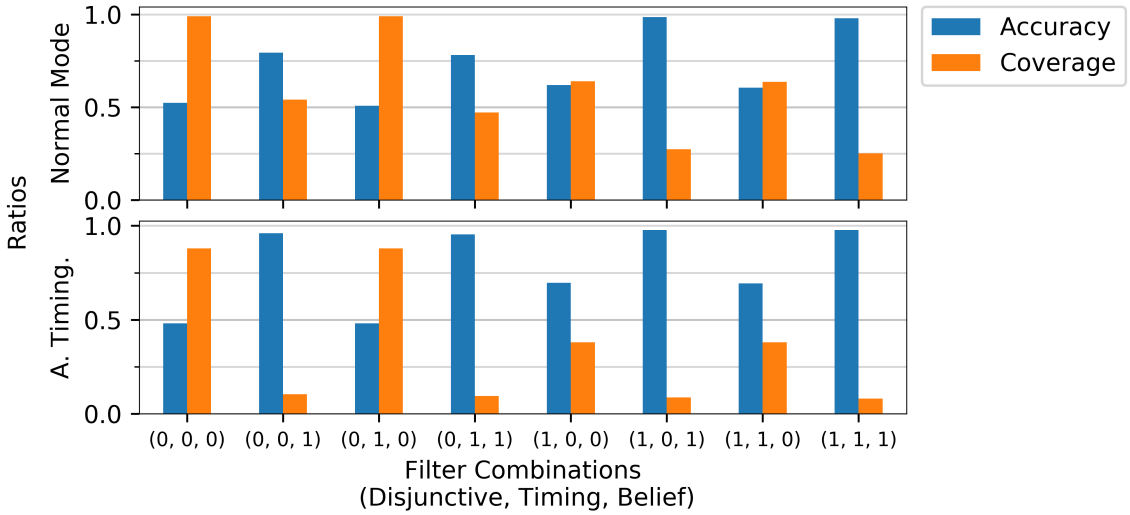


Figure 7.9: Performance results for different filter configuration for the GreenerBuidlings dataset. Results are the average performance of 14 month of recording. Due to the similarities in rule and symbol rate, as well as, differences between symbol frequencies, the combination of the belief and disjunction filter are the most effective. However, the high accuracy comes at the price of reduced coverage. Bayesian Rule Extractor (BRE)'s alternative timing mode rejects rule instance samples by reducing the observation window size $|ow|$. Thus, accuracy increases for the application of individual filters. However, when the filters are combined, accuracy does not reach the level of BRE's normal mode. Overall coverage decreases in BRE's alternative timing mode. Therefore, we found the best results with BRE's normal mode and the belief and disjunction filters because, based on possible applications of these results, accuracy is more important than coverage.

Activity Diary Dataset Evaluation

In addition to the synthetic time-series scenario, we also tested BRE's performance with two real-life datasets. The first test is the activity diary evaluation scenario, where patients change activities at irregular intervals, but follow a regular routine pattern. Fig. 7.10 shows the results for the $|ow|$ sweep. When BRE runs in normal mode and using a five hour $|ow|$, BRE builds up the belief in a rule by adding observations as the activities repeat during a day. For example, "talking to physiotherapist" (TP) and "standing to sit" (SS), are two activities which belong to intense motor training routine. The activity pattern example shown in Eq. 7.18 illustrates how additional rule instances are observed by linking the last TP event of the morning session with the first SS event in the afternoon. BRE's alternative timing mode has the best performance with an $|ow| = 1:00$ hour which represents the intuitive notion of the patient's schedule. In reality, most routines, e.g., focused training sessions, are scheduled for 30 to 45 minutes. Therefore, the one hour $|ow|$ has a 99% accuracy but fails to associate 3% of the activities due to routines which are usually booked for more than one hour, e.g., Lunch. Whereas in BRE's normal mode, all activities were associated at the cost of accuracy.

$$(9:00, \text{TP}), \ldots, (9:08, \text{SS}), \ldots, (9:45, \text{TP}), \ldots, (14:00, \text{TP}), \ldots, (14:10, \text{SS}), \ldots \tag{7.18}$$

Fig. 7.11 shows the effects of the filter configuration on the activity diary evaluation. In general, BRE's normal mode profits from the five hours $|ow|$ to extract enough independent rule instances of the form $a \to b$ and $b \to a$. Because we accept both directions of the rule as correct, it is the equivalent of considering rules and insertions in the synthetic time-series scenario, and we know that with enough instances and as long as the relative frequencies are similar, the belief is always higher than the confidence for the most frequent form of the rule. As a result, and without filters, there is a 100% coverage with 93% accuracy. The timing and belief filters have a marginal negative effect on accuracy of no more than 1 percentage point. However, when the disjunctive filter is included, accuracy drops below 90% reaching a minimum of 88.6% when the three filters are combined. Additionally, coverage drops to 80% with a minimum of 76%. The disjunctive filter is not suited to deal with the activity diary dataset, as the relative frequency of all symbols are similar. Therefore, the filter confuses some rules with noise, i.e., the disjunction probability does not reflect a characteristic of false rules in the activity diary scenario. In BRE's alternative timing mode, the smaller initial $|ow|$ of one hour causes BRE to extract fewer independent rule instances. Therefore, without filter, coverage drops to 97%. However, by focusing on the shorter routines, e.g., basic motor session, the extracted rules are more likely to belong to a single routine session and accuracy increases to 99%. Similar to BRE's normal mode, the belief

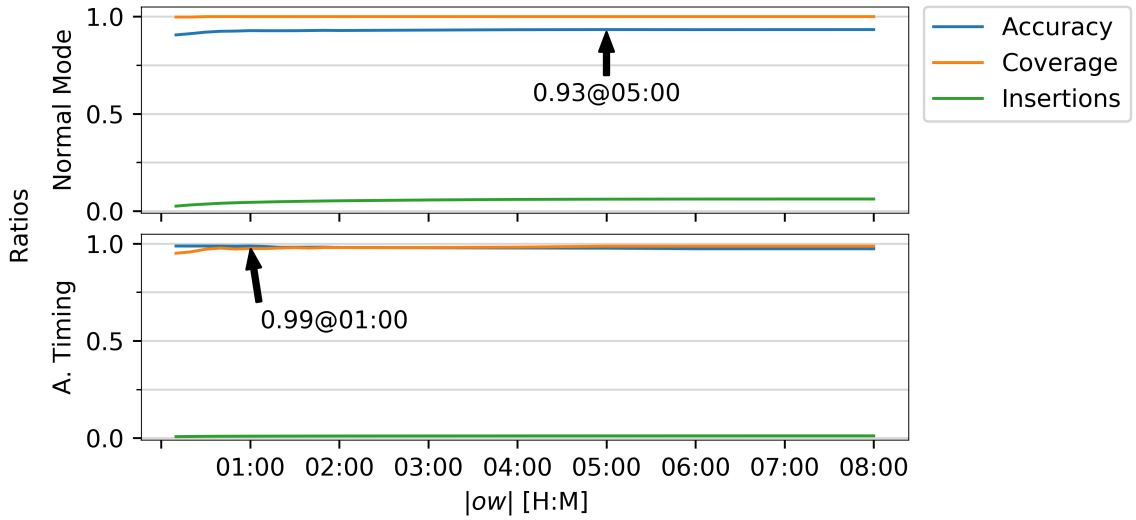


Figure 7.10: Observation window size $|ow|$ sweep for the activity diary evaluation. Results are the average performance of 11 patients. Bayesian Rule Extractor (BRE)'s normal mode maintains good coverage by building rule belief across multiple sessions. However, the large $|ow|$ has an impact on accuracy. BRE's alternative timing mode obtains better results with a slight drop in coverage. $|ow| = 1:00$ Hour reflects an intuitive expectation of how often people will change routines. However, there is a drop in coverage as the result of routines which take longer than one hour, e.g., two hour lunch break.

and disjunctive filter reduced accuracy by 4 percentage points. In contrast, when the disjunctive and belief filters are applied, the few observed instances, which lowers confidence, and the similar symbol's relative rates cause coverage to drop between 57% and 25%. The timing filter improves accuracy by a negligible margin of +0.05 percentage point, although, it also reduces coverage by 1 percentage point.

Insertions are caused by association of activities across multiple routines. The five hour $|ow|$ covers several sessions. Therefore, the likelihood of creating insertions increases. The filters' rejection criteria does not represent a distinguishing factor between rules and insertion. Therefore, insertions are only reduced by ≈ 0.5 percentage points when all filter are applied. BRE's alternative timing mode is a much more effective mode of dealing with insertions. The shorter $|ow|$ reduces the overlap between sessions. Additionally, the automatic $|ow|$ adjusting per symbol focuses on consecutive activities. Therefore, insertions' chances are significantly reduced. Without filters, BRE retrieves only 1% of insertions. The lowest insertion rate, i.e., 0.04%, is achieved when the disjunctive and belief filters are applied, at the cost of coverage and accuracy.

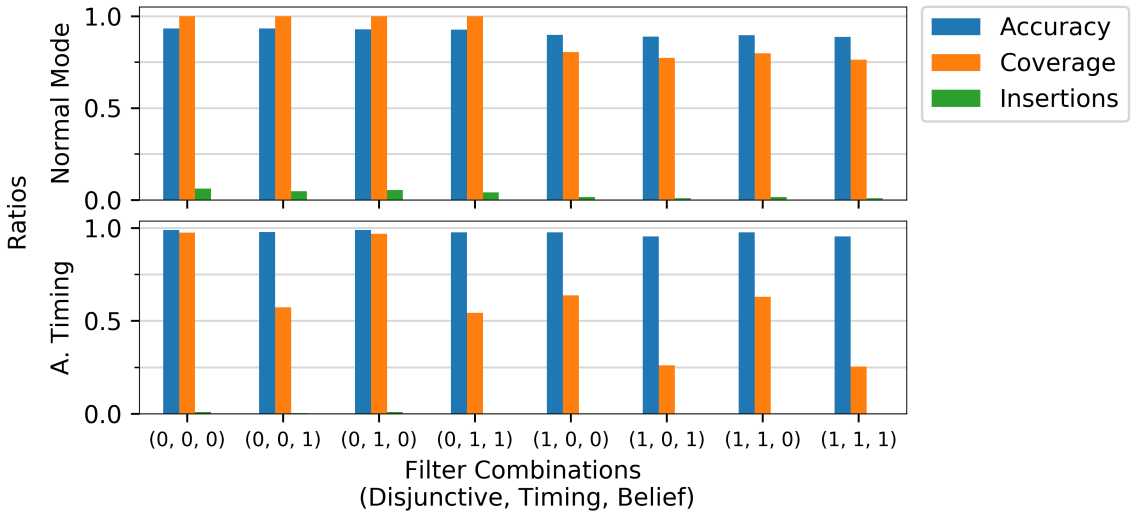


Figure 7.11: Results of applying different filter configurations to the activity diary. The results are the average over 11 patients. The symbols and the extracted rules have similar frequencies. Therefore, the filters are not suitable for the activity diary dataset. The best result was found with the alternative timing mode with no filters, because of the achieved coverage and accuracy.

7.6 Conclusion and further work

We presented the Bayes Rule Extraction algorithm (Bayesian Rule Extractor (BRE)), an association rule-mining technique for time-series. Most algorithms in the field use thresholds called minimum support and confidence in order to select which rules to consider as true. Thresholds on support and confidence are not related to the application and thus cannot be calculated, instead, they are found through parametric searches. BRE is the first rule mining approach to use Bayesian inference as a method to select candidate rules cr, avoiding the use of thresholds. After analysing three evaluation scenarios, we concluded that BRE's simple approach to rule-mining makes it an important addition to the knowledge discovery field. We showed that the observation window size can have an important effect on the extraction performance, but unlike the minimum support, which cannot be calculate beforehand, the observation window size $|ow|$ is closely related to basic knowledge of the time-series generating process, such as, the expected response time from the Building Management System (BMS) system, or average routine length in the schedule. We show that the filters play an important role in the extraction performance, and that they can separate the noise from the rule while extracting rare associations. However, filters are application dependent, as shown by the activity diary dataset, whose best performance was premise without filters. BRE is capable of extracting infrequent rules, and it has tolerance to premise and conclusion insertions. Furthermore, BRE's alternative timing mode is capable of relearning a rule as new instances become available as shown in Fig 7.5. Table 7.2 summarizes the best filter and mode configuration for each dataset. The timing filter generated a minor improvement in the activity diary dataset with respect to no filter configuration. Therefore, we conclude that the activity diary requires no filters.

BRE is not without limitations. The rules extracted in the activity diary dataset require further processing to create objects, which are useful in real-world applications, such as routine discovery. Processes that have a conclusion which depends on multiple premises, e.g., a logic function, will fail to extract all relationships due to Prop. 3. When the time series is the result of multiple generating processes, e.g., people in smart buildings, BRE will extract correct relationships at the expense of coverage. BRE's alternative timing mode is useful when the relationship between symbols are not expected to have self-relations, e.g., $a \rightarrow a$. In future work, we

	Filters			BRE Modes	
Datasets	Belief	Disjunctive	Timing	Normal	A. Timing
Synthetic time-series	•	•			•
GreenerBuidlings	•	•		•	
Activity Diary					•

Table 7.2: Summary of the best filter and mode configurations per dataset.

plan to analyse the behaviour of setting the symbol's observation window size in BRE's alternative running mode by different methods additional to the minimum timing. Additionally, we plan to redesign the timing filter to target specific timing modalities, depending on the application. Finally, we plan to analyse, how to combine atomic rules into complex models and structures.

7.7 Appendix

In this section we provide formalization and the analytical proof of the following Bayesian Rule Extractor (BRE) properties: (1) rejection boundary as Prop, 2, (2) diminishing belief as Prop. 3, and (3) absolute belief in Prop. 1.

Property 1. Absolute belief: Once a rule reaches belief of one it will never be discarded.

Proof. Assume either the prior $B(b|a)_{k-1}$ or the conditional probability $P(a|b)$ are 1.

$$
\begin{aligned}
B(b|a)_k &= \frac{P(a|b)B(b|a)_{k-1}}{P(a|b)B(b|a)_{k-1} + P(a|b)'B(b|a)'_{k-1}} \\
B(b|a)_{k-1} &= 1 \rightarrow B(b|a)'_{k-1} = 0 \\
&\text{or} \\
P(a|b) &= 1 \rightarrow P(a|b)' = 0 \\
&\Rightarrow \\
B(b|a)_k &= \frac{P(a|b)B(b|a)_{k-1}}{P(a|b)B(b|a)_{k-1}} \\
B(b|a)_k &= 1
\end{aligned}
$$

□

Remark. Property 1 implies that if an observation window *ow* contains multiple new symbols, they will all be associated to the first new symbol in persistent rules, i.e., $B(b|a) = 1$. As a result, the disjunctive filter is needed to eliminate final rules which are associations between low and high frequency symbols.

Property 2. Rejection boundary: Let $r = a \rightarrow b$ be the first observation of the rule r in the time-series. The rule r will be rejected as long as the condition shown in Eq. 7.19 is maintained, where $\#b$ is the number of times the premise was observed, and s is the length measured in number of symbols of the time-series.

$$\#b < s \tag{7.19}$$

Proof. For the first observation of a rule to be rejected by the algorithm, the belief needs to be less than the conclusion's probability. For this proof, n is the number of times the rule was observed, $\#b$ is the rule's

conclusion, and s is the length of time-series measured in number of symbols.

$$B(a \to b, prior) < prior \qquad \text{BRE's rule rejection criteria}$$

$$\frac{n/b \cdot prior}{\frac{n}{b} \cdot prior + \left(1 - \frac{n}{b}\right)(1 - prior)} < prior$$

$$n = 1 \text{ and } prior = \frac{b}{s} \qquad \text{BRE's state for first rule observation}$$

$$\frac{1/b}{\frac{1}{b} \cdot \frac{b}{s} + \left(1 - \frac{1}{b}\right)\left(1 - \frac{b}{s}\right)} < 1$$

$$\frac{1/b}{\frac{1}{s} + \frac{1}{bs} \cdot (b-1)(s-b)} < 1$$

$$\frac{1/b}{\frac{b + (b-1)(s-b)}{bs}} < 1$$

$$\frac{s}{b + bs - b^2 - s + b} < 1$$

$$s < b(2+s) - b^2 - s$$

$$2s < b(2+s) - b^2$$

$$\frac{2s}{b} < s + 2 - b$$

$$b < s + 2 - \frac{2s}{b}$$

$$b^2 < bs + 2b - 2s$$

$$b^2 - 2b < s(b-2)$$

$$\frac{b^2 - 2b}{b-2} < s$$

$$b\frac{b-2}{b-2} < s$$

$$b < s$$

□

Remark. Property 2 implies that unless the conclusion of the rule is a new symbol (see Prop. 1) the first observation of the rule will always be discarded, and rules will be accepted in the second time around, where the confidence of the rule is used instead of the conclusion's probability.

Property 3. Diminishing belief: The belief of a rule $a \to b$ decreases if the conditional probability $P(a|b)$ is less than 0.5

Proof. Property 1 is proven by finding the point where $B(b|a)_k = B(b|a)_{k-1}$.

$$B(b|a)_{k-1} = B(b|a)_k$$

$$B(b|a)_{k-1} = \frac{P(a|b)B(b|a)_{k-1}}{P(a|b)B(b|a)_{k-1} + P(a|b)'B(b|a)'_{k-1}}$$

$$1 = \frac{P(a|b)}{P(a|b)B(b|a)_{k-1} + P(a|b)'B(b|a)'_{k-1}}$$

$$P(a|b) = P(a|b)B(b|a)_{k-1} + P(a|b)'B(b|a)'_{k-1}$$

$$P(a|b)\left(1 - B(b|a)_{k-1}\right) = P(a|b)'B(b|a)'_{k-1}$$

$$P(a|b)B(b|a)'_{k-1} = P(a|b)'B(b|a)'_{k-1}$$

$$P(a|b) = P(a|b)' = 1 - P(a|b)$$

$$2P(a|b) = 1$$

$$P(a|b) = 0.5$$

□

Chapter 8

Conclusions

Functional and spatial relationship modelling is a fundamental diagnostic and maintenance tool for the lifespan of Internet of Things (IoT) enabled applications such as building management systems.

When considering the class reference programming paradigm, in combination with the semantic compiler (Chapter 2), the relationships between sensor and actuator devices is defined by functional and spatial relationships, which are characterized in the building's run-time ontology. As a result, when the user provides the locations of sensors, actuators, the semantic compiler creates all instances of inferred context that can be deduced based on the available services and devices per location. The compiler creates lists with context variables and of implementable composers. With the lists the user can write rules using the class rule paradigm, and the compiler will the required composer instances.

In the Building Management System (BMS)'s data stream, i.e., the BMS's time-series, the user's behaviour generates system responses which can be tracked. Thus, all actuators that are used by the system to provide requirements are functionally linked to the user's behaviour patterns, despite the non-deterministic association between user requirement's and available actuators. Additionally, external environmental phenomenon also generate traceable system responses, e.g., the sun's movement associates all outdoor light sensors into a façade group which allows the BMS to replace the functionality of one sensor in the group with any other (Chapter 4). The functional relationships created by both sources of events can be extracted from the building's time-series, the associated variables were correctly grouped by location (Chapter 3). Moreover, spatial relationship can be determined by user behaviour. Specifically, when commissioning spatial sensors that share an area of coverage, in Chapter 6, the relative positions were successfully extracted by observing user's movement patterns.

The rule-mining based approach used in this thesis proved to be successful at extracting the functional and spatial relationships of building variables, despite the noise conditions of real-life installations of development platforms. In this work we showed the downside of using minimum support and confidence in real life applications. Specifically, that noisy variables that trigger frequently, associate with every other variable in the system including infrequent ones (Chapters 3, 4, and 6). Noisy variables create two types of rules with high support and confidence which are actually false: one supported by too many iterations, i.e., exceeding the expected event rate, or by being a random association between a noisy and rare variables with only a few occurrences. The minimum support and commissioning downside is overcome by the introduction, in Chapter 7, of the BRE algorithm, which eliminates the use of thresholds on minimum support and confidence; instead it uses Bayesian inference to decide the truth value of a rule. Depending on the application, filter based on statistical properties of the rule's symbols and their association are used to remove false rules, in accordance to the underlying application process.

8.1 Functional and structural variable relationships

This thesis introduces two methods for extracting functional and structural relationships between building context variables: (1) Weighted Transitive Clustering (WTC) introduced in Chapter3, and (2) Hierarchical tree clustering (HTC) introduced in Chapter 4. Both showed that user generated patterns in the Building Management System (BMS) time-series can be mined and converted into groups which associate variables by location and functionality. WTC uses the confidence to control how rules aggregate into groups. With a high confidence threshold, associations occur at the cell's level. All room variables become associated as the confidence threshold is reduced. A limitation of the method is that the confidence threshold can not be computed beforehand, i.e., WTC does not provide a programmatic method for adjusting the threshold to extract a specific level of association and a parametric search is required to extract a desired structural

level. In contrast, HTC builds a tree which then is clustered to form the hierarchical groups. HTC limitations comes from building the tree; if the rules produce loops in the tree, trimming the tree can have an exponential execution time, rendering it unusable for practical applications such as error detection.

HTC and WTC found unforeseen relations in the BMS time-series. For example the façade group, where the outdoor and desk light sensors were associated into groups, as a result of the natural movement of the sun and that environmental factors, such as cloud cover, influenced all sensors simultaneously. Other example of unforeseen grouping were the windows and the light sensors during the summer months, as users will open the windows in response to increase of the sun's illumination intensity. In the case of the façade group, when a light sensor is broken, the tree can be used to find an appropriate replacement base on closes distance in the façade group. WTC functional associations were used to compute additional context variables, such as the number of people in a room based on the desk presence. The number of people estimation function is defined at the room level and WTC or HTC retrieve the desk presence sensors which belong to each room. A similar application is the automated correction of variable naming convention, when a variable's name has encoded its location, e.g., room number, associations created by HTC or WTC can be used to automatically update any discrepancies in the location section of the variable's name in the group. Deciding which location to use is implemented through a simple majority vote.

8.2 Relative position

Structural relationships are intrinsically related to a location. However, there's no information regarding the absolute or relative physical position of the devices in space. Spatial sensors' relative position are mined by a data driven methodology based on user generated events. The methodology proposed in Chapter 6 calculates the relative position of sensors by finding sensor pairs and their neighbouring regions. As users walk from one sensor's field of view to another, the transition time between fields of view is described by a probability distribution. Sensor:region pairs are selected based on the shortest expected transition value. The methodology proposed in the thesis was able to successfully retrieve the relative positions of sensors in different configurations. Although the methodology did not use any thresholds on the rule support, and it used a weighted rule aggregation to retrieve true connections. Geometries with gaps and varying sensor densities, like the foyer, proved difficult to retrieve in full.

Using a data driven approach to extract the relative position has an advantage over wireless sensor node based methods, as the methodology is independent of the protocol used by the sensor node to communicate with the BMS. Typical node positioning strategies use the node's communication channel to estimate its position in the mesh. With the method proposed in this thesis, nodes can use different communications protocols over any media, i.e., wireless or wired. A limitation of the method is that it requires spatial sensors, which can track object movement in the field of view, additionally it requires a dense area coverage. As a result, installations like public parking garages will not have transitions between sensors, as they are located on the individual parking spaces. Nevertheless, if additional sensors over the garage's driveways are installed, the algorithm can map the relative position of all sensors on the entire parking garage, independent of the sensors' means of communications. In general, the methodology can be used to simplify the commissioning process of spatial sensors.

8.3 Bayesian rule extraction

The Bayesian rule extraction algorithm (Bayesian Rule Extractor (BRE)), introduced in Chapter 7, represents a breakthrough in the development of rule mining algorithms for time-series. BRE drops the thresholds on minimum support and confidence. Instead, BRE uses Bayesian inference to represent the belief in the rule and decide whether to keep the rule or not. The decision to reject a rule is based on the decrease of belief after a new observation. Rule support and confidence are susceptible to noise in the rule's symbols. Therefore, asking rule support to always increase does not work in real-world examples. In contrast, BRE's believe will always increase as long as the ratio between the rule observations and the number of observed conclusions does not drop below 0.5. In Chapter 7, three filters based on statistical properties are developed to compliment the extracted rules' quality. The disjunction filter eliminates rules where the disjunctive probability is greater than the confidence, i.e., rules created by the association of frequent with infrequent symbols. The belief filter removes rules where the confidence is larger than the belief, i.e., rules which associate two frequent symbols that appear in a rule only a fraction of the individual symbol observations. Finally, rules whose timing is uniformly distributed over the observation window are discarded by the timing filter. BRE's only parameter is the observation window which defines how big a splice of the time-series to use for candidate rule extraction. Unlike the minimum support and confidence thresholds, which can not be calculated beforehand,

the observation window is determined by properties of the generating process such as expected symbol rate or inter-symbol timing.

Atomic rules have the power to build arbitrary graphical models. However, BRE is limited by the belief property which states: when the ratio of rule observations over the number of conclusion is 0.5, belief will decay and thus the rule will be discarded. As a result, BRE can not model processes where many symbols relate to one, e.g., $a \rightarrow d$, $b \rightarrow d$, $c \rightarrow d$. Additionally, depending on the application, when filters are applied, BRE becomes very selective, i.e., only few rules are considered true. Therefore, the filter configuration has to be selected according to the application.

8.4 Future directions

In the application realm of assisted living, and beyond the Building Management System (BMS) management applications discussed in this thesis, functional and spatial modelling play a role in the diagnostic and tracking of user patterns. User pattern anomalies can be detected as a change in associations between variables. For example, a non-critical patient, who decides not to take medicine in the morning, will have a disassociation of the medicine taking variable with the morning activities. Unlike alarm approaches, where if the user forgets the medicine once, he/she will receive a reminder, a rule mining approach could evaluate if the occurrence is a one time thing, i.e., an acceptable error rate, or a new user behaviour, which might warrant follow-up by the specialist.

The idea of patient behaviour analysis can be extended to IT security. For example, network security can be roughly summarized as the existence of private, i.e., trusted, and public, i.e., dangerous, spaces. The space separation is usually done through a firewall which physically control the traffic flow from one the public to the private network. When Internet of Things (IoT) devices are used, the network that they connect to is usually perceived as a potentially compromised network from a security perspective. Since IoT devices are usually in public spaces, an attacker could get physical access to the device and take over the computation and communication capabilities. A theoretical scenario is the empty room attack, where the attacker would fake all signals from a room in order to give the appearance that the room is empty. Under such attack, a structural relation mining, like the ones presented in this thesis, would yield that none of the room's context variables associate with the surrounding components, thus the attack would be detected. The research questions are how fast can the attack be detected? And what number of devices can be compromised in the building before the attack is no longer detectable.

This thesis explored BMS diagnostics derived from the functional and structural relationships mined for the data. For example, automatic variable naming when changing a sensor's location, programmatic context variable computation based on location information, e.g., number of people in the room based on desk presence. Broken sensor signal replacement based on group similarities, e.g., replacing outdoor light sensor with another from the automatically detected façade group. Functional and structural association can also be applied to diagnostics in other realms. For example, in compute clusters work load distribution could be analysed using rule mining and structural relations to distinguish misbehaving process, e.g., taking over system resources from. The hypothesis is that the pattern of resource association will indicate if the process belongs to a job from a parallel work load or is a stray process. A similar problem is API tracing, API calls can be arranged in several ways, and detecting what could be considered a misbehaving program is challenging. The functional and structural relationship analysis can be used to extract patterns of API calls and their respective resource consumption. Then applying modern clustering and classification techniques, the patterns can be clustered together, and classified according to the perception if the resources used is misbehaviour.

Bayesian Rule Extractor (BRE) retrieves atomic rules from the time-series. In real world applications, rules are complex and can form arbitrarily long chains. Additionally, the premise of the rule can be the result of a logic function. A specially challenging scenario to model is extracting the *disjunction* , e.g., $a \vee b \rightarrow c$, since in the time series the relative appearance frequencies of $a \rightarrow c$ and $b \rightarrow c$ will trigger BRE's rule rejection mechanism. In this thesis, the time-series was created by extracting individual variable events and then mining rules. Therefore, the disjunction function cannot be easily modelled, because when premise a was observed, the value of b was not considered, and vice-versa. A possible solution could be inspired by Ilyas et al. [44] work in databases, where the rules are found between columns within records. Instead of single events, each event will be a snapshot of the system's status. Then BRE would be applied in time, i.e., current mode, as well as in width, i.e., within the snapshot. This approach will consider, not only the transition moments of key variables, but also, the state of the building. To avoid dimensionality explosion while searching for relationships within the snapshot, a candidate rule reduction based on BRE's original observations can be applied.

One challenge of "graph-based" machine learning algorithms, e.g., Bayesian networks or hidden Markov

models, is that they depend on an arbitrary choice of initial topology to solve any learning problem. By combining the ideas of the three-phase dependency analysis algorithm introduced by Cheng et al. [14], and the idea proposed in this thesis, that any graphical model can be broken down into atomic rules, i.e., the nodes and edges which constitute the graph. A methodology is envisioned where BRE is the first part of a two stage framework to discover the topology of Bayesian and Markov like networks. The framework's objective is to derive the graph's topology without supervision from data. Future work could aim at combining the atomic rules into the different graphs types and topologies. As a result, Hierarchical tree clustering (HTC) and Weighted Transitive Clustering (WTC), and the relative spatial sensor positioning algorithm can be reimplemented using improved topologies.

BRE's approach to rule mining creates rules which are robust to time warping, i.e., the rule is extracted regardless of the number of noise-symbols or elapsed time between the rule's symbols. However, the time warping property alone does not make BRE a robust technique for motif search. The amplitude of a signal is expected to have measurement error, as a result, not only the timing between symbols changes, but the premise and the conclusion of the rule should also accept a symbol distribution. For example, for the rule $a \rightarrow b$ could have symbols represented by a mean voltage value and the limits of a uniform distribution: $a = (10 \pm 0.5)$V and $(b = 5 \pm 0.1)$V. Future work could develop the concept of symbol's distribution on how to use the variability in the symbol to effectively perform rule mining for motif search.

Symbols and Acronyms

Symbols

Chapter 1

D	Database. 3
$\bar{t}$	Average time between events. 2, 4
a	Rule's premise, usually, an event on a time series or an entry in a record. 3, 4
b	Rule's conclusion, usually an event on a time series or an entry in a record. 3, 4
n	Distance between records in the database. 3
r	Database record. 3

Chapter 2

P_r	Presence in room r. 13
λ_i	Light sensor for desk i. 12
λ_l	Light sensor for location l. 13
λ_o	Outdoor light sensor. 13
$\triangle x_l$	Illumination difference between current state and possible desired state. 13
c_1	Ceiling lamp with one fluorescent bulb. 13
c_2	Ceiling lamp with two fluorescent bulbs. 13
d_i	Desk i. 12
l	Location identifier. 13, 89
u_i	Actuator for ceiling lamp i. 12
x_l	Illumination level for each lamp controller u_i. 13

Chapter 3

A	Generic BMS variable. 20
B	Generic BMS variable. 20
C	Generic BMS variable. 20
GC	Group confidence. 21, 23
GT	Group threshold. 21, 24
G_k	Set of all groups of variables at iteration k. 21
L	Office lights event. 18
MCV	Maximum confidence arrays. 21
N	Size of G_k. 21
P	Office presence event. 18
RT	Rule acceptance threshold. 21, 24
W	Computer work event. 18
Θ_i	Event threshold value for changes in variable v_i. 20
τ	Observation window, maximum time laps between events in the rule mining process. 20
bw	Acceptance bandwidth. 21, 24
t	Time. 20
v_i	Data stream for BMS variable i. 20

Chapter 4

root	Root node of the tree. 33
A	Generic BMS variable. 31, 32
B	Generic BMS variable. 31, 32
G_d	Groups extracted from data. 36
N	Set of nodes derived from rules. 33

Γ	Set of groups. 36
Θ_i	Event threshold value for changes in variable v_i. 31
β	Set of all paths of a main branch. 33
Ne	Neighbourhood function of variable x given a group set G. 36
$\overline{L}_B$	Normalised branch length. 33
$\overline{L}_p$	Normalised path length. 33
τ	Observation window, maximum time laps between events in the rule mining process. 32
ξ	Set of edges derived from rules. 33
c_i	Edge i's weight in path p. 33
c	Rule confidence. 33
e_p	Edges of path p. 33
p	Branch path. 33
t	Time. 31
v_i	Data stream for BMS variable i. 31

Chapter 5

A_u	Uncorrected pixel projected view area. 42, 43
C	SVM's kernel penalty parameter. 46
K_n,K_1, K_2	Radial distortion coefficients. 42, 43
T_c	Corrected pixel temperature reading. 42
T_u	Uncorrected pixel temperature reading. 42
T_{amb}	Ambient temperature. 42
α	Area normalization constant. 42
σ	SVM's kernel standard deviation parameter. 46
θ	Average pixel's angle of view. 42, 43
d	Distance for area projection. 42, 43
r_c	Corrected pixel distance to optical axis. 42
r_u	Uncorrected pixel distance to optical axis. 42

Chapter 6

E	East sensor region. 56, 57
N	North sensor region. 56, 57
S	South sensor region. 56, 57
T_f	Tracks in sensor f. 52
T_s	Tracks in sensor s. 52
W	West sensor region. 56, 57
δ	Simulated sensor rotation. 52, 60
ϵ	Sub set of transition events for $s:r$ pairs. 52, 53
$\hat{E}$	Set of all transition events. 52
λ	Set of all link rules for two sensors. 54
$\mathbf{M_{GT}}$	Ground truth sensor matrix. 60
$\mathbf{M}$	Inferred sensor matrix. 54, 55, 57, 60, 90
$\mathbf{X}$	Space for sensor image analysis. 52–54, 56, 57
$\mathbf{o}_l$	Link rule orientation vector. 54
$\mathbf{p}$	Position vector of the sensor in $\mathbf{M}$. 54
$\mathbf{rp_a}$	Relative position vector for the origin. 54
$\mathbf{rp_b}$	Destination's relative position vector. 54
$\mathbf{rp_n}$	Normalized relative position vector. 53, 54
$\mathbf{rpw}_n$	Weight vectors for rule aggregation. 53, 57
$\overline{\mathbf{o}}$	Relative pairwise orientation of sensors. 54
$\overline{d}'$	Maximum observed gap time. 54
$\overline{d}$	Expected transition gap time. 52, 53
$\overline{l}$	Aggregated link rule. 54
σ	Standard deviation of the number of link rules containing $s:r$. 53
c'	Maximum link rule confidence. 54
cm_s	Maximum confidence observed in any link rule involving sensor s. 53
c	Link rule Confidence. 52–54
dw	Gap time weight. 53
d	Transition gap time. 52, 53, 56
e	Transition event. 52, 53, 57

ltd	Local trajectory direction. 56, 57
lt	Track lifetime. 52, 56, 60–62
l	Link rule. 54, 55
ow	Transition observation window. 52, 56, 57, 60–62
o	Detected object. 56
q'	Maximum number of transition events observed. 54
q	Number of transitions events observed. 52, 53
r	Sensor region. 52–56, 60
s, f, i, j, k	Sensors. 52–57, 60, 90
t	Event's time stamp. 52, 56
wc_l	Weighted confidence for link rule l. 54, 55
ws	Sliding window size for aggregating link rules. 60, 61
w	Sensor-region pair weight. 53, 54

Chapter 7

A	Symbol subset. 64
Br	Tree branch. 71
B	Symbol subset. 64
C	Ceiling lamp event. 78
E	Tree of BMS variables. 71
G_r	Rule grade. 71
P	Presence event. 78
R	Data range. 69
S	Subtree. 71
T	Dataset partition. 64
V_r	Vocabulary of rule based event generating process. 70, 71, 73
V_u	Vocabulary of uniformly distributed event generating process. 70, 71, 73–77
V	Vocabulary of distinct symbols. 64, 67
μ_t	Expected time of rule timing distribution. 67, 68, 74
σ	Standard deviation. 68, 69
a	Event on a time series. 64–69, 71, 79, 82, 83
b	Event on a time series. 64–69, 71, 79, 82, 83
cr	Candidate rule set. 65–67, 81
h	Histogram's bin size. 67–69
k	Number of observed symbols. 65, 66
n_h	Number of bins in a histogram. 69
n_r	Number of rule instances. 70
n_s	Number of samples. 70
n	Number of times a rule is observed. 82
ow	Observation window. 63–68, 70–80, 82
p_r	Randomly selected position. 70
p	Prior. 65
r	Mined rule. 67, 68, 70, 82
s_t	Length of the time series at time t. 65, 66
s	Length of the time series. 65, 82, 83
t_b	Time of conclusion b. 65
t	Time. 65, 70
w_m	Number of edges between the premise, the root of the sub-tree, and the conclusion. 71
w	Number of edges between the premise, the nearest ancestor, and the conclusions. 71

Chapter 8

a	Event on a time series. 87, 88
b	Event on a time series. 87, 88
c	Event on a time series. 87

Acronyms

BCG BMS Configuration Groups. 36, 37
BMS Building Management System. iii, 1, 2, 4, 5, 17–20, 22–25, 27, 29–31, 33, 35–39, 49, 63, 66, 69, 71, 77, 78, 81, 85–87
BOC Building's Ontology Compiler. 8, 11–14, 16
BRE Bayesian Rule Extractor. 5, 63–66, 68–83, 85–88

CG Correctness at the group level. 36, 37
CGSC Correctness at the group level with sensor class selection. 37
CHMV Correctness by head majority vote. 36, 37
CMV Correctness by majority vote. 36, 37
CRP Class Rule programming Paradigm. 8, 10–12, 16
CSC Correctness with sensor class selection. 36–38

GB GreenerBuildings. 22, 35
GTG Ground Truth Groups. 36, 37

HAC Hierarchical Agglomerative Clustering. 24, 27
HMM Hidden Markov Model. 44, 72
HTC Hierarchical tree clustering. 30, 32–39, 85, 86, 88
HVAC Heating, Ventilation, and Air-Conditioning. 17, 18, 29, 50, 71

IoT Internet of Things. iii, 1, 2, 7, 9–11, 63, 85, 87
IQR Interquartile range. 69

MBC Manual Building Commissioning. 8

PIR Passive Infra-Red motion detectors. 9, 11–13, 41

RBF Radial Basis Function. 46

SVM Support Vector Machines. 44, 46, 90

TAP Triguer-Action Paradigm. 7, 8, 12
TITArl Temporal Interval Tree Association rule learning. 19, 20, 23, 25, 32
TTC Trivial Transitive Clustering. 21, 23

USR Ultra Sound Ranging sensors. 11–13

WTC Weighted Transitive Clustering. 19, 21, 24, 25, 27, 30, 31, 33–35, 37, 85, 86, 88

Bibliography

[1] M. Achichi, M. Cheatham, Z. Dragisic, J. Euzenat, D. Faria, A. Ferrara, G. Flouris, I. Fundulaki, I. Harrow, and V. Ivanova. "Results of the Ontology Alignment Evaluation Initiative 2016". In: *CEUR Workshop Proceedings*. Vol. 1766. RWTH, 2016, pp. 73–129.

[2] K. Aduda, W. Zeiler, and G. Boxem. "Smart Grid - BEMS: The Art of Optimizing the Connection between Comfort Demand and Energy Supply". In: *Intelligent Systems Design and Engineering Applications, 2013 Fourth International Conference On*. 2013 Fourth International Conference on Intelligent Systems Design and Engineering Applications. IEEE, Nov. 2013, pp. 565–569. DOI: 10.1109/ISDEA.2013.534.

[3] Y. Agarwal, B. Balaji, R. Gupta, J. Lyles, M. Wei, and T. Weng. "Occupancy-driven energy management for smart building automation". In: *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*. ACM, 2010, pp. 1–6.

[4] R. Agrawal and R. Srikant. "Fast Algorithms for Mining Association Rules in Large Databases". In: *Proceedings of the 20th International Conference on Very Large Data Bases*. VLDB '94. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994, pp. 487–499. ISBN: 1-55860-153-8.

[5] J. Aspnes, T. Eren, D. K. Goldenberg, A. S. Morse, W. Whiteley, Y. R. Yang, B. D. Anderson, and P. N. Belhumeur. "A theory of network localization". In: *Mob. Comput. IEEE Trans. On* 5.12 (2006), pp. 1663–1678.

[6] B. Balaji, A. Bhattacharya, G. Fierro, J. Gao, J. Gluck, D. Hong, A. Johansen, J. Koh, J. Ploennigs, Y. Agarwal, M. Berges, D. Culler, R. Gupta, M. B. Kj\aergaard, M. Srivastava, and K. Whitehouse. "Brick: Towards a Unified Metadata Schema For Buildings". In: *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*. BuildSys '16. New York, NY, USA: ACM, 2016, pp. 41–50. ISBN: 978-1-4503-4264-3. DOI: 10.1145/2993422.2993577.

[7] R. J. Bayardo Jr. "Efficiently Mining Long Patterns from Databases". In: *SIGMOD Rec* 27.2 (June 1998), pp. 85–93. ISSN: 0163-5808. DOI: 10.1145/276305.276313.

[8] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. ISBN: 978-0-387-31073-2.

[9] R. Bodor, P. Schrater, and N. Papanikolopoulos. "Multi-Camera Positioning to Optimize Task Observability". In: *Advanced Video and Signal Based Surveillance, 2005. AVSS 2005. IEEE Conference On*. IEEE, 2005, pp. 552–557.

[10] buildingSMART. *IFC4 Release Summary — Welcome to buildingSMART-Tech.Org*. http://www.buildingsmart-tech.org/specifications/ifc-releases/ifc4-release/ifc4-release-summary.

[11] F. Cabitza, D. Fogli, R. Lanzilotti, and A. Piccinno. "Rule-Based Tools for the Configuration of Ambient Intelligence Systems: A Comparative User Study". en. In: *Multimedia Tools and Applications* 76.4 (2017), pp. 5221–5241. ISSN: 1380-7501, 1573-7721. DOI: 10.1007/s11042-016-3511-2.

[12] Y. Chen. "Discovering Time-Interval Sequential Patterns in Sequence Databases". In: *Expert Systems with Applications* 25.3 (Oct. 2003), pp. 343–354. ISSN: 09574174. DOI: 10.1016/S0957-4174(03)00075-7.

[13] J. Cheng, D. A. Bell, and W. Liu. "Learning Belief Networks from Data: An Information Theory Based Approach". In: *Proceedings of the Sixth International Conference on Information and Knowledge Management*. CIKM '97. New York, NY, USA: ACM, 1997, pp. 325–331. ISBN: 978-0-89791-970-8. DOI: 10.1145/266714.266920.

[14] J. Cheng, R. Greiner, J. Kelly, D. Bell, and W. Liu. "Learning Bayesian Networks from Data: An Information-Theory Based Approach". In: *Artificial Intelligence* 137.1 (May 1, 2002), pp. 43–90. ISSN: 0004-3702. DOI: 10.1016/S0004-3702(02)00191-1.

[15] P.-C. Chung and C.-D. Liu. "A Daily Behavior Enabled Hidden Markov Model for Human Behavior Understanding". In: *Pattern Recognition* 41.5 (May 1, 2008), pp. 1572–1580. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2007.10.022.

[16] L. Columbus. *Roundup Of Internet Of Things Forecasts And Market Estimates, 2016*. URL: https://www.forbes.com/sites/louiscolumbus/2016/11/27/roundup-of-internet-of-things-forecasts-and-market-estimates-2016/ (visited on 12/13/2017).

[17] D. J. Cook, A. S. Crandall, B. L. Thomas, and N. C. Krishnan. "CASAS: A Smart Home in a Box". In: 46.7 (2013), pp. 62–69. ISSN: 0018-9162. DOI: 10.1109/MC.2012.328.

[18] F. Corno, L. D. Russis, and A. M. Roffarello. "A Semantic Web Approach to Simplifying Trigger-Action Programming in the IoT". In: 50.11 (2017), pp. 18–24. ISSN: 0018-9162.

[19] S. Cox. *SOSA Ontology - Spatial Data on the Web Working Group*. https://www.w3.org/2015/spatial/wiki/SOSA_Ontology.

[20] R. De Dear. "Thermal Comfort in Practice". In: *Indoor Air* 14 (Aug. 1, 2004), pp. 32–39. ISSN: 1600-0668. DOI: 10.1111/j.1600-0668.2004.00270.x.

[21] A. De Paola, M. Ortolani, G. Lo Re, G. Anastasi, and S. K. Das. "Intelligent Management Systems for Energy Efficiency in Buildings: A Survey". In: *ACM Comput Surv* 47.1 (June 2014), 13:1–13:38. ISSN: 0360-0300. DOI: 10.1145/2611779.

[22] V. Degeler, A. Lazovik, F. Leotta, and M. Mecella. "Itemset-based mining of constraints for enacting smart environments". In: *2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS)*. 2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS). Budapest, Hungary, Mar. 2014, pp. 41–46. DOI: 10.1109/PerComW.2014.6815162.

[23] V. Degeler, L. I. Lopera Gonzalez, M. Leva, P. Shrubsole, S. Bonomi, O. Amft, and A. Lazovik. "Service-Oriented Architecture for Smart Environments (Short Paper)". In: *SOCA 2013: Proceedings of the IEEE 6th International Conference on Service-Oriented Computing and Applications*. 6th IEEE International Conference on Service-Oriented Computing and Applications, SOCA 2013. IEEE, 2013, pp. 99–104. DOI: 10.1109/SOCA.2013.26.

[24] A. K. Dey, G. D. Abowd, and D. Salber. "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications". In: *Human-Computer Interaction* 16.2 (2001), pp. 97–166. ISSN: 0737-0024. DOI: 10.1207/S15327051HCI16234_02.

[25] J. Ding, P. Bouvry, B. J. Krämer, H. Guan, A. Liang, and F. Davoli. "Context-Aware Computing and Self-Managing Systems". In: ed. by W. Dargie. Chapman and Hall, 2009, pp.363–401.

[26] P. Eichholtz, N. Kok, and J. M. Quigley. "Doing Well by Doing Good? Green Office Buildings". In: *The American Economic Review* 100.5 (2010), pp. 2492–2509. ISSN: 0002-8282. JSTOR: 41038771.

[27] "Enhanced Smart Refrigerator Systems and Methods". C. J. Merz, D. Humphreys, O. Cummins, A. Rizzini, S. Elder, and S. Toner. July 2016.

[28] V. L. Erickson, Y. Lin, A. Kamthe, B. Rohini, A. Surana, A. E. Cerpa, M. D. Sohn, and S. Narayanan. "Energy efficient building environment control strategies using real-time occupancy measurements". In: *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*. BuildSys '09. New York, NY, USA: ACM, 2009, pp. 19–24. ISBN: 978-1-60558-824-7.

[29] A. J. Eronen, V. T. Peltonen, J. T. Tuomi, A. P. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi. "Audio-Based Context Recognition". In: *IEEE Trans. Audio Speech Lang. Process.* 14.1 (Jan. 2006), pp. 321–329. ISSN: 1558-7916. DOI: 10.1109/TSA.2005.854103.

[30] A. D. Floarea and V. Sgârciu. "Smart Refrigerator: A next Generation Refrigerator Connected to the IoT". In: *2016 8th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*. 2016 8th International Conference on Electronics, Computers and Artificial Intelligence (ECAI). IEEE, June 2016, pp. 1–6. DOI: 10.1109/ECAI.2016.7861170.

[31] J. Fraden. *Handbook of Modern Sensors: Physics, Designs, and Applications*. 3rd edition. New York: Springer, 2004. 589 pp. ISBN: 978-0-387-00750-2.

[32] D. Freedman and P. Diaconis. "On the Histogram as a Density Estimator:L2 Theory". In: *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete* 57.4 (Dec. 1, 1981), pp. 453–476. ISSN: 0044-3719, 1432-2064. DOI: 10.1007/BF01025868.

[33] gbXML. *gbXML - An Industry Supported Standard for Storing and Sharing Building Properties between 3D Architectural and Engineering Analysis Software*. http://gbxml.org/.

[34] I. Georgievski, V. Degeler, G. A. Pagani, T. A. Nguyen, A. Lazovik, and M. Aiello. "Optimizing Energy Costs for Offices Connected to the Smart Grid". In: *IEEE Transactions on Smart Grid* 3.4 (2012), pp. 2273–2285. ISSN: 1949-3053. DOI: 10.1109/TSG.2012.2218666.

[35] M. Guillame-Bert and J. L. Crowley. "Learning Temporal Association Rules on Symbolic Time Sequences". In: *Proceedings of the 4th Asian Conference on Machine Learning, ACML 2012, Singapore, Singapore, November 4-6, 2012*. 2012, pp. 159–174.

[36] M. Guillame-Bert and J. L. Crowley. "Planning with Inaccurate Temporal Rules". In: *IEEE 24th International Conference on Tools with Artificial Intelligence, ICTAI 2012, Athens, Greece, November 7-9, 2012*. 2012, pp. 492–499. DOI: 10.1109/ICTAI.2012.73.

[37] J. Han and B. Bhanu. "Human Activity Recognition in Thermal Infrared Imagery". In: *IEEE Computer Vision and Pattern Recognition Conference - Workshops*. IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE, 2005, pp. 17–17. DOI: 10.1109/CVPR.2005.469.

[38] S. N. Han, G. M. Lee, and N. Crespi. "Semantic Context-Aware Service Composition for Building Automation System". In: *IEEE Transactions on Industrial Informatics* 10.1 (2014), pp. 752–761. ISSN: 1551-3203. DOI: 10.1109/TII.2013.2252356.

[39] L. Hermes and J. Buhmann. "Feature selection for support vector machines". In: *Proceedings of the 15th International Conference on Pattern Recognition*. Vol. 2. IEEE Comput. Soc, 2000, pp. 712–715. ISBN: 978-0-7695-0750-7. DOI: 10.1109/ICPR.2000.906174.

[40] P. Hevesi, S. Wille, G. Pirkl, N. Wehn, and P. Lukowicz. "Monitoring Household Activities and User Location with a Cheap, Unobtrusive Thermal Sensor Array". In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. UbiComp '14. New York, NY, USA: ACM, 2014, pp. 141–145. ISBN: 978-1-4503-2968-2. DOI: 10.1145/2632048.2636084.

[41] J. Hipp, U. Güntzer, and G. Nakhaeizadeh. "Algorithms for Association Rule Mining — a General Survey and Comparison". In: *SIGKDD Explor. Newsl.* 2.1 (June 2000), pp. 58–64. ISSN: 1931-0145. DOI: 10.1145/360402.360421.

[42] C.-w. Hsu, C.-c. Chang, and C.-j. Lin. "A Practical Guide to Support Vector Classification". In: *Bioinformatics* 1.1 (2010), pp. 1–16.

[43] *IFTTT Helps Your Apps and Devices Work Together*. https://ifttt.com.

[44] I. F. Ilyas, V. Markl, P. Haas, P. Brown, and A. Aboulnaga. "CORDS: Automatic Discovery of Correlations and Soft Functional Dependencies". In: *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*. SIGMOD '04. New York, NY, USA: ACM, 2004, pp. 647–658. ISBN: 978-1-58113-859-7. DOI: 10.1145/1007568.1007641.

[45] A. Jain, L. Hong, and S. Pankanti. "Biometric Identification". In: *Commun ACM* 43.2 (Feb. 2000), pp. 90–98. ISSN: 0001-0782. DOI: 10.1145/328236.328110.

[46] P. Jaramillo-Garcia, L. I. Lopera Gonzalez, and O. Amft. "Using implicit user feedback to balance energy consumption and user comfort of proximity-controlled computer screens". In: *Journal of Ambient Intelligence and Humanized Computing* 6.2 (Feb. 2014), pp. 207–221. DOI: 10.1007/s12652-014-0222-2.

[47] J. D. Jennings, F. M. Rubinstein, D. DiBartolomeo, and S. L. Blanc. "Comparison of Control Options in Private Offices in an Advanced Lighting Controls Testbed". In: *Journal of the Illuminating Engineering Society* 29.2 (2000), pp. 39–60.

[48] S. P. Kavulya, K. Joshi, F. D. Giandomenico, and P. Narasimhan. "Failure Diagnosis of Complex Systems". In: *Resilience Assessment and Evaluation of Computing Systems*. Ed. by K. Wolter, A. Avritzer, M. Vieira, and A. van Moorsel. Springer Berlin Heidelberg, 2012, pp. 239–261. ISBN: 978-3-642-29031-2 978-3-642-29032-9. DOI: 10.1007/978-3-642-29032-9_12.

[49] D. Kawanaka, T. Okatani, and K. Deguchi. "HHMM Based Recognition of Human Activity". In: *IEICE Trans. Inf. Syst.* E89-D.7 (July 2006), pp. 2180–2185. ISSN: 0916-8532, 1745-1361. DOI: 10.1093/ietisy/e89-d.7.2180.

[50] M. M. Khasreen, P. F. G. Banfill, and G. F. Menzies. "Life-Cycle Assessment and the Environmental Impact of Buildings: A Review". In: 1.3 (Sept. 18, 2009), pp. 674–701. ISSN: 2071-1050. DOI: 10.3390/su1030674.

[51] J. E. Kim, T. Barth, G. Boulos, J. Yackovich, C. Beckel, and D. Mosse. "Seamless Integration of Heterogeneous Devices and Access Control in Smart Homes and Its Evaluation". In: *Intelligent Buildings International* 9.1 (2017), pp. 23–39. ISSN: 1750-8975. DOI: 10.1080/17508975.2015.1018116.

[52] R. Kirichek, I. Grishin, D. Okuneva, and M. Falin. “Development of a Node-Positioning Algorithm for Wireless Sensor Networks in 3D Space”. In: *2016 18th International Conference on Advanced Communication Technology (ICACT)*. 2016 18th International Conference on Advanced Communication Technology (ICACT). Jan. 2016, pp. 279–282. DOI: 10.1109/ICACT.2016.7423360.

[53] C. Kittel and H. Kroemer. *Thermal Physics*. 2d ed. San Francisco: W. H. Freeman, 1980. 473 pp. ISBN: 978-0-7167-1088-2.

[54] K. Koperski and J. Han. “Discovery of spatial association rules in geographic information databases”. In: *Advances in spatial databases*. Springer, 1995, pp. 47–66.

[55] N. C. Krishnan, D. J. Cook, and Z. Wemlinger. “Learning a taxonomy of predefined and discovered activity patterns”. In: *JAISE* 5.6 (2013), pp. 621–637. DOI: 10.3233/AIS-130230.

[56] S. Kyriazakos, M. Mihaylov, B. Anggorojati, A. Mihovska, R. Craciunescu, O. Fratu, and R. Prasad. “eWALL: An Intelligent Caring Home Environment Offering Personalized Context-Aware Applications Based on Advanced Sensing”. In: *Wireless Personal Communications* 87.3 (Apr. 1, 2016), pp. 1093–1111. ISSN: 0929-6212, 1572-834X. DOI: 10.1007/s11277-015-2779-2.

[57] D. Kyriazis, T. Varvarigou, D. White, A. Rossi, and J. Cooper. “Sustainable Smart City IoT Applications: Heat and Electricity Management Amp; Eco-Conscious Cruise Control for Public Transportation”. In: *2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*. 2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM). June 2013, pp. 1–5. DOI: 10.1109/WoWMoM.2013.6583500.

[58] J.-B. Lamy. “Owlready: Ontology-Oriented Programming in Python with Automatic Classification and High Level Constructs for Biomedical Ontologies”. eng. In: *Artificial Intelligence in Medicine* 80 (2017), pp. 11–28. ISSN: 1873-2860. DOI: 10.1016/j.artmed.2017.07.002.

[59] Lingling Gao and Xuesong Bai. “A Unified Perspective on the Factors Influencing Consumer Acceptance of Internet of Things Technology”. In: *Asia Pacific Journal of Marketing and Logistics* 26.2 (Apr. 8, 2014), pp. 211–231. ISSN: 1355-5855. DOI: 10.1108/APJML-06-2013-0061.

[60] L. Liu, S. Wang, Y. Peng, Z. Huang, M. Liu, and B. Hu. “Mining Intricate Temporal Rules for Recognizing Complex Activities of Daily Living under Uncertainty”. In: *Pattern Recognition* 60 (Dec. 2016), pp. 1015–1028. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2016.07.024.

[61] L. I. Lopera Gonzalez and O. Amft. “Mining Device Data to Auto-Commission Buildings: Poster Abstract”. In: *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*. BuildSys ’16. New York, NY, USA: ACM, Nov. 2016, pp. 249–250. ISBN: 978-1-4503-4264-3. DOI: 10.1145/2993422.2996413.

[62] L. I. Lopera Gonzalez and O. Amft. “Mining Hierarchical Relations in Building Management Variables”. In: *Pervasive and Mobile Computing*. Pervasive and Mobile Computing 26 (Feb. 2016), pp. 91–101. ISSN: 1574-1192. DOI: 10.1016/j.pmcj.2015.10.009.

[63] L. I. Lopera Gonzalez and O. Amft. “Mining relations and physical grouping of building-embedded sensors and actuators”. In: *PerCom 2015: Proceedings of the International Conference on Pervasive Computing and Communications*. IEEE, Mar. 2015, pp. 2–10. DOI: 10.1109/PERCOM.2015.7146503.

[64] L. I. Lopera Gonzalez, U. Großekathöfer, and O. Amft. “An Intervention Study on Automated Lighting Control to Save Energy in Open Space Offices”. In: *PerEnergy 2015: IEEE International Conference on Pervasive Computing and Communications Workshops*. IEEE, Mar. 2015, pp. 317–322. DOI: 10.1109/PERCOMW.2015.7134055.

[65] L. I. Lopera Gonzalez, U. Großekathöfer, and O. Amft. “Novel stochastic model for presence detection using ultrasound ranging sensors”. In: *ACOMORE 2014: IEEE International Conference on Pervasive Computing and Communications Workshops*. PerCom Workshops. 1st Symposium on Activity and Context Modeling and Recognition. IEEE, 2014, pp. 55–60. DOI: 10.1109/PerComW.2014.6815164.

[66] L. I. Lopera Gonzalez, R. Stier, and O. Amft. “Data Mining-Based Localisation of Spatial Low-Resolution Sensors in Commercial Buildings”. In: *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*. BuildSys ’16. New York, NY, USA: ACM, Nov. 2016, pp. 187–196. ISBN: 978-1-4503-4264-3. DOI: 10.1145/2993422.2993428.

[67] L. I. Lopera Gonzalez, M. Troost, and O. Amft. “Using a thermopile matrix sensor to recognize energy-related activities in offices”. In: *SEIT 2013: Proceedings of the 3rd International Conference on Sustainable Energy Information Technology*. Procedia Computer Science. Recipient of the Best Paper Award at SEIT 2013. Elsevier, 2013, pp. 678–685. DOI: 10.1016/j.procs.2013.06.090.

[68] J. Lu, D. Birru, and K. Whitehouse. "Using simple light sensors to achieve smart daylight harvesting". In: *BuildSys 2010: Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*. BuildSys '10. Zurich, Switzerland: ACM, 2010, pp. 73–78. ISBN: 978-1-4503-0458-0. DOI: 10.1145/1878431.1878448.

[69] J. Lu, T. Sookoor, V. Srinivasan, G. Gao, B. Holben, J. Stankovic, E. Field, and K. Whitehouse. "The smart thermostat: using occupancy sensors to save energy in homes". In: *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*. SenSys '10. New York, NY, USA: ACM, 2010, pp. 211–224. ISBN: 978-1-4503-0344-6.

[70] B. L. W. H. Y. Ma. "Integrating classification and association rule mining". In: *Proceedings of the 4th International Conference on knowledge discovery and data mining(KDD-98)*. 1998.

[71] S. Mashiyama, J. Hong, and T. Ohtsuki. "Activity Recognition Using Low Resolution Infrared Array Sensor". In: *2015 IEEE International Conference on Communications (ICC)*. 2015 IEEE International Conference on Communications (ICC). June 2015, pp. 495–500. DOI: 10.1109/ICC.2015.7248370.

[72] M. Mesbahi and M. Egerstedt. *Graph theoretic methods in multiagent networks*. Princeton Series in Applied Mathematics. Princeton Univ. Press, 2010.

[73] M. Milenkovic and O. Amft. "An opportunistic activity-sensing approach to save energy in office buildings". In: *eEnergy 2013: Proceedings of the Fourth International Conference on Future Energy Systems*. ACM, 2013, pp. 247–258. DOI: 10.1145/2487166.2487194.

[74] M. Milenkovic and O. Amft. "Recognizing energy-related activities using sensors commonly installed in office buildings". In: *SEIT 2013: Proceedings of the 3rd International Conference on Sustainable Energy Information Technology*. Procedia Computer Science. Elsevier, 2013, pp. 669–677. DOI: 10.1016/j.procs.2013.06.089.

[75] R. Minerva, A. Biru, and D. Rotondi. "Towards a Definition of the Internet of Things (IoT)". In: *IEEE Internet Initiative* 1 (2015).

[76] N. Mohd Khairudin, A. Mustapha, and M. H. Ahmad. "Effect of temporal relationships in associative rule mining for web log data". In: *Sci. World J.* 2014 (2014).

[77] M. V. Moreno, L. Dufour, A. F. Skarmeta, A. J. Jara, D. Genoud, B. Ladevie, and J.-J. Bezian. "Big data: the key to energy efficiency in smart buildings". In: *Soft Comput.* (2015), pp. 1–14.

[78] K. Murao, T. Terada, A. Yano, and R. Matsukura. "Detecting Room-to-Room Movement by Passive Infrared Sensors in Home Environments". In: *AwareCast 2012: Workshop on Recent Advances in Behavior Prediction and Pervasive Computing*. 2012.

[79] M. A. Musen. "The Protégé Project: A Look Back and a Look Forward". In: *AI matters* 1.4 (2015), pp. 4–12. ISSN: 2372-3483. DOI: 10.1145/2757001.2757003.

[80] M. Muyeba, M. S. Khan, and F. Coenen. "Fuzzy Weighted Association Rule Mining with Weighted Support and Confidence Framework". In: *New Frontiers in Applied Data Mining*. Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, Berlin, Heidelberg, May 20, 2008, pp. 49–61. DOI: 10.1007/978-3-642-00399-8_5.

[81] M. K. Najafabadi, M. N. Mahrin, S. Chuprat, and H. M. Sarkan. "Improving the Accuracy of Collaborative Filtering Recommendations Using Clustering and Association Rules Mining on Implicit Data". In: *Computers in Human Behavior* 67 (Feb. 2017), pp. 113–128. ISSN: 0747-5632. DOI: 10.1016/j.chb.2016.11.010.

[82] T. A. Nguyen and M. Aiello. "Energy Intelligent Buildings Based on User Activity: A Survey". In: *Energy and Buildings* 56 (2013), pp. 244–257. ISSN: 03787788. DOI: 10.1016/j.enbuild.2012.09.005.

[83] T. A. Nguyen, A. Raspitzu, and M. Aiello. "Ontology-Based Office Activity Recognition with Applications for Energy Savings". In: *Journal of Ambient Intelligence and Humanized Computing* 5.5 (2014), pp. 667–681.

[84] Q. Ni, A. B. García Hernando, and I. Pau de la Cruz. "A Context-Aware System Infrastructure for Monitoring Activities of Daily Living in Smart Home". In: *Journal of Sensors* 2016 (2016).

[85] M. Niknam and S. Karshenas. "A Shared Ontology Approach to Semantic Representation of BIM Data". In: *Automation in Construction* 80.Supplement C (2017), pp. 22–36. ISSN: 0926-5805. DOI: 10.1016/j.autcon.2017.03.013.

[86] M. Nikolić. "Measuring Similarity of Graph Nodes by Neighbor Matching". In: *Intell Data Anal* 16.6 (Nov. 2012), pp. 865–878. ISSN: 1088-467X. DOI: 10.3233/IDA-2012-00556.

[87] D. Nistér. "Reconstruction from Uncalibrated Sequences with a Hierarchy of Trifocal Tensors". In: *Computer Vision - ECCV 2000*. Lecture Notes in Computer Science 1842. Springer Berlin Heidelberg, June 26, 2000, pp. 649–663. ISBN: 978-3-540-45054-2. DOI: 10.1007/3-540-45054-8_42.

[88] J. O'Connor. "Survey on Actual Service Lives for North American Buildings". In: *Woodframe Housing Durability and Disaster Issues Conference, Las Vegas*. 2004.

[89] N. Oliver, A. Garg, and E. Horvitz. "Layered representations for learning and inferring office activity from multiple sensory channels". In: *Comput. Vis. Image Underst.* 96.2 (Nov. 2004), pp. 163–180. ISSN: 1077-3142. DOI: 10.1016/j.cviu.2004.02.004.

[90] Panasonic Electric Works Corporation. "Infrared Array Sensor: Grid-EYE." In: (June 2012).

[91] N. Patwari, J. Ash, S. Kyperountas, A. Hero, R. Moses, and N. Correal. "Locating the nodes: cooperative localization in wireless sensor networks". In: *Signal Process. Mag. IEEE* 22.4 (July 2005), pp. 54–69. ISSN: 1053-5888. DOI: 10.1109/MSP.2005.1458287.

[92] N. Patwari, A. O. Hero, M. Perkins, N. S. Correal, and R. J. O'dea. "Relative location estimation in wireless sensor networks". In: *Signal Process. IEEE Trans. On* 51.8 (2003), pp. 2137–2148.

[93] A. Payal, C. Rai, and B. Reddy. "Analysis of Radial Basis Function network for localization framework in Wireless Sensor Networks". In: *Confluence The Next Generation Information Technology Summit (Confluence), 2014 5th International Conference* -. Sept. 2014, pp. 432–435. DOI: 10.1109/CONFLUENCE.2014.6949349.

[94] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. "Scikit-learn: Machine Learning in Python". In: *J. Mach. Learn. Res.* 12 (2011), pp. 2825–2830.

[95] R. Priyadarshini, S. Reddy, and R. Mehra. "Design, Development And Deployment Of Multi-Sensor Node Based Wireless Sensor Network For Campus Monitoring". In: *Int. J. Appl. Eng. Res.* 10.3 (2015), pp. 5599–5615.

[96] D. Ramanan, D. A. Forsyth, and A. Zisserman. "Strike a pose: Tracking people by finding stylized poses". In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005. 2005. DOI: 10.1109/CVPR.2005.335.

[97] Resource Efficient Scotland. *Building Management System Procurement Guide*. 2014. URL: http://www.resourceefficientscotland.com/sites/default/files/Building%20Management%20System%20BMS%20Technical%20Guide%20Resource%20Efficient%20Scotland%202014.pdf (visited on 06/29/2017).

[98] S. Rollins and N. Banerjee. "Using rule mining to understand appliance energy consumption patterns". In: *IEEE International Conference on Pervasive Computing and Communications, PerCom 2014, Budapest, Hungary, March 24-28, 2014*. 2014, pp. 29–37. DOI: 10.1109/PerCom.2014.6813940.

[99] A. Rosenberg and J. Hirschberg. "V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure". In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. 2007, pp. 410–420.

[100] Y. Rui and Y. Chen. "Better Proposal Distributions: Object Tracking Using Unscented Particle Filter". In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001. CVPR 2001*. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001. CVPR 2001. Vol. 2. 2001, II–786–II–793 vol.2. DOI: 10.1109/CVPR.2001.991045.

[101] A. Savvides, H. Park, and M. B. Srivastava. "The Bits and Flops of the N-Hop Multilateration Primitive for Node Localization Problems". In: *Center for Embedded Network Sensing* (Sept. 28, 2002).

[102] J. Seiter, A. Derungs, C. Schuster-Amft, O. Amft, and G. Tröster. "Activity Routine Discovery in Stroke Rehabilitation Patients without Data Annotation". In: *Proceedings of the 2nd ICTs for Improving Patient Rehabilitation Research Techniques Workshop*. ACM, 2014. DOI: 10.4108/icst.pervasivehealth.2014.255275.

[103] J. Seiter, A. Derungs, C. Schuster-Amft, O. Amft, and G. Tröster. "Daily Life Activity Routine Discovery in Hemiparetic Rehabilitation Patients Using Topic Models". In: *Methods of Information in Medicine* 54.3 (2015), pp. 248–255. DOI: 10.3414/ME14-01-0082.

[104] M. Shokoohi-Yekta, Y. Chen, B. Campana, B. Hu, J. Zakaria, and E. Keogh. "Discovery of Meaningful Rules in Time Series". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, 1085 {–1094}.

[105] H. A. Sturges. “The Choice of a Class Interval”. In: *Journal of the American Statistical Association* 21.153 (1926), pp. 65–66. ISSN: 0162-1459. JSTOR: 2965501.

[106] E. M. Tapia, S. S. Intille, and K. Larson. “Activity Recognition in the Home Using Simple and Ubiquitous Sensors”. In: *Pervasive*. Vol. 4. Springer, 2004, pp. 158–175.

[107] B. Ur, E. McManus, M. Pak Yong Ho, and M. L. Littman. “Practical Trigger-Action Programming in the Smart Home”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’14. New York, NY, USA: ACM, 2014, pp. 803–812. ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557420.

[108] F. Wahl, M. Milenkovic, and O. Amft. “A Distributed PIR-Based Approach for Estimating People Count in Office Environments”. In: *EUC 2012: Proceedings of the 10th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*. IEEE, 2012, pp. 640–647. DOI: 10.1109/ICCSE.2012.92.

[109] G. J. Wang and S. G. Jin. “Application of Association Rule Mining Technology in Collection and Management of Wireless Sensor Network Node”. In: *Applied Mechanics and Materials*. Vol. 685. Trans Tech Publ, 2014, pp. 575–578.

[110] G. M. Weiss. “Mining with Rarity: A Unifying Framework”. In: *SIGKDD Explor. Newsl.* 6.1 (June 2004), pp. 7–19. ISSN: 1931-0145. DOI: 10.1145/1007730.1007734.

[111] M.-Y. Weng, C.-L. Wu, C.-H. Lu, H.-W. Yeh, and L.-C. Fu. “Context-Aware Home Energy Saving Based on Energy-Prone Context”. In: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference On*. IEEE, 2012, pp. 5233–5238.

[112] *Works with Nest*. https://www.nest.com/works-with-nest/.

[113] C. R. Wren and E. M. Tapia. “Toward scalable activity recognition for sensor networks”. In: *LoCA 2006: Proceedings of the 4th International Symposium on Location*. Vol. 3987. Lecture Notes in Computer Science. Springer, 2006, pp. 168–185. DOI: 10.1.1.65.6272.

[114] R. Yang and M. W. Newman. “Learning from a Learning Thermostat: Lessons for Intelligent Systems for the Home”. In: *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. UbiComp ’13. New York, NY, USA: ACM, 2013, pp. 93–102. ISBN: 978-1-4503-1770-2. DOI: 10.1145/2493432.2493489.

[115] C. C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, D. F. Silva, A. Mueen, and E. Keogh. “Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets”. In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. 2016 IEEE 16th International Conference on Data Mining (ICDM). Dec. 2016, pp. 1317–1322. DOI: 10.1109/ICDM.2016.0179.

[116] K.-C. Yin, Y.-L. Hsieh, D.-L. Yang, and M.-C. Hung. “Association Rule Mining Considering Local Frequent Patterns with Temporal Intervals”. In: *Appl Math* 8.4 (2014), pp. 1879–1890.

[117] P. S. Yu, X. Li, and B. Liu. “Adding the Temporal Dimension to Search, A Case Study in Publication Search”. In: *Web Intell. IEEE WIC ACM Int. Conf. On* 0 (2005), pp. 543–549. DOI: http://doi.ieeecomputersociety.org/10.1109/WI.2005.21.

[118] R. Zetik, S. Crabbe, J. Krajnak, P. Peyerl, J. Sachs, and R. Thomä. “Detection and Localization of Persons behind Obstacles Using M-Sequence through-the-Wall Radar”. In: vol. 6201. 2006, pp. 62010I–62010I–12. DOI: 10.1117/12.667989.

[119] J. Zhao and S.-c. Cheung. “Multi-Camera Surveillance with Visual Tagging and Generic Camera Placement”. In: *Distributed Smart Cameras, 2007. ICDSC’07. First ACM/IEEE International Conference On*. IEEE, 2007, pp. 259–266.

[120] Y. Zhu, Z. Zimmerman, N. S. Senobari, C. C. M. Yeh, G. Funning, A. Mueen, P. Brisk, and E. Keogh. “Matrix Profile II: Exploiting a Novel Algorithm and GPUs to Break the One Hundred Million Barrier for Time Series Motifs and Joins”. In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. 2016 IEEE 16th International Conference on Data Mining (ICDM). Dec. 2016, pp. 739–748. DOI: 10.1109/ICDM.2016.0085.