Proceedings of the

Workshop on AI Problems and Approaches for Intelligent Environments

(AI@IE 2012)

Sebastian Bader, Anika Schumann and Stephan Sigg (eds.)

27. August 2012

in conjunction with
the *20th European Conference on Artificial Intelligence (ECAI 2012)*
Montpellier, France

Workshop on AI Problems and Approaches for Intelligent Environments (AI@IE 2012)

Organisers

- Sebastian **Bader**, MMIS & MuSAMA, Computer Science, University of Rostock, Germany
- Anika **Schumann**, IBM Research, Dublin, Ireland
- Stephan **Sigg**, National Institute of Informatics, Tokyo, Japan

Invited Speaker

- Olivier **Boissier**, Ecole Nationale Supérieure des Mines of Saint-Etienne, France

Program Committee

- Juan Carlos **Augusto**, University of Ulster, UK
- Ken **Brown**, Cork Constraint Computation Centre, Ireland
- Vic **Callaghan**, University of Essex, UK
- Patricia **Dockhorn Costa**, Federal University of Espirito Santo, Brazil
- Marcus **Handte**, University of Duisburg-Essen, Germany
- Pascal **Hitzler**, Wright State University, USA
- Gerrit **Kahl**, DFKI, Germany
- Thomas **Kirste**, University of Rostock, Germany
- Zico **Kolter**, Massachusetts Institute of Technology, USA
- Rico **Kusber**, Kassel University, Germany
- Brian **Lim**, Carnegie Mellon University, USA
- Ilaria **Lombardi**, Dipartimento di Informatica, University of Torino, Italy
- Yannick **Pencolé**, CNRS-Toulouse, France
- Gregory **Provan**, University College Cork, Ireland
- Alex **Rogers**, University of Southampton, UK
- Alessandro **Saffiotti**, Örebro University, Sweden
- Johannes **Schöning**, Germany
- Rene **Schumann**, University of Applied Sciences Western Switzerland
- Sylvie **Thiébaux**, The Australian National University, Australia
- Ilaria **Torre**, Dipartimento di Informatica, Sistemistica e Telematica, University of Genova, Italy
- Fabiana **Vernero**, Dipartimento di Informatica, University of Torino, Italy
- Lena **Wiese**, University of Hildesheim, Germany
- Nic **Wilson**, Cork Constraint Computation Centre, Ireland
- Kristina **Yordanova**, University of Rostock, Germany

Acknowledgements

Sebastian Bader is supported by DFG graduate school 1424 MuSAMA at the University of Rostock, Germany.

Contents

Preface

The first *international workshop on AI Problems and Approaches for Intelligent Environments (AI@IE)* is a one day event co-located with the *European Conference on Artificial Intelligence 2012*. It encourages the interaction between researchers in the area of artificial intelligence and smart environments to identify and discuss problems at the intersection of the two research areas, and to transfer the technical results to researchers applying AI methods in intelligent environments.

Researchers in the area of intelligent environments aim to embed intelligence into everyday working and living spaces. To reach this goal they investigate options to integrate smart technologies into ordinary objects within the environment or by controlling the available infrastructure in some clever way. The scale of tackled environments ranges from single rooms up to complete houses and whole cities. The ultimate goal is the creation of environments which support their users proactively and optimise themselves, for example with respect to energy usage. Advances in the area of artificial intelligence as well as in computer science in general should already enable researchers to build truly intelligent environments. In the last decade, numerous projects in industry and academia have targeted at providing intelligent environments and produced an impressive count of showcase-rooms or even buildings. Researchers from diverse disciplines, most notably Pervasive Computing, have been attracted by the chances and challenges in applying AI in IE. However, so far most intelligent environments are not yet intelligent from an AI perspective, but only instrumented environments providing some intelligent interaction modalities and some support for maintenance tasks. This workshop bridges the gap between AI researchers and developers of intelligent environments in various disciplines. It provides a forum for discussion between the different communities and encourages:

1. The establishment of new research collaborations between researchers from the areas of artificial intelligence, intelligent environments and pervasive computing.

2. The identification of open AI problems within the area of intelligent environments.

3. The identification of problems occurring while applying AI techniques in practice (e.g., the availability of scalable and robust implementations), which need to be solved when building intelligent environments.

4. The Collection of a set of case studies of smart environments with a particular focus on the used AI techniques and open AI problems and the lessons learned while building them.

5. The collection of benchmark data sets for the evaluation of AI methods within the area of intelligent environments.

6. The development of a suitable notion of intelligence while considering intelligent environments.

Sebastian Bader
Anika Schumann
Stephan Sigg

(organisers of AI@IE 2012)

Invited Talk

Olivier Boissier from the *Ecole Nationale Supérieure des Mines of Saint-Etienne, France* will give an invited talk on

Multi-Agent Oriented Programming and Intelligent Environments.

Abstract. These last years, the multi-agent domain has produced different proposals, such as agent-oriented programming, environment-oriented programming, interaction-oriented programming or organisation-oriented programming, for programming decentralized and open systems. In this talk we will present and discuss a seamless integration of these different programming approaches in what we call "multi-agent oriented programming" (MAOP). We discuss how this approach brings the full potential of multi-agent systems as a programming paradigm. We illustrate its use in the context of different applications and discuss how it opens interesting perspectives for developing intelligent environments.

Technical Papers

Introducing Conviviality as a New Paradigm for Interactions among IT Objects

Assaad Moawad and **Vasileios Efthymiou** and **Patrice Caire** and **Grégory Nain** and **Yves Le Traon** [1]

Abstract.
The Internet of Things allows people and objects to seamlessly interact, crossing the bridge between real and virtual worlds. Newly created spaces are heterogeneous; social relations naturally extend to *smart* objects. Conviviality has recently been introduced as a social science concept for ambient intelligent systems to highlight soft qualitative requirements like user friendliness of systems. Roughly, more opportunities to work with other people increase the conviviality. In this paper, we first propose the conviviality concept as a new interaction paradigm for social exchanges between humans and Information Technology (IT) objects, and extend it to IT objects among themselves. Second, we introduce a hierarchy for IT objects *social interactions*, from low-level one-way interactions to high-level complex interactions. Then, we propose a mapping of our hierarchy levels into dependence networks-based conviviality classes. In particular, low levels without cooperation among objects are mapped to lower conviviality classes, and high levels with complex cooperative IT objects are mapped to higher conviviality classes. Finally, we introduce new conviviality measures for the Internet of Things, and an iterative process to facilitate cooperation among IT objects, thereby the conviviality of the system. We use a smart home as a running example.

1 Introduction

Two decades ago, Mark Weiser coined the term ubiquitous computing. Ubiquitous computing "enhances computer use by making many computers available throughout the physical environment, while making them effectively invisible to the user" [22].

Today, microelectronic devices have become so small and inexpensive that they can be embedded in almost everything, making everyday objects "smart" [15]. The new paradigm of the Internet of Things (IoT) has emerged. The basic idea behind it is the pervasive presence around us of a variety of smart objects which, "through unique addressing schemes, are able to interact with each other and cooperate with their neighbors to achieve common goals" [1].

Smart objects carry chunks of application logic. They sense, log, and interpret what is happening to them and the world, they act on their own, interact with each other, and exchange information with human users. They know what "has happened to them in the past" [15]. In this heterogeneous world, consisting of both human users and objects, social relations naturally extend to objects.

The concept of *conviviality*, defined by Illich as "individual freedom realized in personal interdependence" [12], focuses on the cooperative aspects of the interactions among humans. It has recently been introduced as a social science concept for multi-agent and ambient intelligent systems to highlight soft qualitative requirements like user friendliness of systems [5].

In this paper, we extend conviviality as a new paradigm for IoT Information Technology (IT) objects in two ways. First, convivial relations among IT objects and human users allow the latter to fulfill their needs for social interactions, and second, convivial relations among IT objects facilitate cooperation among participants. The aim is to enable knowledge sharing for the collective achievement of common objectives among entities which form various groups or coalitions [3]. The challenge of capturing social relations among IT objects breaks down into the following research questions: 1) How to distinguish the different kinds of social interactions of IT objects? 2) How to map the social interactions among IT objects to conviviality classes? 3) How to measure the conviviality of an individual IT object? and 4) How to use conviviality in the Internet of Things?

Tools for conviviality are concerned in particular with dynamic aspects of conviviality, such as the emergence of conviviality from the sharing of properties or behaviors whereby each member's perception is that their personal needs are taken care of [12]. In such dynamic circumstances, the conviviality of each participating member is a key criterion.

In [4], conviviality measures were introduced by counting, for each pair of agents, the possible ways to cooperate, indicating degree of choice or freedom to engage in coalitions. In this paper we build on these measures to define conviviality measures for each agent. Our coalitional theory is based on dependence networks [6, 19], labeled directed graphs where the nodes are agents, and each labeled edge represents that the former agent depends on the latter one to achieve some goal. Furthermore, in order to increase the conviviality of the system, we establish an iterative process through which the least cooperative IT objects are identified, then, upgrades for these objects are proposed to enhance their cooperations and increase their inclusions into more coalitions.

Our motivation lies in the vision that IT objects will be endowed with all the capabilities needed for a society of objects fully integrated into human society. In [14] smart objects differ from simple tracking objects such as RFIDs, in that they are autonomous physical/digital objects augmented with sensing, processing and networking capabilities. Here, we refer to both kinds of objects as IT objects.

The structure of this paper is the following: In Section 2, we provide the background for our IT object interaction classification, we then introduce our motivating example, in Section 3. We propose our mapping between IT objects interaction classes in Section 4, and the conviviality measures for individual IT objects in Section 5. We discuss these measures in Section 6 and present some related work in Section 7, and conclude in Section 8.

[1] Interdisciplinary Center for Security, Reliability and Trust, University of Luxembourg, email: firstname.lastname@uni.lu

2 IoT Evolution and conviviality issues

The *Internet of Things* relates to the interconnection of communication enabled-IT objects [9, 1]. IT objects from our everyday life are getting more communication abilities every day. TVs, phones or cars, are now able to share information and offer services to each other.

New services taking advantage of these communication links and shared data are emerging from these new abilities of IT devices. But the way toward seamlessly interacting devices and smart services is still long.

The miniaturization of hardware material for computation made it possible to introduce programs in electronic devices. That is how autonomous regulation devices made it possible to automate several household tasks and duties (e.g.: in Heating Ventilating and Air Conditioning (HVAC) systems). This automation of basic tasks resulted in an increase of the comfort and security for users. These autonomous regulation systems were the foundation of Internet of Things.

Along with the democratization of computers, the Internet and communication technologies, autonomous regulation systems got enriched with customization capabilities and sometimes remote accesses [18]. Now, users can specify the behavior of such configurable devices to enhance their own comfort and usage. Many devices have been equipped with bi-directional communications links, for reading and writing their configurations. With simple user interfaces, non electronic-specialists are now allowed to configure and/or remotely use their devices.
The availability of Internet everywhere and at any time, opens the door to remote accesses to IT devices, being at the office or at home. One can cite media centers, alarm or heating systems, or video camera for example. However, the configuration of such communication-enabled devices can sometimes turn into a nightmare for the uninitiated. As a consequence, protocols have been set up to allow automatic device recognition and connections. Also, zero-configuration devices[20] that are able to self-configure and get ready for use are more and more present in the IT environment.

The paradigm of *Cloud* tend now to get rid of the precise location to access a device, a service or a content. Resources can be accessed at any time, from anywhere and in several ways, with no idea about the precise location of this resource.

Today, Things (i.e. IT Objects) are able to communicate, are remotely accessible and are available from anywhere at any time [21]. But the services offered by these devices do not adapt or evolve with the presence of other services from other IT Objects. The next generation of Internet connected Things should be able to autonomously collaborate, adapt their behavior and services offered, according to their capabilities and to surrounding objects' capabilities and needs. They will participate at a time, in a community of devices by providing a new service, and integrate later another community as a backup for an already existing service.

Some classification or measures have to be developed to categorize these interactions among Things from a simple data provision to a collaborative decision making capability. As a social interaction measure, the conviviality can be applied to interactions between IT objects, and with humans, and provide a first set of tools for the next generation of smart devices. They could then be able to make more accurate decisions when adapting to their surroundings and evolving. They could be able to choose the community of devices to connect to by maximizing the benefit for both the community and themselves. They could even be able to improve their social involvement by acquiring new skills or taking charge of some duties.

3 Running Example: Smart-House

In this section we present a scenario of a smart-house automation system, regulating the temperature of a room. The IT devices in this scenario communicate, trying to figure out the cause of a heating problem. Such automation systems could be used to improve the energy-efficiency of a house and also reduce the cost of living. In similar ways, smart-home or smart-city automation systems could achieve a better quality of life, improved public services, ambient assisted living, or simply entertainment.

In our example, illustrated in Figure 1, we use five types of IoT objects that can communicate with each other; a refrigerator and its log, a heating system, door sensors and a phone. To accomplish their goal and find the source of the heating problem, the devices exchange information, query their logs and perform reasoning.

The heating system is responsible for keeping the room in a specified temperature at all times. However, in the last several minutes it has not been able to reach this temperature. Therefore it informs the refrigerator that it has problems heating the room (step 1). Like the heating system, the refrigerator is responsible for keeping its interior in a specified temperature. In other words, they have similar tasks. Hence, if the refrigerator has encountered a similar problem and solved it in the past, then there is a possibility that this solution could also work for the current problem of the heating system. Consequently, every time the heating system encounters a problem that it cannot solve, it "consults" the refrigerator.

The refrigerator receives and processes this transmission. It discovers from its log (step 2) that the last time it had a problem reaching a specified temperature, this was because its door was open. After its door was closed, the refrigerator could function properly, so this was a confirmed solution to this problem. Therefore, the refrigerator searches for a signal from the door sensors and receives that one of the house doors is open (step 3). The heating system is informed by the refrigerator that the problem comes from an open door (step 4).

Finally, the heating system stops functioning, until the problem is resolved (step 5). It also informs the phone that there is a heating problem and the recommended action is to close the door (step 6).

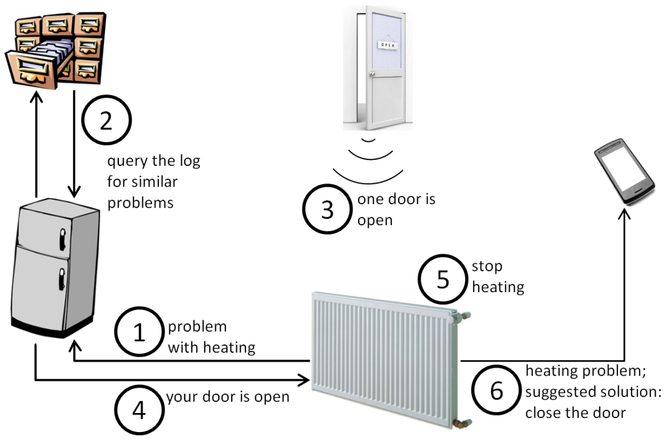


Figure 1. IT objects cooperate to solve a heating problem.

This is a typical example in Ambient Intelligence, where devices with different interaction capabilities have to cooperate. We now formalize the levels of this interaction, that we call *social interaction* of IT objects, by using the notion of conviviality.

4 IT Objects Classification and Mapping

In this section we discuss how IT objects can have a social interaction and how these interactions can be classified. For this classification, we use the notion of conviviality and Dependence Networks.

Definition 4.1 (Dependence networks) . *A dependence network (DN) is a tuple $\langle A, G, dep, \geq \rangle$ where: A is a set of agents, G is a set of goals, $dep : A \times A \rightarrow 2^G$ is a function that relates with each pair of agents, the sets of goals on which the first agent depends on the second, and $\geq$: $A \rightarrow 2^G \times 2^G$ is for each agent a total pre-order on sets of goals occurring in his dependencies: $G_1 >_{(a)} G_2$.*

Moreover, a Dependence Network can be represented by a directed graph, where the agents are the nodes of the graph, and the dependencies form the directed edges. For example, Figure 2 illustrates the graph that represents the Dependence Network derived from our motivating example of Section 3. Note that dependencies are potential, i.e, not all of them are actualized in our scenario. It should be also clarified that DNs are not equivalent to data flow networks; the latter model information exchange, not dependencies. The heating system h depends on the door d to be close, in order to function properly, but h does not have the capability to interact with d.

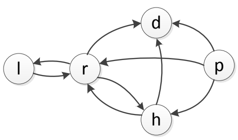


Figure 2. The DN of our example. h is the heating system, r is the refrigerator, l is the refrigerator's log, d is the door sensor and p is the phone.

Conviviality has been introduced as a social science concept for multi-agent systems to highlight soft qualitative requirements like user friendliness of systems [4]. The idea of conviviality is based on the notion of interdependency; *Cycles* denote the smallest graph topology expressing interdependence, and are considered as atomic relations conveying conviviality. When referring to *cycles*, we are implicitly signifying *simple cycles* (as defined in [8]), without repetition, with order and discarding self-loops.

In [4], conviviality is classified as presented in Figure 3, through a ranking of the DNs. Briefly, (W) is the worst class of conviviality because all agents are isolated. On the opposite side, (P) achieves perfect conviviality because the corresponding graph is a *clique*. For the in-between classes, (AWe) class has some dependencies but no cycles, (N) class has at least one isolated node and one cycle and in (APe) class, all the agents are participating in at least one cycle.

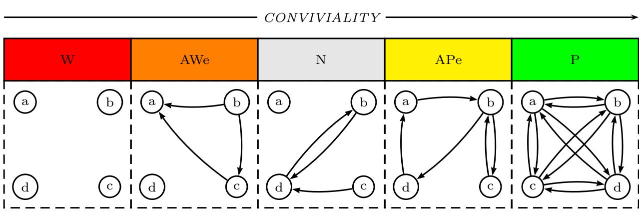


Figure 3. Conviviality Classes.

We use the term *social interaction* for IT objects, in a way similar to the human social interaction, as their ability to communicate with other IT objects and exchange information.

In Table 1, we illustrate the different levels of social interaction that an IT object can have. Level 0 IT objects are those who can only receive information from other IT objects. The phone, in our scenario belongs to Level 0, as it only receives alerts from other devices.

Level 1 is about the objects that only share their information with other objects. The door sensors are of Level 1, because they can only transmit the state of the doors to other devices.

Level 2 IT objects are programmed explicitly to interact with specific objects. The refrigerator's log is of Level 2, as it interacts only with the refrigerator. The heating system is on the same level. Level 2 is the current maximum social interaction level of IT objects.

Finally, Level 3 IT objects have the potential to interact with any other object, in order to achieve a goal. In our scenario, the refrigerator is of Level 3, since it is not explicitly programmed to interact with a specific set of devices.

We ignore IT objects operating only autonomously. The heating system could also work autonomously and try to keep a stable temperature in the room. However, its social interaction led to an improved, a more efficient functionality.

The social interaction level of IT objects can be associated with the conviviality of a network, in which they participate. To present this association in Table 1, we first suggest four possible DNs, each of them including at least one node of the specific level and then analyze the maximum conviviality in such a DN.

The maximum conviviality class of a network that includes a Level 0 or 1 object is N, since this node cannot be a part of a cycle. Level 0 objects have no incoming edges in a DN and Level 1 objects have no outgoing edges. For networks that include an object of Level 2, the conviviality cannot be better than APe, since such an IT object is not able to interact with every other device. Hence the graph of DN cannot be a clique. The maximum conviviality of a network that includes an IT object of Level 3 is P. Furthermore, P conviviality is achieved only if every node of the DN is a Level 3 IT object. W conviviality can exist if all nodes are Level 1. The maximum conviviality of a graph with only Level 0 and 1 objects is AWe.

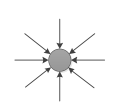
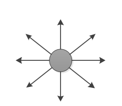
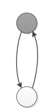
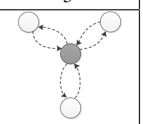
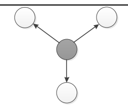
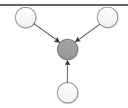
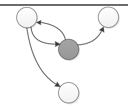
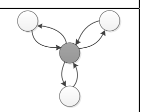
Interaction	0	1	2	3
Data				
DN				
max conv	N	N	APe	P

Table 1. Social interaction level of IT objects and maximum conviviality if at least one such object appears in a DN.

In this section, we have introduced a novel approach to classifying the social interaction of IT objects. We have mapped the social interaction level of an object with the maximum conviviality that can be achieved if this object is included in a DN. To do this mapping, we have established correspondences between DNs and IT objects interaction level. This way, we can have a maximum conviviality estimation, just by knowing the social interaction level of IT objects that participate in our system. However, it is sometimes necessary to get a more accurate measure to improve the conviviality of a system.

5 Conviviality Measures

Social network analysis has been providing many measures to reflect social interactions among agents [16]. However none of these considers cycles as basis. Our measurements meet the following requirements and assumptions.

5.1 Assumptions and Requirements

In this work, the cycles identified in a dependence network are considered as coalitions. These coalitions are used to evaluate conviviality for the network and for each agent.

In our second assumption, we consider the conviviality of a dependence network or a specific agent to be evaluated in a bounded domain, i.e., over a $[min; max]$ interval. This allows reading the values obtained by any evaluation method.

In terms of requirements, the first requirement for our conviviality measures concerns the size of coalitions. It is captured by the statement that larger coalitions are more convivial than smaller ones.

Our second requirement concerns the number of coalitions. It is captured by the statement that the more coalitions in the dependence network, the higher the conviviality of DN would be (all else being equal). Similarly, the more coalitions an agent is participating at, the higher its conviviality measure would be. This requirement is motivated by the fact that a large number of coalitions indicates more interactions among agents, which is positive in term of conviviality according to our definition based on interdependence.

5.2 Conviviality of a dependence network

The *conviviality of a dependence network* DN is defined in [4] as

$$\mathrm{Conv}(DN) = \frac{\sum_{a,b \in A, a \neq b} \mathrm{coal}(a,b)}{\Omega} \tag{1}$$

where $\mathrm{coal}(a,b)$ for any distinct $a,b \in A$ is **the number of cycles that contain both** a **and** b **in** DN and Ω is the maximum the sum in the numerator can get, over a dependence network of the same set of goals and the same number of agents but with all dependencies (fully-connected graph).

This way, the conviviality measurement of a dependence network which is a rational number in [0,1], can be used to compare different dependence networks, with 0 being the conviviality of a dependence network having no cycles at all (class W, class AWe) and 1 the conviviality of a fully-connected dependence network (class P).

However, this measurement just reflects the conviviality of the whole dependence network and does not allow to compare, inside the same dependence network, the conviviality of two different agents.

5.3 Conviviality of an agent

In this work, we extend the conviviality measures of a dependence network DN, by defining the conviviality of each agent inside DN. First, Let $C_{DN}(a)$ be the set of all cycles in DN that contains the agent a.

We define the *conviviality of an agent* $a \in A$ as

$$\mathrm{conv}_{\mathrm{DN}}(a) = \frac{\sum_{c \in C_{DN}(a)} (\mathrm{Len}(c) - 1)}{\omega} \tag{2}$$

where $\mathrm{Len}(c)$ is the length of the cycle c and ω is the maximum number the sum in the numerator can get, over a dependence network of the same size but with all possible dependencies (a clique). Moreover, ω is related to the Ω measured in Section 5.2 by the formula: $\omega = \Omega/|A|$ because of the symmetry between all agents in a clique.

This measurement per agent is also a rational number bounded in [0,1]. An agent participating in no cycle at all would have 0 conviviality, and all agents in a fully-connected dependence network would have a conviviality of 1.

Finally, the conviviality measurement for the whole dependence network defined in Section 5.2 can be deduced by calculating the average conviviality of all agents in the dependence network:

$$\mathrm{Conv}(DN) = \frac{\sum_{a \in A} (\mathrm{conv}_{\mathrm{DN}}(a))}{|A|} \tag{3}$$

5.4 Computation

We apply our computation on the dependence network of the running example illustrated in Figure 2. In this example, the set of all cycles is $C = \{(h,r),(r,l)\}$

The pairs participating in one cycle are $(h,r),(r,h),(l,r),(r,l)$ and there are no pairs participating in more than one cycle, thus the conviviality of the dependence network, according to Equation 1 is $\mathrm{Conv}(DN) = 4/\Omega$ with $\Omega = 980$ calculated over a clique of 5 nodes.

Now, to calculate the conviviality of each agent, we need to list the cycles containing that agent and applying Equation 2. We get:
$C_{DN}(h) = \{(h,r)\}$, $\mathrm{conv}(h) = 1/\omega$
$C_{DN}(r) = \{(h,r),(r,l)\}$, $\mathrm{conv}(r) = 2/\omega$
$C_{DN}(l) = \{(r,l)\}$, $\mathrm{conv}(l) = 1/\omega$
$C_{DN}(p) = \{\emptyset\}$, $\mathrm{conv}(p) = 0/\omega = 0$
$C_{DN}(d) = \{\emptyset\}$, $\mathrm{conv}(d) = 0/\omega = 0$
Where $\omega = \Omega/5 = 196$.

Note that, by taking the average of the convivialities of all the agents, we get $avg = (1/\omega + 2/\omega + 1/\omega + 0 + 0)/5 = 4/(5\omega) = 4/\Omega = \mathrm{Conv}(DN)$ as stated in Equation 3. Figure 4 shows our computation and the IT level of the objects of DN.

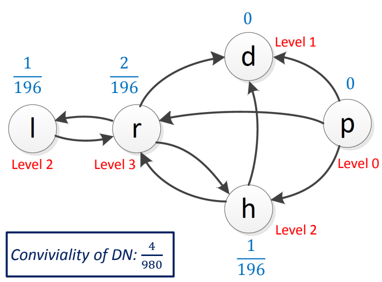


Figure 4. IT levels and conviviality measurements for the agents of DN.

As a conclusion, these measurements provide a way to compare agents to each other according to their social interactions and therefore they can be used to find potential improvements in the dependence network. For instance, in this example, we can deduce that agents d and p are the least convivial and can be seen as bottlenecks for the conviviality of DN.

6 Using Conviviality in IoT

6.1 Iterative Process

Improvement of the conviviality of a system is an iterative process. First, we identify the less participating agents in the network. Then, we try to involve them in more coalitions, which will increase their conviviality and consequently the conviviality of the system. If this solution is not applicable, then we upgrade these agents, when possible, to increase their participation. The overall conviviality of the system can thus be improved by iterating these steps.

6.2 Computation Examples

In the previous scenario, agents d and p are the least convivial and cannot do better because of their IT interaction levels of 0 and 1. We suggest as an alternate scenario S' to upgrade them by other IT objects (d' and p') having an IT interaction level of 2. But this is not enough. If the upgraded objects do not have the possibility to participate in more coalitions, the measurements will still remain the same, as the number of coalitions is unchanged. In the alternate scenario S', the smartphone p' (level 2) can have a more important role than just being a display device: it has a very good computation power and the ability to connect to the Internet to get updates and some information for example on potential solutions to a problem in the smart home context. In particular, the refrigerator and the heating system can potentially depend on its computation and connectivity capabilities. Figure 5 illustrates the dependence Network with the conviviality computation for scenario S'.

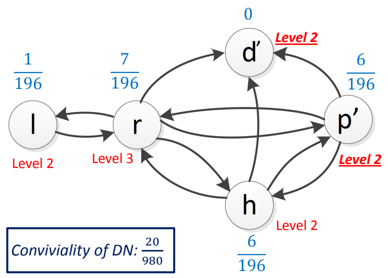


Figure 5. The alternate scenario S' with the new conviviality computation.

Comparing to Figure 4, we can deduce that conviviality of the refrigerator, heating system and the phone has improved. On the other hand, conviviality of the door and the log remain unchanged. Globally, the conviviality of the whole dependence network in S' has improved 5 times (4/980 to 20/980). Note that, the more dependencies we add to DN, the faster the conviviality increases, exponentially, to reach the 980/980 for a fully-connected DN because of the combinatorial nature of the measurements.

Finally, having the maximum conviviality is not always the best option for an IT system, because it might have other requirements and other constraints like security, privacy, efficiency, power management requirements, costs, etc. For instance, in a secured location, a smartphone might not be allowed to connect to the Internet for security reasons. In a camping context, a smartphone might not be able to do a lot of computations due to power saving measures. A good trade-off between conviviality and other requirements is the key to have a better system. Smart IT objects should be capable to adapt to different situations and contexts, selecting between different trade-offs accordingly in order to optimize their utility.

6.3 Conviviality as an Incentive

Conviviality can be used in agent theory to satisfy requirements on user-friendly systems and ensure that considerations such as the usability of a system get the same importance as the functionality.

In this section we discuss how conviviality measures can be used by agents as an incentive for cooperation, using a game-theoretic framework [17]. IT objects with a social interaction level of 3, as defined in Section 4, are not programmed explicitly to interact with specific objects. Depending on their needs, they have the ability to cooperate with any other object that will help them, or that needs to be helped. In order to decide on the form of cooperation, Level 3 objects have first to find out from which coalition they will benefit more, or, to which coalition they can contribute more.

The conviviality measures, introduced in Section 5, can be used to calculate the payoff of each agent participating in a coalition. Thus, agents have a formal way to calculate the gain that their participation in a coalition infers and therefore, decide which coalition to join.

A *co-operative game* is determined by a set Ag of agents wherein each subset of Ag is called a coalition, and a characteristic function V, assigning each coalition its maximum gain, the expected total income of the coalition (the so-called coalition value). The *payoff distribution*, P, assigns each agent its utility out of the value of the coalition it is member of in a given coalition structure [13]. In other words, P is the gain of the agent and V is the gain of the coalition.

The main idea behind these notions is to find out if the agents have an incentive to form coalitions. If the agents are not motivated to form coalitions, or if they find another, better coalition for them, then the current coalition is at risk; it is unstable. If the payoff of the agents is greater when they are in a specific coalition, than what they would gain otherwise, then that makes this coalition stable. We propose conviviality measures as a way to quantify this gain.

In our example, let's consider that the Level 3 refrigerator r is not yet a part of a network and it is trying to decide which of the networks, Figure 4 or Figure 5, to join. Then r can calculate what its conviviality would be if it joined each of these networks. It finds out that in the network of Figure 4 its conviviality would be $2/196$, whereas in the one of Figure 5 its conviviality would be $7/196$. Therefore, it decides to join the second network.

6.4 Computational Challenges

In our vision of the IoT, each IT object has the ability to act autonomously, in the sense of decision making. This means that IT objects can perform computations before joining a coalition, like the refrigerator in the previous paragraph. This is different from what usually exists today; a centralized system that makes all the computations. In today's systems, devices are usually programmed to interact only with the central computer and get these computational results, or request an available service.

The problem with our Level 3 objects is that smaller IT objects usually have a limited capability of processing. The computational complexity of our conviviality measurements is prohibiting such small devices to perform this calculation, especially as the number of agents in the network increases. This also limits the potential size of coalitions that can be created, since for larger coalitions it is harder to compute the conviviality in a reasonable time.

One possible solution is to revise these measurements and make them computationally easier for such devices. The new measurements should also meet the requirements introduced in [4]. Another approach would be to consider new definitions of conviviality.

7 Related Works

Many measures exist in graph theory domain that can be used to reflect the "social importance" of a node and the "structural importance" of a graph [10, 11, 16]. Some of the most relevant measures for a node are: *clustering coefficient* of the node which is the ratio of existing links connecting the node's neighbors to each other, to the maximum possible number of such links, *closeness centrality* of the node which is the reciprocal of the sum of distances to all other nodes in the graph. For a graph, we have the *clustering coefficient of the graph* which is the average of the clustering coefficients of all the nodes. However, these measurements do not take into consideration *cycles* in the graph. Our conviviality measures are based on the number and size of cycles in the graph which reflect interdependence.

In [7], Castelfranchi et al. use dependence networks to represent trust among agents. They build a socio-cognitive model of trust and present measurements for the degree of trust and trust capital.

The i-dud property [2] is a reciprocity property, saying that "an agent sees to a goal of another agent only if this enables it to obtain, directly or indirectly, the satisfaction of one of its own goals". This is also a desired approach for our goal-directed agents, similar to what we refer to as interdependency, or conviviality.

The notion, issues, and challenges of dynamic coalition formation (DCF) among rational software agents are introduced in [13]. For the formation of a dynamic coalition, the coalitions are represented by a coalition leader, who continuously attempts to improve the value of its coalition, by building re-configurations and suggesting them to its coalition members. This is different to what we vision for the future IoT, as discussed in 6.4, where there is no coalition leader. However, this approach could solve the computational issues that are discussed in the same section.

8 Conclusion

In this paper, we extend the social concept of conviviality as a new paradigm for IoT IT objects in two ways. First, convivial relations among IT objects and human users support the latter in fulfilling their needs for social interactions, and second, conviviality among IT objects facilitates their cooperation.

We first introduce a hierarchy for IT objects *social interactions*, from low-level one-way interactions to high-level complex interactions. Second, we propose a mapping of our hierarchy levels into dependence networks-based conviviality classes. In particular, low levels without cooperation among objects, are mapped to lower conviviality classes, and high levels with complex cooperative IT objects are mapped to higher conviviality classes. Third, we define new measures, since conviviality measures introduced in [4] are over the whole network, and do not differentiate among objects.

Fourth, in order to increase the conviviality of the system, we establish an iterative process through which the least cooperative IT objects are identified, then, upgrades for the identified objects are proposed to allow more cooperations among them, by increasing their inclusions into a greater number of coalitions. The process iterates to satisfy the system requirements, in which the tradeoffs among potentially conflicting requirements have been set, for example between conviviality, efficiency, privacy and security.

In future works, we plan to define the requirements needed for communications and negotiations among level three objects. We also want to provide a first set of tools for the next generation of smart devices and IT objects. More specifically, we plan to endow such objects with the capability of making more accurate decisions, for example, when adapting to their surroundings, and while evolving. Furthermore, we will focus on the capability for smart devices and IT objects to choose the community of devices and objects they may connect to. This choice may be guided by maximizing both their own benefit as well as their communities', i.e., the coalitions they belong to. Finally, we plan to enable devices and objects with the possibility to improve their social involvement through new skill sets acquisition and the adoption of new goals.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, 'The internet of things: A survey', *Computer Networks*, **54**(15), 2787–2805, (2010).

[2] Guido Boella, Luigi Sauro, and Leendert van der Torre, 'Algorithms for finding coalitions exploiting a new reciprocity condition', *Logic Journal of the IGPL*, **17**(3), 273–297, (2009).

[3] P. Caire, S. Villata, G. Boella, and L. van der Torre, 'Conviviality masks in multiagent systems', in *7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), Estoril, Portugal, May 12–16, 2008, Volume 3*, pp. 1265–1268, (2008).

[4] Patrice Caire, Baptiste Alcade, Leendert van der Torre, and Chattrakul Sombattheera, 'Conviviality measures', in *10th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, May 2–6, 2011*, (2011).

[5] Patrice Caire and Leendert van der Torre, 'Convivial ambient technologies: Requirements, ontology and design', *The Computer Journal*, **3**, (2009).

[6] C. Castelfranchi, 'The micro-macro constitution of power', *Protosociology*, **18**, 208–269, (2003).

[7] Cristiano Castelfranchi, Rino Falcone, and Francesca Marzo, 'Being trusted in a social network: Trust as relational capital', in *iTrust*, pp. 19–32, (2006).

[8] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, *Introduction to Algorithms*, The MIT Press, 2nd edn., 2001.

[9] *The Internet of Things - 20th Tyrrhenian Workshop on Digital Communications*, eds., Daniel Giusto, Antonio Iera, Giacomo Morabito, and Luigi Atzori, Springer, 2010.

[10] D. Gómez, 'Centrality and power in social networks: a game theoretic approach', *Mathematical Social Sciences*, **46**(1), 27–54, (August 2003).

[11] Robert A. Hanneman and Mark Riddle, *Introduction to social network methods*, 2005.

[12] Ivan Illich, *Tools for Conviviality*, Marion Boyars Publishers, London, August 1974.

[13] Matthias Klusch and Andreas Gerber, 'Dynamic coalition formation among rational agents', *IEEE Intelligent Systems*, **17**, 42–47, (2002).

[14] G. Kortuem, F. Kawsar, D. Fitton, and V. Sundramoorthy, 'Smart objects as building blocks for the internet of things', *Internet Computing, IEEE*, **14**(1), 44–51, (2010).

[15] Friedemann Mattern, 'From smart devices to smart everyday objects (extended abstract)', in *Proceedings of sOc'2003 (Smart Objects Conference)*, pp. 15–16, Grenoble, France, (may 2003).

[16] Alan Mislove, Massimiliano Marcon, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee, 'Measurement and analysis of online social networks', in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, IMC '07, pp. 29–42, New York, NY, USA, (2007).

[17] Martin J. Osborne and Ariel Rubinstein, *A Course in Game Theory*, The MIT Press, July 1994.

[18] Timothy I Salsbury, 'A survey of control technologies in the building automation industry timothy i. salsbury', *Proc IFAC World Congress*, (2005).

[19] Jaime Simão Sichman and Rosaria Conte, 'Multi-agent dependence by dependence graphs', in *Procs. of The First Int. Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS 2002*, pp. 483–490. ACM, (2002).

[20] Daniel Steinberg and Stuart Cheshire, *Zero Configuration Networking: The Definitive Guide*, O'Reilly Media, Inc., 2005.

[21] International Telecommunication Union, 'The internet of things', ITU Internet Reports 7, International Telecommunication Union, (2005).

[22] Mark Weiser, 'Some computer science issues in ubiquitous computing', *Commun. ACM*, **36**(7), 74–84, (1993).

Storage of information on manufactured products using "communicating materials"

Sylvain Kubler[1] and **William Derigent**[1] and **Éric Rondeau**[1] and **André Thomas**[1]

Abstract. The amount of data output into our environment is increasing each day, and the development of new technologies constantly redefines how we interact with this information. It is therefore necessary to control the different ways information is diffused. As a result, a data dissemination methodology in the framework of the Supply Chain Management is introduced. A specific stage of the methodology is detailed in this paper which aims at storing directly on the product, relevant data to the subsequent users. To do so, a new type of product is presented referred to as "communicating material".

1 Introduction

New challenges and opportunities arise with concepts such as Internet of Things (IoT), Ubiquitous/Pervasive Computing [15] or still Artificial Intelligence [12]. Through these concepts, objects of the real world are linked with the virtual world. Thus, connections are not just people to people or people to computers, but people to things and most strikingly, things to things [14, 13]. Ley [8] quotes the example of clothes able to carry their own information, and thus enabling the washing machine to automatically adapt its washing program. Such applications rely on ever more complex information systems combined with ever increasing data volumes, which are stored in a large number of information vectors. These vectors may be fixed (computers) or mobile (wireless devices, RFID).

Any product, during its life, passes through numerous companies and undergoes various operations (manufacturing, transportation, recycling...). Technical, semantic and organizational interoperability between these companies is not always ensured, thus, conducing to information loss. If one considers the product as an information vector (on which information could be stored), it would contribute to improve interoperability all along its life cycle. Meyer et al. [9] provides a complete survey on *intelligent products*, i.e. products carrying their own information and intelligence. Främling et al. [2] argue that it is a formidable challenge to link the product related information to the products themselves, making the information of all the product components easily achievable. However, most of the time, products are only given an identifier (e.g. via a RFID tag) which provides a network pointer to a linked database and decision making software agent [10]. Moreover, this kind of product is still limited on some points: risk of tag damage, small memory capacity, problem of data transfer (e.g. when the product is cut), etc.

As a result, we propose a new concept referred to as *communicating material*, which considers the material as intrinsically communicating. First, recent works carried out on this concept are discussed in section 2. This corpus of works mainly focus on the development of a data dissemination process to identify what the relevant information to users is and where it should be stored: on databases or on the product themselves? One important step of this framework deals with the storage/retrieval of data on/from the communicating materials, which is the subject of the paper. An appropriate architecture of communication is developed in section 3 which aims at splitting data over the communicating material and, at determining where this information is (or will be) located. An applicative scenario is finally presented in section 4.

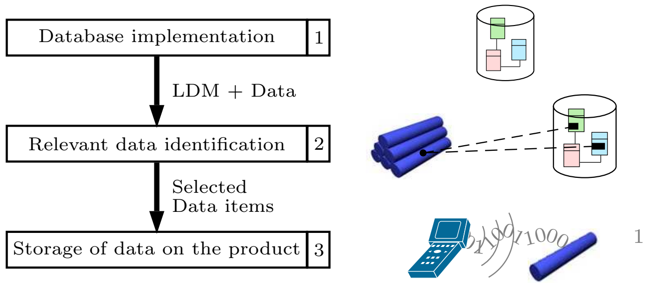
2 Data dissemination process

As said previously, the product passes through numerous companies during its life cycle. Each actor/operation requires product related information (e.g., for decision-making, production orders) which is not always available (inaccessible information, e.g. owned by another supplier) and is not always up-to-date [4] (unavailable information, e.g not shared by the supplier). Accordingly, solutions and platforms have emerged to link the product related information to the products themselves such as EPCglobal, ID@URI or WWAI [9]. However, information is generally deported on the network through these solutions because products are memory-constrained and the question of what information is relevant to users and where information needs to be stored is not answered. For this reason, we have proposed in recent publications [6, 7] a data dissemination process consisting of 3 steps as shown in Figure 1(a):

➪ Process step 1 consists in implementing the database system architecture, which can be either centralized or distributed. Many works in the literature help the designer to choose the more suitable system according to the application constraints [11].

➪ Process step 2 aims at selecting *context-sensitive information*. When users want to write information on the product at a given time (e.g. before the product leaves the company), it is necessary to identify information that is relevant (e.g. for subsequent users). This identification is achieved thanks to the process step 2. Our approach, detailed in [6], consists in assessing what is the relevance of storing a given data

[1] Centre de Recherche en Automatique de Nancy, Université de Lorraine, CNRS UMR 7039
Campus Sciences, BP 70239, F-54506 Vandœuvre-lès-Nancy Cedex, France, email: Sylvain.Kubler@cran.uhp-nancy.fr



(a) Data dissemination process

	1 (★)	2	3 (☆)
	ID_Material	Description	Value
1	MDWP0	Wood plank with a nominal 3/4"...	$4m$ of...
2	MDP-B	Textile with a high developed pol...	$3mm$...
3	MD06 **0.6**	Textile which is provided with... **0.05**	15°C... **0.41**
4	MDH-V1	Vehicle headrests which conform...	2 × ...

This corresponds to the data item noted $T_{\mathrm{Mat}\{3,1\}}$
The relevance value of $T_{\mathrm{Mat}\{3,1\}}$ is equal to 0.6

(b) Relational table *Material* & Data item relevance

Figure 1. Overview of the data dissemination process and the data assessment (data from a relational table)

item on the product according to the user concerns, the environment details, etc. One data item corresponds to a cell of a relational table (i.e. the intersection between a column, named "attribute", and a row, named "tuple") as emphasized in Figure 1(b). For instance, the data item located at row 3-column 1 in table `Material`, noted $T_{\mathrm{Mat}\{3,1\}}$, has the value `MD06`. Only data items which have an interest of being stored on the product are assessed (such a selection is also proposed in [6]). For instance, only the tuples 3 in `Material` is assessed (see dashed background in Figure 1(b)) and the relevance of each data item from this tuple is computed (e.g. the relevance of $T_{\mathrm{Mat}\{3,1\}} = 0.6$). To compute the relevance value, the approach uses the notion of priorities which are numerical values either supplied or generated through observation and experimentation and are assigned through a multi criteria evaluation [6]. At the end, all data items from all relational tables of the database are assessed and then, classified in order of relevancy. The higher the relevance value, the higher the probability that these data items will be stored on the product. The storage/retrieval of data items on/from the communicating material is done via the process step 3, as depicted in Figure 1(a).

➪ Two things are needed in process step 3: (i) first, the product must be instrumented in order to carry such information. As introduced previously, [5] propose a new concept referred to as *communicating material*, which considers the material as intrinsically and wholly communicating thanks to a huge amount of RFID μtags scattered in the material. Different textile prototypes were designed[2]through an industrial process with different types of RFID tags (μtags from Hitachi that can only store an identifier or Omron tags that can store up to 64 bytes). An example of communicating textile is shown in Figure 2. (ii) Secondly, it is necessary to design an architecture of communication to be able to store data fragments (i.e. data items) on a communicating material.

3 Data fragment storage/retrieval using communicating materials

Process step 3 of the data dissemination process deals with the storage of data items (identified across the process step 2) on the product. In this section, an appropriate architecture combined with a protocol of communication is developed in order to split data items over the material and then, to rebuild them. Subsequently, we develop a method to determine where data are located on the communicating material.

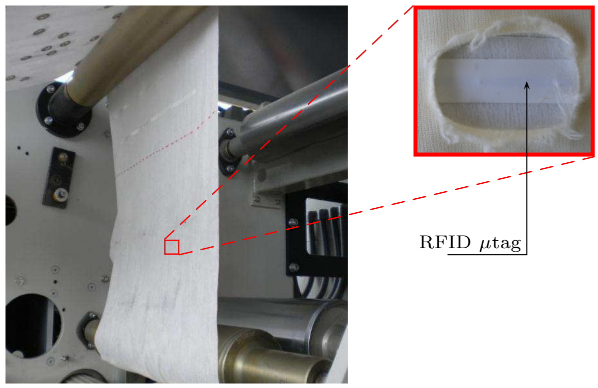
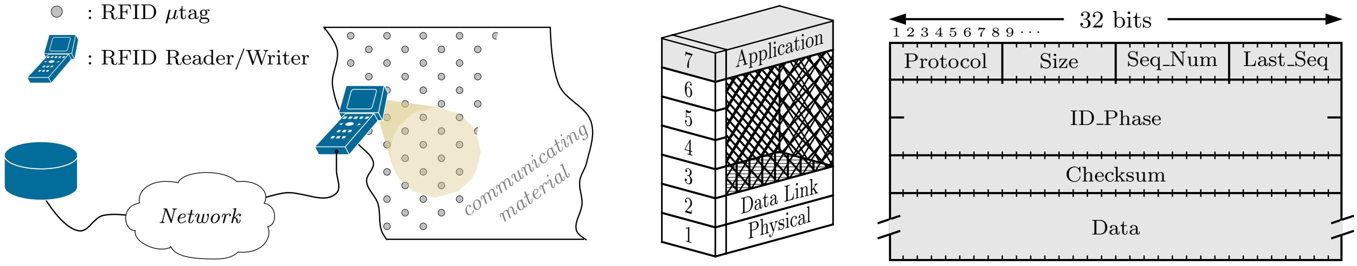


Figure 2. Prototype communicating textile designed in [5]

3.1 Data storage on communicating materials

First, it is necessary to have a communicating material, as the communicating textile designed in our previous work [5]. Let us remind ourselves that a huge quantity of RFID tags are spread over/in the material. Since the RFID tags are memory-constrained, the idea is to split the set of data items over several tags. To do so, a specific architecture must be implemented and a specific application protocol is developed (named *protocol of splitting* in our paper). Figure 3(a) depicts the global architecture with a RFID reader, a communicating material and a database (containing data which are assessed and which may be store on the material). The *protocol of splitting* respects the RFID standard ISO/IEC 18000-1 [3], which relies on the 1^{st}, 2^{nd} and 7^{th} OSI layer as shown in Figure 3(b). Layer 1 corresponds to the physical part of the RFID (i.e antenna, analog part). Layer 2 deals with the communication protocol and especially the collision mechanisms. Layer 7 deals with the application data (this is the memory portion in which data can be added or modified by users). A RFID tag may store more or less information according to the technology and, therefore, one data item may require more memory space than that available in a unique tag. That is why we propose a protocol of splitting. This application protocol is obviously defined at layer 7 of the OSI model (cf. Figure 3(b): gray background). The application data consists of 7 fields, 6 are reserved to the header (used to rebuild the

[2] Designed in collaboration with the CETELOR laboratory.



(a) Global communication architecture

(b) Frame structure used in the *protocol of splitting*

Figure 3. Protocol of communication developed to read/write data fragments from/on communicating materials

data item) and the last one contains the data item value:

1. **Protocol** (8 *bits*)**:** Integer from 0 to 255 which enables to know which fields compose the packet. The value 255 is defined in our application which refers to the frame structure defined in Figure 3(b) (specific to our application),
2. **Size** (8 *bits*)**:** Integer from 0 to 255 which indicates the size of data included into the field `Data` (7^{th} field),
3. **Seq_Num** (8 *bits*)**:** Integer from 0 to 255. It provides the sequence number of the current frame (1 frame/tag). The sequence number is used to determine the order of the different frames that have been written over the RFID tags (needed to rebuild the set of data items),
4. **Last_Num** (8 *bits*)**:** Integer from 0 to 255. It provides the sequence number of the last frame which contains data related to the same writing phase. The notion "same writing phase" is important because one data item may be written at two different times in its life cycle and then, data inconsistency/conflict may occur,
5. **ID_Phase** (64 *bits*)**:** Integer from 0 to 2^{64} which is the identifier of the writing phase (the date of writing is currently used). Several frames may have the same `ID_Phase` but a couple `ID_Phase/Seq_Num` is unique.
6. **Checksum** (32 *bits*)**:** Integer from 0 to 2^{32}. Used for data error-checking (it does not detect errors in the header),
7. **Data** ($n - 128$ *bits*)**:** The content of the data item is added in this field, which is a string. This string may or may not be stored integrally in a unique RFID tag according to the technology (where n is the number of writable data bits in one RFID tag). Let us note that an index is added for locating each data item in the database (i.e. the table name, the attribute name and the instance concerned). In our method, the index is coded as follows: `Tablename.Attributename.PrimaryKeyValue`. For instance, when the data item $T_{\text{Mat}\{3,3\}}$ is written (see Figure 1(b)), the index `Material.Value.MD06` is added. Then, the data item content is added (i.e. `15°C...`).

Regarding the application layer, 128 bits (32×4) out of n useful are used in each tag for the application header. The application logic rebuilds each data item thanks to the `Seq_Num` and the `Last_Num`, which indicate in what order data items have been split. In section 4, the tag memory is itemized after having been written. Let us note that according to the set of data items carried by the product, queries may be answered or unanswered. Nevertheless, some methods can be deployed to know in advance (i.e. before data items will be written on the product) if queries may be answered or not (e.g. by transforming queries to corresponding bitmaps [1]).

3.2 Data location on communicating materials

In this section, we develop an approach to know where information will be written on the material (in case of writing) or where information is located (in case of reading). In certain production processes like textile manufacturing, materials are transformed with highly automatised machines, at high manufacturing speed. However, some defects may occur on specific material zones (e.g., holes, stains). Because machines are not aware of these defaults, end products could not be sold. In our vision, materials are consider hard disks, able to store information about themselves. If a defect is detected, this information is then stored on the material and can be reused ad libitum when needed. The type of data manipulated is area-related : consider a grease spot on the textile. Information related to the grease spot must be stored as close as possible to the real grease spot. As a result, information must be located. Our method is designed to help locating information on the material, thus enabling machines not to be blindfolded and to adapt their behavior. To do so, a more complex architecture is required than previously. Indeed, in the previous section, the data location over the material was not taking into account and therefore, data was written/read anywhere ('on the fly') as illustrated in Figure 3(a). However, if the user desires to know where a specific information is located over the material, he needs to use a specific method which is detailed hereafter.

First, let $\mathcal{C}$ be the set of RFID tags present in the material. Let $\mathcal{R}$ be the set of RFID readers which are aligned as depicted in Figure 4 (e.g. on a ramp), with $\mathcal{R} = \{\mathrm{r}_1, \mathrm{r}_2, \mathrm{r}_3\}$). Each RFID tag and RFID reader has a reference number respectively noted $\mathtt{ID}_c$ and $\mathtt{ID}_r$ with $c \in \mathcal{C}$ and $r \in \mathcal{R}$. A reader r generates an event $e_{r,c}$ if the presence of the tag c is detected. This event is made up of $\{\mathtt{ID}_r, \mathtt{ID}_c, \mathrm{t}_{r,c}\}$, with $\mathrm{t}_{r,c}$ the acquisition time of the event. All the events form the set $\mathcal{E}$.

Let us focus now on the algorithm which calculates the theoretical positions of the tags in the material (i.e. $\in \mathcal{C}$). First, it is necessary to model the *reading zone* (i.e. zone in which a RFID reader and a RFID tag can communicate) which depends on the RFID technology. This *reading zone* must be modeled as a cylinder[3] as depicted in Figure 4 (the three RFID readers have the same cylindrical shape). Once the *reading zone* is modeled, the algorithm for computing the location $2D$ of tags takes as input the set of events $\mathcal{E}$. Since the *reading zone* is modeled as a cylinder, it is possible to calculate

[3] A methodology to model the *reading zone* of a given RFID technology is presented in [5].

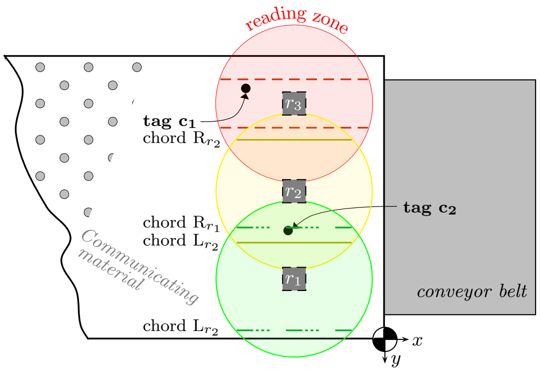


Figure 4. Architecture needed to locate information

the tag position based on the chord of the circle. The RFID reader transmits events as long as one tag is in its *reading zone* and as many events as acquisition cycles (i.e. if a tag c stays during n acquisition cycles, n events are generated). By this way, it is possible to calculate the chord based on the difference between the date of the last and the first event. Thus, we obtain two possible positions on y-axis via equation 1 for each RFID reader : r_1 and r_2 detect together c_2 and compute respectively the right chord (noted chord R_{r_1} and chord R_{r_2} in Figure 4) and the left chords (noted chord L_{r_1} and chord L_{r_2}) so as to locate c_2 ; r_3 only detects c_1 and proceeds to the same computation. Let us note that in computing the average between the two chords, the measured error is reduced by two. The tag coordinate in x axis is calculated via equation 2.

$$y_{c_t} = y_r \pm \sqrt{rad^2 - \left[\left(\frac{t_{r,c_L} - t_{r,c_F}}{2}\right) \times v\right]^2} \qquad (1)$$

$$x_{c_t} = \left(\frac{t_{r,c_F} + t_{r,c_L}}{2} - t_s\right) \times v \qquad (2)$$

x_{c_t}, y_{c_t}	theoretical coordinates x-y of the tag c ($c \in \mathcal{C}$)
x_r, y_r	coordinates in x and y-axis of the reader ($r \in \mathcal{R}$)
t_{r,c_F}	first date to which the reader r detected the tag c
t_{r,c_L}	last date to which the reader r detected the tag c
v	conveyor speed (or material speed)
t_s	textile detection date (thanks to a presence sensor)
rad	radius of the *reading zone*

The error made on the chord computation depends on both the acquisition time cycle and the conveyor speed. In fact, the higher the acquisition time cycle, the lesser the error. Obviously, the error depends also on the modeled *reading zone* (modeled as a perfect cylindrical shape). Figure 4 illustrates that the computed chords do not exactly overlap with the RFID tag c_1 and c_2. By using the chord approach, two possible chords are determined, which means that two possibilities are returned. If one tag is detected by only one RFID reader, it implies two possible coordinates y_{c_t} (impossible to know if the real tag is located on the left or right chord). This is illustrated in Figure 4 with the tag c_1 (only detected by $r3$). This problem does not occur when a tag is detected by several readers, as illustrated with c_2 (detected by r_1 and r_2). Indeed, the tag might most likely be located between ch_{r1_r} and ch_{r2_l}, as shown in Figure 4. The higher the number of readers detect one tag, the higher the precision.

4 Applicative scenario

In this scenario, we use the communicating textile reel designed in [5] (cf. Figure 2). This textile reel is preparing to leave the company X to a company Y as depicted in Figure 5 (it will be used for designing vehicle headrests). Accordingly, the supplier X decides to implement the data dissemination process for storing on the textile reel, the product related information which are useful for the supplier Y. This is done at `Activity X` as shown in Figure 5: implementation of step 1, 2 and 3 for identifying and storing the relevant data from the database, on the communicating textile reel. Let us note that the supplier X does not mind where information will be stored on the textile (i.e. it is unnecessary to use the architecture from Figure 4). Then, the textile reel will continue its transformation and will arrive at `Activity Y` as depicted in Figure 5. In this activity, the machine retrieve information carried by that one and wants to locate information on the textile (i.e. it is necessary to use the architecture from Figure 4). Both activities must implement the *protocol of splitting* detailed in section 3.1. Moreover, `Activity Y` must implement the architecture and algorithm detailed in section 3.2 for locating data on the textile.

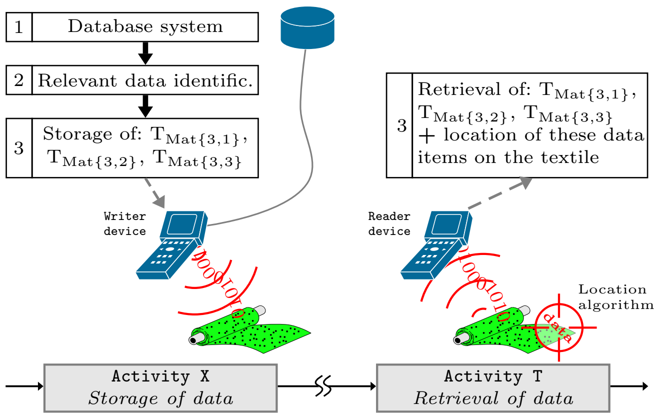
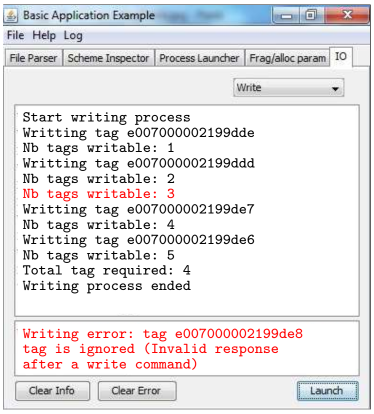


Figure 5. Description of the scenario applicative activities

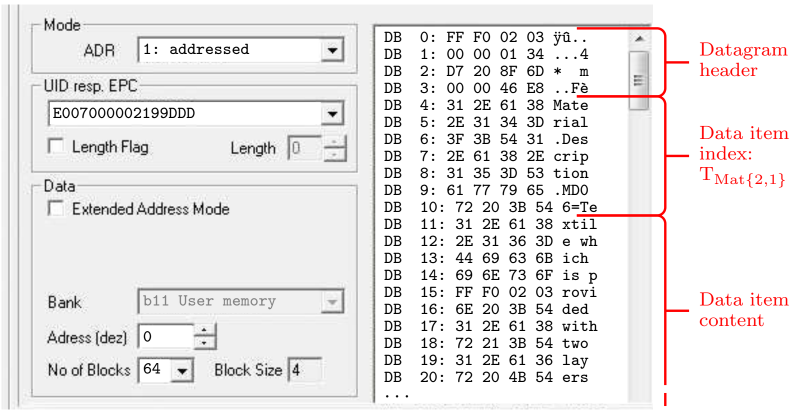
4.1 Data storage on the communicating textile prototype: Activity X

The communicating textile arrives at `Activity X`, where the machine starts the relevant data identification (i.e. step 2 of the data dissemination process). This step returns the set of data items that need to be stored on the textile. Let us assume that among these data items, the ones composing the tuple 3 of table `Material` (see Figure 1(b)) are included. The machine then starts to write $T_{Mat\{3,1\}}$, $T_{Mat\{3,2\}}$ and $T_{Mat\{3,3\}}$ over the communicating textile thanks to the *protocol of splitting* described in section 3.1, as depicted in Figure 5.

As said before, the supplier in `Activity X` does not mind where information is written over the textile. As a result, we use the architecture given in Figure 3(a) (i.e. data is written 'on the fly'). Figure 6(a) shows the log events generated when writing the three data items over the textile. It can be observed that 4 tags are required for splitting the three data items. However, 5 tags have been written because one



(a) Log events of the writing process



(b) List of data items ordered from the highest P_d to the smallest

Figure 6. Results of the data item relevance

write operation failed (tag `e007000002199de8`). This highlights that all data items are stored on the material even if writing errors occur[4]. Now, let us focus on the datagram content of a given RFID tag. Figure 6(b) details the datagram contained in the tag `e007000002199ddd` after having written the three data items. It is important to note that a RFID tag memory (whatever the RFID technology) consists of several Data Blocks (denoted `DB` in Figure 6(b)). The RFID tags disseminated in our textile are the *Omron's V720-D52P03* and their memory is divided in 64 blocks of size 4 bytes[5]. Accordingly, the datagram header occupies the four first bytes of each tag memory as highlighted in Figure 6(b) (`DB0` to `DB3`). Then, the remaining `DB` are exclusively reserved to the content of data items. However, let us remind that one index is added at beginning of each data item in order to locate it in the database (cf. section 3.1). Figure 6(b) shows the index related to $T_{Mat\{3,2\}}$, which is `Material.Description.MD06`. Then, the content of this data item (string of value: `Textile which is provided with two layers of protective coatings`) is added (cf. Figure 6(b)).

Then, the textile reel arrives at `Activity Y` in which the machine requires data carried by that one and must identify where it is located over the textile.

4.2 Data retrieval and data location: Activity Y

Since information must be located over the textile, the architecture described in Figure 4 is implemented. Then, the communicating textile passes under the ramp of RFID readers and is read. The three data items are retrieved and their content is displayed via an application software (JAVA programming) as shown in Figure 7. Then, services can be programmed: for instance, queries may be directly performed via the JAVA software based on the set of data items retrieved from the communicating textile (unanswered queries could happen). Many services may therefore be imagined as the example of Ley [8] with the washing machine (cf. section 1).

[4] This depends on the technology but most RFID technologies implement error correction mechanisms.

[5] Let us remind that one ASCII character occupies 1 byte.

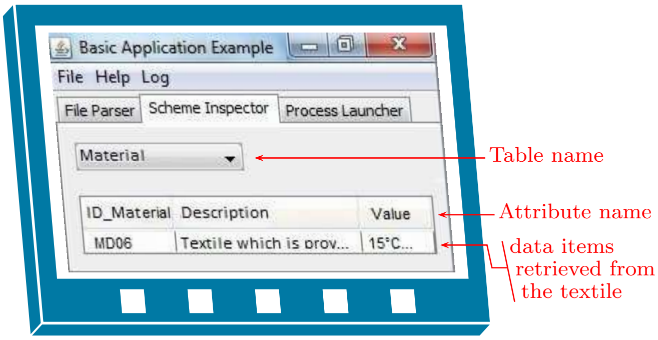
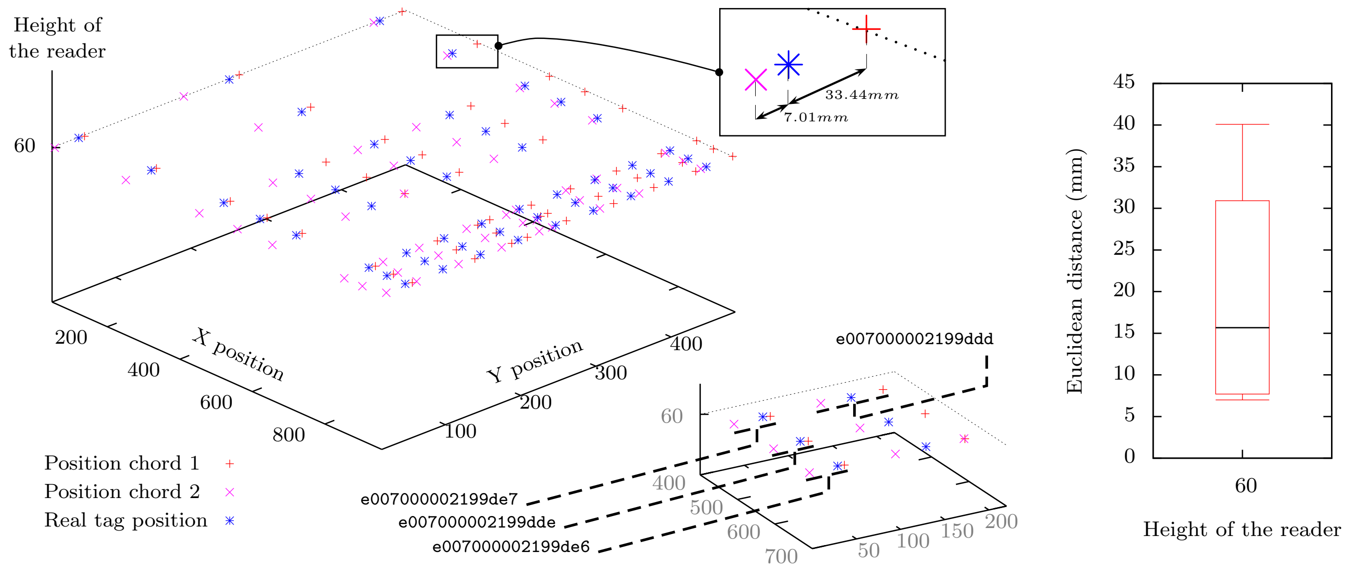


Figure 7. Display of tuples related to `Material`

Until this point, data items are retrieved from the communicating textile. To locate them over the textile, we implement the algorithm described in section 3.2. To assess the algorithm precision, the theoretical results will be confronted to the real tag positions. The virtual map given in Figure 8(a) details the real tag position as well as those returned by the algorithm (two possible solutions for one tag). These results have been obtained when the textile is read with a distance of $60mm$ from the ramp of readers[6]. If we look at the detailed case in Figure 8(a), the euclidean distance between the real tag position and those computed by the algorithm is equal to $7.01mm$ for the nearest position (chord 2) and $33.44mm$ for the furthest (chord 1). The focus in Figure 8(a) emphasized where the tags `e007000002199de6`, `e007000002199de7`, `e007000002199ddd` and `e007000002199dde` (which have been used for storing the 3 data items) are located on the textile.

For each tag, the minimal error is measured between the real tag position and the nearest position among the two computed by the algorithm (euclidean distance). The set of all the computed errors obtained is synthesized in a box-and-whisker diagram in Figure 8 (the reader is positioned $60mm$ above the textile). We can see that the minimal error (i.e. the minimal euclidean distance) is $\simeq 7mm$. 25% of the errors are inferior to $8mm$ (cf. the 1^{st} quartile) and 25% are superior to $31mm$ (cf. the 3^{rd} quartile). In average, the error is about $15mm$ with a reader positioned $60mm$ above the textile, with the conveyor speed equals to $4m/s$ and with an acquisition time

[6] The maximum distance depends on the modeled *reading zone*.



(a) Comparison between real and calculated tag positions

(b) Errors synthesized in form of whisker-box plots

Figure 8. Assessment of the precision of the location algorithm

cycle fixed to $5ms$ (errors depending on these 3 parameters).

5 Conclusion

New challenges and opportunities arise with concepts such as Internet of Things, Ubiquitous Computing and Artificial Intelligence. Nowadays, products are more and more fitted with electronic devices (e.g. sensors, RFID tags) which give them abilities such as data storage, decision making, monitoring. Some authors argued the usage of intelligent products in the framework of the supply chain management. Indeed, these products are able to control their own life, evolution and could serve as an interoperability hub between the supplier members. However, most of the time, products are only given an identifier (e.g. via a RFID tag) which provides a network pointer to a linked database and decision making software agent. As a result, a new kind of material is discussed in this paper referred to as "communicating material" which allows to embed significant proportion of data directly on the manufactured product. In order to answer the question of what information is relevant to store on the product during its lifecycle, a data dissemination process (consisting of three steps) is presented and makes reference to previous works. One important step deals with the storage and retrieval of data on/from the communicating material. In this paper, an appropriate architecture of communication is developed which aims at splitting information over the "communicating material" and at determining where this information is located.

REFERENCES

[1] D. Chan and J.F. Roddick, 'Context-sensitive mobile database summarisation', in *26th Australasian computer science conference*, volume 16, pp. 139–149, (2003).

[2] K. Främling, T. Ala-Risku, M. Kärkkäinen, and J. Holmström, 'Agent-based model for managing composite product information', *Computers in Industry*, **57**(1), 72–81, (2006).

[3] ISO/IEC. Information technology aidc techniques – rfid for item man- agement – air interface, part 1 – generic parameters for air interface communication. ISO/IEC 18000-1, 2004.

[4] M. Kärkkäinen, J. Holmström, K. Främling, and K. Artto, 'Intelligent products–a step towards a more effective project delivery chain', *Computers in Industry*, **50**(2), 141–151, (2003).

[5] S. Kubler, W. Derigent, A. Thomas, and E. Rondeau, 'Prototyping of a communicating textile', in *International Conference on Industrial Engineering and Systems Management*, pp. 1333–1342, (2011).

[6] S. Kubler, W. Derigent, A. Thomas, and E. Rondeau, 'Information dissemination process for context-aware products', in *14th Symposium of Information Control Problems in Manufacturing*, (2012).

[7] S. Kubler, W. Derigent, A. Thomas, and E. Rondeau, 'Key factors for information dissemination on communicating products and fixed databases', in *Service Orientation in Holonic and Multi-Agent Manufacturing Control*, ed., Springer, volume 402, (2012).

[8] D. Ley, 'Ubiquitous computing', *Ubiquitous Computing, emerging technologie*, **2**, 64–79, (2007).

[9] G.G. Meyer, K. Främling, and J. Holmström, 'Intelligent products: A survey', *Computers in Industry*, **60**(3), 137–148, (2009).

[10] G. Morel and B. Grabot, 'Editorial of special issue', *Engineering Applications of Artificial Intelligence*, **16**(4), 271–275, (2003).

[11] MT Ozsu and P. Valduriez, 'Principles of Distributed Database Systems', *Prentice Hall*, (1991).

[12] S. Russell and P. Norvig, 'Artificial intelligence: A modern approach', *Prentice-Hall, Egnlewood Cliffs*, (1995).

[13] M. Schneider and A. Kroner, 'The smart pizza packing: An application of object memories', in *Intelligent Environments, 2008 IET 4th International Conference on*, pp. 1–8, (2008).

[14] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelfflé, 'Vision and challenges for realising the Internet of Things', *Cluster of European Research Projects on the Internet of Things, European Commision*, (2010).

[15] M. Weiser, 'The computer for the 21st century', *Scientific American*, **265**(3), 94–104, (1991).

Domain Recompilation-based Approach towards Hierarchical Task Network Planning with Constraints

Plaban Kr Bhowmick, Debnath Mukherjee, Prateep Misra,[1] and Anupam Basu[2]

Abstract. In this paper, we present a domain recompilation-based approach towards specifying and handling state-trajectory constraints in Hierarchical Task Network (HTN) planning paradigm. The proposed constraint specification language is inspired by the PDDL3.0 constructs. The domain-recompilation technique is based on automata templates for the PDDL modal operators. To implement automata embedding we have introduced conditional effect construct in HTN. Introduction of dead automata state has helped in reducing amount of backtracking that was required originally. The constraint specification and handling strategy has been tested with a city tour and travel domain modelled using HTN.

1 INTRODUCTION

Design of intelligent environment rests upon three important activities: sense, analyze and respond. The sensors acquire relevant data from the environment, the analyze phase processes the sensor data to generate some contextual information and the respond phase needs to plan the actions for some assigned task given the context information. Thus planning is an important component of intelligent environment as far as the actuation part is concerned. Below we present an example scenario that points out the role of planners in the context of intelligent environment.

Example 1 ***City tour and travel***
A city tour and travel planner provides the travellers with smart plan based on the traveller specified intents and state of the environment. The state of the environment may be described with information related to the Points of Interests (PoIs), traffic updates, weather update and others. The state information can be obtained or updated by implanting related sensors (traffic sensors, rss feeds, twitter feeds etc.) and extracting contexts (blocked road segment, accident, inclement weather etc.) out of the sensor data.

To deploy planners in practical settings, they need to be scalable a feature that is lacking in most of the "first principle" planners [9]. Hierarchical Task Network (HTN) planning [4] paradigm handles the scalability issue by consuming knowledge regarding plan search process. This indicates that HTN paradigm is a natural fit to the domains where a planning task can be achieved through some standard process modules.

With the constraint specification standard set in PDDL3.0 [5], research in constraint-based planning has gained a considerable momentum. The classical planners that follow PDDL standard have made progress towards handling constraints imposed on goal states as well as entire set of states visited by a plan. The primary challenges in planning with constraints are

- processing of the constraint expressions for the consumption by the planning algorithm, and
- designing planning algorithm to handle constraints.

One of the popular ways to handle constraints is to model planning as a model checking problem [3]. The plan-space defines the model which needs to be validated against the stated constraints. One important decision here is the use of external model checkers which in turn may call for the adaptation of the planning algorithm that does not handle constraints. Another approach that avoids external model checkers and adaptation of planning algorithm is domain recompilation-based technique. In this approach, the effects of constraints are simulated by including them in the planning domain with no changes in the planning algorithm. The external model checker based approaches can handle complex and large number of constraints. However, model checking is expensive in terms of time and space. On the other hand, domain recompilation-based technique is efficient by limiting the scope in terms of complexity and number of constraints.

As compared to its classical counterpart, efforts towards constraint specification and handling in HTN planning is sparse. Thus the constraint or preference specification scheme, the constructs required to implement them and corresponding planning algorithm have not received the required level of attention.

In this work, we aim at incorporating constraint processing feature in HTN planner. This is achieved through extending a state-of-the-art open-source HTN planner, namely, JSHOP2[3] [7]. We have adopted the constraint specification syntax and semantics provided in PDDL3.0. A domain recompilation-based strategy has been adopted to transform state-trajectory constraints into goal constraints. The specific contributions of this paper are as follows:

- Constraint specification language
- Automata template-based approach for conversion of constraint expression to automata
- Constructs required to embed constraint automata into planing domain
- Reduction of backtracking effort

The paper is organized as follows: Section 2 provides background on HTN planning and related works in constraint-based HTN planning. Section 3 presents the constraint specification issues in HTN.

[1] Innovation Labs, Tata Consultancy Services, Kolkata, India, email: {plaban.bhowmick,debnath.mukherjee,prateep.misra}@tcs.com
[2] Department of Computer Science & Engoneering, Indian Institute of Technology Kharagpur, India, email: anupambas@gmail.com

[3] `http://www.cs.umd.edu/projects/shop/`

The constraint compilation technique for HTN is described in section 4. Section 5 describes the implementation details and some results with respect to a city tour and travel domain developed by us.

2 BACKGROUND & RELATED WORKS

Since our primary objective is to incorporate constraint specification in HTN planning paradigm, brief backgrounds on PDDL3.0 constraint specification and HTN planning paradigm are provided in this section followed by some previous efforts towards constraint and preference-based planning.

2.1 PDDL3.0 Constraint Specification

Among other features introduced in PDDL3.0, constraint specification language perhaps is the most important one. Through this language, both the soft and hard constraints can be specified over the goal state as well as entire state trajectory generated by a plan. To represent the *state-trajectory* constraints, some modal operators were introduced in PDDL3.0. The modal operators are of two types: *relative temporal* and *metric temporal*. The relative temporal operators are those that do not have any explicit mention of time unlike the metric temporal operators.

In this paper, we deal with constraints that can be specified with relative temporal modal operators. PDDL3.0 supported relative temporal modal operators are `at end`, `always`, `sometimes`, `at most once`, `sometime after` and `sometime before`. The semantics of the relative temporal modal operators are captured through Linear Temporal Logic (LTL). PDDL3.0 does not support nested modal operators as the nested construct in a constraint may contribute to the exponential growth of the corresponding automata required to process the constraint.

2.2 HTN Planning

The HTN planners differ from the classical notion of planning in various aspects. Firstly, apart from standard classical operators or *primitive tasks*, HTN makes use of *methods* for specifying domain specific plan search knowledge. Secondly, the goal in HTN is presented as a *complex task* whereas classical planning specify goal as a set of propositions.

A *task network* is defined by a set of task nodes (T) and set of precedence relations (P). Each of the task nodes contains a task and a precedence relation establishes a precedence constraint between two tasks. If all the tasks in the task network are primitive then the task network is called primitive.

An HTN domain ($D = (O, M)$) consists of primitive tasks or operators (O) and methods (M). The primitive tasks are *executable* and has precondition, add list and delete list. The complex tasks are decomposed by their corresponding methods. Depending upon different preconditions different branches of decomposition are followed. The decomposed tasks are more simpler than the original one and they again form a network depicting the precedence relations among them.

For a given task, there exists a plan *plan* $\pi = o_1 o_2 \dots o_n$ if there is a primitive decomposition (T_p) of the initial task network T_0 of the planning problem $\mathcal{P} = (S_o, T_o, D)$ and π is an instance of T_p.

The planning process starts by solving the initial task network which in turn is decomposed into more basic task network until the whole task network becomes primitive (existence of plan) or no more decomposition is possible (non-existence of plan).

2.3 Constraints in HTN Planning

As compared to classical planning, efforts towards specifying constraints or preferences have been started recently. A constraint-based HTN planning is defined as follows

Definition 1 *A constraint-based planning problem is defined as* $\mathcal{P} = (S_0, T_0, D, C)$ *where* C *is the set of constraints. The plan* π *generated for* $\mathcal{P}$ *is valid if all constraints in* C *are satisfied by state trajectory generated by* π.

There have been few efforts towards incorporating PDDL constraint specification in HTN. HTNPLAN-P [11] is an HTN-based planner that can plan with soft constraints or preferences. A best-first search technique and different heuristics have been used in order to generate preferred plans. SCUP [8] is an algorithm to perform web service composition by modelling it as a preference-based HTN planning problem. Other notable non-HTN planning algorithm that can handle PDDL constraint constructs are HPlan-P [1] and SGPLAN [6].

3 PROPOSED CONSTRAINT SPECIFICATION FOR HTN

In order to keep the constraint specification compliant to PDDL standard, the proposed constraint specification borrows PDDL constructs with minor modifications. The constraints are specified in problem description in a separate block. The syntax for constraint specification is as follows

```
(:constraint constraint_name (:modal_operator
operands))
```

A constraint expression starts with a `:constraint` tag followed by the constraint name. The modal operator used to encode the temporal modality is specified followed by the list of arguments required by the modal operator. The modal operators can be anyone of the relative temporal modal operators proposed in PDDL.

Some examples of constrains are as follows:

Example 2 *There should always 3000 amount of cash available to the traveller.*

```
(:constraint c1 (:always ((call > avail-cash
3000))))
```

Example 3 *Sometime in the plan poi1 has to be visited*

```
(:constraint c2 (:eventually (visited poi1)))
```

Example 4 *Payment by credit card will be done at most once*

```
(:constraint c3 (:at-most-once (pay-via
creditCard)))
```

Example 5 *poi7 has to be visited sometime before poi2*

```
(:constraint c4 (:s-before(visited
poi2)(visited poi7))
```

Example 6 *poi7 has to be visited sometime after poi2*

```
(:constraint c5 (:s-after(visited
poi2)(visited poi7))
```

4 COMPILATION OF CONSTRAINTS

After specification, the next step is constraint processing. In this work, we adopted a domain recompilation based technique for processing constraints. A fully automated domain recompilation technique has the following steps.

- Conversion of the constraint expressions into LTL formulae.
- Generating Büchi automata for LTL formulae.
- Embedding the automata into planning problem by changing the problem and domain description.

As modal operator set in PDDL is a closed one and PDDL does not support nesting of operators, the generation of automata from a constraint expression can be simplified by defining automata templates for the modal operators. Thus the conversion process is adapted into the following steps.

- Defining automata templates for modal operators.
- Instantiating transition labels of the automata by operands of the modal operators in the specified constraints.

4.1 Automata Templates

The automata templates are generic representation of the modal operators. An automata template consists of start state, set of transitions and set of accepting states. A transition is described by a source-destination state pair and a transition label. The automata templates are stored as xml specification. An example automata for `at-most-once` is shown in Figure 1.

On encountering a constraint expression, the corresponding automata template is retrieved and the labels of the transitions are instantiated with the logical expressions formed out of the operands in constraint expression.

4.2 Conditional Effects (CE)

After instantiation of automata, the next task is to embed it into planning problem. Thus the transitions has to be translated into a form which the original planner can process. *Conditional effect* is a construct that supports switch-case like syntax in operator definition. Apart from the standard effects, the application of an operator may impose one of multiple effects depending on the branch of precondition that is being satisfied.

Conditional effects are like operators as they have similar components like precondition, delete list and add list. The automata states are represented by state predicates like `(state-C S0)`, `(state-C S1) ... (state-C Sn)`. The plan state and the automata states are synchronized by adding automata state predicate to the current plan state. Thus for a constraint `C`, a plan state will contain no more than one of the automata state predicates. A transition from one state `Sm` to another state `Sn` with label ϕ can be modelled by a CE having

- Precondition: logical conjunction of `(state-C Sm)` and ϕ
- Delete list: current automata state i.e., `(state-C Sm)`, `(accepting-C)` if `Sn` is not an accepting state.
- Add list: next automata state i.e., `(state-C Sn)`, `(accepting-C)` if `Sn` is an accepting state.

The conditional effect form for automata representing the constraint `always (call > avail-cash 3000)` (available cash should be greater than 3000) is presented below.

```
(:when (!cond_c3_0)
     ((state-c3 S0)(avail-cash ?VARC0)(call >
      ?VARC0 3000.0 ))
     ((state-c3 S0))
     ((state-c3 S1)(accepting-c3))
) (:when (!cond_c3_1)
     ((state-c3 S1)(avail-cash ?VARC0)(call >
      ?VARC0 3000.0 ))
     ((state-c3 S1))
     ((state-c3 S1)(accepting-c3))
)
(:when (!cond_c3_2)
     ((state-c3 S1)(avail-cash ?VARC0)(not
      (call > ?VARC0 3000.0 )))
     ((state-c3 S1)(accepting-c3))
     ((state-c3 S2)(dead-c3))
)
(:when (!cond_c3_3)
     ((state-c3 S0)(avail-cash ?VARC0)(not
      (call > ?VARC0 3000.0 )))
     ((state-c3 S0)(accepting-c3))
     ((state-c3 S2)(dead-c3))
)
```

4.3 Automata Simulation

After representing the automata into required construct, one needs to simulate the automata correctly to test the validity of the constraints. This is achieved through the following changes in the original planning problem.

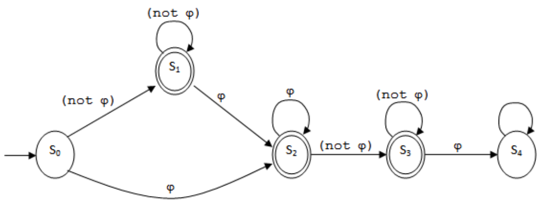
- Start state initialization: The start states of the automata have to be initialized and the corresponding state predicates have to be inserted in the plan state prior to start solving the actual task network. This is achieved through the inclusion of a new operators `!start` in the compiled domain description. The `!start` operator for two constraints is shown below.

  ```
  (:operator (!start)
       ()
       ()
       ((state-c1 S0)(state-c2 S0))
  )
  ```

  It is to be noted that the operator has empty precondition and delete list. The add list consists of the predicates stating that the automata are in start state.
- Probing plan validity: A constraint is satisfied if the corresponding automata is in an accepting state at the end of the plan. For a constraint `C`, the `!finish` operator performs this task by checking whether `(accepting-C)` is there in the final plan state. If so the plan satisfies `C`; otherwise it is violated. An example finish operator is shown below.

  ```
  (:operator (!finish)
       (accepting-c1)(accepting-c2)
       ()
       ()
  )
  ```

- Changes in operators: After applying an operator, some state predicates will be changed which in turn ask for changes in the current states of some automata. As discussed earlier, the transitions can be modelled with CEs. The union of the set of CEs corresponding to the automata for all the constraints will decide the next automata state after applying one action. Thus the union of the CEs




```
<automata>
    <name value="At-Most-Once"></name>
    <start state="S0"></start>
    <transition from="S0" to="S1" label="not X"></transition>
    <transition from="S1" to="S1" label="not X"></transition>
    <transition from="S1" to="S2" label="X"></transition>
    <transition to="S2" from="S2" label="X"></transition>
    <transition from="S0" to="S2" label="X"></transition>
    <transition from="S2" to="S3" label="not X"></transition>
    <transition from="S3" to="S3" label="not X"></transition>
    <transition from="S3" to="S4" label="X"></transition>
    <accept state="S1"></accept>
    <accept state="S2"></accept>
    <accept state="S3"></accept>
    <dead state="S4"></dead></automata>
<automata>
```

Figure 1. Automata template for at-most-once modal operator

are added to the domain operators in compiled domain description.

- Modified initial task network: Apart from the tasks in the original initial task network, two additional tasks `!start` and `!finish` have to be performed. The modified initial task network in the compiled problem description is `(!start)(original initial task network)(!finish)`.
- Removal of constraint block: As the constraints have already been taken care of by the automata, they are removed in the compiled problem description.

5 IMPLEMENTATION AND RESULTS

In this section, we describe the implementing details of constraint processing in existing HTN framework. We study different aspects of constraint-based HTN planning in the purview of a domain called city tour and travel.

5.1 Constraints in JSHOP2

JSHOP2 is a java implementation of Simple Hierarchical Ordered Planner (SHOP2) [10]. It is an open source tool for generating problem specific planner. Current implementation does not have the facility to specify and process constraints. In this work, we have extended JSHOP2 to incorporate PDDL3.0 supported relative temporal operators. Here we describe the implementation details of the said extension. The modifications are as follows:

- Constraint block: A constraint block has been added in the problem description to specify constraints. The constraints are expressed by using modal operators. The original ANTLR[4] JSHOP2 grammar has been modified to add rules for parsing constraint expressions.
- Conditional Effect: JSHOP2 does not support conditional effect construct. Looking at the similarity of the constructs, the CE extension has been implemented by modelling CEs as operators.
- Translation of CEs: The CEs are automatically translated into JSHOP2 operator format and appended into the original operators.
- Inserting `!start` and `!finish`: Depending on the expressed constraints, the `!start` and `!finish` operators are constructed and added to the operator list in domain description.

Next we provide a brief description of HTN domain developed for city tour and travel application and this domain is used to test the proposed extension of JSHOP2.

[4] `http://www.antlr.org/`

5.2 City Tour and Travel Domain

City tour and travel is a service provided by the city authority to aid the prospective tourists with smart travel plans. The travellers may specify their travel intents in terms of points of interest to be visited, other activities. The application in response generates valid plans with respect to the traveller's intents and constraints imposed by him/her. Here, we briefly describe the domain the detail of which is given in [2].

The operators and the methods in the domain description are given in Table 1 and Table 2 respectively.

Table 1. Operators in city tour and travel domain

Operators	Synopsis
`!load-pref`	This is used in loading a traveller intent
`!set-cost`	This is used to set the tour cost based on available amount and estimated cost
`!mark-stay-hotel`	Marking one hotel of traveller's choice and updates time of day
`!lunch`	This is used to represent the fact that the user has taken lunch
`!dinner`	Mark the fact that the traveller has taken dinner
`!set-loc`	The traveller has visited a particular location
`!end-day`	Marks the end of day and reset time of day for the next day

Table 2. Methods in city tour and travel domain

Methods	Synopsis
`ini-pref`	a generic method that loads all the traveller intents
`end-day`	To probe the end of day condition
`find-hotel-go`	To find the hotel that matches traveller specified features
`travel`	To visit the traveller provided points of interest
`find-rest`	To find the restaurant that matches traveller specified features

The state information consists of facts related to hotels and their services, restaurants and information about points of interest. The initial task network consists of one task that specifies choices regarding hotels, restaurants and points of interest as arguments.

5.3 Experimental Setup & Results

The results presented in [2] were based on original JSHOP2. Here, we modify the experimental setup in order to test the implemented extension over JSHOP2. The experiments have been performed on a 2.99 GHz core 2 duo Intel processor with 2GB RAM machine. The problem description consists of 6 hotels having 7 different services (total 15 facts), information about 17 restaurants (17 facts) and information about 28 locations (28 facts). In experimental study, we have considered the following test scenarios.

- Experiment I: Order of PoIs and constraint
- Experiment II: The effect of initial available cash with constraint on PoIs.
- Experiment III: The effect of constraint over traveller's wallet.

Experiment I: In this experiment, we test the performance of the planner based on the setup where the traveller has specified the POIs in some order and placed constraint on a particular PoI that needs to be visited. All the PoIs are placed in order in the initial task network description. Figure 2 depicts the effect of selecting different PoIs and number of days to be planned. The x-axis plots the values of the position of the constrained PoI in the PoI list specified in the initial task network and y-axis plots the number of plan steps[5] for each constrained PoI. This experiment also presents the comparison of planner performance in case of one-day and two-day travel planning.

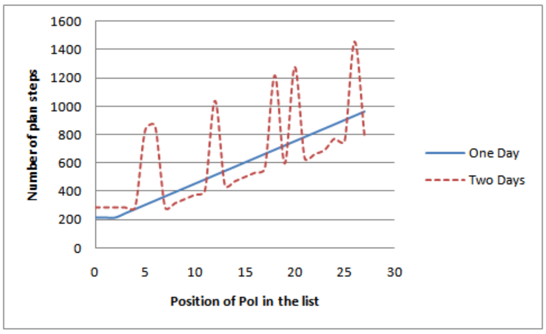


Figure 2. Effect of position of constrained PoI

From the specified list, the number of PoIs selected by the planner is limited by the number of days to be visited and available cash. It is to be noted that in both one-day and two-day plan the number of plan steps increases linearly with the relative position of the constrained PoI with notable spikes in two-day plan. The explanations behind the observations are as follows.

- Linearity: To include the constrained PoI in the plan, the planner needs to backtrack, remove the PoI that was selected last and include the constrained PoI. This takes same amount of steps to be performed for each PoI preceding the constrained PoI. This attributes to the linearity of the plan step size growth.
- Spikes: A spike is observed when a PoI with high cost is selected as constrained PoI. To satisfy the budget, the planner needs to remove more than one PoIs or select some other PoIs to make room for the constrained PoI. Hence the amount of backtracking increases non-linearly.

[5] The total number of steps including backtracking steps required by the planner

Experiment II: In this experiment, we study the effect of initial cash on the number of plan steps. The planner was asked to generate plan for two-day trip with constraint on one particular PoI to be visited. Figure 3 presents the variation of number of plan steps with initial cash available to the traveller.

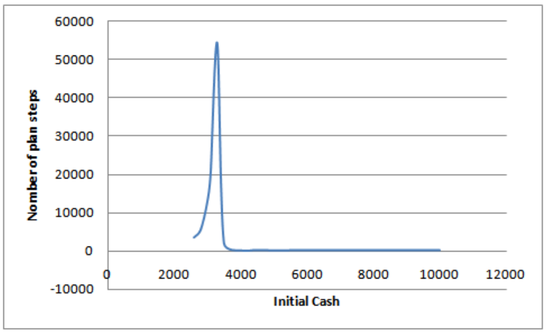


Figure 3. Effect of initial cash and constrained PoI

Similar experiment has been performed on the planner without any constraints. The variation was similar as in Figure 3 with minor decrease in number of plan steps.

In both the cases, no plan was getting generated below a certain limit on initial cash. After that the number of plan steps was increasing in rapid rate with increasing initial cash upto a certain limit and was decreasing again with increasing cash.

Experiment III: In this experiment, we put constraint like 'the traveller should always have greater than 3000 amount of cash available'. With this constraint, the variation in the number of plan steps with initial cash has been studied and presented in Figure 4.

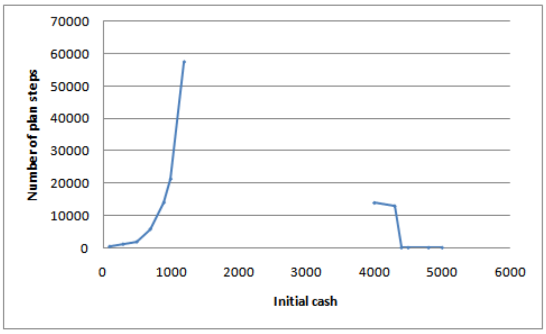


Figure 4. Effect of constraint over traveller's wallet with 'always' modal operator

It is noticed that for smaller and larger amount of cash the planner has been able to generate plan within reasonable time. However, for moderate amount of cash (900 – 4000) planner failed to generate plan with the given computing power. This is due to huge amount of backtracking in case moderate initial cash.

5.4 Avoiding some Pitfalls

In experiment III, the planner should indicate that the constraint is violated for initial cash less than 3000. Instead, the planner backtracks enormously with different available options. A close look at the automata for the modal operators reveals an interesting state of the automata which we call as *dead state*. A dead state is defined as the run phase of the automata in which the automata is in a state that does not have any outgoing transition. For example, S_2 is a dead state of the automata for *always* operator (Figure 5).

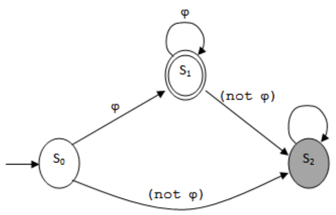


Figure 5. Dead state of *always* automata

For any value less than 3000 the automata will be in a dead state and will never reach the accepting state. Consequently, the acceptance check operator `!finish` will fail and the planner will backtrack with other instantiations of variables, operators and methods.

This huge amount of backtracking can be avoided by making the planner aware of the dead state. Whenever an automata is in a dead state `(dead-C)` predicate is added to the aggregated state information. The precondition of the `!finish` operator for a constraint `C` is changed to `(or (accepting-C)(dead-C))`.

The improvement in performance of the planner after the inclusion of dead state is shown in Figure 6.

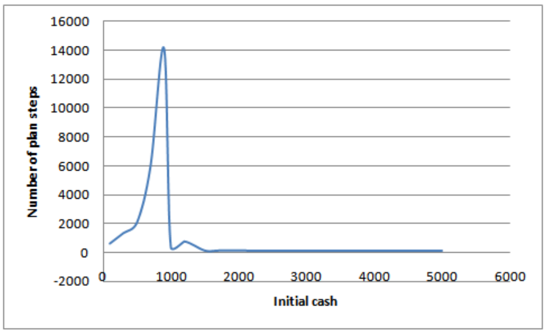


Figure 6. Performance improvement due to inclusion of dead state

6 Discussion

In this paper, we have adopted a domain recompilation-based technique towards extending HTN planner for handling temporal constraints. The constraints are processed by representing them into automata corresponding to the modal operators used to express them. The resulting automata have been embedded into the original planning domain thus generating new domain and problem descriptions that the original planner can consume without the knowledge of the existence of any constraints. We have tested different aspects of the constraint-based HTN planner with a city tour and travel domain. Different issues and limitations of the proposed extension are discussed below.

- The automata for the modal operators have been specified through generic templates. This limits the scope of expressive power of the constraint specification. The inclusion of expressions involving nested modal constructs calls for automatic translation of constraint expressions into automata.
- The present extension deals with only untimed (LTL) category of the modal operators where the automata template-based constraint processing is feasible. However, the constraint expression involving the timed modal operators cannot be captured through the current scheme.

We will take up the above mentioned issues as our future research challenges.

REFERENCES

[1] Jorge A. Baier, Fahiem Bacchus, and Sheila A. McIlraith, 'A heuristic search approach to planning with temporally extended preferences', *Artificial Intelligence*, **173**(5-6), 593–618, (April 2009).
[2] Plaban Kumar Bhowmick, Debnath Mukherjee, Prateep Misra, and Arunasish Sen, 'A study on applicability of hierarchical task network planner in smart city scenario: A smart city tour and travel planning use case', in *27th International Conference on Computers and their Applications (CATA-2012)*, pp. 252–257, (2012).
[3] Stefan Edelkamp, Shahid Jabbar, and Mohmmed Nazih, 'Cost-Optimal Planning with Constraints and Preferences in Large State Spaces', in *In International Conference on Automated Planning and Scheduling (ICAPS) Workshop on Preferences and Soft Constraints in Planning*, pp. 38–45, (2006).
[4] Kutluhan Erol, James Hendler, and Dana S. Nau, 'UMCP: A sound and complete procedure for hierarchical task-network planning', in *International Conference on AI Planning Systems (AIPS)*, pp. 249–254, (June 1994).
[5] Alfonso E. Gerevini, Patrik Haslum, Derek Long, Alessandro Saetti, and Yannis Dimopoulos, 'Deterministic planning in the fifth international planning competition: Pddl3 and experimental evaluation of the planners', *Artificial Intelligence*, **173**(5-6), 619–668, (April 2009).
[6] Chih-Wei Hsu, Benjamin W. Wah, Ruoyun Huang, and Yixin Chen, 'Constraint partitioning for solving planning problems with trajectory constraints and goal preferences', in *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pp. 1924–1929, San Francisco, CA, USA, (2007). Morgan Kaufmann Publishers Inc.
[7] Okhtay Ilghami and Dana S. Nau, 'A General Approach to Synthesize Problem-Specific Planners', Technical Report CS-TR-4597, UMIACS-TR-2004-40, University of Maryland, (2003).
[8] Naiwen Lin, Ugur Kuter, and Evren Sirin, 'Web service composition with user preferences', in *Proceedings of the 5th European semantic web conference on The semantic web: research and applications*, ESWC'08, pp. 629–643, Berlin, Heidelberg, (2008). Springer-Verlag.
[9] Alexander Nareyek, Eugene C. Freuder, Robert Fourer, Enrico Giunchiglia, Robert P. Goldman, Henry Kautz, Jussi Rintanen, and Austin Tate, 'Constraints and ai planning', *IEEE Intelligent Systems*, **20**(2), 62–72, (March 2005).
[10] Dana Nau, Tsz-Chiu Au, Okhtay Ilghami, Ugur Kuter, J. William Murdock, Dan Wu, and Fusun Yaman, 'Shop2: an htn planning system', *J. Artif. Int. Res.*, **20**(1), 379–404, (December 2003).
[11] Shirin Sohrabi, Jorge A. Baier, and Sheila A. McIlraith, 'Htn planning with preferences', in *Proceedings of the 21st international jont conference on Artifical intelligence*, IJCAI'09, pp. 1790–1797, San Francisco, CA, USA, (2009). Morgan Kaufmann Publishers Inc.

A Collaborative Model for Participatory Load Management in the Smart Grid

Matteo Vasirani[1] and **Sascha Ossowski**[2]

Abstract. The major task of a grid operator is perfectly balancing the demand of all customers at any instant with supply. One of the facets of the Smart Grid vision is tackling the problem of balancing demand with supply using strategies that act on the demand side. With the deployment of intelligent ICT devices in domestic environments, homes are becoming smarter and able to optimise the electricity consumption to minimise costs and/or meet supply constraints. In this work, we envision a scenario where smart homes actively participate in the balancing of demand with supply by forming groups of electricity consumers that agree on a joint demand profile to be contracted with a retail electricity provider. We put forward a novel business model as well as an optimisation model for collaborative load management, showing the economic benefits for the participants.

1 INTRODUCTION

A power system needs to perfectly balance at any instant the demand of all customers with supply in order to keep voltage and frequency stable and guarantee a safe functioning of the system. This task is carried out by the grid operator. The traditional approach is intervening from the supply side, by increasing or decreasing the supply to continuously match demand. Base load demand (i.e., the amount of electricity required on a continuous basis) is usually covered by power stations with low generation costs, but long start-up times. These power stations are therefore not able to quickly adjust their generation capacity to match unexpected peak load demand. Balancing power is therefore provided by expensive and carbon-intensive power plants, which are responsible for most part of consumer electricity bill.

One of the facets of the Smart Grid vision is tackling the problem of balancing demand with supply using strategies that act on the demand side [6][7]. For instance, the grid operator may use demand dispatch schemes that remotely turn (industrial) intensive loads off for a limited period of time in order to reduce demand. Also peak-shaving strategies, such as real-time pricing, may be used to encourage off-peak consumption, thus flattening demand [1].

All these strategies do not conceive an active and participatory role for the consumers. With the deployment of intelligent ICT devices in domestic environments, homes are becoming smarter and able to optimise the electricity consumption to minimise costs and/or meet supply constraints [8]. The participation of consumers into the management of demand is quite a recent line of research. For instance, Vinyals *et al.* proposed the formation of coalitions among energy consumers with near-complementary consumption restrictions [9]. In this work, a coalition of consumers can act in the market as a single virtual energy consumer (VEC), buying electricity directly from the day-ahead market and the future market. The experimental results show that the coalition of consumers obtains noticeable gains if the (average) price of electricity in the future market is half the (average) price of electricity in the day-ahead market, while using realistic prices the gains account only for less than 2%. Nevertheless, there is a growing consensus towards a more active role for the consumers.

In this work, we envision a business model where smart homes actively participate in the balancing of demand with supply by forming groups of electricity consumers that agree on a joint demand profile to be contracted with a retail electricity provider. By doing so, the consumers are able to get better prices from the retail electricity provider, since the management of balancing the demand with the contracted supply (and the eventual penalties) is responsibility of the consumers.

This paper is structured as follows: Section 2 presents our optimisation model and the computation of payments and penalties; Section 3 defines the scenario used for the evaluation of the model; in Section 4 the experimental results are reported; finally we conclude in Section 5.

2 COLLABORATIVE LOAD MANAGEMENT MODEL

This work envisions a scenario such as that depicted in Figure 1. Let $\mathcal{H}$ be a set of smart homes, represented by an aggregator, which interacts with a retail electricity provider (REP) to contract power on a daily basis. On a given day, each smart home is assumed to estimate its power consumption for the next day. Provided with the data of each home, the aggregator optimises the energy consumption of the whole group of smart homes and purchases the power to be delivered the next day from the REP.

2.1 Consumer load

We classify loads into two categories: those that can be shifted in time and those that cannot. The sum of the power consumption of the latter type of loads forms the *base load*, while all the others loads are individually modelled as *shiftable loads*. Each consumer has exactly one base load and several shiftable loads. Shiftable loads are further classified in loads that can be interrupted and resumed (*shiftable interruptible loads*) and loads that can be shifted but once they start they cannot be interrupted (*shiftable atomic loads*)[3]. Let $\mathcal{S} = \mathcal{I} \cup \mathcal{A}$

[1] University Rey Juan Carlos, Spain, email: matteo.vasirani@urjc.es
[2] University Rey Juan Carlos, Spain, email: sascha.ossowski@urjc.es

[3] Examples of shiftable interruptible loads are plug-in (hybrid) electric vehicles, or heating/air conditioning (AC) devices, which can be switched on and off while maintaining the temperature between the desired limits. Examples of shiftable atomic loads are washing machines or dryers.

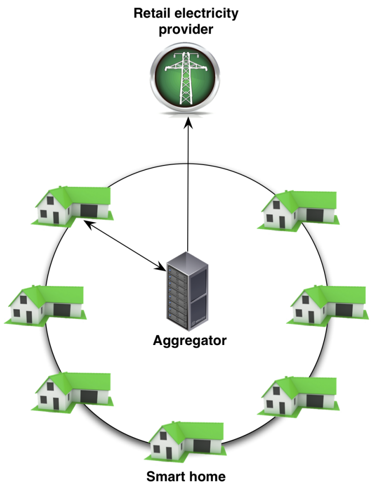


Figure 1. Collaborative load management scenario

be the set of shiftable loads of a consumer, where $\mathcal{I}$ is the set of shiftable loads that can be interrupted and resumed, while $\mathcal{A}$ is the set of shiftable atomic loads.

Definition 1: base load
The base load is defined as:

$$\mathbf{w}^B = [w_1^B \; w_2^B \; \dots \; w_N^B]^T$$

where $\mathcal{T} = \{1, \dots, N\}$ *is the number of time slots in a day, and* $w_t^B \in \mathbb{R}^+$ *is the base load power (expressed in kW) for time slot t.*

Definition 2: shiftable interruptible load
A shiftable interruptible load is defined as:

$$\mathbf{x}^{S_I} = [x_1^{S_I} \; x_2^{S_I} \; \dots \; x_N^{S_I}]^T, W^{S_I}, d^{S_I}, t_s^{S_I}, t_f^{S_I}$$

where $x_t^{S_I} \in \{0, W^{S_I}\}$ *is the load power for time slot t,* W^{S_I} *is the power rate (expressed in kW) of the load,* $d^{S_I} \in \{1, \dots, |\mathcal{T}|\}$ *is the duration of the load,* $t_s^{S_I} \in \mathcal{T}$ *is the earliest time slot for the load to start, and* $t_f^{S_I} \in \mathcal{T}$ *is the latest time slot for the load to finish,*

subject to

$$t_f^{S_I} - t_s^{S_I} + 1 \geq d^{S_I} \tag{1}$$

$$\sum_{t \in \mathcal{T}} x_t^{S_I} = d^{S_I} W^{S_I} \tag{2}$$

$$\sum_{\substack{t < t_s^{S_I} \\ t > t_f^{S_I}}} x_t^{S_I} = 0 \tag{3}$$

Constraint (1) ensures that the number of available time slots between the earliest time slot and the latest time slot is enough for the shiftable load to run for its entire duration d^{S_I}. Constraint (2) ensures that the shiftable load runs for d^{S_I} time slots. Constraint (3) prevents the shiftable load from running before the earliest time slot $t_s^{S_I}$ or after the latest time slot $t_f^{S_I}$.

Definition 3: shiftable atomic load
A shiftable atomic load is defined as:

$$\mathbf{x}^{S_A} = [x_1^{S_A} \; x_2^{S_A} \; \dots \; x_N^{S_A}]^T, W^{S_A}, d^{S_A}, t_s^{S_A}, t_f^{S_A}$$

where $x_t^{S_A} \in \{0, W^{S_A}\}$ *is the load power for time slot t,* W^{S_A} *is the power rate (expressed in kW) of the load,* $d^{S_A} \in \{1, \dots, |\mathcal{T}|\}$ *is the duration of the load,* $t_s^{S_A} \in \mathcal{T}$ *is the earliest time slot for the load to start, and* $t_f^{S_A} \in \mathcal{T}$ *is the latest time slot for the load to finish,*

subject to Constraints (1), (2), (3) and

$$x_t^{S_A} + x_{t+n}^{S_A} \leq W^{S_A} + x_{t+1}^{S_A} \tag{4}$$

$$\forall n \in \{2, \dots, N-1\}, \; \forall t \in \{t_s^{S_A}, \dots, t_f^{S_A} - n\}$$

Constraint 4 ensures that there exists a set of d^{S_A} consecutive slots when the load is running[4].

Definition 4: overall shiftable loads
We define the overall shiftable loads vector $\mathbf{x}$ *as:*

$$\mathbf{x} = [x_1 \; x_2 \; \dots \; x_N]^T = \tag{5}$$

$$= \left[\left(\sum_{S_k \in \mathcal{S}} x_1^{S_k} \right) \left(\sum_{S_k \in \mathcal{S}} x_2^{S_k} \right) \dots \left(\sum_{S_k \in \mathcal{S}} x_N^{S_k} \right) \right]^T$$

2.2 Joint load optimisation

The economic model for participatory load management that we proposed is based on two components: energy and power. The group of smart homes, represented by the aggregator, must pay the REP for the purchased energy as well as for the power capacity that is needed by the smart homes.

Let $\mathbf{p}^e = [p_1^e \; p_2^e \; \dots \; p_N^e]^T$ be the price of electricity (expressed in €/kWh) defined by the REP to supply energy over the N time slots. Let p^c be the price that is charged for the required capacity (expressed in €/kW). The goal of the aggregator is defining the consumer load of each smart home so as to minimise the total cost. This goal is defined by the following optimisation problem[5]:

$$\underset{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^m, x^c}{\text{minimise}} : \sum_{t \in \mathcal{T}} p_t^e \Delta t \sum_{i \in \mathcal{H}} \left(w_t^{B^i} + \sum_{S_k^i \in \mathcal{S}^i} x_t^{S_k^i} \right) + p^c x^c \tag{6}$$

subject to

$$x^c \geq \sum_{i \in \mathcal{H}} \left(w_t^{B^i} + \sum_{S_k^i \in \mathcal{S}^i} x_t^{S_k^i} \right), \; \forall t \in \mathcal{T} \tag{7}$$

where Δt is the duration of a time slot. Constraint (7) sets the variable x^c to the peak power consumption of the group of smart home throughout the set of time steps, which represents the required power capacity.

[4] Less formally, Constraint 4 ensures that if two slots are equal to W^{S_A}, then there is no slot in between that is equal to 0.

[5] The problem described in Eq. 6 can be modelled as a standard mixed integer linear programming problem, which has been solved with IBM ILOG CPLEX 11.0

2.3 Computing day-ahead payments

The value of the objective function described in Eq. 6 is the total cost $c(\mathcal{H})$ incurred by the group of smart homes $\mathcal{H}$. This cost must be shared among the participants. To do that, we have to model what would be the cost that an individual home with the same demand would pay if it did not participate in the group. In this case, we assume a situation where electricity is paid for at a fixed per unit price p^{fix} (expressed in €/kWh), as it happens with the regulated tariffs currently used in most countries. In this case, there is no need to defer loads, since the priced of electricity is fixed.

Let $\widehat{\mathbf{w}}^i = [\widehat{w}_1^i \; \widehat{w}_2^i \; \dots \; \widehat{w}_N^i]^T$ be the consumer load vector such that:

$$\widehat{w}_t^i = \begin{cases} w_t^{B^i} + \sum\limits_{S_k^i \in \mathcal{S}^i} W^{S_k^i} & \text{if } t \in \{t_s^{S_k^i}, \dots, t_s^{S_k^i} + d^{S_k^i} - 1\} \\ w_t^{B^i} & \text{otherwise} \end{cases} \tag{8}$$

Constraint (8) ensures that each shiftable load of i is executed at the earliest time slot without interruption. The load vector $\widehat{\mathbf{w}}^i$ could then be considered as the preferred load of a home that does not join the collaborative group. Let $c(\{i\})$ be the cost incurred by smart home i for demanding the load vector $\widehat{\mathbf{w}}^i$:

$$c(\{i\}) = \sum_{t \in \mathcal{T}} p^{\text{fix}} \Delta t \, \widehat{w}_t \tag{9}$$

The task of the aggregator is therefore defining a vector of payments $\mathbf{z} = [z^1 \; z^2 \; \dots \; z^m]^T$, such that:

$$\sum_{i \in \mathcal{H}} z^i = c(\mathcal{H}) \tag{10}$$

$$z^i \leq c(\{i\}), \; \forall i \in \mathcal{H} \tag{11}$$

Constraint (10) ensures that the sum of the payments equal the total cost. Constraint (11) is needed to satisfy the *individual rationality* condition (i.e., a smart home will not join the group if the cost of doing that is greater than the cost of acting on its own).

In cooperative game theory, the set of all the vectors $\mathbf{z}$ that satisfy constraints (10) and (11) is a solution concept called Core. We remark that in this model we assume that there is no discomfort cost derived from running a shiftable load over any set of d^S time slots between t_s^S and t_f^S. In fact, if a consumer is sensitive to discomfort, they may impose $t_f^S = t_s^S + d^S - 1$, so that the load cannot be shifted at all.

2.4 Computing imbalance penalties

Once the aggregator has solved the optimisation problem described in Section 2.2, it contracts with the retail electricity provider a certain power profile $\mathbf{w}^{\text{buy}} = [w_1^{\text{buy}} \; w_2^{\text{buy}} \; \dots \; w_N^{\text{buy}}]^T$, where:

$$w_t^{\text{buy}} = \sum_{i \in \mathcal{H}} \left(w_t^{B^i} + \sum_{S_k^i \in \mathcal{S}^i} x_t^{S_k^i} \right) \tag{12}$$

The group of smart homes is therefore committed to consume exactly the contracted amount of power. However, on the day of the delivery of the contracted power, it is possible that the real consumption differs from the contracted one. Let $\mathbf{w}^{\text{real}} = [w_1^{B^i_{\text{real}}} \; w_2^{B^i_{\text{real}}} \; \dots \; w_N^{B^i_{\text{real}}}]^T$ be the real base load of a smart home during the day of the delivery. In this work we assume that only the base load may differ from the predicted one, since all the shiftable loads are scheduled automatically according to the optimal plan. The power consumption mismatch is therefore defined as:

$$\boldsymbol{\epsilon} = [\epsilon_1 \; \epsilon_2 \; \dots \; \epsilon_N]^T = \tag{13}$$

$$= \left[\left(w_1^{B^i_{\text{real}}} - w_1^{B^i} \right) \; \left(w_2^{B^i_{\text{real}}} - w_2^{B^i} \right) \; \dots \; \left(w_N^{B^i_{\text{real}}} - w_N^{B^i} \right) \right]^T$$

If $\epsilon_t > 0$, the smart home is in a short position (i.e., it has been contracted less power than what is needed), while if $\epsilon_t < 0$, the smart home is in a long position (i.e., it has been contracted more power than what is needed). In the first case, the aggregator may be required to buy the missing power in the balancing market, while in the second case the aggregator may be required to sell the excess power in the balancing market.

Let $\mathbf{p}^{\text{bal-up}} = [p_1^{\text{bal-up}} \; p_2^{\text{bal-up}} \; \dots \; p_N^{\text{bal-up}}]^T$ be the price of electricity for "balancing-up" adjustments (i.e., when more power must be purchased in the balancing market), and let $\mathbf{p}^{\text{bal-down}} = [p_1^{\text{bal-down}} \; p_2^{\text{bal-down}} \; \dots \; p_N^{\text{bal-down}}]^T$ be the price of electricity for "balancing-down" adjustments (i.e., when excess power must be sold in the balancing market). During the day of the delivery of electricity, each smart home must pay the aggregator, as imbalance penalty, the following amount:

$$\sum_{t \in \mathcal{T}} p_t^{\text{bal}} \Delta t |\epsilon_t| \tag{14}$$

$$p_t^{\text{bal}} = \begin{cases} p_t^{\text{bal-up}} & \text{if } \epsilon_t \geq 0 \\ p_t^{\text{bal-down}} & \text{if } \epsilon_t < 0 \end{cases}$$

The imbalance penalty is intended to incentivise smart homes to adhere to their contracted power, by better predicting the day-ahead consumption. We remark that the fact that a smart home pays the aggregator for its short (or long) position does not automatically imply that the aggregator on its turn will cover the position in the balancing market. For example, it is possible that a short position of a smart homes is cancelled out by a long position of another smart home, for the same amount of kW. Therefore, although both smart homes pay the aggregator for their mismatches, the aggregator is not required to buy or sell power in the balancing market. In this case, we assume that the aggregator keeps the money that has been paid as imbalance penalty by the two smart homes.

3 EVALUATION SCENARIO

We define the evaluation scenario as follows. The duration Δt of a time slot is 10 minutes, and the number of smart homes in $\mathcal{H}$ is 10. For reasons of computational complexity, we kept the number of homes relatively small in order to solve optimally the problem described in Eq. 6. In this work, we assume that $\mathbf{p}^e = (1+\alpha)\mathbf{p}^{\text{mkt}}$, where $\mathbf{p}^{\text{mkt}}$ is the price of electricity in the day-ahead electricity market and $\alpha > 0$ is a parameter that ensures a profit margin for the retail electricity provider. The price of electricity of the day-ahead ($\mathbf{p}^{\text{mkt}}$) and balancing ($\mathbf{p}^{\text{bal-up}}$ and $\mathbf{p}^{\text{bal-down}}$) markets are taken from the July 2012 and January 2012 bulletin of the Spanish market operator[6]. The capacity price p^c is set to 0.07 €/kW, which is the capacity price in Spain for power delivery greater than 15 kW.

Each smart home is equipped with a certain number of electric equipments, such as heaters, washing machines, plug-in electric vehicles, etc. The probability that a smart home has a particular electric

[6] http://www.omie.es

equipment has been obtained from a study of the Institute for Diversification and Energy Saving, in collaboration with Eurostat [5]. This study analysed the electricity consumption of the residential sector in Spain. Table 1 resumes, for each electric equipment, the associated probability of being present in a smart home. The type of load can be base (B), shiftable interruptable (I) or shiftable atomic (A). Although nowadays the penetration of the plug-in electric vehicle (EV) is negligible, we assume a scenario where 10% of households has a plug-in (hybrid) electric vehicle, which is the projected penetration by 2020 that is reported by many studies [3, 4]. For simplicity we also assume that the probability of an equipment of being present in a smart home is statistically independent of the presence of other equipments[7].

Once the electric equipments that are present in a smart home has been defined, it is necessary to instantiate the *predicted* base load and shiftable loads for the next day, used by the aggregator to define the optimal consumer load $\mathbf{w}$, and the *real* base load $\mathbf{w}^{\mathrm{real}}$ during the day of the delivery. These instantiations are based on an elaboration of the results of the INDEL project, carried out by the Spanish grid operator, which assessed the electric demand of the residential sector in Spain [2].

Table 1. Loads and probability ($p(load)$) of being present in a smart home.

Type of load	Load	$p(load)$
B	Water	0.2
B	Lighting	1
B	Kitchen	0.53
B	Fridge	1
B	Freezer	0.23
B	Oven	0.77
B	Microwave	0.9
B	TV	1
B	Desktop computer	0.52
B	Laptop computer	0.41
I	Heating	0.41
I	AC	0.49
I	EV	0.1
A	Washing machine	0.93
A	Dryer	0.28
A	Dishwasher	0.53

3.1 Base load

3.1.1 Water

The power demand of an electric water heater is characterised by high peaks of power at regular intervals. The typical consumption cycle is turning the water heater on for half an hour (or 3 time slots) every two hours between 0:00 and 18:00. Between 18:00 and 24:00 the interval between two consecutive half-hour of usage decreases to one hour. The reason of this functioning is that the heat loss are negligible when the consumer does not use hot water, while during intense usage (between 18:00 and 24:00) the equipment needs to heat water more frequently. The water heater contribution to the base load is modelled as follows. An initial time slot t is randomly selected from the set $\{1,\ldots,6\}$ (i.e., the first hour of the day). Starting from time slot t, the water heater is turned on at regular intervals for 3 consecutive time slots, consuming 1.2 kW for every time slot it is turned on. The regular interval is set to 12 time slots (i.e., 2 hours) between 0:00 and 18:00, and 6 time slots (i.e., 1 hour) between 18:00 and 24:00.

[7] In reality, this may not be the case. For example, usually the presence of a dryer is conditioned to the presence of the washing machine.

3.1.2 Lighting

The average power consumption of lighting in a working day is taken from [2], using for every time slot a normal distribution with mean equal to the average consumption and variance equal to 5% of the average.

3.1.3 Kitchen

The average power consumption of an electric kitchen in a working day is taken from [2]. The electric kitchen is not used at all until 6 in the morning. A normal distribution with mean equal to the average consumption and variance equal to 5% of the average is used to stochastically generate different power consumptions.

3.1.4 Fridge and freezer

The fridge and the freezer are two appliances that are always running at a constant power rate. For these two loads, we use a fixed power rate of 0.08 kW and 0.07 kW respectively.

3.1.5 Oven

According to the surveys collected in [2], when the electric oven is used it runs between 20 minutes and 1 hour, around 14:00 $\pm$ 1h (lunch time) and/or around 21:00 $\pm$ 1h (dinner time). The probability of using the oven at lunch time is 0.8, while the probability of using it at dinner time is 0.2. The oven is used on average 2 times a week, and its power rate is 1.2 kW.

3.1.6 Microwave

The microwave is used repeatedly throughout a day for short periods of time (10 minutes). Analysing the data of [2], the microwave is mainly used around 9:00 $\pm$ 1h, 11:00 $\pm$ 1h, 15:00 $\pm$ 1h and 22:00 $\pm$ 1h, with probability 0.12, 0.2, 0.25 and 0.43 respectively. The microwave is used every day, and its power rate is 1.3 kW.

3.1.7 TV, desktop computer and laptop computer

The study carried out in [2] did not analyse the usage of TVs, desktop computers or laptop computers. In this works we assume that each device is used twice a day, at 14:00 $\pm$ 1h and at 20:00 $\pm$ 1h, and each usage takes between 1 and 3 hours. The power rate of the TV, the desktop computer and the laptop computer is set to 0.01, 0.1 and 0.02 kW respectively.

3.2 Shiftable interruptible loads

3.2.1 Heating

The power demand of an electric heating system with a thermostat is characterised by high peaks of active power. A typical heater is usually off before 8:00 in the morning and after 23:00 in the night. Between 8:00 and 23:00 the heater is turned on for a total of 3.5 hours. Although the functioning of the heater depends on the number of people inside the home, the external weather conditions and the thermal leakage of the home, in this work we rely on the typical power consumption reported in [2]. A heater must be on for 10 minutes in every hour (1 time slot out of 6) between 8:00 and 20:00, and for 30 minutes in every hour (3 time slot out of 6) between 20:00 and 23:00. Thus between 8:00 and 20:00 a heater is on for 2 hours

Table 2. Experimental results

	w/o load mgmt. (€)	REP margin	with load mgmt. (€)	gain (€)
Winter	50.65 ± 1.39	$\alpha = 0.1$	32.04 ± 1.79	18.60
		$\alpha = 0.3$	35.97 ± 2.51	14.68
		$\alpha = 0.5$	43.91 ± 2.28	6.74
		$\alpha = 0.7$	45.18 ± 2.76	5.46
Summer	43.02 ± 3.46	$\alpha = 0.1$	16.37 ± 1.16	26.63
		$\alpha = 0.3$	32.31 ± 2.55	10.69
		$\alpha = 0.5$	40.88 ± 2.86	2.11
		$\alpha = 0.7$	42.56 ± 2.68	0.44

of the total usage of 3.5 hours, and the remaining 1.5 hours is placed between 20:00 and 23:00. Each smart home is equipped with 1 to 3 heaters, each of them with a power consumption of 1 kW.

On the basis of the aforementioned assumption, we instantiated in our model the load of each heater as follows. For each heater we define 15 shiftable loads, representing each hour τ between 8:00 and 22:00 inclusive. Each load $\mathbf{w}^{S_{H_\tau}}$ is characterised by a power rate of $W^{S_{H_\tau}} = 1$kW, a earliest time slot $t_s^{S_{H_\tau}} = 6\tau + 1$, a latest time slot $t_f^{S_{H_\tau}} = 6\tau + 6$, and a duration equal to:

$$d^{S_{H_\tau}} = \begin{cases} 1 & \text{if } 8{:}00 \le \tau \le 20{:}00 \\ 3 & \text{if } 20{:}00 < \tau \le 22{:}00 \end{cases} \tag{15}$$

3.2.2 *AC*

A typical AC system is turned on for a certain amount of time between 13:00 and 18:00, consuming an amount of energy that varies between 1.6 and 5.6 kWh per day. Similarly to a heating system, the power consumption of an AC system depends on environmental conditions. However, for simplicity we define the load $\mathbf{w}^{S_{AC}}$ of an AC system as follows. The earliest time slot is set to $t_s^{S_{AC}} = 6 \cdot 13 + 1$, while the latest time slot is set to $t_f^{S_{AC}} = 6 \cdot 18 + 6$. We then draw the amount of energy e that is consumed from a uniform distribution over the interval $[1.6, 5.6]$ kWh. Given the power rate $W^{S_{AC}} = 1.5$ kW, the number of time slots when the AC is running is therefore defined as:

$$d^{S_{AC}} = \frac{e}{W^{S_{AC}} \Delta t} \tag{16}$$

3.2.3 *EV*

In this work we assume that a plug-in electric vehicle uses Level 1 charging, with a power rate of $W^{S_{EV}} = 1.92$ kW. We assume that the EV owner arrives at home at 19:00 ± 1h, and needs the EV charged at 8:00 ± 1h of the next day. We assume a battery size of 24kWh (such as that of the Nissan Leaf), and state of charge SOC at the time the EV is plugged-in uniformly distributed between $[0.3, 0.8]$. Since the charging is spread over two consecutive days, for every EV we instantiate two shiftable interruptible loads: one for the charging between the arrival time and 24:00 ($\mathbf{w}^{S_{EV_1}}$) and one for the charging between 24:00 and the departure time ($\mathbf{w}^{S_{EV_2}}$). For $\mathbf{w}^{S_{EV_1}}$, the earliest time slot $t_s^{S_{EV_1}}$ is equal to the time slot corresponding to the arrival time, while the latest time slot $t_f^{S_{EV_1}}$ is equal to N (i.e., the last time slot in $\mathcal{T}$). For $\mathbf{w}^{S_{EV_2}}$, the earliest time slot $t_s^{S_{EV_2}}$ is equal to 1, while $t_f^{S_{EV_2}}$ is equal to the time slot corresponding to the departure time. For the definition of the two durations, $d^{S_{EV_1}}$ and $d^{S_{EV_2}}$, we use the following heuristic. Let $k_1 = 144 - t_s^{S_{EV_1}} + 1$ be the number of time slots between the arrival time and 24:00, and let $k_2 = t_f^{S_{EV_2}}$ be the number of time slots between the 0:00 of the next day and the departure time. Given that the amount of energy needed by the EV is $e = 24(1 - SOC)$ kWh, the EV tries to charge $e_1 = ek_1/(k_1 + k_2)$ kWh between the arrival time and 24:00, and the remaining $e_2 = ek_2/(k_1 + k_2)$ between 0:00 and the departure time. The durations of the two loads are therefore:

$$d^{S_{EV_1}} = \frac{e_1}{W^{S_{EV}} \Delta t} \quad d^{S_{EV_2}} = \frac{e_2}{W^{S_{EV}} \Delta t} \tag{17}$$

3.3 Shiftable atomic loads

3.3.1 *Washing machine*

According to the study we use as a reference [2], the washing machine is run on average 3 times a week. The earliest time slot $t_s^{S_{WM}}$ is set at 11:00 ± 1h (with probability 0.78) or at 19:00 ± 1h (with probability 0.22). In this work we assume that the latest time slot $t_f^{S_{WM}}$ is set at 15:00 ± 1h (if the washing machine is run in the morning), or at 23:00 ± 1h (if the washing machine is run in the evening). A typical washing machine operates for one to two hours at a power rate $W^{S_{WM}}$ of 0.19 kW. The duration $d^{S_{WM}}$ (in time slots) is therefore drawn uniformly from the set $\{6\ldots,12\}$ (from 1 to 2 hours).

3.3.2 *Dryer*

The smart homes that have a dryer installed are assumed to run this device 3 times a week on average. The earliest time slot $t_s^{S_D}$ is set at 17:00 ± 1h, while the latest time slot $t_f^{S_D}$ is set at 21:00 ± 1h. A typical dryer operates at a power rate W^{S_D} of 1.24 kW, while the duration d^{S_D} (in time slots) is drawn uniformly from the set $\{6\ldots,9\}$ (1 to 1.5 hours).

3.3.3 *Dishwasher*

A dishwasher is run on average 4 times per week, either at 15:00 ± 1h (with probability 0.5) or at 19:00 ± 1h (with probability 0.5). Here we assume that the latest time slot $t_f^{S_{DW}}$ is set at 19:00 ± 1h (if the dishwasher is run in the afternoon), or at 23:00 ± 1h (if the dishwasher is run in the night). A typical dishwasher operates at a power rate $W^{S_{DW}}$ of 0.66 kW. The duration $d^{S_{DW}}$ (in time slots) is drawn uniformly from the set $\{6\ldots,12\}$ (from 1 to 2 hours).

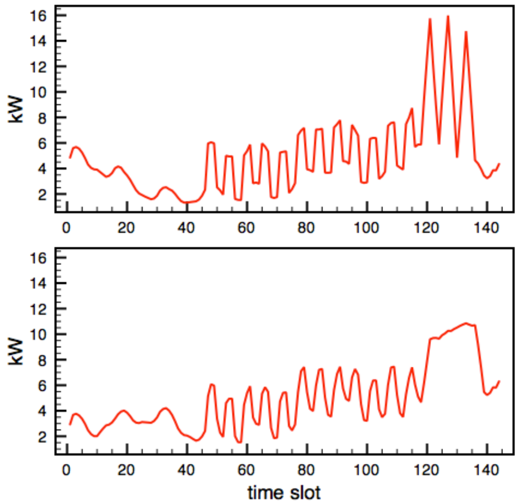


Figure 2. Load profile with (bottom) and without (top) load management

4 EXPERIMENTAL RESULTS

We compute the average monthly payment that an individual home will pay if it does not participate in a load management collaborative group. Then, depending on the REP's profit margin (α), we compute the average monthly payment of an individual home that participates in a load management collaborative group. The difference between the payments without and with load management gives us the monthly monetary gain of a individual home. Table 2 shows the results of the experimental simulations. In winter, the average monthly payment without load management is about 50€. With participatory load management, an individual home is able to save 18€ per month, when REP's margin over the spot price of electricity is 10% ($\alpha = 0.1$). For bigger profit margins, of course the advantages of a load management scheme decrease, and the gain falls to 5€ per month, when REP's profit margin is 70% ($\alpha = 0.7$). In summer, the average monthly payment without load management is 43€, slightly lower than winter's payment. The gain obtained from participating in a load management group spans from about 26€ per month ($\alpha = 0.1$) to about 0€ when REP's profit margin very high ($\alpha = 0.7$).

We are also interested in computing the gains that the aggregator obtains. We define the daily gain of the aggregator as the difference between the imbalance penalties paid by the smart homes to the aggregator during the day of delivery and the net financial position of the aggregator after covering short and long positions of the group of smart homes in the balancing markets (see Eq. 18).

$$G = \overbrace{\sum_{t \in \mathcal{T}} p_t^{\mathrm{bal}} \Delta t \sum_{i \in \mathcal{H}} |\epsilon_t^i|}^{\text{home}\rightarrow\text{aggregator}} - \overbrace{\sum_{t \in \mathcal{T}} p_t^{\mathrm{bal}} \Delta t \sum_{i \in \mathcal{H}} \epsilon_t^i}^{\substack{\text{aggregator}\rightarrow \\ \text{balancing market}}} \tag{18}$$

The monthly net financial position of the aggregator after after covering short and long positions in the balancing markets is on average negative, accounting for a loss of -5.79€ ± 0.52. Nevertheless, since smart homes pay the aggregator for their imbalances, even if they may cancel out and therefore do not require any buying or selling the balancing market, the average monthly gain of the aggregator is 38.12€ ± 6.4. Since the aggregator could be a mere coordinating entity with no profit maximising interests, this gain could be shared among the smart homes and therefore increase the benefit of each participant.

Figure 2 plots the winter load profile when the smart homes are organised in a collaborative group (bottom), compared to the same set of homes that do not participate in the load management (top). It is possible to appreciate how load management smooths the evening power peak, since power capacity is part of the cost function to be minimised (see Eq. 6). This fact does not only translate into lower costs for the consumers, but also lower installation costs for the grid operator, since less capacity is needed to serve the set of homes involved in the participatory load management scheme.

5 CONCLUSIONS

In this work, we put forward a model for participatory load management, where smart homes actively participate in the balancing of demand with supply by forming groups of electricity consumers that agree on a joint demand profile to be contracted with a REP. We defined an economic model where electricity is priced by the REP above the spot market price but below the fixed per unit price paid by conventional consumers. In this way the REP obtain a profit margin and it does not have to take care of balancing the demand of its consumers with supply, since it is direct responsibility of the collaborative group of smart homes. These homes, represented by an aggregator, optimise electricity consumption and power capacity, while trying to sticking to the contracted supply on the day of the delivery. The experimental evaluation shows that an individual smart home may gain up to 18€ per month (in winter) and up to 26€ per month (in summer). At the same time, by putting a price on the needed power capacity, the group of smart homes is able to shave the peak power consumption, thus reducing installation costs.

As future work, the complexity of the optimisation model must be tackled in order to increase scalability, either by distributing the optimisation or by means of meta-heuristics methods. Furthermore, more sophisticate techniques to model the stochasticity of the problem can be employed, such as agent-based simulations.

REFERENCES

[1] A. J. Conejo, J. M. Morales, and L. Baringo, 'Real-time demand response model', *IEEE Transactions on Smart Grid*, **1**(3), 236–242, (2010).

[2] *Atlas de la demanda eléctrica española*, ed., Red Eléctrica de España, REE, 1999.

[3] *Guía del Vehículo Eléctrico*, ed., Fundación de la Energía de la Comunidad de Madrid, FENERCOM, 2009.

[4] F. Nemry and M. Brons, 'Plug-in hybrid and battery electric vehicles. market penetration scenarios of electric drive vehicles', Technical report, JRC Technical Notes, (2010).

[5] *Análisis del consumo energéetico del sector residencial en España*, ed., Instituto para la Diversificación y Ahorro de la Energía, IDAE, 2011.

[6] G. Strbac, 'Demand side management: Benefits and challenges', *Energy Policy*, **36**(12), 4419–4426, (2008).

[7] C. Su and D. Kirschen, 'Quantifying the effect of demand response on electricity markets', *IEEE Transactions on Power Systems*, **24**(3), 1199–1207, (2009).

[8] S. L. Tompros, N. P. Mouratidis, M. Draaijer, A. Foglar, and H. Hrasnica, 'Enabling applicability of energy saving applications on the appliances of the home environment', *IEEE Network*, **23**(6), 8–16, (2009).

[9] M. Vinyals, F. Bistaffa, A. Farinelli, and A. Rogers, 'Stable coalition formation among energy consumers in the smart grid', in *Proceedings of the 3rd International Workshop on Agent Technologies for Energy Systems (ATES 2012)*, (2012).

Occupant Location Prediction Using Association Rule Mining

Conor Ryan and **Kenneth N. Brown**[1]

Abstract. HVAC systems are significant consumers of energy, however building management systems do not typically operate them in accordance with occupant movements. Due to the delayed response of HVAC systems, prediction of occupant locations is necessary to maximize energy efficiency. In this paper we present an approach to occupant location prediction based on association rule mining, which allows prediction based on historical occupant movements and any available real time information. We show how association rule mining can be adapted for occupant prediction and show the results of applying this approach on simulated and real occupants.

1 INTRODUCTION

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

We believe that in most office buildings, the behaviour of the occupants tends to be regular. The regularity may be based on the time of day, or particular days of the week or times of the year. Behaviour within a day may also be dependent on behaviour earlier in the day, or dependent on the behaviour of other associated individuals. We require a system which is able to recognise these time and feature based patterns across different levels of granularity from observed data. Further, many office users now use electronic calendars to manage their schedules, and information in these calendars may support or override the regular behaviour. The reliability of the calendar data will depend on the individual maintaining it, and so the prediction system needs to be able to learn occupant-specific patterns from the calendars.

We propose the use of association rule mining for learning individual occupant behaviour patterns, using the Apriori algorithm [2]. From the individual patterns, we then aggregate behaviour to produce bulk occupancy predictions. We show how the algorithm can be extended to represent time series, incorporating calendar entries. We then propose a number of transformations of the learning mechanism, pruning itemsets and rules to focus in on useful rules, and extending the generation of itemsets in areas where useful patterns will be found. We evaluate the performance empirically on both simulated and real observed data, and show a 14% increase in accuracy over the base algorithm, reaching up to 79% accuracy in predicting occupant locations on real data and up to 85% accuracy in predicting whether rooms are occupied on simulated data.

The remainder of the paper is organized as follows: Section 2 provides an overview of association rules and the existing work on location prediction. Section 3 details the modifications we make to the mining process. In Section 4 we outline the datasets we use for evaluation and present our results. We conclude the paper in Section 5.

2 RELATED WORK

2.1 Location Prediction

Existing methods for predicting occupant locations include bayesian networks[3], neural networks[4], state predictors[5], context predictors[6], eigenbehaviors [7].

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

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

[1] Cork Constraint Computation Centre, Department of Computer Science, University College Cork, Ireland.

The state predictor approach in [5] uses a two-level context predictor with two-state predictors. This method selects a two-state predictor based on the occupant's sequence of previous locations. Each state within the selected predictor is a prediction; the current state is used as the prediction, and the state may then change depending on whether the prediction was accurate. Being a two-state predictor, each possible location has two corresponding states, so a maximum of two incorrect predictions for any given sequence is necessary to change future predictions, resulting in fast retraining if an occupant changes their behavior.

The second level of this predictor can store the frequencies of the possible next locations for each sequence, instead of state predictors. This makes it equivalent to a markov model approach.

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

2.2 Association Rules

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

Let U be a universe of items. A dataset D is a set of instances $I_1 \dots I_n$, where each instance is a set of items from U. An itemset X is a subset of U. The frequency of X, $freq(X)$, is the number of instances I in D for which $X \subseteq I$, while the support is:

$$supp(x) = freq(X)/|D| \qquad (1)$$

An association rule is an implication of the form $X \Rightarrow Y$ where X and Y are itemsets such that $X \cap Y = \emptyset$. This rule states that each instance which contains X tends to contain Y. The support of the rule is $supp(X \cup Y)$. The confidence of the rule is how often it is correct as a fraction of how often it applies:

$$conf(X \Rightarrow Y) = supp(X \cup Y)/supp(X) \qquad (2)$$

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

Apriori uses breadth first search to find all frequent itemsets. First all itemsets of size 1 are enumerated. Itemsets whose support falls below the support threshold (infrequent itemsets) are removed, as any superset of an infrequent itemset will also be infrequent. Candidate itemsets of size 2 are then generated by combining all frequent itemsets of size 1, and infrequent itemsets of size 2 are removed. This process continues, finding frequent itemsets of size n by generating candidates from the itemsets of size n-1 and removing infrequent itemsets, until an n where no frequent itemsets exist is reached.

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

3 ADAPTING ASSOCIATION RULE MINING FOR OCCUPANT PREDICTION

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

Although this format is all that is needed to run Apriori, it is unlikely to produce usable results. The items in our dataset have semantics which are critical for the eventual application, but Apriori by default treats them all as equivalent.

The location attributes $l_i \dots l_j$ represent an ordered list of time/location pairs which it is our objective to predict. However, Apriori has no concept of the importance of or ordering over these items, so it will produce rules which run counter to the order, i.e. rules which use later locations to predict earlier locations, and which make useless predictions, e.g. predicting timetable entries.

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

A problem with regard to the content of the data is that the many common patterns tend to be the least interesting, while we require low frequency patterns to be found in order to make predictions in unusual circumstances. Consider for example an occupant who has a 90% chance of being in their office in any timeslot from 9am to 5pm. In this case, any pattern of the form "in at N implies in at M" where N and M are between 9-5 will have support of at least 80%, thus all such patterns will be found. But there is no real correlation there; all these patterns could be summarized simply as "the occupant is likely to be in". At the extreme opposite end, we have days when the occupant does not turn up at all, due to illness or other reasons – a very obvious pattern which would be represented by rules such as "out at 9,10,11 implies out at 12". Such rules could have confidence close to 100% if the occupant tends to be in in the morning, but if absences are rare the itemset behind the rule will have such low support it won't even be a candidate. Since enumerating every itemset is not feasible, we wish to eliminate the common uninteresting ones and focus on the less common but interesting ones.

3.1 Candidate/Rule Pruning

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

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

3.2 Support Modification

In 2.1 we provided the typical definition of support, the proportion of the instances which contain the itemset/rule. To deal with the reduction in support for itemsets which contain metadata items, we redefine support as follows:

$$supp(X) = freq(X)/max(freq(X)) \tag{3}$$

For market basket items, which can in principle occur in every instance, this is the same definition. In the case of our metadata attribute/value pairs however, this definition results in a different value which is normalized such that the maximum value of supp(X) is always 1 for comparison to other support values.

Using this modified support threshold in Apriori allows it to find itemsets when have a lower support due to their metadata attributes. However this greatly increases the area of the itemset lattice which is explored for any given support threshold. Thus, in order to conserve memory, we mine each possible combination in a separate pass. For every combination of metadata attributes/values C, we initialize Apriori with all itemsets of size $|C| + 1$ which are a superset of C, instead of standard 1-itemsets. This allows the generation of every itemset which contains that metadata combination in a separate pass.

3.3 Windowing

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

We define a window as: $Win(n,m) =< d, o, l_{n \dots n+m}, s_{i \dots j} >$ where i and j denote the first and last timeslots, and n and m denote the beginning and length of the window respectively. In the windowing phase, we search within every window of the chosen length. This approach ignores patterns which span times which do not fit within a window. We choose to focus on patterns which occur in consecutive time slots as predicting occupant locations based on their most recent movements has been shown to work by the other approaches discussed in 2.2.

For distinct patterns windowing is sufficient to find rules which will make the correct predictions should the pattern recur. Taking the example of an occupant who is absent all day, within each window we will learn that consecutive hours of absence imply absence in the next hour. Taken in combination, these rules will state that at any hour of the day, consecutive absence implies continued absence, although we are still not learning sequences in the same sense as the approaches in 2.2, as the individual rules are still tied to specific time slots. These rules are added to the rules mined from the complete instances.

3.4 Rule Selection

Once the rules are generated we need a mechanism to choose a rule to make a prediction. When a prediction is required, values for any subset of the possible attributes can be supplied in the form of an itemset V. A target for the prediction l_t is also given. We search the generated rules for all rules $X \Rightarrow Y$ where $X \subseteq V$ and $Y = \{l_t\}$. From these we select the rule with the highest confidence to make the prediction[2].

4 EXPERIMENTAL EVALUATION

To test our approach we use three different datasets: (i) data obtained from a simulator which generates occupancy patterns, (ii) data recorded by occupants of the 4C lab in UCC, and (iii) data from the Augsburg Indoor Location Tracking Benchmarks [8].

4.1 Occupancy Simulator

We have developed an occupancy simulator loosely based on the work of [9]. We model the activities which the occupants engage in during the day to determine their location. The model can also be viewed as a markov chain, where each state is a set of generated events and each transition adds a new event to the set.

[2] Confidence provided the highest accuracy over several evaluated metrics, though the difference was marginal

We have a set of agents A for whom we wish to generate locations over a period of time. To determine these locations we have a set of tasks T which can be assigned to agents. Each task has a set of attributes, enumerated below, which determine the agent's location for a period of time. To assign tasks to agents we have a set of roles R, each of which contains a set of tasks. Each agent then has a set of roles which apply to them. A role is chosen for each agent from their respective set at the beginning of each day based on a probability distribution over their possible roles.

Roles are defined as $R_i = < A_i, F_i, D_i, m_i, P_i, S_i, V_i >$ where A_i and $F_i$are sets of possible arrival and finish times outside which the agent is absent, D_i is a set of possible durations for the role, m_i is a flag indicating whether the agent may attend meetings in this role, and P_i, S_i, V_i are sets of tasks of different types.

There are three different types of task in order to represent different kinds of activities which can occur. These types differ in how the task's start time is selected and the order in which the tasks are evaluated:

Primary tasks are evaluated first: $P_i = < T_i, D_i, L_i, p_i >$ where T_i is a set of possible start times, D_i is a set of possible durations, L_i is a set of possible locations, and p_i is the base probability of the task occurring. Values are chosen from T_i, D_i and L_i based on associated probability distributions. A primary task will not occur if the agent is not available at the time chosen.

Secondary tasks are evaluated once all primary tasks are done: $S_i = < T_i, D_i, L_i, p_i >$ where the attributes are as above, except that times are selected from T_i in a set order until a free time slot is found. A secondary task will not occur if the agent is not free at any of the possible times.

For every empty slot which is left unassigned at the end of the process, a tertiary task is select to fill the slot: $V_i = < l_i >$.

Tertiary tasks are selected based on an associated probability distribution in the role. They have fixed duration and can appear multiple times, and so their only property is a location.

Primary tasks represent activities which must occur at a fixed time if they occur and take priority over other tasks, while secondary and tertiary tasks represent less important and miscellaneous activities which are scheduled around other tasks.

We also model meetings, which are essentially tasks involving multiple agents. Because they involve multiple agents, each meeting has a set of relevant agents rather than each agent having a set of meetings. We have two types of meetings:

Primary meetings are evaluated before all other tasks and meetings: $PM_i = < T_i, D_i, L_i, p_i, R_i, O_i >$. Primary meetings have the same attributes as primary tasks. In addition, they have a set of required agents R_i, whom all must be available for the meeting to occur, and a set of optional agents O_i, who will attend if possible.

Secondary meetings are evaluated after primary meetings and primary tasks: $SM_i = < T_i, D_i, L_i, p_i, A_i, m_i >$. Secondary meetings have the same properties as secondary tasks. In addition, they have a set of agents A_i a minimum number of available agents required for the meeting to occur m_i.

The model used to generate the data used in this evaluation includes 8 agents. 3 of these are lecturers, the remaining 5 are students. The lecturers each have a one person office, which they leave to give lectures and labs at fixed times with high probability. The students share an open plan office, which they leave to have one-on-one meetings with their supervisor (one of the lecturers). All agents also go to lunch, choosing whether to remain in the building or leave based on individual probabilities, attend a weekly group meeting which is dependent on certain agents being present to run it, and have a chance of missing an entire day due to illness. To simulate how a system using this approach would actually function, we use a contiguous dataset which is split such that there is 2 months of training data with the following 3 weeks as test data.

4.2 UCC Data

To gather data to test our approach, five occupants of the 4C lab in University College Cork including the authors manually recorded their movements over a period of 1-2 months using google calendar. Each occupant recorded their location by room code if within a campus building, or marked themselves as 'away' if off campus. The data was recorded from 8am to 6pm with half-hour granularity, with any occupancy of significantly shorter duration than 30 minutes filtered out. The occupants also recorded their timetables for the time period, which recorded the locations they were scheduled to be in in the same format as the record of their actual movements. 20 locations were frequented by the occupants including the 'away' location. The test set for this evaluation was the most recent two weeks of data for each occupant, while the training set was all the preceding data each occupant had recorded, which covered between 3 and 7 weeks.

4.3 Augsburg Dataset

This external dataset contains data on 4 occupants for 1-2 weeks in summer and 1-2 months in fall. The format of the dataset is a series of timestamped locations for each occupant. In order to be able to apply Apriori to the data, we converted it to the same timeslot format as our gathered data. An occupant's location in each timeslot is the location they spent the majority of that timeslot in according to the original data. Following this conversion there are 7 locations frequented by the occupants including 'away'.

4.4 Experiments

We generate rules from each training set using a minimum support and confidence of 0.2 and 0.5 respectively. During windowing we use a window size of 6 slots and a minimum support of 0.05. We evaluate the predictions of the association rules for each test set by predicting the location of each occupant at each time slot in the test set and recording the following statistics:

- Overall Accuracy: The percentage of predictions made which were correct
- Exact Occupancy: The percentage of room/time combinations for which the correct occupancy was exactly predicted
- Binary Occupancy: The percentage of room/time combinations for which the room was correctly predicted to be occupied or not

Occupancy level prediction accuracy is only available on the simulated dataset, as the collected dataset does not feature shared rooms, rendering occupant location prediction and occupancy level prediction essentially the same.

We test the association rules on their ability to predict with and without the timetable data available, and their ability to predict next-hour and next-day. The former determines whether the values of $s_i \dots s_j$ are available when predicting, and is marked 'no

Timetable' if they are not. The latter determines whether $l_i \ldots l_{n-1}$ are available, where n is the time slot being predicted, 'Next Hour' if this information is available, and 'Next Day' if it is not.

4.5 Results

4.5.1 *Algorithm Modification*

Table 1. Algorithm Evaluation Results (UCC Dataset)

	No Support Mod / NoWindowing	No Windowing	All
Overall Accuracy	65%	74%	79%

Table 1 shows accuracy gains due to the extra areas of the itemset lattice explored due to Windowing and Support Modification. Support Modification adds the most accuracy, as it allows searching for more occupant-specific patterns. Windowing provides a smaller boost by allowing us to find patterns which describe rarer events.

The predictor highlights any instances where less than 50% of time slots were predicted correctly as problem instances. The addition of Support Modification and Windowing reduce the number of these problem instances by approx. 66% on all datasets.

4.5.2 *Generated Data*

Table 2. Generated Data Evaluation Results

Prediction	Overall	Exact	Binary
Next-Hour	69%	67%	85%
Next-Day	66%	66%	83%
Next Hour (No Timetable)	69%	66%	85%
Next Day (No Timetable)	64%	64%	81%

Table 2 shows the prediction accuracy for the agents in the generated data. These agents have a high probability of attending their timetabled tasks and narrow windows within which they arrive at and leave the building. Due to this their overall behavior is quite predictable, resulting in only a 5% drop in accuracy when predicting next-day locations without timetable data compared to next-hour predictions. The variations in their behavior, such as missing tasks or entire days, are purely random and thus unpredictable, reducing the accuracy in all tests.

The exact and binary occupancy level accuracy values vary due to the open plan office which the students share. The exact occupancy is the same as or slightly lower than the overall accuracy as the occupancy for the open plan office can be predicted incorrectly if any one student is predicted incorrectly. The binary occupancy accuracy is significantly higher due to the high probability of the open plan office being occupied by at least one student coupled with the high probability of correctly predicting the presence of at least one student.

4.5.3 *UCC Data*

Table 3. UCC Data Evaluation Results

	Next Hour	Next Day	Next Hour (no Timetable)	Next Day (no Timetable)
Overall Accuracy	79%	66%	79%	65%

Table 3 shows higher overall accuracy on the UCC data than on the generated data, which may indicate that the regular habits of individual people are less random than the probability model in our simulator. The drop in accuracy for next-day predictions compared to next-hour predictions shows that more intelligent predictions of the real occupants' movements later in the day can be made based on their movements earlier on, if that information is available.

Figure 1 shows the overall accuracy across the day. The next day and next hour predictions are of equal accuracy at the beginning and end of the day and at lunch time, as these are the times at which people's movements are most reliable.

Outside these times, data on the occupant's movements that day are needed to make accurate predictions. The next-day predictions drop in accuracy at 14:00. This is due to the system being unable to make a prediction for 14:00 without information on earlier movements, indicating that relevant rules based solely on historical movements at that time were below the confidence threshold.

Figure 2 shows per-occupant overall accuracy for next-hour and next-day predictions. In all cases next-day predictions are equal or worse than next-hour predictions, but the degree varies between the agents. One occupant is predictable enough that the overall accuracy is equal, though this does not necessarily mean that exactly the same predictions were made in both tests.

The availability of timetable data makes no difference to the accuracy of the predictions in either case. In general the timetable entries in this dataset are weekly meetings. Without the presence of irregularly scheduled meetings no extra useful correlations can be found. Furthermore meetings can be cancelled, relocated or rescheduled without sufficient notice to update the timetable, or the occupant can be absent, all of which will reduce the reliability of the timetable-related rules.

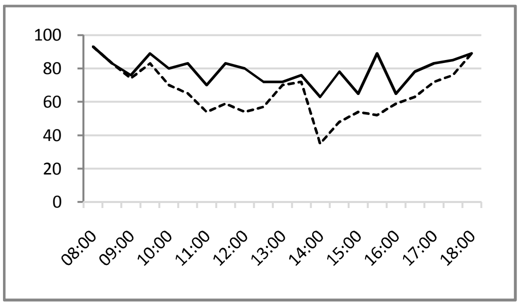


Figure 1. Prediction accuracy across the day for next-hour (solid) and next-day (dashed) predictions

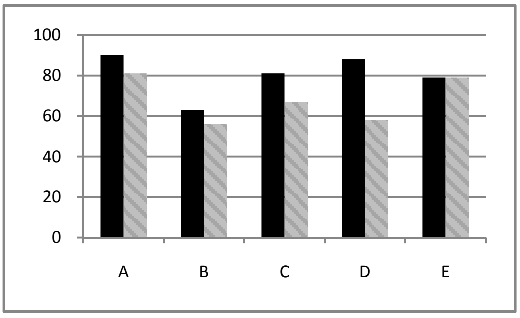


Figure 2. Prediction accuracy by occupant for next-hour (solid) and next-day (dashed) predictions

A recurring pattern in the selection of rules is the use of timetable entries to predict movements before the meeting, and the use of those predicted movements to predict attendance at the meeting. This does not help accuracy on next-day predictions, as in these cases the meetings are always scheduled every week and thus do not allow discrimination between different patterns of movement. However, this does show that in cases where the meeting may or may not be scheduled, the timetable data could improve accuracy for times before the scheduled event, as well as for the event itself.

4.5.4 Augsburg Dataset

Table 4. External Data Evaluation Results

	Two Season	Fall
Overall Accuracy	40%	56%

Table 4 shows overall accuracy on two evaluations on this data. The 'Two Season' evaluation trains on the summer data and tests on the fall data. Our prediction accuracy is low while accuracy levels of 70-80% are achieved in [10]. We believe there are two reasons for this. First, the occupant patterns are predictable sequences but at irregular times of day and our approach cannot learn a sequence of movements independently of the time at which it occurs. If an occupant repeats the same sequence of movements at different times, other approaches will treat this as reinforcement of a single sequence, whereas our approach will attempt to learn rules representing multiple separate sequences. Second, the evaluation in [10] only predicts the destination of an occupant when they move rather than predicting their location at each time slot.

The 'Two Season' evaluation limits the training data to a maximum of two weeks for each occupant, therefore we also evaluated a split of the fall data where the final week is used for testing, leaving approximately a month of preceding data per occupant for training. The increased training set results in a significant increase in accuracy, however our approach still has difficulty predicting these occupants.

5 CONCLUSIONS AND FUTURE WORK

In this paper we presented an approach for applying association rule mining to the problem of predicting future occupant locations. We implemented our approach using a modification of a standard association rule mining algorithm, and presented experimental results which show that our modification of the algorithm can predict actual occupant movements with a high degree of accuracy, but is dependent on the types of patterns in the occupancy data.

In comparison to standard approaches, association rule mining has some benefits and drawbacks. While most other approaches predict an occupant's next location from a sequence representing the occupant's recent movements, our aim is to predict for any time slot using whatever information is available, whether it be the occupant's recent movements the same day, or simply their historical patterns. This is successful on two of the datasets, but our approach's inability to learn time-independent sequences means we fall short of the existing approaches on the third dataset. We intend to perform a deeper investigation of how our approach compares to existing approaches on our datasets.

There is existing work on extending Apriori to add new functionality, including sequence mining and mining with taxonomies. As part of our future work we intend to integrate these features into our modified Apriori. Sequence mining will allow it to perform better on datasets similar to the Augsburg dataset, as well as improving performance on the other datasets, by finding time-independent patterns. Taxonomies would allow the specification of a hierarchy over the possible locations an occupant may occupy. For example, we may generalize all locations to a simple in or out of office value for each occupant, and then find that in some cases where we cannot predict an occupant's exact location with high confidence, we can be highly confident that they will not be in their office.

We also wish to consider inter-instance mining. In many cases an occupant's movements will depend on other occupants, for example a meeting not occurring due to someone being absent. By learning patterns between occupants we could recognize these cases and improve our prediction accuracy.

The eventual goal is to integrate this approach with occupant localization systems such as [11], and predictive control systems such as [12]. Using occupant localization data, a system based on our approach could provide the predictions necessary for more energy efficient building control.

REFERENCES

[1] World Business Council for Sustainable Development, *Energy Efficiency in Buildings: Facts and Trends – Full Report*, (2008).

[2] R. Agrawal and R. Srikant, *Fast Algorithms for Mining Association Rules in Large Databases*, Proceedings of the 20th International Conference on Very Large Data Bases, 487-499, (1994).

[3] J. Petzold, A. Pietzowski, F. Bagci, W. Trumler and T. Ungerer, *Prediction of Indoor Movements Using Bayesian Networks*, LoCA 2005, LNCS 3479, 211–222, (2005).

[4] L. Vintan, A. Gellert, J. Petzold, and T. Ungerer, *Person Movement Prediction Using Neural Networks*, Technical Report, Institute of Computer Science, University of Augsburg, (2004).

[5] J. Petzold, F. Bagci, W. Trumler, T. Ungerer, and L. Vintan, *Global State Context Prediction Techniques Applied to a Smart Office Building*, The Communication Networks and Distributed Systems Modeling and Simulation Conference, San Diego, CA, USA, (2004).

[6] C. Voigtmann, Sian Lun Lau, K. David, *A Collaborative Context Prediction Technique*, Vehicular Technology Conference, (2011)

[7] N. Eagle and A. S. Pentland, *Eigenbehaviors: Identifying structure in routine*, Behavioral Ecology and Sociobiology, vol. 63, no. 7, 1057–1066, (2009).

[8] Jan Petzold, *Augsburg Indoor Location Tracking Benchmarks*, Context Database, Institute of Pervasive Computing, University of Linz, Austria. http://www.soft.uni-linz.ac.at/Research/Context Database/index.php, (2005).

[9] G. Zimmermann, *Modeling and Simulation of Individual User Behavior for Building Performance Predictions*, Proceedings of the Summer Computer Simulation Conference (SCSC), 913–920, San Diego, California, (2007).

[10] J. Petzold, F. Bagci, W. Trumler, and T. Ungerer. *Comparison of Different Methods for Next Location Prediction*, EuroPar 2006, LNCS 4128, 909–918, (2006).

[11] W. Najib, M. Klepal and S.B. Wibowo, *MapUme: Scalable middleware for location aware computing applications*, International Conf. on Indoor Positioning and Indoor Navigation (IPIN), (2011).

[12] A. E.D. Mady, G. Provan, C. Ryan and & K. N. Brown, *Stochastic Model Predictive Controller for the Integration of Building Use and Temperature Regulation*, AAAI (2011).

This work is funded by Science Foundation Ireland as part of the ITOBO project under grant No. 07.SRC.I1170.

Situation-Aware Energy Control by Combining Simple Sensors and Complex Event Processing

Leonard Renners and **Ralf Bruns** and **Jürgen Dunkel** [1]

Abstract. In recent years, multiple efforts for reducing energy usage have been proposed. Especially buildings offer high potentials for energy savings. In this paper, we present a novel approach for intelligent energy control that combines a simple infrastructure using low cost sensors with the reasoning capabilities of Complex Event Processing. The key issues of the approach are a sophisticated semantic domain model and a multi-staged event processing architecture leading to an intelligent, situation-aware energy management system.

1 Introduction

In recent years, global warming, greenhouse effect, as well as escalating energy prices have lead to enhanced efforts for reducing energy usage and carbon emission. Especially, buildings offer high potentials in reducing energy consumption: electric lighting, heating and air conditioning are highly energy-intensive and not well-adjusted to actual usage necessities.

In many buildings, there are already first approaches of more effective control mechanisms to improve energy consumption. Typically, light in public areas like corridors is controlled by using motion sensors instead of classic switches, preventing unnecessarily turned on lamps. On the contrary, heating is mostly controlled on base of fixed heating plans that determine the heating period by a predefined timetable. Often the heating schedule is not related to single rooms but to the entire building or larger parts of it (e.g. total floors). Overall, this leads to the situation that many rooms are heated although they are not in use, thus causing a huge waste of energy. In summary, we can conclude that in general energy control of buildings is not situation-aware.

In the following we present an approach for intelligent energy control that combines a simple infrastructure using cheap sensors with event stream processing of the plain sensor data. Our approach should reach the following goals:

(a) *Individual energy control* for every single room (instead of considering the entire building).
(b) *Low-cost solution* by utilizing the existing infrastructure as well as cheap and simple sensors (as opposed to new, sophisticated and expensive sensors).
(c) *Situation awareness:* the energy consumption should be controlled according to actual usage (in contrast to predefined and fixed schedules).
(d) *Proactive control*: the control mechanisms should exploit the knowledge about normal room occupancy, for instance based on room schedules and normal user behavior patterns.

[1] Hannover University of Applied Sciences and Arts, Germany, email: forename.surname@fh-hannover.de

To achieve these goals, our approach uses two different models:

1. A *Semantic Domain Model* describes the expert knowledge about the domain, i.e. in our case the structure of the building and its expected usage.
2. An *Event Model* defines all those events occurring in the building that are relevant for energy control. Different event sources can be distinguished: low cost sensors yield information about the current incidents in a building. Furthermore, domain-specific events are created by various knowledge sources, like heating plans or lectures schedules.

To react on relevant situations in real-time we apply *Complex Event Processing* (CEP) that has been proposed as a new architectural paradigm for in-memory processing continuous streams of events. CEP is based on declarative rules to describe event patterns, which are applied to the event stream to identify relevant situations in a certain domain.

The integration of the Semantic Domain Model with the Event Model allows the enrichment of event processing rules with domain knowledge. It is a key issue of our approach and provides an intelligent sensor data processing, leading to a reasonable, situation-aware heating and energy management.

The remainder of the paper is organized as follows. The next section 2 describes an application scenario for our approach. In section 3 the related work is discussed. In the subsequent section 4 we introduce our approach using CEP techniques to implement an intelligent energy management. Section 5 shows an example set-up and yields an evaluation. The final section 6 contains some concluding remarks and provides an outlook to future directions of research.

2 Scenario: Energy Control in Buildings

We selected the energy management of buildings as our application scenario. In particular, the scenario helps to discuss in some more detail the benefits that intelligent energy control provides. As already mentioned in the introduction, buildings have high potentials in improving energy efficiency, since they have many energy consumer units that are often unnecessarily turned on. In the following, we will use an university campus as a concrete example of our approach. Universities exhibit the following features that are common to most public buildings and which will influence energy management significantly:

- *Various room types* with different energy consumption profiles can be distinguished: For instance, server rooms have to be air conditioned below 20 degrees Celsius. Instead, offices and lecture rooms must be heated to achieve a temperature above 22 degrees, but normal storage room must be neither cooled nor heated. In the following, we will focus on office and lecture rooms.

- *Non-uniform usage profiles*: Each room exhibits its individual occupancy depending on the room type and the specific behavior of different user groups. For instance, lecture rooms are occupied according to a prefixed schedule that might be changed only exceptionally. Instead, offices or cube farms are used according to the personal behavior of their occupants including absences due to vacation times or illness.
- *Spontaneous occupancies*: Furthermore, rooms are spontaneously and individually used, e.g. due to ad-hoc meetings, rescheduled lectures or unplanned project work. Note that these individual occupancies are inherently unpredictable.

To deal with non-uniform and spontaneous usage of rooms, which deviates significantly from predicted average occupancies, an intelligent and situation-aware energy control mechanism is required.

On the one hand, expert knowledge about the building and the behavior of its users should be exploited for adjusting the energy control to realistic usage patterns.

On the other hand, actual behavior must be monitored to react adequately on spontaneous and individual usage actions. Especially, if unexpected occupancies or periods of absence are detected in a certain room, the corresponding energy consumption units (like heating and lighting) can be switched on or off, respectively.

To provide an individual control, we have to observe the incidents and states in every single room. To achieve this goal we will incorporate already installed sensors in the building as well as we will equip the rooms with simple and cheap sensors: *motion sensors* for detecting movements, *temperature sensors* to measure the heating, and *contact switches* that register when a door or window is opened or closed, respectively.

3 Related Work

Our approach is based on the exploitation of fine-grained sensor data emitted by networks of simple sensors in buildings. Sensor networks possess intrinsic problem properties that are perfectly addressed by complex event processing. Several published approaches prove the suitability of event processing for sensor networks (e.g. [8, 11]).

Determining the current status of usage is an important topic in smart homes and intelligent facility management systems. Several approaches have shown how occupancy detection can be used to implement more effective and powerful behaviour [1, 2]. Other approaches put the main focus on occupancy and movement prediction algorithms instead of real-time reactions [6, 5, 9].

There is also increasing interest using CEP technologies for energy management. Holland-Moritz and Vandenhouten examined different solutions for an intelligent management system (in general) and identified CEP as one very suitable and suggestive concept [7].

Xu et al. introduced a CEP approach with ontology and semantics supporting occupancy detection for an intelligent light management system [12, 14].

Another approach in a similar direction was made by Wen et al. within their industrial experience report about using CEP for energy and operation management, while focussing on predictive elements and adaptable behavior [13].

In summary, on the one hand some approaches address energy control adapted to the current occupancy/usage, but do not rely on event processing techniques, thus not taking advantage of features like in-memory processing and real-time capabilities. On the other hand first approaches report the employment of event processing technologies in energy management, but do not target the same field and way of application.

However, none of them is presenting a comprehensive approach of real-time situational awareness for the whole energy management, based on the integration of an extensive semantic model of the application domain in the event processing reasoning process. In particular, the investigated control of the heating process requires more sophisticated semantic models and more advanced event processing rules.

4 Intelligent Energy Control Using CEP

In this section we present our approach for a situation-aware energy control by applying intelligent sensor data processing on simple buildings. After giving a short overview of *Complex Event Processing* we present our general software architecture and a *Domain Event Model* that integrates knowledge about domain specific concepts and events. Finally, we illustrate the intelligent reasoning part of our approach by showing some event processing rules.

4.1 CEP Overview

Complex Event Processing (CEP) is a software architectural approach for processing continuous streams of high volumes of events in real-time [10]. Everything that happens can be considered as an *event*. A corresponding *event object* carries general metadata (event ID, timestamp) and event-specific information, e.g. a sensor ID and the measured temperature. Note that single events have no special meaning, but must be correlated. CEP analyses continuous streams of incoming events in order to identify the presence of complex sequences of events, so called *event patterns*.

A *pattern match* signifies a meaningful state of the environment and causes either creating a new *complex event* or triggering an appropriate action.

Fundamental concepts of CEP are an *event processing language* (EPL), to express *event processing rules* consisting of *event patterns* and *actions*, as well as an *event processing engine* that continuously analyses the event stream and executes the matching rules.[2] Complex event processing and event-driven systems generally have the following basic characteristics:

- *Continuous in-memory processing*: CEP is designed to handle a consecutive input stream of events and in-memory processing enables real-time operations.
- *Correlating Data*: It enables the combination of different events from distinct sources including additional domain knowledge. Event processing rules transform fine-grained simple events into complex (business) events that represent a significant meaning for the application domain.
- *Temporal Operators*: Within event stream processing, timer functionalities as well as sliding time windows can be used to define event patterns representing temporal relationships.
- *Distributed Event Processing*: Event processing can be distributed on several rule engines (physically or logically). Thereby scalability and the separation of different functionalities can be realized.

4.2 Event Processing Architecture

Luckham introduced the concept of *event processing agents* (EPA) [10]. An EPA is a software component specialized on event

[2] Sample open source CEP engines are Esper and Drools Fusion.

stream processing with its own rule engine and rule base. An *event processing network* (EPN) connects several EPAs to constitute a software architecture for event processing. Event processing agents communicate with each other by exchanging events.

EPAs provide an approach for modularizing and structuring rules: Light-weighted agents with few rules fulfill a coherent domain-specific task and improve comprehensibility and maintainability. Furthermore, distributing the EPAs on different computing nodes enhances system performance and scalability [3].

Thus, the event-driven architecture of our energy control system is based on a multi-staged EPN for structuring and organizing the event processing rules. Figure 1 depicts the different EPAs and illustrates the flow of events:

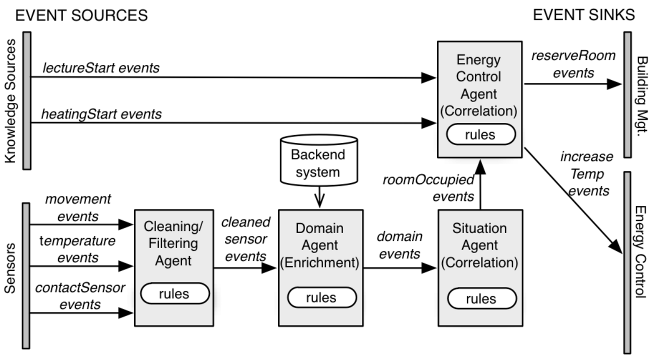


Figure 1. Event Processing Network (EPN) for Energy Management

Event Sources: We can distinguish different types of *event sources* that correspond to the information that is used by our energy control system (as already mentioned in section 2).

- *General knowledge sources*: There are general knowledge sources that can emit application-specific events relevant for the buildings energy management. For instance, a calendar containing the lecture schedules might create *lectureStart* events that signal the scheduled starting time of a teaching session.
- *Sensors*: Low cost sensors as described in section 2 are used to monitor the incidents in the building. For instance, motion sensors and temperature sensors emit *movement* events as well as *temperature* events. The contact switches produce *contactSensor* events that signal if doors or windows are opened or closed, respectively.

Event Processing Network (EPN): Event processing is nothing else than event transformation: the simple events emitted by the event sources are transformed into more abstract application-specific events for inferring appropriate control steps. The event transformations are processed by the EPAs depicted in figure 1.

- *Cleaning/Filtering Agent*: Due to technical problems, sensor data is often inconsistent: e.g. duplicated readings or outliers must be compensated. Therefore, in a cleaning step all sensor events have to be pre-processed to overcome inconsistencies [3].
  Furthermore, not all events are required in subsequent processing stages. For instance, motion sensors may emit many *movement* events within a small time interval, which are related to the same incident. Therefore irrelevant events are filtered out to reduce the total number of events. Using various processing rules, the Cleaning/Filtering Agent forwards *cleaned sensor events* events to the Domain Agent.
- *Domain Agent*: The *cleaned sensor* events contain only low-level technical information, e.g. sensor IDs that have no specific meaning in the application domain, and are often incomplete for further processing. Therefore, they should be transformed to domain events by mapping plain sensor event data to domain concepts. For instance, a measured temperature should be related to a certain room and to the desired temperature of the corresponding room type. The information necessary for this content enrichment step is retrieved from the backend systems. The Domain Agent transforms *cleaned sensor* events into enriched *domain* events and forwards them to the Situation Agent.
- *Situation Agent*: In a diagnosis step various domain events are synthesized to a new (complex) situation event that characterizes a particular state of the building. For example, *contactSensor* events and *movement* events are correlated to a new *roomOccupied* event signalizing that somebody is staying in a certain room. In summary, the Situation Agent processes a correlation step to create new types of complex events that are propagated to the Energy Control Agent.
- *Energy Control Agent*: Finally, the situation diagnosed from the stream of sensor events must be correlated with the information received from the general knowledge sources. The Energy Control Agent emits an *action event* to trigger a certain control action that reacts appropriately on the actual state of the building.
  For instance, *lectureStart* events emitted by a calendar are combined with *roomOccupied* events generated by the the Situation Agent to trigger an appropriate control actions by creating an action event of type *increaseTemperature*.

Event Sinks: The backend systems of the building management serve as event sinks of the events produced by the Energy Control Agent. Figure 1 shows two examples: an *increaseTemperature* event could be sent to the energy control system to change directly the heating of a certain room. As another example, we can consider a *reserveRoom* event that could be forwarded to the building management system for generating automatically an entry into the occupancy plan of the corresponding room.

4.3 Domain Event Model

A main contribution of our approach is integrating general *domain knowledge* with *sensor events* in a *Domain Event Model*. Figure 2 shows the general structure of the Domain Event Model that distinguishes two dimensions, and thus yielding four different quadrants:

	WORLD MODEL	EVENT MODEL
DOMAIN MODEL	Domain Concepts	Domain Events
SENSOR MODEL	Sensors	Sensor Events

Figure 2. Structure of the Domain Event Model

- The *World Model* describes the structural or static concepts regarded in the system: First, it defines the *domain concepts* like buildings, rooms or class schedules. Secondly, it defines the *sensor* infrastructure the building is equipped with. For instance, what different kind of sensors are used and where they are installed.
- The *Event Model* defines the dynamic aspects of the system, i.e. all types of events that are considered in the system [4]. First, all *sensor events* emitted by the different sensor types are described. Secondly, it considers all *domain events* in the system: On the one

hand, these are the application events that are generated by CEP rules and have a certain meaning in the application domain. For instance, that a room is occupied for a certain time. Furthermore, it defines context events, that are produced by general knowledge sources, like calendar applications containing room schedules.

Of course, there are interrelations between the concepts of the different model parts. For instance, the sensor concept 'contact sensor' is related to the domain concept 'room' specifying the concrete position of a certain sensor.

Figure 3 shows an excerpt of the Domain Event Model for our university building scenario. Note that for simplicity and clarity, no attributes are depicted in the diagram.

The **Model of the Domain Concepts** (upper left quadrant) describes the hierarchy of different types of rooms in an university, such as course rooms, offices, and server rooms. Course rooms can be further refined into lecture rooms, labs, etc. Other concepts modeled in figure 3 are heating plans that are related to each room and class schedules for each course room. Neighbouring rooms are specified by the *adjacent* relationship. Furthermore, the model defines room equipment as windows and doors.[3]

The **Model of the Sensors** (lower left quadrant) specifies the different types of sensors installed in the building. Note that in figure 3 sensor characteristics like measured variables, and quality of measures, like availability or accuracy, are omitted for clarity. However, the model represents the location of the sensors by relating them to a physical item of the domain model.

The **Model of the Sensor Events** (lower right quadrant) defines the types of sensor events. Depending on the specific type of a sensor different data can be produced. For instance, a contact sensor might produce data for signalizing that a door has been opened. Furthermore, each sensor event is related by the *sensed-by* relationship to a corresponding sensor, and thereby to a certain position. Note that this relation shows the connection between the world and the event model.

The **Model of the Domain Events** (upper right quadrant) presents all *application events* considered in the system. On the one hand, there are the *context events* that are produced directly by software components. Figure 3 defines *heatingStart* events and *lectureStart* events as specific examples of context events, which might be produced by a heating plan and a class schedule, respectively. On the other hand, *situation events* signal that a certain situation has occurred in the building, e.g. a room has been occupied or freed.

Furthermore, action events are considered like *increaseTemperature* or *reserveRoom* events. These events trigger some actions in the backend system, e.g. turning up the heating. They are produced by CEP rules that might correlate situation events and context events.

4.4 Event Processing Rules

In the following, we present some exemplary rules in a pseudo language to provide a better understanding of the intelligent reasoning capabilities our approach. An event processing rule contains of two parts: a *condition* part describing the requirements for the rule to fire and an *action* part to be performed if the requirements are matched. The *condition* is defined by an event pattern using several operators and further constraints.

[3] Note that this is not a very sophisticated model: many aspects are not shown, e.g. different user types and their behavior defined by working times and the usually used rooms.

Operators

`AND`	Combination of events or constraints
`NOT`	Negation of a constraint
`->`	Followed-by operator. Sequence of conditions.
`Timer`	*Timer(time)* defines a time to wait *Timer.at(daytime)* is a specific (optionally periodic) point of time.
`.within`	defines a time window for an event in which the event has to happen to be considered.

The following two rules are part of the rule base of the Situation Agent (see Figure 1) and produce a *situation event*. Note that these rules detect a certain situation in the building that can be exploited in different application domains. Here, we show how the Energy Control Agent can use identified situations to derive energy control actions. But also other kind or agents, for instance Security Agents can make use of *situation events* for detecting security risk or incidents.

The first rule produces a *situation event* of type *RoomOccupiedEvent* indicating that a certain room is currently occupied.

```
rule: "room occupied"
CONDITION DoorOpenEvent AS d ->
          Timer(5 minutes) ->
          MovementEvent AS m
          AND (d.room = m.room)
ACTION    new RoomOccupiedEvent(d.room)
```

A room is assumed to be occupied if the door is opened and five minutes later a movement is still observed. The delay will prevent false positives, like only cleaning the room or just quickly picking up some things. Note that the rule correlates two sensor events to derive a new complex event of type *situation event* with a new application-specific meaning.

The next rule considers the opposite situation: a room is not occupied if the door is closed and there is no movement within the following 10 minutes.

```
rule: "room not occupied"
CONDITION DoorCloseEvent AS d AND
          NOT MovementEvent.
              within(10 minutes) AS m
          AND (d.room = m.room)
ACTION    new RoomNotOccupiedEvent(d.room)
```

The following rules reside in the rule base of the Energy Control Agent (see Figure 1) and correlate *situation events* to derive some *action events* triggering some reactions in the backend system.

The first occupancy per day of an office is of special importance, since from then on the office is in use and needs to reach its operating temperature. Before that, room temperature could be a bit lower yielding a reduction of the heating costs.

```
rule: "first usage"
CONDITION Timer.at(06:00 AM) ->
          NOT RoomOccupiedEvent AS n ->
          RoomOccupiedEvent AS r
          AND (r.room.type = office)
          AND (n.room = r.room)
ACTION    IncreaseTemp(r.room)
```

The rule considers the situation in a certain room after 6:00 AM. If then a *RoomOccupied* event *r* occurs and there was no other *RoomOccupied* event *n* (between 6:00 and the occurrence of event

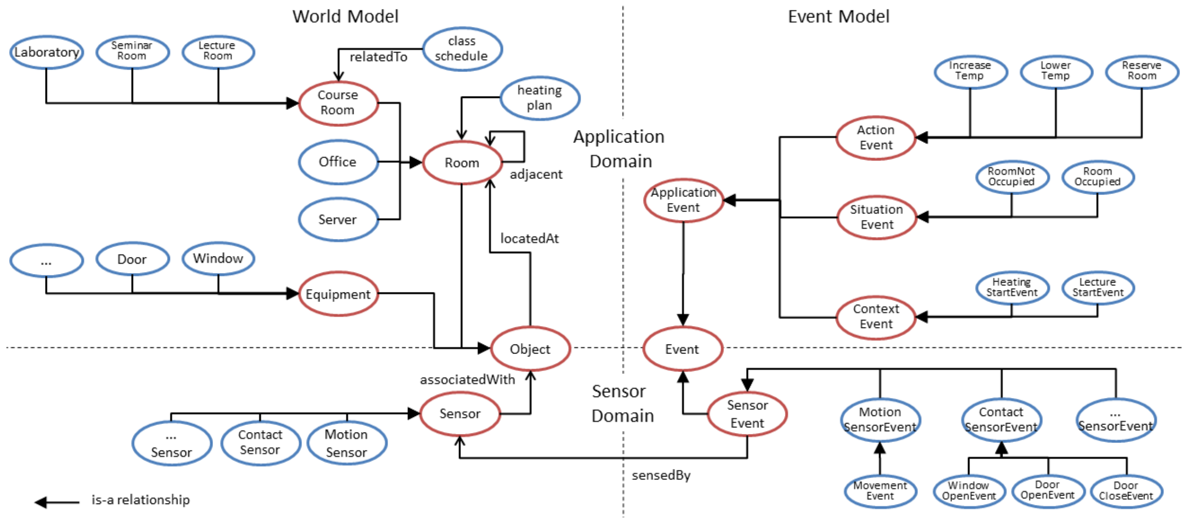


Figure 3. Domain Event Model

r) then the room is used for the first time that day and the temperature should be increased.

Another situation of interest is the 'final' absence of an employee. A shorter break, e.g. having lunch or a meeting, should not have the effect of cooling down the employees office. But after the typical end of the workday the probability that the room will be in use again is very low. Therefore, the heating can now be lowered to reduce the energy consumption.

```
rule: "after hour"
CONDITION Timer.at(06:00 PM) ->
          RoomNotOccupiedEvent AS r
          AND (r.room.type = office)
ACTION    LowerTemp(r.room)
```

If after 6:00 PM a *RoomNotOccupied* event is captured in an office room the temperature will be reduced.

The next rule illustrates how *context events* from general knowledge sources and *sensor events* are correlated to derive an *action event*. In particular, the rule describes the situation that though a lecture is scheduled, the lecture room is not occupied.

```
rule: "planned, but not used"
CONDITION LectureStartEvent AS l ->
          NOT RoomOccupiedEvent.
              within(15 minutes) AS r
          AND (l.room = r.room)
ACTION    lowerTemp(r.room)
```

The rule will fire if a *LectureStart* event is captured for a certain room, but within the following 15 minutes no *RoomOccupied* event occurs. This leads to the assumption that the lecture will not take place and accordingly the room will not be in use and the temperature can be lowered to the idle state.

Finally, we present a simple rule that exploits further domain-specific knowledge in the event reasoning. Several semantic relationships are represented in the Domain Event Model, which can be exploited to enhance the reasoning capabilities of event processing. For example, lecture rooms, seminar rooms and laboratories are all of type course room as specified by a 'is-a' relationship in the Domain Event Model. The semantical meaning of the 'is-a' relationship can be used in event processing: A rule for course rooms is implicitly valid for all subtypes as well.

```
rule: "course room not used for more
      than 1 hour"
CONDITION NOT RoomOccupiedEvent.
               within(60 minutes) as r
          AND (r.room.type = course)
ACTION    lowerTemp(r.room)
```

If a course room is not used for at least 60 minutes, then the temperature of the room can be decreased. This rule will match for a lecture room, but not for other rooms as offices.

Note that we will investigate the modeling of much more semantic relationships using an appropriate formalism in further researches. For instance with OWL, relationships between concepts can be described much more precisely. For OWL object properties ranges and domains can be specified as well as further property characteristics (as transitivity, symmetry or reflexivity). This information can be exploited by more sophisticated reasoning, for instance by using C-SPARQL query language.

5 Case Study / Evaluation

We have equipped one room of our university building with a sample setup of different physical sensors and implemented a prototype of our event-driven energy control system. As sensor hardware we used a *Phidget*[4] *Interface-Kit* and corresponding motion sensor and contact switches.

The event processing is implemented with the open source CEP engine *Esper* [5]. Esper provides the essential features of typical CEP systems like time windows, external method calls, and event pattern operators. The event processing rules are defined in a SQL-like rule language , the so-called *Continuous Query Language* (CQL). In contrast to SQL the CQL queries are not executed on a Database, but directly in-memory on the continuously arriving event stream.

Our experimental evaluation has proven the capabilities of CEP to correlate the sensor data and achieve a real-time analysis and reaction based on the sensor data stream and event patterns.

Since we could only realize an example installation for one distinct room and only including sensors and no actuators, the useful-

[4] http://www.phidgets.com
[5] http://esper.codehaus.org

ness of our approach is evaluated on the basis of assumptions about the real usage. We assume a typical heating behavior in a static manner starting heating at 6 AM until 9 PM. The typical workday of an university lecturer (as an example of a non-uniform user type) may be structured as followed: *start of work* at 8 AM, *lecture* between 10 and 11:30 AM, *lunch break* between 11:30 AM and 12 PM, *exercise lesson* between 12 and 1:30 PM, and *end of work* at 5:30 PM.

Figure 4 visualizes the different heating behaviors by example of a lecturer's office room: (a) dynamic heating with our approach based on situational awareness compared to (b) the static solution with a fixed heating plan. The *Human Presence* depicts the occupancy of

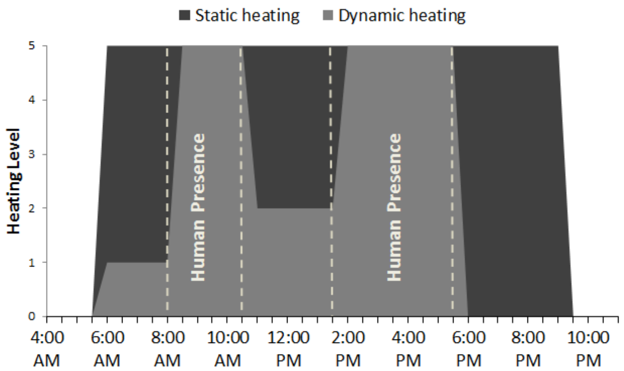


Figure 4. Static versus dynamic heating

the room according to the typical workday defined above. The two curves describe the heating level on the y-axes with respect to the time.

As can be seen, the biggest differences, and therefore energy savings, appear during the time before and after the workday. Notice that the usual hours of work may differ from lecturer to lecturer and thus the static schedule can not be fitted to the typical behavior of one lecturer.

In contrast, our approach provides a room-specific and situation-aware control mechanism enabling a precise energy management that additionally enables the heater to turn lower during temporary absence. In order to keep the room in a comfortable state, if the user returns, the heaters level is only decreased and not completely turned off for temporarily unoccupied rooms.

Based on these assumptions a calculation comparing the two different heating behaviors (static versus dynamic) results in a heating reduction up to 30% and, accordingly, lower energy consumption and carbon emission.

6 Conclusion

In this paper, a novel approach for intelligent energy management by means of complex event processing and simple sensors has been presented. The approach is different from other approaches in that is based on a sophisticated representation of domain as well as sensor knowledge and a multi-staged event processing architecture. By the integration of domain knowledge and semantic information into to the reasoning process we achieve an intelligent, situation-aware behavior.

The approach allows an individualized, situation-aware energy management of buildings according to the current occupancy status of the separate rooms. By means of complex event processing an existing infrastructure with everyday sensors can be expanded into an intelligent environment.

Directions of future research are, among others, the further enhancement of the semantic Domain Event Model as well as the development of advanced concepts for the incorporation of the semantic knowledge in event processing languages. Furthermore, development towards an automated rule creation, by the means of removing the necessities to hand-code the scenarios, could be considered.

7 Acknowledgement

This work was supported in part by the European Community (Europäischer Fonds für regionale Entwicklung – EFRE) under Research Grant EFRE Nr.W2-80115112.

REFERENCES

[1] 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 (BuildSys)*. ACM, (2010).

[2] J. C. Augusto and C. D. Nugent, 'The use of temporal reasoning and management of complex events in smart homes', in *Proceedings of the 16th Eureopean Conference on Artificial Intelligence (ECAI)*, pp. 778–782. IOS Press, (2004).

[3] R. Bruns and J. Dunkel, *Event-Driven Architecture: Softwarearchitektur für ereignisgesteuerte Geschäftsprozesse*, Springer-Verlag, Berlin Heidelberg, 2010.

[4] J. Dunkel, A. Fernández, R. Ortiz, and S. Ossowski, 'Injecting semantics into event-driven architectures', in *Proceedings of the 11th International Conference on Enterprise Information Systems (ICEIS)*, pp. 70–75. Springer, (2009).

[5] Varick L. Erickson, Yiqing Lin, Ankur Kamthe, Rohini Brahme, Amit Surana, Alberto E. Cerpa, Michael D. Sohn, and Satish 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*, pp. 19–24. ACM, (2009).

[6] Ge Gao and Kamin Whitehouse, 'The self-programming thermostat: optimizing setback schedules based on home occupancy patterns', in *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, pp. 67–72. ACM, (2009).

[7] R. Holland-Moritz and R. Vandenhouten, 'A flexible architecture for intelligent management systems', in *Proceedings of the 3rd International Symposium on Logistics and Industrial Informatics (LINDI)*, pp. 83–86. IEEE Computer Society, (2011).

[8] S. R. Jeffery, G. Alonso, M. J. Franklin, and J. Widom W. Hong, 'A pipelined framework for online cleaning of sensor data streams', in *Proceedings of the International Conference on Data Engineering (ICDE)*, p. 140, (2006).

[9] J. Landay, Y. Shi, D. J. Patterson, Y. Rogers, X. Xie, J. Scott, A.J. Bernheim Brush, J. Krumm, B. Meyers, M. Hazas, S. Hodges, and N. Villar, 'Preheat: Controlling home heating using occupancy prediction', in *Proceedings of the 13th International Conference on Ubiquitous Computing (UbiComp)*, pp. 281–290. ACM, (2011).

[10] D. C. Luckham, *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*, Addison-Wesley, 2002.

[11] S. Selvakennedy, U. Rohm, and B. Scholz, 'Event processing middleware for wireless sensor networks', in *Proceedings of the International Conference on Parallel Processing Workshops (ICPPW)*, p. 65, (2007).

[12] N. Stojanovic, D. Milenovic, Y. Xu, L. Stojanovic, D. Anicic, and R. Studer, 'An intelligent event-driven approach for efficient energy consumption in commercial buildings: Smart office use case', in *Proceedings of the International Conference on Distributed Event-based System*, pp. 303–312. ACM, (2011).

[13] J. Y. C. Wen, G. Y. Lin, T. Sung, M. Liang, G. Tsai, and M. W. Feng, 'A complex event processing architecture for energy and operation management', in *Proceedings of the 5th International Conference on Distributed Event-Based Systems (DEBS)*, pp. 313–316. ACM, (2011).

[14] Y. Xu, N. Stojanovic, L. Stojanovic, D. Anicic, and R. Studer, 'An approach for more efficient energy consumption based on real-time situational awareness', in *The Semanic Web: Research and Applications*, pp. 270–284. Springer, (2011).

Learning Individual Thermal Comfort using Robust Locally Weighted Regression with Adaptive Bandwidth

Carlo Manna and **Nic Wilson** and **Kenneth N. Brown**[1]

Abstract.
Ensuring that the thermal comfort conditions in offices are in line with the preferences of the occupants, is one of the main aims of a heating/cooling control system, in order to save energy, increase productivity and reduce sick leave days. The industry standard approach for modelling occupant comfort is Fanger's Predicted Mean Vote (PMV). Although PMV is able to predict user thermal satisfaction with reasonable accuracy, it is a generic model, and requires the measurement of many variables (including air temperature, radiant temperature, humidity, the outdoor environment) some of which are difficult to measure in practice (e.g. activity levels and clothing). As an alternative, we propose Robust Locally Weighted Regression with Adaptive Bandwidth (LRAB) to learn individual occupant preferences based on historical reports. As an initial investigation, we attempt to do this based on just one input parameter, the internal air temperature. Using publicly available datasets, we demonstrate that this technique can be significantly more accurate in predicting individual comfort than PMV, relies on easily obtainable input data, and is fast to compute. It is therefore a promising technique to be used as input to adpative HVAC control systems.

1 INTRODUCTION

One of the primary purposes of heating, ventilating and air conditioning (HVAC) systems is to maintain an internal environment which is comfortable for the occupants. Accurately predicting comfort levels for the occupants can enable one to avoid unnecessary heating or cooling, and thus improve the energy efficiency of the HVAC systems. A number of thermal comfort indices (indicators of human comfort) have been studied for the design of HVAC systems [1,2], the most widely used of which is the Predicted Mean Vote (PMV) index, which was developed by Fanger [1]. This conventional PMV model predicts the mean thermal sensation vote on a standard scale for a large group of people in a given indoor climate. It is a function of two human variables and four environmental variables, i.e. clothing insulation worn by the occupants, human activity, air temperature, air relative humidity, air velocity and mean radiant temperature. The values of the PMV index have a range from -3 to +3, which corresponds to the occupants thermal sensation from cold to hot, with the zero value of PMV meaning neutral.

However, PMV has some drawbacks: *(i)* it requires many environmental data whose retrieval is costly due to the sensors needed, and it requires precise personal dependent data (i.e., clothing and activity level) which are often difficult to obtain in practice; *(ii)* it is a statistical measure which assumes a large number of people experiencing the same conditions, and so may be inaccurate for small groups, or for variable conditions and behaviours within the space, and *(iii)* it requires an expensive iterative evaluation to compute the root of a nonlinear relation.

In this paper, we propose an alternative approach tailored to individual occupants, which relies on historical data on individual responses to internal environment conditions. We apply Robust Locally Weighted Regression [23] with an Adaptive Bandwidth (LRAB), a statistical pattern recognition methods, to learn, automatically, the comfort model of each user based on their history. As a preliminary study, we applied this method with only one input variable (internal air temperature) and compared with PMV, using publicly available datasets [18]. Our experimental results show that LRAB outperforms PMV in predicting individual comfort, and hence it is a promising technique to be used as input to heating/cooling control systems in office environment.

The paper is organised as follows: in the next section, some background on PMV and on alternative techniques are reported. Then, in the section 3, the proposed method is described and in the section 4, the experimental results using a public dataset [18] are shown. Finally, in the section 5, conclusions and future directions are reported.

2 BACKGROUND

The conventional PMV model has been an international standard since the 1980s [3,4]. It has been validated by many studies, both in climate chambers and in buildings [5,6,7]. The standard approach to comfort-based control involves regulating the internal environment variables to ensure a PMV value of zero [8,9,10,11,12].

The PMV model parameters are based on field studies over large populations experiencing the same conditions. For small groups of people within a single room or zone in a building, however, PMV may not be an accurate measure. Kumar and Mahdavi in [17] analysed the discrepancy between predicted mean vote proposed in [1] and observed values based on a meta-analysis of the field studies database made available under ASHRAE RP-884 [18] and finally proposing a framework to adjust the value of thermal comfort indices (a modified PMV). The large field studies on thermal comfort described in [27], have shown that PMV does not give correct predictions for all environments. de Dear and Brager [28] found PMV to be unbiased when used to predict the preferred operative temperature in the air conditioned buildings. PMV did, however, overestimate the subjective warmth sensations of people in warm naturally ventilated buildings. Humphreys and Nicol in [29] showed that PMV was less closely correlated with the comfort votes than were the air temperature or the mean radiant temperature, and that the effects of errors

[1] Cork Constraint Computation Centre, Department of Computer Science, University College Cork, Ireland, e-mail: c.manna@4c.ucc.ie, n.wilson@4c.ucc.ie, k.brown@cs.ucc.ie

in the measurement of PMV were not negligible. Finally the work in [30] also showed that the discrepancy between PMV and the mean comfort vote was related to the mean temperature of the location.

In addition to the relative inaccuracy, the PMV model is a nonlinear relation, and it requires iteratively computing the root of a nonlinear equation, which may take a long computation time. Therefore, a number of authors have proposed alternative methods of calculation to the main one proposed in [1]. Fanger [1] and ISO [4] suggest using tables to determine the PMV values of various combinations between the six thermal variables. Sherman [13] proposed a simplified model to calculate the PMV value without any iteration step, by linearizing the radiation exchange term in Fanger's model. This study indicated that the simplified model could only determine precisely when the occupants are near the comfort zone. Federspiel and Asada [14] proposed a thermal sensation index, which was a modified form of Fanger's model. They assumed that the radiative exchange and the heat transfer coefficient are linear, and they also assumed that the clothing insulation and heat generation rate of human activity are constant. They then derived a thermal sensation index that is an explicit function of the four environmental variables. However, as the authors said, the simplification of Fanger's PMV model results in significant error when the assumptions are not respected. On the other hand, in [15] and [16] different approaches have been proposed in order to compute PMV avoiding the difficult iterative calculation. The former proposes a Genetic Algorithm—Back Propagation neural network to learn user comfort based both on historical data and real-time environmental measurements. The latter proposes a neural network applied to the iterative part of the PMV model that, after a learning phase, based on historical data, avoids the evaluation of such iterative calculation in real-time.

Finally, recent trends in the study of the thermal environment conditions for human occupants are reported in the recently accepted revisions to ASHRAE Standard 55, which includes a new adaptive comfort standard (ACS) [19]. According to de Dear and Brager [20] this adaptive model could be an alternative (or a complementary) theory of thermal perception. The fundamental assumption of this alternative point of view states that factors beyond fundamental physics and physiology play an important role in building occupants expectations and thermal preferences. PMV does take into account the heat balance model with environmental and personal factors, and is able to account for some degrees of behavioral adaptation such as changing one's clothing or adjusting local air velocity. However, it ignores the psychological dimension of adaptation, which may be particularly important in contexts where people's interactions with the environment (i.e. personal thermal control), or diverse thermal experiences, may alter their expectations, and thus, their thermal sensation and satisfaction. In particular, the level of comfort perceived by each individual also depends on their degree of adaptation to the context and to the environmental changes, and therefore the specificity of each individual should be taken into account to learn and predict comfort satisfaction.

For this reason, some authors have proposed techniques based on learning the perception of comfort by individuals. For example, in [21] the author proposes a system able to learn individual thermal preferences using a Nearest Neighbor Classifier, taking into account only four variables (air temperature, humidity, clothing insulation and human activity), acquired by means of wearable sensors. In [22], a Nearest Neighbor Classification-like method was implemented in order to learn individual user preferences based on historical data, using only one variable (air temperature).

In this study we consider such alternative and more practical approaches to predicting thermal comfort through the automatic learning of the comfort model of each user based on his/her historical records. We apply the Robust Locally Weighted Regression [23] technique with an Adaptive Bandwidth (LRAB), one of the family of statistical pattern recognition methods. Non-parametric regression methods, or kernel-based methods, are well established methods in statistical pattern recognition [24]. These methods do not need any specific prior relation among data. Hence, there are no parameter estimates in non-parametric regression. Instead, to forecast, these methods retain the data and search through them for past similar cases. This strength makes non-parametric regression a powerful method due to its flexible adaptation in a wide variety of situations. The Robust Locally Weighted Regression is one of a number of non-parametric regressions. It fits data by local polynomial regression and joins them together. This method was first introduced by Cleveland [23] and further developed for multivariate models [25].

3 THE PROPOSED METHOD

The proposed method is largely inspired by the work in [23]. In the following, we will only describe the proposed LRAB method, while for a more general description of the robust locally weighted regression, the readers should refer to the work in [23].

Before giving precise details on the LRAB procedure, we attempt to explain the basic idea of the method. Let (x_i, y_i) denote a response, y_i, to a recorded value x_i, for $i = 1, \ldots, n$. In this paper x_i denotes an environmental variable (in our case air temperature) and the response y_i represents the satisfaction degree (integer-valued on a 7-points scale from -3 to $+3$) that the user has given in response to the condition x_i, and then stored in a database. The aim is to assess the response $\hat{y_k}$ (i.e. predict the degree of satisfaction) for a input value x_k. The approach aims to estimate a local mean, fitting the recorded data by means of a local linear regression centered at x_k. This involves, for a fixed entry point x_k, solving a least squares problem, where α_k and β_k are the values that minimize:

$$\sum_{i=1}^{n} \big(y_i - \alpha_k - \beta_k(x_i - x_k)\big)^2 \omega(x_i - x_k; h) \qquad (1)$$

Then α_k is the response $\hat{y_k}$ for the point x_k. The kernel function $\omega(x_i - x_k; h)$, is generally chosen to be a smooth positive function which peaks at 0 and decreases monotonically as $|x_i - x_k|$ increases in size. The smoothing parameter h controls the width of the kernel function and hence the degree of smoothing applied to the data. This procedure computes the initial fitted values. Now, for each (x_i, y_i), a different weight, ψ_i is defined, based on the residual $(\hat{y_i} - y_i)$ (the larger the residual, the smaller the associated weight). Then, the function (1) is computed replacing $\omega(x_i - x_k; h)$ with $\psi_i * \omega(x_i - x_k; h)$. This is an iterative procedure.

3.1 Kernel function and Adaptive bandwidth

In the following we introduce the concepts of kernel function (and adaptive bandwidth) and residuals, then we describe the algorithm in details. There are many criteria to choose the kernel function based on the theoretical model of the function that has to be fitted. For a locally weighted regression, a common choice is a tri-cubic function, which generally can be written as: $\omega(u) = (1 - |u|^3)^3$, for $|u| \leq 1$, and $\omega = 0$ otherwise [23]. Starting from these considerations, we propose a similar kernel function:

$$\omega(x_i - x_k; h) = \left(1 - \left(\frac{|x_i - x_k|}{h}\right)^3\right)^2 \tag{2}$$

for $|x_i - x_k| \leq h$; otherwise $\omega = 0$. The outer exponent is 2 (in place of 3 as in the standard tri-cubic function), because of empirical considerations (preliminary experiments on smaller set of data were carried out to select the shape of the kernel function).

Finally, we need to choose the bandwidth h. The choice here needs to take into account the fact that the density of the recorded data may be variable. In particular, there may be areas in which the data are clustered closely together (which suggests that a narrow bandwidth would be appropriate), while, on other hand, other areas may be characterised by sparse data (in which case a choice of a large bandwidth is better). In view of this, it would be appropriate to have a large smoothing parameter where the data are sparse, and a smaller smoothing parameter where the data are denser (Figure 1). In this situation an adaptive parameter has been introduced. Let the ratio ν/n (where $\nu < n$), describes the proportion of the sample which contributes strictly positive weight to each local regression (for example if the ratio is 0.7, it means that 70% of the recorded data contributes to the regression). Once we have chosen ν/n (that means we have chosen ν, as n is fixed), we select the ν nearest neighbours from the new entry point x_k. Then, the smoothing parameter h is denoted by the distance of the most distant neighbour among the ν neighbours selected. It should be noted that the entire procedure requires the choice of a single parameter setting.

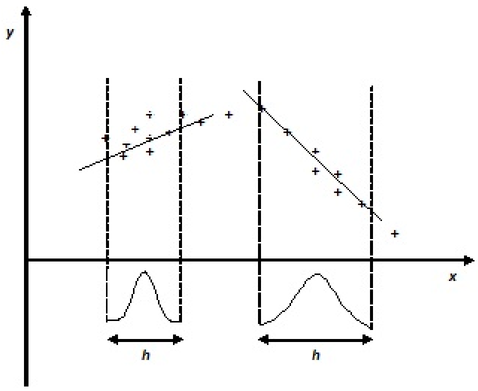


Figure 1. In locally weighted regression, points are weighted by proximity to the current x_k in question using a kernel function. A linear regression is then computed using the weighted points. Here, an adaptive bandwidth h based on the density of the recorded data is proposed.

3.2 Computing the residuals and weights update

In this section we introduce the update mechanism for the weighted function (2), based on the residuals $(\hat{y}_i - y_i)$ for $i = 1, \ldots, n$, as mentioned at start of Section 2. Define the bisquare function:

$$\Gamma(\xi) = (1 - \xi^2)^2 \tag{3}$$

for $|\xi| < 1$; otherwise $\Gamma = 0$.

Then, for a fixed new entry point x_k, let:

$$\rho_i = (\hat{y}_i - y_i) \tag{4}$$

be the residuals for $i = 1, \ldots, n$, between the original points y_i and the estimated points $\hat{y}_i$ (i.e. by means of α_k and β_k), and let m be the median of the $|\rho_i|$. As described in [23], we now choose robustness weights by:

$$\psi_i = \Gamma(\rho_i/6m) \tag{5}$$

At each step of the proposed procedure, the equation (5) is used to update the weight of the function (2) based on the residual ρ_i. In this way the value of the kernel function (2) at each recorded points x_i, is decreased (increased) where the residual value in x_i (i.e. ψ_i) is too high (too low), so as to improve the regression for the next step.

3.3 The algorithm

The proposed method can be described by the following sequence of operations:

LRAB

1: **Initialize**: set parameters ν
2: **For each** entry point x_k:
2.1: **minimise** (1)
2.2: **while** *iterations* $<$ *max iterations* **do**:
2.2.1: **for each** i compute (5)
2.2.2: **minimise** (1) replacing ω with $\psi_i * \omega$
2.3: **end while**
3: **end**

The algorithm is initialized by setting only one parameter (step 1). Then, for each new entry point x_k (step 2), it first computes an initial fitting (step 2.1), then it strengthens the initial regression by the steps 2.2.1 to 2.2.2, performing the sub-procedure described in the previous section, iteratively. If we have K new entry points x_k in total, the steps from 2.1 to 2.3 are repeated K times (one time for each new entry point).

4 EXPERIMENTS

This section describes the experimental results obtained from a comparison between the proposed method and the PMV. Although PMV is not based on a learning approach, in this paper, we compare our method with PMV since the latter is the international standard used to predict comfort in current building design and operation [32,33].

In particular, LRAB has been compared with the PMV index on real data from ASHRAE RP-884 database [18]. This collection contains 52 studies with more than 20,000 user comfort votes from different climate zones. However, some of these field studies contain only a few votes for each user. Thus they are not well suited for testing the proposed algorithm. This is because our approach seeks to learn the user preferences based on their votes, and it requires sufficiently many data records. For this reason, only the users with more than 5 votes have been used to compute the proposed LRAB. After removing the studies and records as described above we were left with 5 climate zones, 226 users and 7551 records (Table 1).

As a starting point, we consider only one environmental variable (i.e. inside air temperature) to evaluate the proposed LRAB. LRAB has been implemented in Matlab™, using the *trust-region method* to minimize the problem in (1), with a termination tolerance of 10^{-6}. The experiments have been performed through *leave-one-out validation*, for each user (i.e., using a single observation from the original

Climate zone	users	records
Hot arid	59	2594
Mediterranean	51	1899
Semi arid	21	2185
Tropical Savana	54	476
Continental	41	397

Table 1. Number of users and records divided by climate zones, and used in the experiments

sample as the validation data, and the remaining observations as the training data).

As with the field study [22, 31], the algorithms are evaluated considering the difference ΔV between the computed votes by both LRAB (evaluated) and PMV (reported in the database) and the actual vote (reported in the database) on a three-level accuracy scale [22, 31] as reported below:

- *Precise*: $\Delta V < 0.2$
- *Correct*: $0.2 \leq \Delta V < 0.5$
- *Approximation*: $0.5 \leq \Delta V < 0.7$

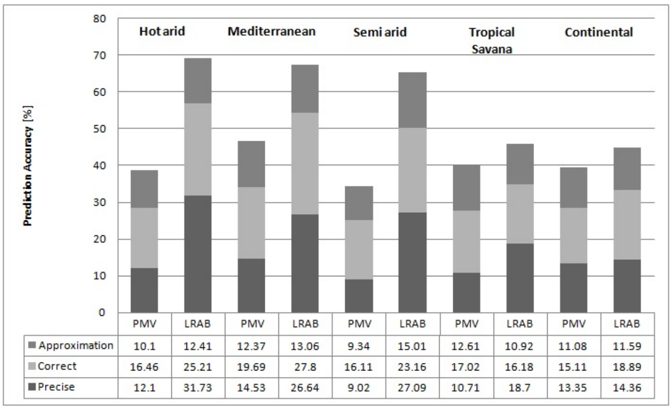


Figure 2. Average accuracy of predicting user comfort in 5 different climate zones.

Figure 2 illustrates how accurately the LRAB predicts the actual comfort vote of each user compared with PMV. In all 5 climate zones, LRAB predicts the actual vote better than PMV especially in the accuracy level $\Delta V < 0.2$ and has up to 200% of the number of occupants for whom a precise value is predicted. However, having a close look to the Figure 2, we notice that in the first three climate zones (i.e. hot arid, mediterranean and semi-arid), LRAB achieves more than 25% accuracy ($\Delta V < 0.2$), while in the last two climate zones (i.e. tropical savana and continental) it achieves only 16% and 18% of the same accuracy level respectively. We think that this discrepancy is because there is a lack of data for the last two climate zones compared the first three. In fact, in the former we have 476 records on 54 users (tropical savana) and 397 records on 41 users (continental), as shown in Table 1. Hence, in these case studies we have 8.8 and 9.6 records per user on average, respectively. Conversely, in the first three climate zones, we have 44, 37 and 104 records per user on average, respectively (table 1). As the proposed LRAB essentially *learns from the data*, it requires a sufficient *amount of data* to give the best results.

5 CONCLUSIONS AND FURTHER STEPS

In the present paper, we have applied robust locally weighted regression with an adaptive bandwidth to predict individual thermal comfort. The approach has been characterized and compared with the standard PMV approach. The experiments were carried out using publicly available datasets: they have shown that our LRAB outperforms the traditional PMV approach in predicting thermal comfort. Since LRAB can be computed quickly, and requires only a single setting parameter that is easily obtained, then if individual comfort responses are available, this method is feasible for use as a comfort measure in real time control.

The next step will be:*(i)* the comparison of our LRAB to other (nonparametric) regression mechanisms (e.g., CART, neural networks, k-NN) and *(ii)* the extension of the method to accept multiple environment variables (for example humidity, external air temperature etc.) in order to improve the above results. This mainly means the choice of a different kernel function to the one used here, in order to avoid a bias problem on the boundaries of the predictor space, a kind of problem that may be arise especially in the multidimensional case [26].

This work is part of the Strategic Research Cluster project ITOBO (supported by the Science Foundation Ireland), for which we are acquiring occupant comfort reports and fine grained sensor data, and constructing validated physical models of the building and its HVAC systems. The intention is then to use the comfort reports and sensor data as input to our LRAB method, and then to use the output of LRAB as the input to intelligent control systems which optimise the internal comfort for the specific individual occupants.

6 ACKNOWLEDGEMENTS

This work is supported by Intel Labs Europe and IRCSET through the Enterprise Partnership Scheme, and also in part by SFI Research Cluster ITOBO through grant No. 07.SRC.I1170.

REFERENCES

[1] Fanger, P. O. Thermal comfort: analysis and applications in environmental engineering. New York: McGraw-Hill; 1972.
[2] Gagge, A.P., Fobelets, A.P., Berglund, L.G. A standard predictive index of human response to the thermal environment. ASHRAE Trans 1986;92(2B):70931.
[3] ASHRAE-55. Thermal environmental conditions for human occupancy. American Society of Heating, Refrigerating and Air-conditioning Engineers Inc.; 1992.
[4] International Standard ISO 7730, Moderate thermal environments-estimation of the PMV and PPD indices and specification of the conditions for thermal comfort, Geneva; 1987.
[5] Busch, J.F., A tale of two populations: Thermal comfort in air conditioned and naturally ventilated offices in Thailand. Energy Buildings 1992;18:23549.
[6] de Dear, R. J., Leow, K.G., Ameen, A., Thermal comfort in the humid tropics. Part I. Climate chamber experiments on temperature preferences in Singapore. ASHRAE Trans 1991;97(1):8749.
[7] de Dear, R.J., Auliciems, A., Validation of the predicted mean vote model of thermal comfort in six Australian field studies. ASHRAE Trans 1985;91(2):45268
[8] Funakoshi, S., Matsuo, D., PMV-based train air-conditioning control system. ASHRAE Trans 1995;101(1):42330.
[9] MacArthur, J.W., Humidity and predicted-mean-vote-base (PMV-based) comfort control. ASHRAE Trans . 1986;92(1B):517.
[10] Scheatzle, D.G., Arch, D., The development of PMV-based control for a residence in a hot and arid climate. ASHRAE Trans 1991;97(2):100219.
[11] Wai, L.T., Albert, T.P.S., Implementation of comfort-based air-handling unit control algorithms. ASHRAE Trans . 2000;106(1):2944.

[12] Yang, K.H., Su, C.H., An approach to building energy savings using the PMV index. Building Environment 1997;32(1):2530.
[13] Sherman, M., A simplified model of thermal comfort. Energy Buildings 1985;8:3750.
[14] Federspiel, C.C., Asada, H., User-adaptable comfort control for HVAC systems. Trans ASME 1994;116:47486.
[15] Bingxin, M., Jiong, S., Yanchao, W.,Experimental design and the GA-BP prediction of human thermal comfort index, Seventh International Conference on Natural Computation,vol.2, pp. 771 - 775, 2011
[16] Atthajariyakul, S.,Leephakpreeda, T., Neural computing thermal comfort index for HVAC systems, Energy Conversion and Management, Volume 46, Issues 1516, September 2005, Pages 2553-2565, 10
[17] Kumar, S.,Mahdavi, A., Integrating thermal comfort field data analysis in a case-based building simulation environment, Building and Environment, Volume 36, Issue 6, pp 711-720, 2001
[18] de Dear, R.J., A global database of thermal comfort experiments, ASHRAE Trans., 104(1):1141-1152, 1998.
[19] ASHRAE. ASHRAE standard 55: thermal environmental conditions for human occupancy. American Society of Heating, Refrigerating and Air-conditioning Engineers; 2004.
[20] de Dear, R.J., Brager, G.S., Thermal comfort in naturally ventilated buildings: revisions to ASHRAE Standard 55, Energy and Buildings, Volume 34, Issue 6, Pages 549-561, 2002
[21] Feldmeier, M.C., Personalized Building Comfort Control, PhD Thesis, Massachusetts Institute of Technology, Sept. 2009
[22] Schumann, A., Wilson, N., Burillo, M., Learning User Preferences to Maximise Occupant Comfort in Office Buildings, IEA/AIE (1) 2010: 681:690.
[23] Cleveland, W.S., Robust Locally Regression and Smoothing Scatterplots, J. of the American Statistical Association,Vol.74, No. 368, 829-836, 1979.
[24] Hastie , T., Tibshirani, R., Friedman, J., The elements of Statistical Learning, Springer, 2001
[25] Cleveland, W.S., 1988. Locally weighted regression: an approach by local fitting. Journal of the American Statistical Association, 88, 596-610.
[26] Hastie , T. and Loader, C., Local regression: automatic kernel carpentry, Statist. Science 8, 120-143, 1993.
[27] Humphreys, M. A., Nicol, J. F., The validity of ISO-PMV for predicting comfort votes in every-day thermal environments, 2002, Energy and Buildings, 34(6), 667-684
[28] de Dear, R.J., Brager, G.S., Developing an adaptive model of thermal comfort and preference, Field studies of thermal comfort and adaptation, ASHRAE Technical Data Bulletin 14 (1) (1998) 27-49; ASHRAE Transactions 104 (1) (1998).
[29] Humphreys, M.A., Nicol, J.F., Effects of measurement and formulation error on thermal comfort indices in the ASHRAE database of field studies, ASHRAE Transactions 106 (2) (2000) 493-502.
[30] Humphreys, M.A., in: Proceedings of the World Renewable Energy Congress on the Recent Progress in the Adaptive Approach to Thermal Comfort, Brighton, UK, 2000, 438-443.
[31] IS0/DIS 7730:2003, Ergonomics of the thermal environment Analytical determination and interpretation of thermal comfort suing calculation of the PMV and PPD indices and local thermal comfort, 2003.
[32] ASHRAE-55. Thermal environmental conditions for human occupancy. American Society of Heating, Refrig- erating and Air-conditioning Engineers Inc.; 1992.
[33] International Standard ISO 7730, Moderate thermal environments-estimation of the PMV and PPD indices and specification of the conditions for thermal comfort, Geneva; 1987.