Noname manuscript No.
(will be inserted by the editor)

A Survey on Managing Users' Preferences in Ambient Intelligence

C. L. Oguego · J. C. Augusto · A. Muñoz · M. Springett

Received: date / Accepted: date

Abstract Understanding the importance of preference management in ambient intelligent environments is key to providing systems that are better prepared to meet users' expectations. This survey provides an account of the various ways that preferences have been handled in Artificial Intelligence (AI). Our analysis indicates that most of those techniques lack the ability to handle ambiguity and the evolution of preferences over time. Further exploration shows that argumentation can provide a feasible solution to complement existing work. We illustrate our claim by using an intelligent environment case study.

Keywords User Preferences · Preferences Handling · Ambient Intelligent · Argumentation

1 Introduction

The balancing of users' preferences is one of the most important [28,31]factors in designing successful Ambient Inteligence Systems (AmI), particularly in Ambient Assisted Living (AAL) [8]. For a system to be effective enough to support the user needs, it needs to know about users' expectations. This research aims to understand how to enhance user benefits from AAL technology through effective handling of preferences. Due to the impact of AAL on human lives [32,8] these system require complex problem solving and intelligent decision making capabilities. Preferences have a number of complexities. This may change over time, clash or conflict and be modified by experience. For example watching movies or listening to music may make us change our mind on an opinion we have about a product and we may decide to consume more or less of it. Preferences can even be imposed to some extent, such as lifestyle adjustment requested by doctors or insurance companies, e.g., the need to take medicines [6]. These changes are what the proposed solution should handle, as the system needs to observe changes in user's behaviour and have the ability to adapt to those changes. The system receives input from the user and various other sources (e.g. sensors and internet services), and if the system needs to provide feedback or help in making decisions, some real-time mechanism will be required to keep the system updated and to react appropriately. Because of the reasons above, our analysis of the systems in this area will be made on the basis of:

- Conflict Resolution
- Application to complex problem
- Decision Making
- Ability to reason and represent user's preferences
- Ability to handle time

Preference handling is one of the core issues in the design of any system that automates and supports decision making [19]. There have been various preference handling techniques proposed in artificial intelligence (e.g. CP-Net, UCP-Net, etc.) that address preference recommendation and preference-based representation

Chimezie Leonard Oguego
Research Group on Development of Intelligent Environments, Department of Computer Science, Middlesex University London, UK. E-mail: co527@live.mdx.ac.uk

Juan Carlos Augusto
Research Group on Development of Intelligent Environments, Department of Computer Science, Middlesex University London, UK. E-mail: j.augusto@mdx.ac.uk

Andrés Muñoz
Department of Polytechnic Sciences, Universidad Católica San Antonio de Murcia, Spain. E-mail: amunoz@ucam.edu

Mark Springett
Design for All Research Centre, Department of Computer Science, Middlesex University London, UK. E-mail: m.springett@mdx.ac.uk

problems. These techniques are to some extent useful in expressing users' preference and they have been implemented in various ways. However, they lack certain core aspects, such as not having the ability to reason and represent users' preference over time and not being able to handle inconsistencies. We illustrate these limitations of previous systems using an AmI system case study that deals with the automatic control of lights.

The paper is organised as follows. Section 2 describes the importance of preferences in AmI and the motivation behind the survey. Section 3 discusses some of the notable classical preference techniques in AI, and were tested to know if they are able to handle the kind of problems required by AmI systems. We continue our survey in section 4 turning our attention towards argumentation. Then the criteria defined above in this section were applied in comparing the preferences handling in classical AI and argumentation, and result is shown in Table 9. Finally we conclude in Section 5, along with a discussion on further work.

2 Motivation

A key mission of AmI is to enhance the way individuals interact with their environment and to promote safety which will enrich their lives [5]. AmI systems are meant to act proactively to anticipate preferences, in order to support users in making decisions [11]. Users should be empowered to personalize systems according to their preferences and this should be reasonably easy to do [7].

Preference handling can naturally lead to conflicts, such as when we have feelings or desire about what we want that conflict with what needs to be done (as will be seen in the case study description). These needs can be resolved if the system has the ability to understand such situations and present solutions to users which are perceived as natural. In addition, these preferences change with time, such as temperature preferences during seasons or lighting preferences during day and night.

Given that this survey focuses on finding the most suitable approach to handle preferences, we simplified the analysis as much as possible to illustrate some important points. In doing so we focus on managing the preferences of one user. Various works focus on one user and so we follow this line, leaving as further work to consider more than one user [35]. The following example will be used throughout this article to compare different features needed in managing preferences and to assess the extent to which current formalisms cover those desirable features.

2.1 AAL Case Study

Let us consider a smart home with a light management system capable in understanding the activities in a room, in order to make decisions for the user. The following description depicts a representative situation;

Dr. Bob is a 65 years old man who lives alone and loves reading at night. He usually falls asleep during reading process, leaving the lights on. Bob does not have any problem sleeping with the lights on, but he knows that keeping the light on when it is not needed increases the lighting bills and can also lead to other risks (such as, electrical issues) he does not want.

This description implies specifiable preferences such as stating how long he wants the light to be 'on' for when the system has detected that he is asleep. The idea is to have a system that will be intelligent enough to understand and react to significant changes. From the above description, three scenarios will be created, to illustrate and compare possible solutions to handle this type situation.

- **Scenario 1 (Bob comes home and prepares to go to bed):** *The light can be on until the system detects that the user is asleep, and then it turns the lights off after some time (specified by the user).*
- **Scenario 2 (Bob wakes up in the middle of the night):** *The user is asleep (light should be off at this point), then if the user wakes up (e.g. to use the toilet, etc.), the light should come on. If the user goes back to sleep, the light goes off after 10minutes.*
- **Scenario 3 (Bob leaves home):** *The user wakes up from sleep then the lights comes on. Then the user leaves home (e.g. to go to work). The system should turn the lights off after some time, if the user forgets to switch off the lights before leaving home.*

Table 1 summarise the highlights of the scenario above, with added sample times associated with its main stages.

2.2 Problem Statement

The scenarios above describe a light management system in operation within a bedroom of a smart house. Modern sensor can respond to human movement [34]. For example some systems typically used at offices will turn lights off when there is no movement for some time. However, this is unhelpful when we stay still absorbed in reading and suddenly the lights go off, breaking our concentration and forcing us to wave our arms to turn lights back on again. Conversely, as soon as movement is detected the system brings the lights back on. This

Table 1 Summary of Scenarios

Scenarios	Times	Significant Developments
Scenario 1	10pm	Bob enters the room and the light comes on (system detects movement and detects that its dark outside)
	11pm	Bob goes to bed (lights goes after some time (e.g. 10mins), if no movement is detected and pressure is detected on the bed sensor)
Scenario 2	2am	Bob wakes up in the middle of the night to use the toilet (lights comes on gradually up to 50%as soon as he gets out of bed)
	2:05am	Bob goes back to bed (lights goes off again after detecting that Bob is asleep).
Scenario 3	7am	Bob wakes up in the morning (lights comes on gradually when movement is detected out of bed).
	8am	Bob leaves home for work (light goes off automatically after a specified time, when Bob forgets to switch off the lights).

is fine for an office but not for a bedroom as moving during the night will cause the lights to go on and off intermittently several times.

There are two problems with the above type of systems which our work will try to address. One is that those office systems are set in such a way that (whilst not impossible to change), modifying the waiting time is usually beyond most typical users' capabilities. The other problem is that the system's notion of context is very limited. The only context they recognize is time without movement. Our research into these systems aims at providing ways for users to easily personalize the behaviour of the system through parameters which represent their preferences. The parameters which can facilitate this personalization, depend on the technology available in a given environment. We will keep the system functionality, the technology and the type of personalization simple. We still hope to demonstrate our system is more intelligent and capable enough to detect whether the person is sleeping or not and whether lights should be turned on or off in a sensible and flexible way.

A system like the one described can be created with current technology based on wireless sensors [34]. For example movement within a room can be perceived by the system using Passive Infrared Sensors (PIR) which measure spatial variations of heat. These sensors are commonly used in domestic alarm systems as a way to detect the movement of intruders. The type system this research aim to provide will enable the user to perform usual activities of moving around, getting in or out of bed, without going to turn off or on the lights. They can also set up preferences which affect the way the system reacts, for example how long should the system wait when there is no movement to turn lights off. The system can react by turning the lights on or off. This includes turning them half way (dimmed) when the user gets up from bed during the night. A similar system was used in [9], although that system was more centred on measuring quality of sleep and to detect dangerous situation that threaten the safety of the user. However, the system did not allow for preference personalization.

Table 2 Input-output to the smart lighting system

	input				Output
	Sensors		Human		Actuator
Type	Pressure Pad	PIR	Bob	Bob	Light Bulb
Values	on-off	on-off	actions	preferences	on-off dimmed

Table 2 summarizes the main parameters of the environment which can be perceived by the system to feed the context-awareness module along with the main ways the system can act upon the environment.

3 Preference in Classical AI

Preferences guide the choices of the user. So understanding several aspects of preference handling is important both for supporting active user control and designing systems that act on behalf of users. Preference is known as a core issue in the design of automated systems that aims to support the decision making of the users. It is therefore crucial to understand preference handling and the tools needed to help develop a system that can handle inconsistencies and deal with time.

One of the main aims of this paper is to address some existing classical preference in AI and then investigate their ability to deal with conflicting situations and represent users preference over time. These classical AI models will be discussed and analysed with the light case scenario to assess whether they are suitable for addressing the problem described. The classical preferences techniques include:

- CP-nets
- UCP-nets
- TCP-nets
- LCP-nets

Note that these are not all the preference handling techniques that exist in AI. In this survey we focus on these because they relate closely to the proposed solution this study aims to provide.

3.1 Conditional Preference Network (CP-Nets)

CP-net is known to be the most prominent qualitative approach for presenting preferences. Its clear graphical structure unifies an easy representation of user desires with cordial computational properties when computing the best outcome [33].

CP-nets is a directed graph representation of conditional preferences, where nodes represent variables and edges express preference links between variables. CP-nets exploits the power of conditional ceteris paribus rules [2] which enables a compact representation of human preferences. CP-net is naturally suited to simple applications (e.g. recommender systems to buy books on the web) in which preferences can easily be approximated by lexicographic rules on attributes with small domains [29]. It represents a complex preference over objects, using a set of atomic preferences each of which is a preference over a single object attributes given that the values of the other attributes are equal (the ceteris paribus principles). Such as: *Bob prefers X = x1 to X = x2*.

Example of how CP-net expresses preferences could be that of light choice. Figure 1 expresses preference of light choice in a house. This network consists of two variables B and R, standing for Bulb and Room respectively. A user might prefer coloured Bulb (Bc) to white Bulb (Bw), and their preference of whether the user wants the white bulb or a coloured one, could be conditioned based on the sitting room (Rs) or the bedroom (Rb): Bob prefers coloured bulb than white bulb in his bedroom and the white bulb in his sitting room than the coloured one.

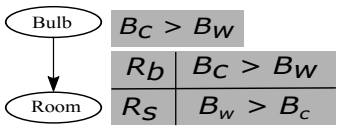


Fig. 1 CP-net for Light Choice: Bulb and Room

According to [18], "tools for representing and reasoning about Ceteris paribus preferences are important because they should aid in elicitation process for naive users".

Various studies of CP-nets are restricted to preferences that are strict, binary, known and complete [2]. This means all the features which an outcomes depends on are known. For instance, an individual who lives alone may prefer the light to be off during the day and wants the light on at night as long as it not her bed time (which can vary for users). These are strict preferences because this is a user who works during the day and sleeps at night. An example of a complete, strict and acyclic CP-net is illustrated in figure 2. The diagram illustrates a user (e.g. student) who prefers to have the light on when she studies at night and off when she studies during the day.

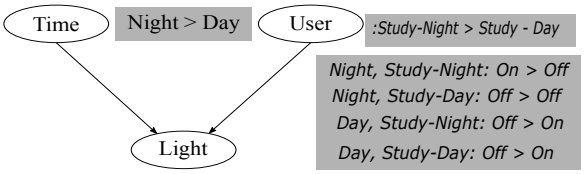


Fig. 2 A Strict, Complete, Binary, Acyclic CP-net.

However, when the users' preference is unknown, especially given that users preferences do change more often, a method that has the ability to handle the change over time and resolve conflicting situations will be needed, and these capabilities are not present in CP-net.

Formally, a preference relation is a partial pre-order on a set of alternatives (or outcomes) O. The expression $O > P$ means that O is preferred to P. If neither outcome is preferred to the other, they are said to be incomparable. CP-nets have been developed for such problems, rather than to compare alternatives in bits, as decision makers consider how the preference over one feature depends on the values of the other in the decision domain.

Let us consider the example of a student who lives alone and studies every night to prepare for an upcoming exam. She falls asleep almost every night without turning the lights off. This means that she falls asleep any time during the reading night, so it is unknown when the student actually falls asleep. It will be difficult to represent this using CP-net of *ceteris paribus* statements as the time when the student falls asleep is unknown.

As we have shown above, CP-nets has not advanced in a sufficient way for widespread use in complex, real world engineering applications [2] like AAL systems we want to address in this paper. Considering this, using CP-nets to represent the preferences of a user over time to help in dealing with conflicting situations will not be feasible.

3.2 Utility Conditional Preference Networks (UCP-Nets)

This model was proposed by [17] in 2001 by combining the appealing aspects of two existing preference models which are: GAI (a graphical model used to represent and manage independences among attributes [29]) and CP-nets. UCP-nets can be viewed as an extension of the CP-network that allows representation of qualitative utility information rather than simple preference

ordering. UCP-nets facilitate an incremental elicitation process, as they have a number of conceptual and computational advantages over GAI and CP-nets models, providing leverage with respect to interference and elicitation. The model is directed like CP-nets but preferences are quantified with utilities and by extending CP-nets with quantitative utility information. The expressive power is enhanced and dominance queries become computationally efficient. By introducing directionality and a ceteris paribus semantics to GAI, it allows utility functions to be expressed more naturally and optimization queries to be answered much effectively. Furthermore, this model allows for more powerful statements that are often more natural. This leads to more effective inference, and can be used in interactive elicitation processes in determining relevant parameters of UCP models in a specific decision scenario.

Despite identifying how UCP-net has various conceptual and computation advantages over CP-nets and GAI model, the authors emphasised in the concluding part of their study that practical experience and empirical studies are needed to gauge the ultimate effectiveness of UCP-nets [17]. This model has not currently been applied to the type of problem (light scenario) we are trying to solve.

In addition, one of the crucial problems faced in the use of a decision theoretic model is the elicitation of preference information [17]. This is one key motivation behind the development of the UCP-nets model. However, the problem this research aim to address goes beyond eliciting and representing of qualitative utility information, because our research aims to resolve conflicts and represent users' preference over time.

3.3 Tradeoffs-Enhanced Conditional Preference Networks (TCP-Nets)

This is another extension of CP-nets that can be referred to as a *relative important* statement for conditional preference networks with trade-off [20]. It is a graph based representation that encodes statement of (conditional) preferential independence and (conditional) relative importance [22]. To better understand this, Using our light scenario, a Bed-Room (B_R) can consists of both a White-Bulb (W_B) and Coloured-Bulb (C_B) (as values) and the Sitting-Room (S_R) consists of White-Fluorescent (W_F) and Coloured-Fluorescent (C_F) lights. The user may prefer to want to read in the Bed-Room than the Sitting-Room and wants to use Brighter Light (BL), (knowing that Bed-Room and Sitting-Room are preferentially independent), then the preference order over Bed-Room can be specified as White Bulb > Coloured Bulb, independently of the value of the Sitting-Room. In a similar way preference values over the Sitting-Room (if the user wants to read in the Sitting-Room), will be of White-Fluorescent > Coloured-Fluorescent, independent of the values of the Bed-Room. We can infer from this that W_B and W_F is a more preferred outcome than C_B and C_F.

TCP-net model basically empowers users to express trade-offs, which they are willing to concede among various preference criteria. The idea of conditional relative importance complements the one of conditional ceteris paribus independence [20] so as to provide for a richer conceptual framework and reason about the user's preferences. Figure 3 illustrates how TCP-nets extends CP-net by adding an ^{i}-arc from (B_R) to (BL) and (S_R) to (BL) (which describes the relative importance from B_R to BL and S_R to BL) and also ^{ci}-arc (which is for *conditional importance*) between (W_B) and (C_B) as well as (W_E) and (C_B). The relative importance of (W_B) and (C_B) or (W_E) and (C_B) depends on the assignment to (B_R) and (BL) or (S_R) and (BL) respectively.

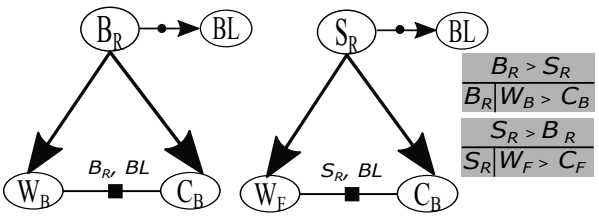


Fig. 3 Illustrations for Example TCP-Net

TCP-net has been used to propose a heuristic for estimating the preference ordering over the different choices at each stage in the composition to improve the efficiency of an algorithm (TCP-Compose*) [42]. This algorithm was presented to generate a set of composite services that achieve the desired functionality and constitute a non-dominated set of solutions with respect to user specified preferences and trade-offs over non-functional attributes [42]. Given that preference elicitation can be a bottleneck in many applications, TCP-net was suggested [20] as an enhancement of CP-nets for structuring, representing and reasoning about quality preference statements. It helps to make an optimally desirable solution for users who lack the knowledge, time or expert support required to specify complex multi-attribute functions.

In other words, TCP-nets provide a richer framework for representing users preferences, allowing stronger conclusions to be drawn among two variables. However, this research aims for more, such as providing a solution to resolve conflict, as well as representing and reasoning with users' preference over time rather than trading-off a less preferred outcome among two attributes.

3.4 Linguistic Conditional Preference Networks (LCP-Nets)

This model was proposed as a result of two important weaknesses spotted in *CP-nets models (including extended ones) [23], in expressing preferences in a Quality of Service setting (QoS). QoS dimensions are defined on continuous domain and *CP-nets only deal with finite domain variable. Using fuzzy linguistic terms [47], LCP-net was proposed to discretize continuous domains instead of crisp sets, so as to better capture user intentions, eliminating the need for the user to express preferences among values of a continuous domain. The other limitation is that, getting precise utility from non-speciality users is difficult, so giving numbers to express preferences is not always feasible. Current *CP-net models provide two alternative in this case. The original CP-nets expresses preferences through a more simple and intuitive relation, but suffer from low performance when comparing two assignments. On the other hand, UCP-nets perform the comparison more efficiently, but it is harder to get precise numeric utility values.

This version of CP-nets model (LCP) was developed to address the problem of expressing preferences, including non-functional properties. It provides programmers with an intuitive tool to express their preferences among services via their various qualities of services monitored at run-time. The advantage of fuzzy linguistic approaches in LCP-nets was acquired by combining UCP-nets and TCP-nets techniques, allowing preference modelling of more qualitative statement such as *I prefer the more or less V1 value for property X over exactly V2 if properties Y equals approximately VY and Z equals a bit more than VZ.* [22] expresses how LCP nets are easier to establish than writing several set of fuzzy rules that can be interdependent but qualitative to deal with user or QoS sensor imprecision. They further stated that LCP-nets allow users to express trade-offs among variables using i-arcs from TCP-nets and have CPTs (conditional preference table) similar to that of UCP-nets, but they express utilities with linguistic terms rather than numeric values. With LCP-nets it is possible to:

- Reveal relative importance of non-functional properties
- Elicit preferred assignment for specific QoS domain
- Indicate trade-offs between non-functional properties

Consider figure 4 where user preference on having the lights on (such as for security purpose) is detailed. The main goal here is for the user to have the light on at night. The goal is translated into preferences according

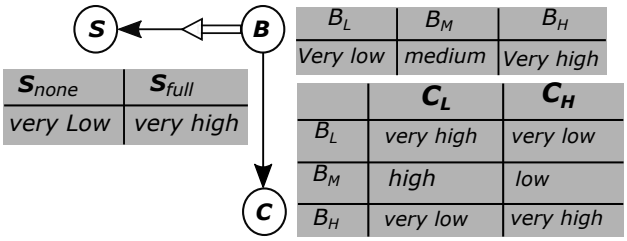


Fig. 4 The Imaging We service QoS preferences example using LCP-nets

to three of its QoS properties: security (S), Bright-light (B) and Colour-Bulb (C). The user would always prefer Bright-light over security but if the light is low, colour-bulb would be preferred so as to still have light at night.

In a different study, the same authors [23] that introduced the LCP-nets framework, applied the framework in tackling the multi-criteria decision making. This arises from run time choice among candidate service and several unrelated Quality of Service (QoS) properties. This was applied to select best service among a set of offers, given their dynamic non-functional properties. Generally, this new variant of CP-nets aids non-specialist programmers to express preferences in a qualitative way among values of the different QoS properties in this multi-criteria decision making process. This decision making process does not include resolving conflicting situations nor dealing with time, both of which are crucial in developing a system that will reason with users so as to assist in making vital decisions. Furthermore, according to the conclusion of [23], one of the limitations of LCP-nets is that it does not have the flexibility to share common preference among complex business processes decision sites, which indicates this method cannot address the complex scenario provided by this survey.

3.5 Other Related Approaches

Further research identified a system that facilitates web service selection when dealing with incomplete or inconsistent users' preferences [46]. The system explores the information of historical users to modify the active users' preference, improving the results of the selected services. Simulation conducted certifies the efficiency and effectiveness of the technique in conflict removal. The approach uses a CP-net model, similar to LCP-nets, which is used for the same reason to provide QoS-based late-binding of service invocations, adding extra agility to business process execution [22]. However, there is no evidence of the work having ability to manage user preferences over time.

Table 3 Table summarizing the pros and cons of preferences in classical AI

AI Preference Models	Pros	Cons
CP-Nets: Conditional Preference Networks	The promising approach for representing preferences in a qualitative and quantitative way is CP-nets [33]. CP-net-s offers a compact and arguably natural representation of preference information, necessary for solving many simple real world problems. Partial order can be created from small set of alternatives. Aids elicitation process for naive/non-expert users	Consistency of cyclic CP-net is not guaranteed. CP-nets are restricted to preferences that are strict, complete and binary and the dependency graph are usually assumed to be acyclic It will not be practical to create a partial order from large number of features Does not allows for the comparison or the ordering of all its alternatives
UCP-Nets: Utility Conditional Preferences Networks	UCP-nets facilitates an incremental elicitation process Has a number of conceptual and computational advantages over the CP-nets model, providing leverage as regards to inference and elicitation. Allows one to make more powerful statements that are often more natural and lead to more effective inferences.	Practical experience and empirical studies are needed as to gauge its effectiveness. To the best of our knowledge, there is not implementation of UCP-nets
TCP-Nets: Tradeoffs-enhanced Conditional Preference Network	With the limitation in CP-nets that does not express preferences over the variables themselves, TCP-nets was introduced to represent relatively importance between variables. Adds more important relations and conditional relative importance statement to ceteris Paribus statement	There is no research work reporting on the implementation of TCP-net as a solver [48]. To the best of our knowledge and that of [48], there is no implementation of TCP-nets. TCP-nets only deal with preferences (soft constraints) as hard constraints are not considered explicitly, which can be a real limitation when dealing with a wide real life problems that includes both constraints and preferences. The challenge of consistency of TCP-nets that is not conditionally acyclic.
LCP-Net: Linguistic Conditional Preference Network	This is a variant of CP-nets that has the same service ranking with all CP-nets extensions, but expressing CP-nets is easier LCP-nets. Applies to select the best service among a set of offers. Indicates trade-off between non-functional property and revealing relative important of non-functional property.	Focuses more on the mathematical modelling, allowing to aggregate the LCP-nets compared to CP-nets and TCP-nets that catches the eye due to their simplicity and expressiveness ([45]).
Service Selection Framework	This system was developed to utilize the information of historical users to enhance the preferences of the active users, improving the service selection results as the simulation results verified the effectiveness and efficiency in conflict removal [46].	Using CP-nets models, the approach tends to handle incomplete and inconsistent user preferences and but does not demonstration the ability to handle users' preferences over time.

4 Argumentation

The previous section provided an overview of several theoretical methods which can capture the process of selection based on preferences. However, from the point of view of Ambient Intelligence there are some further dimensions to the concept which are not explicitly addressed by those methods. Preferences sometimes are in conflict with each other. For example, sometimes there may be reasons to keep the lights on and also reasons to keep them off. Time also plays an important practical role, in particular preferences changing over time. For example, we prefer different levels of lighting at night or day and through different seasons we prefer different ambient temperatures. Computer Science has long investigated both these features of handling conflicts and inconsistencies. For example, this constitutes an interesting feature of Argumentation Systems [13, 36, 14]. Time has also been an important topic in various areas of CS and AI [4] and in particular in AmI [12, 37]. For all these reasons we believe argumentation is an option worth exploring, offering advantages which the methods in the previous section could not. We use this section to introduce some basics of argumentation, and in particular temporal argumentation, and later show with example scenarios how AmI desirable features are more naturally captured by the Argumentation System we describe.

Argumentation started to attract attention within CS during the 80's as a branch of AI focusing on finding ways to represent the processes humans follow when using common sense reasoning, particularly, taking into account exceptions and the way our conclusions adapt to the continuous influx of new information. Previously, Argumentation Systems appeared as an alternative to so-called 'non-monotonic reasoning', 'default reasoning' and 'defeasible reasoning' [24] [15]. The basic idea of argumentation is to create arguments in favour of and against a statement in order to determine if that statement can be acceptable or not and why [21]. Here we only briefly mention the concepts we need for the following subsections below and we refer the reader to [10] for full technical details and definitions.

Amongst other features Argumentation offers a way to represent defeasible reasoning, characterizing the skill that allows us to reason about a changing world where available information is incomplete or not very reliable. Argumentation systems have the ability to change conclusions according to the new information that comes to the system. The conclusions obtained by the system are "justified" through "arguments" supporting their consideration. In addition, an argument could be seen as a "defeasible proof" for a conclusion. The knowledge of new facts can lead to prefer a conclusion to a previous one, or to consider a previous inference no longer correct. In particular, there could exist an argument for a conclusion C and a "counter-argument", contradicting in some way the argument for C. An argument is a justification for a conclusion C if it is better than any other counter-argument for C. To establish the preference of an argument over the others, definition of preference criteria is required. Although several preference methods are possible, one that is widely used is "specificity": more specific information, i.e., better informed arguments. It is important to highlight that Argumentation Systems emphasize the role of inference justification and the dialectical process related to reasoning activities.

Given the limitations we have noticed in the handling of preferences by state of the art systems, including both handling of inconsistency and time-related information, we will use an Argumentation System which allows us to explicitly refer to time [10]. We refer to the reader to the original article for a full and detailed description of the underlying theoretical framework and here we provide only a short overview of the notation which is required to understand the description of the three scenarios further down in our article.

The system presented in [10] is actually an extension of a previous well-known argumentation framework [43]. The extension includes the addition of a temporal language $\mathcal{L}^{\mathbb{T}}$. This temporal language allows reification over time, properties, events and actions, which have been considered in the AI literature as key concepts to model a rational agent in a dynamic world. The system used to represent knowledge is based on a many-sorted logic [27], where different sorts are used to formalize the different groups of concepts represented in the system. The fundamental building blocks such as time, properties, events and actions listed above are only examples of possible sorts. Others can be added depending on needs.

The temporal language allows the association of knowledge to either "instants" ($\mathcal{T}$) or "intervals" ($\mathcal{I}$) so that we can express developments in real-world scenarios which happen (or are perceived to happen) instantaneously as well as developments which take time to complete. Example of an instant could be something that happened in a second in a system where seconds is the minimum time granularity, and an example of an interval will be a whole minute in that system. So if a PIR sensor is triggered only once in a second, e.g. at 17:06PM, then we can describe that as an instantaneous occurrence. If the same sensor is activated continuously for 15 seconds we can say that the activation of the sensor lasted for a while and those 15 seconds will become

an interval of time, e.g., from 17:06PM to 17:21PM. We can define familiar order relationships between units of time, so for example the following relationship between instants represent the notion of 'earlier time' $<: \mathcal{T} \times \mathcal{T}$ such that we can say $17:06\text{PM} < 17:21\text{PM}$. We can also define the notion of interval as a sequence of consecutive instants $\mathcal{I} = \{[i_1, i_2] \in \mathcal{T} \times \mathcal{T} | i_1 < i_2\}$ so that, for example, [17 : 06PM, 17 : 21PM] can be the interval where the sensor was continuously active. Auxiliary useful functions like $begin, end : \mathcal{I} \to \mathcal{T}$ can be defined to obtain the beginning and ending points of an interval: $begin([i_1, i_2]) =_{def} i_1$ and $end([i_1, i_2]) =_{def} i_2$.

We will consider a set of well-known relations in the literature as those between intervals defined by Hamblin [30] and later adopted by Allen [1] (see table 4). Although we have adopted Interval Logic as it is by far the most widespread way to represent and reason about time in CS, especially within AI, we understand other developers may wish to use other time handling options such as the one proposed in [44].

We will consider events as noticeable occurrences of

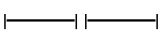
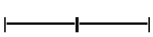
Relation	*Conditions*
BEFORE(X,Y)	
MEETS(X,Y)	
OVERLAP(X,Y)	
STARTS(X,Y)	
DURING(X,Y)	
FINISHES(X,Y)	
EQUAL(X,Y)	

Table 4 Interval-Interval relations (where X and Y represent two intervals). [1]

the real-world which can have an effect on a given situation. So for example the system sending a command to the light system causes it to produce light in the room. We will use a predicate $\text{Occurs}_{on}(e, i)$ ($\text{Occurs}_{on}(e, I)$) to indicate that an event e has occurred in an instant i (interval I).

For example: $\text{Occurs}_{on}(TurnOnLight, 7:00\text{AM})$.

We will assume the following is true of events:

$$\text{Occurs}_{on}(e, I) =_{def} \forall_{\mathcal{T}}\ i\ (\text{In}(i, I) \to \neg\text{Occurs}_{at}(e, i))$$

where $\text{In}(i, I) =_{def} Start(i, I) \vee Divides(i, I) \vee Ends(i, I)$ where these three predicates are true when an instant is at the beginning, 'inside' or the end of an interval. The definition given above for $\text{Occurs}_{on}(e, I)$ means the occurrence of an event in an interval implies it does not occurs inside the interval (this is usually called "non-homogeneity"). Also we consider "weak negation" over durative events in the following sense:

$$\neg\text{Occurs}_{on}(e, I) =_{def} \exists_{\mathcal{T}}\ i\ (\text{In}(i, I) \wedge \neg\text{Occurs}_{at}(e, i))$$

That is, consequently with the concept of non-homogeneity explained above, an event will be considered not to have occurred if a fragment (even just an instant) of it has not occurred.

We assume the world can be described as a set of elements or entities with specific properties for which we will use the following predicate: $\text{Holds}_{at}(p, i)$, $\text{Holds}_{at} \subseteq \mathcal{P} \times \mathcal{T}$, and $\text{Holds}_{on}(p, I)$, $\text{Holds}_{on} \subseteq \mathcal{P} \times \mathcal{I}$, denoting that p is a property that is true in the moment i or interval I respectively. Holds_{on} and Holds_{at} are related in the following way:

$$\text{Holds}_{on}(p, I) =_{def} \forall_{\mathcal{T}}\ i\ (\text{In}(i, I) \to \text{Holds}_{at}(p, i))$$

We will assume "homogeneity" of properties over an interval, meaning that if a property holds in an interval then it also holds in any of its subintervals. For example, if a sensor was activated during 15 minutes in a row, in particular it was activated in each minute of that interval (and each second of each minute):

$$\forall_{\mathcal{T}}\ i\ \forall_{\mathcal{I}}\ I\ (\text{Holds}_{on}(p, I) \wedge \text{In}(i, I) \to \text{Holds}_{at}(p, i)$$

$$\forall_{\mathcal{I}}\ I, I'\ (\text{Holds}_{on}(p, I) \wedge I' \sqsubseteq I) \to \text{Holds}_{on}(p, I'))$$

We consider "weak negation" of properties over intervals that can be obtained directly from the negation of the previous definition:

$$\neg\text{Holds}_{on}(p, I) =_{def} \exists_{\mathcal{T}}\ i\ (\text{In}(i, I) \wedge \neg\text{Holds}_{at}(p, i))$$

We will ascribe actions only to humans, so humans usually acting on their free will perform actions which typically causes some events to occur which in turn potentially change some properties of the world. We will consider that each human agent a from the sort of agents $\mathcal{A}$ has a repertoire $\mathcal{W}$ of possible actions g:

$$\forall_{\mathcal{A}}\ a\ \exists_{\mathcal{W}}\ g\ \text{Agent}(a, g) \tag{1}$$

There could be instantaneous actions Do_{at} (e.g., switching the light on) and durative actions Do_{on} (e.g., getting up from bed).

The explanations above mostly refer to the time related representation of the world. Now we turn focus more properly to inconsistency through the Argumentation System. That is how that information about a dynamic world can be grouped together to form arguments, reasons to believe or support the view of specific states of affairs in the real world we are describing.

We will assume our Knowledge Base is composed of a non-defeasible knowledge part $\mathcal{K}^{\mathbb{T}}$ which in turn is organized in two subsets, one set of facts $\mathcal{K}_G^{\mathbb{T}}$ (general knowledge) and one set of rules $\mathcal{K}_P^{\mathbb{T}}$ (particular knowledge), where $\mathcal{K}_P^{\mathbb{T}} \cup \mathcal{K}_G^{\mathbb{T}} = \mathcal{K}^{\mathbb{T}}$ and $\mathcal{K}_P^{\mathbb{T}} \cap \mathcal{K}_G^{\mathbb{T}} = \emptyset$. $\mathcal{K}_P^{\mathbb{T}}$ represents the safe facts of the world such as the existence of a specific bedroom in a specific house and a week in the calendar having seven days, and $\mathcal{K}_G^{\mathbb{T}}$ represents general laws, e.g. that if Monday is a day of a week then it has 24hours. There is also a finite set $\Delta^{\mathbb{T}}$ of *temporal defeasible rules* representing knowledge that our AmI system agent $a^{\mathbb{T}}$ is prepared to accept unless it finds counter-evidence. Rules in $\Delta^{\mathbb{T}}$ have the form $\alpha \succ\!\!- \beta$, where α and β are sets of literals of $\mathcal{L}^{\mathbb{T}}$. $\Delta^{\mathbb{T}\downarrow}$ will denote the set of basic instances of members of $\Delta^{\mathbb{T}}$. Given space restrictions, our simplified explanation of later sections will actually only use $\Delta^{\mathbb{T}\downarrow}$ instead of the usually preferable $\Delta^{\mathbb{T}}$ as we merely want to illustrate the potential of argumentation to capture certain key aspects of preferences handling and we postpone a more formal explanation for a future article.

We will largely adhere to the notation used in [10] and use $(\mathcal{K}^{\mathbb{T}}, \Delta^{\mathbb{T}})$ to denote a *temporal defeasible structure*, where $\mathcal{K}^{\mathbb{T}}$ is a temporal context and $\Delta^{\mathbb{T}}$ is a finite set of temporal defeasible rules. We will also adopt the same notion of *temporal defeasible consequence*, " $\mid\!\sim$ ", and the notion of A of $\Delta^{\mathbb{T}\downarrow}$ as a *temporal argument* for a temporal literal h and the associated notion of a subargument. Our explanations in the next section will actually be based on *grounded arguments*, $A^{\downarrow}$. Let $(\mathcal{K}^{\mathbb{T}}, \Delta^{\mathbb{T}})$ be a temporal defeasible structure of $a^{\mathbb{T}}$. $\mathbb{T}\mathbf{AStruc}(\Delta^{\mathbb{T}\downarrow})$ will be the set of temporal arguments that can be constructed from $(\mathcal{K}^{\mathbb{T}}, \Delta^{\mathbb{T}\downarrow})$.

Our notion of disagreement is related to time, so given a temporal function $\rho(\{h_1, h_2\})$ which determines whether two temporal literals h_1 and h_2 intersect in their time references, and given two temporal arguments $\langle A_1, h_1\rangle$ and $\langle A_2, h_2\rangle$, A_1 for h_1 and A_2 for h_2 *are in disagreement at least about an instant i*, $\langle A_1, h_1\rangle \bowtie_{\mathbb{T}} \langle A_2, h_2\rangle$, if and only if $\rho(\{h_1, h_2\}) \neq \emptyset$ and $\mathcal{K}^{\mathbb{T}} \cup \{h_1, h_2\} \vdash \bot$. So at least a common temporal reference is required between the temporal references of the arguments involved in the conflict.

A temporal argument $\langle A_1, h_1\rangle$ *counterargues* another temporal argument $\langle A_2, h_2\rangle$ in a basic literal h, if and only if there exists a subargument $\langle A, h\rangle$ of $\langle A_2, h_2\rangle$ such that $\langle A_1, h_1\rangle$ and $\langle A, h\rangle$ are in disagreement (in at least an instant i). Let $\succ$ be a partial order defined over elements of $\mathbb{T}\mathbf{AStruc}(\Delta^{\mathbb{T}\downarrow})$, we will say that a temporal argument $\langle A_1, h_1\rangle$ *defeats* another $\langle A_2, h_2\rangle$, $\langle A_1, h_1\rangle \gg_{\mathbf{tdef}} \langle A_2, h_2\rangle$, if and only if there exists a subargument $\langle A, h\rangle$ of $\langle A_2, h_2\rangle$ such as $\langle A_1, h_1\rangle$ counterargues $\langle A_2, h_2\rangle$ in h and $\langle A_1, h_1\rangle \succ \langle A, h\rangle$.

When there is a conflict between arguments, preference criteria are used to understand whether some arguments may be preferable to others. Specificity is one of such criterion which is widely used and it assesses whether one of the arguments is better informed than the rest (i.e., considers the information the others do plus something additional). Specificity is based on the structure of the arguments. It has the advantage of being independent from the application domain. Still, there are several other criteria which can be used to compare and select arguments. In some cases *Persistency* over time could be used as a reason to prefer an explanation over another. We assume properties persist unless we have reasons to believe otherwise. We will use predicates $\text{Change}_{at}^{+\ -}(p, i))$ and $\text{Change}_{in}^{+\ -}(p, I))$ to indicate that a proposition p changes its truth value from being true to false at an instant i or in an interval I respectively. The following axioms allow the detection of these situation:

$$\forall_{\mathcal{P}}\ p\ \forall_{\mathcal{T}} i(\text{Holds}_{at}(p, i-1) \wedge \neg\text{Holds}_{at}(p, i) \rightarrow \text{Change}_{at}^{+\ -}(p, i))$$

$$\forall_{\mathcal{P}}\ p\ \forall_{\mathcal{I}} I, I'(\text{MEETS}(I, I') \wedge \text{Holds}_{on}(p, I) \wedge \neg\text{Holds}_{on}(p, I') \rightarrow \text{Change}_{in}^{+\ -}(p, I'))$$

We can also consider analogous axioms for $\text{Change}_{at}^{-\ +}$ and $\text{Change}_{in}^{-\ +}$ for properties changing from being false to being true. Let $\langle A_1, h_1\rangle, \langle A_2, h_2\rangle \in \mathbf{TAStruc}(\Delta^{\mathcal{T}\downarrow})$, we say that A_1 for h_1 is *preferred under persistency* to A_2 for h_2, noted $\langle A_1, h_1\rangle \succ_{\mathbf{tpers}} \langle A_2, h_2\rangle$, if and only if $\langle A_2, h_2\rangle$ use persistency and $\langle A_1, h_1\rangle$ does not.

In the next sections we assume the following *precedence order* [40] between the preference criterion: $\Re = \{\succ_{\mathbf{tspec}}, \succ_{\mathbf{tpers}}\}, \succ_{\mathbf{tspec}} > \succ_{\mathbf{tpers}}$. This means we always try to apply specificity first. When the arguments are incomparable under specificity or they are equi-specific we apply the persistency criteria.

4.1 Light case study illustrated using Argumentation

The case study which has been described in section 2 in three different scenarios has been translated into a more technical form in table 5. Further below, we will illustrate how argumentation can handle users'preferences over time and potential conflictive scenarios. Tables 6, 7, and 8 show at the beginning the initial state of the world and then the evolution of the scenario through the *grounded arguments*, $A^{\downarrow}$.

At the end of each scenario we also illustrate the arguments in a tree format. However, the formal language on each table was not strictly used to create the tree, because we only wanted to demonstrate the basic idea behind each arguments. The time measurement assumed in the three scenarios is in minutes.

Table 5 Dynamics evolution of the Light Case Scenario as regards time

Scenario 1	Interval Relationship	$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge MEETS(I_3, I_4)$								
	Initial Stage	$Holds_{on}(Movement, I_0) \wedge \neg Holds_{on}(Sleeping, I_0) \wedge \neg Holds_{on}(OnBed, I_0) \wedge Holds_{on}(LightsOn, I_0)$								
	Properties	MoveDetected	MoveDetected	¬MoveDetected	¬MoveDetected	¬ MoveDetected				
		¬ Sleeping	¬ Sleeping	¬ Sleeping	Sleeping	Sleeping				
		¬ OnBed	OnBed	OnBed	OnBed	OnBed				
		LightsOn	LightsOn	LightsOn	LightsOn	¬ LightsOn				
Transition Cause		Do_{on} ($GoingToBed$, I_0)	$\neg Occurs_{at}$ ($MoveDectected$, $end(I_1)$)	$Holds_{on}(OnBed$, $I_2) \wedge \neg Holds_{on}$ ($MoveDectected$, $I_2) \wedge Length(I_2) > 10$	$Occurs_{on}$ ($SystemTurns$ $LightOff, I_1$)					
Intervals		I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8
Scenario 2	Interval Relationship	$MEETS(I_0, I_1) \wedge OVERLAP(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge BEFORE(I_3, I_4) \wedge MEETS(I_4, I_5) \wedge MEETS(I_5, I_6) \wedge MEETS(I_6, I_7) \wedge MEETS(I_7, I_8)$								
	Initial Stage	$Holds_{on}(Movement, I_0) \wedge \neg Holds_{on}(Sleeping, I_0) \wedge \neg Holds_{on}(OnBed, I_0) \wedge Holds_{on}(LightsOn, I_0)$								
	Properties	¬ MoveDetected	MoveDetected	MoveDetected	MoveDetected	MoveDetected	MoveDetected	¬ MoveDetected	¬ MoveDetected	¬ MoveDetected
		Sleeping	Sleeping	¬ Sleeping	¬ Sleeping	¬ Sleeping	¬ Sleeping	¬ Sleeping	Sleeping	Sleeping
		OnBed	OnBed	OnBed	¬ OnBed	¬ OnBed	OnBed	OnBed	OnBed	OnBed
		¬ LightsOn	¬ LightsOn	¬ LightsOn	¬ LightsOn	LightsOn	LightsOn	LightsOn	LightsOn	¬LightsOn
Transition Cause		$Occurs_{at}$ ($Movement$, $begin(I_1)$)	$Holds_{on}$ ($Movement$, $I_1) \wedge Length$ $(I_1) > 2$	Do_{on} ($GettingOut$ $OfBed$, I_2)	$Occurs_{on}$ ($MoveDetected$, $I_3) \wedge \neg Holds_{at}$ $(OnBed, begin, (I_3))$	Do_{on} ($Going$ $ToBed$, I_4)	$\neg Occurs_{at}$ ($MoveDectected$, $begin(I_6)$)	$Holds_{on}(OnBed, I_6)$ $\wedge \neg Holds_{on}$ $(Movement, I_6)$ $\wedge Length(I_6) > 10$	$Occurs_{at}$ ($SystemTurns$ $LightOff$, $begin(I_7)$)	
Intervals		I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8
Scenario 3	Interval Relatioship	$OVERLAP(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge MEETS(I_3, I_4) \wedge MEETS(I_4, I_5) \wedge MEETS(I_5, I_6)$								
	Initial Stage	$Holds_{on}(Movement, I_0) \wedge \neg Holds_{on}(Sleeping, I_0) \wedge \neg Holds_{on}(OnBed, I_0) \wedge Holds_{on}(LightsOn, I_0)$								
	Properties	¬ MoveDetected	MoveDetected	MoveDetected	MoveDetected	MoveDetected	¬ MoveDetected	¬ MoveDetected		
		Sleeping	Sleeping	¬ Sleeping	¬ Sleeping	¬ Sleeping	¬ Sleeping	¬ Sleeping		
		OnBed	OnBed	OnBed	¬ OnBed	¬ OnBed	¬ OnBed	¬ OnBed		
		¬ LightsOn	¬ LightsOn	¬ LightsOn	¬ LightsOn	LightsOn	LightsOn	¬ LightsOn		
Transition Cause		$Occurs_{at}$ ($AlarmRings$, $end(I_0)$)	$Holds_{on}$ ($Movement$, $I_1) \wedge Length$ $(I_1) > 2$	Do_{on} ($GettingOut$ $OfBed$, $begin(I_2)$)	$\neg Holds_{at}$ ($OnBed, begin$ $(I_3)) \wedge Holds_{on}$ $(Movement, I_3)$	Do_{on} ($Leaving$ $Home$, I_4)	$\neg Holds_{at}(Movement$, $I_5) \wedge Length(I_5)$ $> 15 \wedge \neg Holds_{at}$ $(OnBed, I_5)$	NotAtHome		
Intervals		I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8

Table 5 shows the progression in time of the three scenarios. The time "Intervals" row a the end of each scenario states the different relevant time periods for the scenarios, for example for Scenario 1 we use 5 different intervals $I_0, \ldots I_4$. The first "Interval Relationship" row in each scenario states how they relate to each other in time). For the first scenario it states all the different time intervals mentioned are consecutive to each other.

The "Initial Stage" row states how the system is supposed to be at the time the scenario is considered. For Scenario 1 it states that for the interval I_0 there is movement being detected by the PIR sensor, that the system believes the person is not sleeping and is not in bed and through the light sensor the system detects the lights are on in the bedroom.

The "Properties" section consist of a number of rows, one for each relevant property which depicts the state of the system under consideration. In Scenario 1 we can trace the evolution of movement detection (MoveDetected) as it evolves through time, and we can see movement is detected through the PIR sensor during I_0 and I_1 but movement is not detected ($\neg$ MoveDetected) in the whole of I_2, I_3 and I_4.

The "Transition Cause" row explains how the world transitions from one state to the next one, it explains change. For example, in Scenario 1, we can notice in I_0 the system believes the person is not in bed, and then at I_1 it believes the person is in bed. This change of believe is actually triggered by the action of the person going to bed ($\text{Do}_{on}(GoingToBed, I_0)$).

So to understand how the scenario evolves the reader has to see the values of the properties in two consecutive states of the system of the "Properties" area of the table, and look at the Transition cause under the first state which will explain how the system transitioned to the next state. In Scenario 1 the transition from I_1 to I_2 is caused by an event (hence the use of an $Occcurs$ predicate), then the transition from I_2 to I_3 is caused by a condition which triggers a rule in the system modifying the current belief of the system (hence the use of an $Holds$ predicate), the transition from I_3 to I_4 is caused by an event (hence the use of an Occurs_{on} predicate).

In summary we adopted the convention that the states of the system can change due to an action of the user (Do), an event related to a sensor (Occurs) or an update in the system's beliefs (Holds). Scenarios 2 and 3 evolve in similar fashion.

4.1.1 First Scenario

Table 6 focuses on the formalization of the first scenario. An informal description of what happened in the first scenario is given in Table 1 in section 2, then in Table 5 we provided the formalization of the evolution of that scenario in time through different states as well as of the actions, events and conditions which triggered those changes. Table 6 focuses on the defeasible rules which allows the system to reason with the knowledge of the world as it changes so that is context-aware and can react to the right contexts with sensible actuations.

The first line of the table shows the relationship of the intervals of time, these are the same as they were stated in table 5. The first column associates labels to the rules, for example (S1, R4) refers to the fourth rule of the first scenario.

The interpretation of the rules is according to the syntax and semantics given for the knowledge representation language given in [10] so we invite readers not familiar with it to use that publication as a support to understand the rules in this article.

For example R1 states that when the user performs the durative action of going to bed, it will have as a result the occurrence of the event getting on bed. R2 states that this event in turns has as an effect on the holding of the property of being on bed. R3 states if the system detects through sensors there is no movement detected at an instant (in this case at the end of I_1) then the system infers there is no movement at that time. R4 states if the systems has information the person is in bed and there is no movement for more than 10 units of time (for example 10 minutes) these are reasons to believe the person is sleeping. R5 states the believe the person is sleeping is a reason for the system to turn the lights off. R6 states when lights went off the consequence is that the lights are not on anymore (we assume as a simplification there is not other source of light and the room is dark).

Argument A for the first scenario: As known from the initial facts, the user turns the lights on when he enters the room. So there is a possibility, because of persistency, that the lights will remain on as reflected in the following argumentation tree in figure 5.

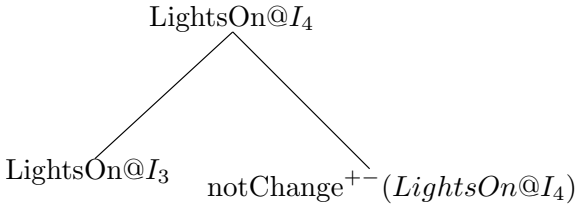


Fig. 5 First Scenario Argument A Tree for *LightsOn*

Argument B for the first scenario: There is an alternative explanation which is better informed than the previous one, given that the system has been pro-

Table 6 Knowledge Representation for First Scenario $\neg LightsOn$ *and* $LightsOn$

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge MEETS(I_3, I_4)$	
$\neg Holds_{on}(Movement, I_0) \wedge Holds_{on}(Sleeping, I_0) \wedge Holds_{on}(OnBed, I_0) \wedge \neg Holds_{on}(LightsOn, I_0)$	
(S1, R1)	$Do_{on}(GoingToBed, I_0) \succ\!\!- Occurs_{at}(GettingOnBed, begin(I_1))$
(S1, R2)	$Occurs_{at}(GettingToBed, begin(I_1)) \succ\!\!- Holds_{at}(OnBed, begin(I_1))$
(S1, R3)	$\neg Occurs_{at}(MoveDectected, end(I_1)) \succ\!\!- \neg Holds_{at}(Movement, end(I_1))$
(S1, R4)	$Holds_{on}(OnBed, I_2) \wedge \neg Holds_{on}(Movement, I_2) \wedge Length(I_2) > 10 \succ\!\!- Holds_{at}(Sleeping, end(I_2))$
(S1, R5)	$Holds_{on}(Sleeping, I_3) \succ\!\!- Occurs_{on}(SystemTurnsLigthsOff, I_3)$,
(S1, R6)	$Occurs_{on}(TurnLigtsOff, I_3) \succ\!\!- \neg Holds_{on}(LightsOn, I_4)$

grammed to understand when the lights are not needed ($\neg Holds_{on}(LightsOn, ...)$. The tree in figure 6 indicates that Bob was going to bed at I_0 and at I_1 Bob was in bed and stayed in bed till at I_2 as seen in the lower left part of the tree. Since there was no movement detected at I_2 (lower right part of the tree), the system has reasons to believe that Bob is asleep at I_2. Bob persists on sleeping all through I_3. At that moment the system infers that it is reasonable to turn the lights off. As a result, the lights are off at I_4.

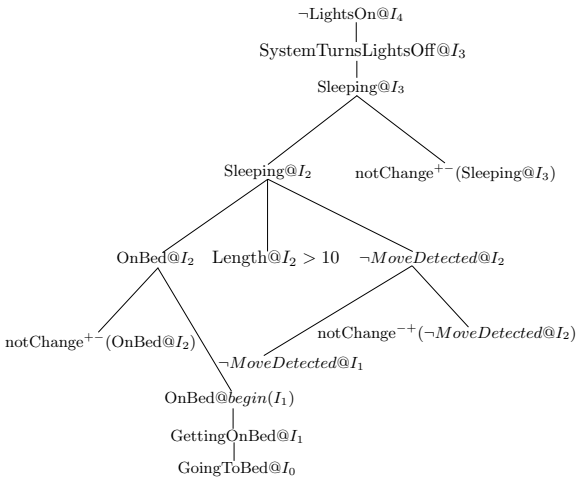


Fig. 6 First Scenario Argument B Tree for $\neg LightsOn$

From the first scenario, $A \bowtie_{\mathbb{T}} B$ about I_4, $B \succ_{\mathbf{tspec}} A$ because there is more information to support the reason that the user is asleep. Therefore, $B \gg_{\mathbf{tdef}} A$, now the system can state $\Delta^{\mathbb{T}\downarrow} \mid\sim \neg\text{Holds}_{on}(LightsOn, I_4)$.

4.1.2 Second Scenario

Table 7 focuses on the formalization of the second scenario. An informal description of what happened in the second scenario was given in Table 1 section 2. As previously stated, Table 5 provides the formalization of the evolution of that scenario in time through different states and also of actions, events and conditions that triggered the changes. Table 7 also focusses on defeasible rules just like Table 6 (same conventions apply for all rule tables).

Row labelled (S2, R1) states that when the system detects movement (maybe the user wakes up in the middle of the night to use the toilet), the property movement holds. Row labelled (S2, R2) states that if the movement continues over the next two minutes then it is believe that the user is not sleeping. Row labelled (S2, R3) states the durative action of the user getting out of bed, it will have as a result of the occurrence of the user is out of bed. This in turn has an effect in (S2, R4) that the user is not in bed anymore. S2, R5 states that if movement is detected via sensor, and if the user is not on bed, then the system turns the light on. Row labelled (S2, R6) states when the system turns the light on, then the lights stays on. Row labelled (S2, R7) states the durative action of going back to bed (after using the toilet) causes the event of the user being on bed. Rows labelled (S2, R7) to (S2, R12) are similar to rows labelled (S1, R1) to (S2, R6) of the first scenario.

Argument A for the Second scenario: As seen from the initial facts, the user turns on the light when he wakes up in the middle of the night, for example to use the toilet, so there is a possibility that the light will remain on at I_8until he turns it off again.

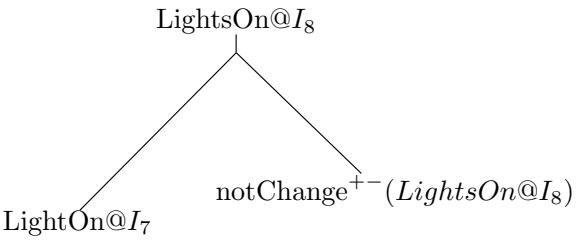


Fig. 7 Second Scenario Argument A Tree for $LightsOn$

Argument B for second scenario: There is an alternative description for the second scenario which is more informed than argument A. Thus, knowing that,

Table 7 Knowledge Representation for the Second Scenario of $\neg LightsOn$ *and* $LightsOn$

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge MEETS(I_3, I_4) \wedge MEETS(I_4, I_5)$ $\wedge MEETS(I_5, I_6 \wedge MEETS(I_6, I_7) \wedge MEETS(I_7, I_8)$	
$\neg Holds_{on}(Movement, I_0) \wedge Holds_{on}(Sleeping, I_0) \wedge Holds_{on}(OnBed, I_0) \wedge \neg Holds_{on}(LightsOn, I_0)$	
(S2, R1)	$Occurs_{at}(MoveDetected, begin(I_1)) \succ\!\!- Holds_{on}(Movement, I_1)$
(S2, R2)	$Holds_{on}(Movement, I_1) \wedge Length(I_1) > 2 \succ\!\!- \neg Holds_{on}(Sleeping, I_1)$
(S2, R3)	$Do_{on}(GettingOutOfBed, I_2) \succ\!\!- Occurs_{at}(GetsOutOfBed, end(I_2))$
(S2, R4)	$Occurs_{at}(GetsOutOfBed, end(I_2)) \succ\!\!- \neg Hold_{at}(OnBed, begin(I_3))$
(S2, R5)	$Occurs_{on}(MoveDetected, I_3) \wedge \neg Holds_{at}(OnBed, begin, (I_3)) \succ\!\!- Occurs_{on}(SystemTurnLightsOn, I_3)$
(S2, R6)	$Occurs_{on}(SystemTurnLightsOn, I_3) \succ\!\!- Holds_{on}(LightsOn, end(I_3))$
(S2, R7)	$Do_{on}(GoingToBed, , I4) \succ\!\!- Occurs_{at}(GettingOnBed, begin(I_5))$
(S2, R8)	$Occurs_{at}(GettingToBed, begin(I_5)) \succ\!\!- Holds_{at}(OnBed, end(I_5))$
(S2, R9)	$\neg Occurs_{at}(MoveDectected, begin(I_6)) \succ\!\!- \neg Holds_{at}(Movement, end(I_6))$
(S2, R10)	$Holds_{on}(OnBed, I_6) \wedge \neg Holds_{on}(Movement, I_6) \wedge Length(I_6), > 10 > -Holds_{on}(Sleeping, I_6)$
(S2, R11)	$Holds_{on}(Sleeping, I_6) \succ\!\!- Occurs_{on}(SystemTurnLightsOff, I_7)$
(S2, R12)	$Occurs_{on}(SystemTurnLightsOff, I_7) \succ\!\!- \neg Holds_{at}(LightsOn, end(I_8))$

the system has been programmed to understand that the lights are not needed $\neg Holds_{on}(LightsOn, ...)$. Figure 8 signifies that if Bob was going back to bed, such as at I_4, and was in bed at I_5 (as seen in the lower right hand side of the table) then Bob will be in bed from this interval onwards. Then for the system to have reason to believe that Bob is asleep at I_6, the system will not have detected any movement at I_6 and if this situation persists for the next 10 minutes, then the system concludes that Bob is now sleeping. Also if Bob persists on sleeping all through at I_6, then system assume at I_7 that it is reasonable to turn off the lights, as a result of that, the lights are off at I_8.

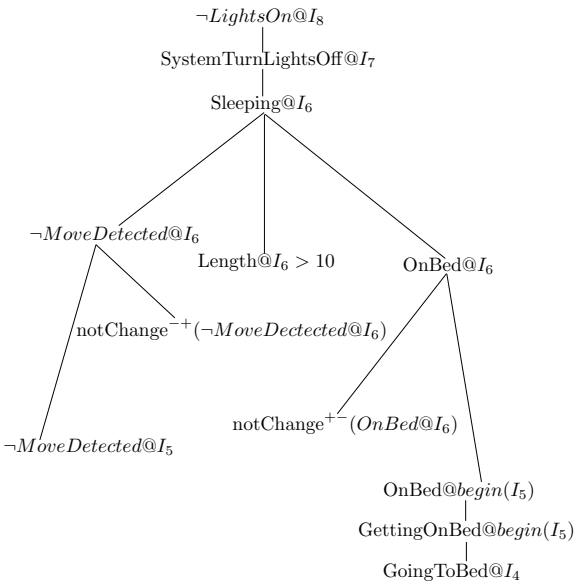


Fig. 8 Second Scenario Argument B Tree for $\neg LightsOn$

From the second scenario, $A \bowtie_{\mathbb{T}} B$ about I_8, $B \succ_{\mathbf{tspec}} A$ because there is more information to support the reason that the user has gone back to sleep so the system turns the light off. Therefore, $B \gg_{\mathbf{tdef}} A$, now the system can state $\Delta^{\mathbb{T}\downarrow} \mid\sim \neg\text{Holds}_{on}(LightsOn, I_8)$.

4.1.3 Third Scenario

Table 8 focues on the formalization of the third scenario. An informal description of the third scenario given in Table 1 section 2. Table 5 provides the formalization of the evolution of that scenario in time whilst Table 8 focusses on defeasible rules.

Row labelled (S3, R1) states the occurrence of the alarm ringing which will lead to awakening the user who will then begin to move. Row labelled (S3, R2) states if the movement continues for more than two minutes, then the system believes that the user is not sleeping. Row labelled (S3, R3) states the durative action of getting out of bed out being performed by the user, will result in the occurrence of the event getting off bed. Row labelled (S3, R4) states that this event in turns has an effect on the holding property of not being on bed. Row labelled (S3, R5) states when property states that user is not on bed and movement is detected with the use of sensors, then the system turns the light on. Row labelled (S3, R6) reflects the effect of the event which turns the light on. Row labelled (S3, R7) states that the durative action of the user leaving home will will lead to the occurrence of event left home. Row labelled (S3, R8) states that when the user has left no movement is expected ($\neg\text{Occurs}_{on}(Movement, I_5)$). Row labelled (S3, R9) states that if the property holds no movement and this state remains the same for over 15 units of

Table 8 Knowledge Representation for Third Scenario $\neg LightsOn$ *and* $LightsOn$

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge MEETS(I_3, I_4) \wedge MEETS(I_4, I_5) \wedge MEETS(I_5, I_6)$	
$\neg Holds_{on}(Movement, I_0) \wedge Holds_{on}(Sleeping, I_0) \wedge Holds_{on}(OnBed, I_0) \wedge \neg Holds_{on}(LightsOn, I_0)$	
(S3, R1)	$Occurs_{on}(AlarmRings, end(I_0)) \succ\!\!- Holds_{on}(Movement, I_1)$
(S3, R2)	$Holds_{on}(Movement, I_1) \wedge Length(I_1) > 2 \succ\!\!- \neg Holds_{on}(Sleeping, end(I_1))$
(S3, R3)	$Do_{at}(GettingOutOfBed, begin(I_2)) \succ\!\!- Occurs_{at}(GetsOutofBed, end(I_2))$
(S3, R4)	$Occurs_{at}(GetsOutofBed, end(I_2)) \succ\!\!- \neg Holds_{at}(OnBed, begin(I_3))$
(S3, R5)	$\neg Holds_{at}(OnBed, begin(I_3)) \wedge Holds_{on}(Movement, I_3) \succ\!\!- Occurs_{on}(SystemTurnLightsOn, I_3)$
(S3, R6)	$Occurs_{on}(SystemTurnLightsOn, I_3) \succ\!\!- Holds_{at}(LightsOn, I_3)$
(S3, R7)	$Do_{on}(LeavingHome, I_4) \succ\!\!- Occurs_{at}(LeftHome, end(I_4))$
(S3, R8)	$Occurs_{at}(LeftHome, end(I_4)) \succ\!\!- \neg Hold_{on}(Movement, I_5)$
(S3, R9)	$\neg Holds_{at}(Movement, I_5) \wedge Length(I_5) > 15 \wedge \neg Holds_{at}(OnBed, I_5) \succ\!\!- Occurs_{on}(SystemTurnLightsOff, I_6)$
(S3, R10)	$Occurs_{on}(SystemTurnLightsOff, I_6) \succ\!\!- \neg Holds_{on}(LightsOn, I_6)$

time (for example 15 minutes) and the bed sensor does not detect anyone on bed, this will make the system to infer that the user has left home and then turns off the light. Row labelled (S3, R10) reflect the effect of the system turning the light off.

Argument A for the third scenario: The initial facts show that the user turns the light on at I_5 when he wakes up in the morning, and as a result, there is a possibility that the light will remain on at I_6 as shown in the argumentation tree in figure 9.

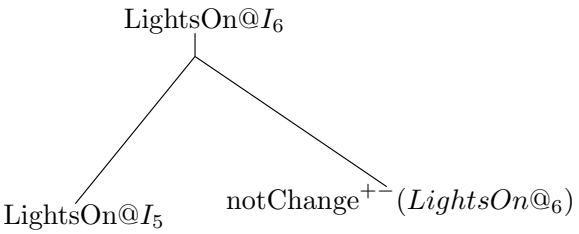


Fig. 9 Third Scenario Argument A Tree for $LightsOn$

Argument B for the third scenario: The alarm rings at I_0 which will awake the user. As he begins to move, this movement is detected by the system at I_1 and persists for the next 10 minutes, then the system understands that the user is awake as seen at the lower middle of the tree figure 10. When the user gets out of bed at I_2, then he is no longer on bed at I_3, as shown in the low right of the argumentation tree. This informs the system which then turns the light on at I_3. As the persistence of not being in bed continues from I_3 to I_4 the system continues to keep the lights on (unless the user turns the light off). The user is about to leave home at I_4, then at end of I_4 the user is out of home. It is possible that the user forgets to switch off the lights before he leaves home (which happened in this case).

As a result, the system turns the lights off at I_6 after no movement is detected at I_5 and not persistent state of $\neg Holds_{OnBed}$ remains at I_5. The resulting argument is explained in figure 10.

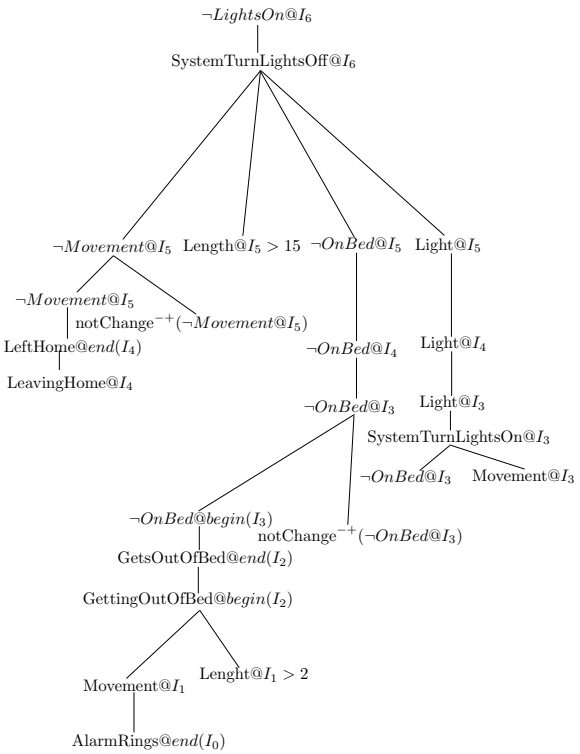


Fig. 10 Third Scenario Argument B Tree for $\neg LightsOn$

From the second scenario, $A \bowtie_{\mathbb{T}} B$ about I_6, $B \succ_{\mathbf{tspec}} A$ because there is more information to support the reason that the user has left home and then the system turns the light off. Therefore, $B \gg_{\mathbf{tdef}} A$, now the system can state $\Delta^{\mathbb{T}\downarrow} \mid\!\sim \neg\text{Holds}_{on}(LightsOn, I_6)$.

Table 9 Comparison of Classical Preferences in AI and Argumentation

	Preferences in Classical AI	**Argumentation**
Conflict Resolution	Preference methods in AI aim at decision-support systems which include web-based recommender systems, solving automated problems [39] and other interactive systems that aim to elicit and satisfy the users preference in order to give satisfactory recommendation.	Argumentation has been shown to handle complex situations in the previous work ([36]; [13]; [14]; [3]) especially in dealing with conflicts, and this has made researchers channel attention to this popular conflict resolution approach. Argumentation was shown to be a very relevant topic in AmI domain [31].
Application to complex problems	Most preferences handling methods in AI (CP-nets specifically) are restricted to preferences that are strict/complete (which a limitation identified by [2] in his study), as the outcome is already known. Strict or binary valued preference occurs in everyday life (such as, Bob prefers the light to be off at 10pm) but multivalued preference are not common (Bob prefers the light to be switched off in the evening). The latter is neither strict nor complete as the term "evening" is ambiguous thereby arising conflicting questions like when in the evening?	Argumentation covers wide range of disciplines just like preferences in AI, but has been applied in wider domains ([16]; [38]; [35]; [26]) in AmI as a knowledge representation and reasoning paradigm, for dealing with incomplete and inconsistent (contradictory) knowledge. Though, one of its main challenges is to design a formal system that enjoys desirable semantic properties and tractable computational complexity, while being easy to understand.
Decision Making	Preferences in AI are known to express preferential dependencies between attributes [28], such as when a Bob prefers to by hard cover mathematics book (which he reads often) and a paperback survey book (which might be read not more than twice). This indicates that the choice is dependent on the book type. This limits preferences in AI in the sense that they cannot model an arbitrary preference over a combinatorial domain.	In a usual context, once a decision is made a course of action is taken leaving behind other possible choices. However, decision making in argumentation is supported by reasoning, which will account for the characteristics of the various available alternatives [25]. This shows the ability that argumentation has to reason in a changing world where information is not complete. When new information surfaces, it gives considerations to obtain new reason to further conclusions or better reasons to sustain previous one.
Ability to reason and represent users' preferences	One main important factor of preferences in AI is that it aids elicitation of preference information from non-expert users directly or indirectly. However, certain questions are yet to be addressed, which include: How can these preferences be represented? How will they be used for reasoning? Can they be actually computed? [28].	Argumentation handles problems in AI which includes defeasible reasoning, (see [24]; [41]; [10]; [16]; [26]). Using the notion of instant or interval or both, demonstration has been made [10] to show how known problems of defeasible reasoning can be solved.
Ability to handle time	Despite the apparent importance of preferences in AI, as it has been applied to handle challenges pose in AI (such as: cognitive challenges, computational challenges, conceptual challenges and representational challenges) [19], there has been no recognition of preferences in AI having the ability to represent users' preference over time	Apart from the fact that argumentation is now a popular conflict resolution approach, and has been applied successfully in [31], it has also been theoretically proven that argumentation can be used to represent users' preferences over time [10].

5 Conclusion and Further Work

Although significant research has been conducted within Ambient Intelligence and despite being an area which in essence is user-centred, it has not been enough to facilitate a fluent inter-relation between AmI systems and user preferences.

The research we are conducting investigates ways to improve the understanding and management of preferences. Our analysis of existing work in preferences handling looked at various strategies developed to represent and reason with partial orders of various types to explain how humans choose amongst alternatives. We looked at several well-known alternatives like CP-nets and UCP-nets, which are seen as promising in other applications.

Our experience based on development of real Ambient Intelligence systems highlights the importance of some aspects which are not well supported in AI formalisms for preference handling. One feature which is naturally expected to deal with human preferences is the tension amongst these as sometimes we wrestle with what "we would like but we can't have". A preference linked to tasty food may be also associated with a preference from a health perspective advising against its consumption. Another feature of preferences is that they are dynamic, they change with time. It could be that we internally change our preferences based on repeated experiences, or that a change of preference is imposed externally to us, for example by health professionals or by weather.

This leads to the consideration of other formalisms which have been designed to be able to handle conflict and inconsistent information as well as knowledge in relation to time. We explain in section 4 how argumentation systems can be used to deal not only with conflictive preferences and with how preferences change with time but also based on those exercises, we started to understand that preferences can actually be embedded in arguments as a sort of constraint. We provided a summary of the pros and cons of the classical preferences in AI in table 3 and also a comparison between the classical AI approaches and argumentation in table 9. Although we provide one formal description which were illustrated in to three scenarios, as we tried to keep it simple, this is also consistent with what we have seen in the scenarios we explored.

Motivated by earlier reflections on the importance of preferences and the challenge they posse technically [6], (see figure 11), we made a first attempt at trying to find specific ways to manage this concept. Our initial investigations are positive. We believe we have found a useful tool to study the computational management

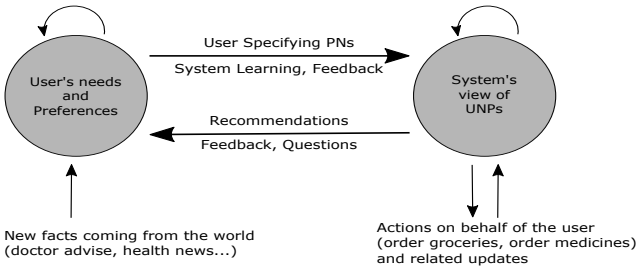


Fig. 11 Main interactions among user, system and real world affecting the dynamics of preference

of preferences and we will now be exploring ways to generalize our findings as well as on creating suitable bridges between users and systems, that is interfaces which can facilitate the flow of preferences from user to system and vice versa.

Acknowledgements : This work is supported by the Spanish MINECO under grant TIN2016-78799-P.

References

1. Allen, J.F.: Towards a general theory of action and time. Artificial intelligence **23**(2), 123–154 (1984)
2. Allen, T.E.: Making cp-nets (more) useful. In: AAAI, pp. 3057–3058 (2014)
3. Amgoud, L., Maudet, N., Parsons, S.: Modelling dialogues using argumentation. In: MultiAgent Systems, 2000. Proceedings. Fourth International Conference on, pp. 31–38. IEEE (2000)
4. Augusto, J.C.: The logical approach to temporal reasoning. Artificial Intelligence Review **16**(4), 301–333 (2001)
5. Augusto, J.C.: Intelligent Computing Everywhere, chap. Ambient Intelligence: The Confluence of Ubiquitous/Pervasive Computing and Artificial Intelligence (2007)
6. Augusto, J.C.: Reflections on ambient intelligence systems handling of user preferences and needs. In: Intelligent Environments (IE), 2014 International Conference on, pp. 369–371. IEEE (2014)
7. Augusto, J.C., Callaghan, V., Cook, D., Kameas, A., Satoh, I.: Intelligent environments: a manifesto. Human-centric Computing and Information Sciences **3**(1), 1–18 (2013)
8. Augusto, J.C., Huch, M., Kameas, A., Maitland, J., McCullagh, P.J., Roberts, J., Sixsmith, A., Wichert, R. (eds.): Handbook of Ambient Assisted Living - Technology for Healthcare, Rehabilitation and Well-being, *Ambient Intelligence and Smart Environments*, vol. 11. IOS Press (2012)
9. Augusto, J.C., Mulvenna, M.D., Zheng, H., Wang, H., Martin, S., McCullagh, P.J., Wallace, J.G.: Night optimised care technology for users needing assisted lifestyles. Behaviour & IT **33**(12), 1261–1277 (2014)
10. Augusto, J.C., Simari, G.R.: Temporal defeasible reasoning. Knowledge and information systems **3**(3), 287–318 (2001)
11. Aztiria, A., Izaguirre, A., Basagoiti, R., Augusto, J.C.: Learning about preferences and common behaviours of the user in an intelligent environment. In: BMI Book, pp. 289–315 (2009)

12. Aztiria, A., Izaguirre, A., Basagoiti, R., Augusto, J.C.: Learning about preferences and common behaviours of the user in an intelligent environment. In: Behaviour Monitoring and Interpretation - BMI - Smart Environments [an outgrow of BMI workshops], pp. 289–315 (2009)
13. Bandara, A.K., Kakas, A., Lupu, E.C., Russo, A.: Using argumentation logic for firewall policy specification and analysis. In: Large Scale Management of Distributed Systems, pp. 185–196. Springer (2006)
14. Bentahar, J., Alam, R., Maamar, Z., Narendra, N.C.: Using argumentation to model and deploy agent-based b2b applications. Knowledge-Based Systems **23**(7), 677–692 (2010)
15. Besnard, P., Hunter, A.: Elements of Argumentation. MIT Press (2008). URL http://mitpress.mit.edu/books/elements-argumentation
16. Bikakis, A., Antoniou, G.: Defeasible contextual reasoning with arguments in ambient intelligence. Knowledge and Data Engineering, IEEE Transactions on **22**(11), 1492–1506 (2010)
17. Boutilier, C., Bacchus, F., Brafman, R.I.: Ucp-networks: A directed graphical representation of conditional utilities. In: Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence, pp. 56–64. Morgan Kaufmann Publishers Inc. (2001)
18. Boutilier, C., Brafman, R.I., Hoos, H.H., Poole, D.: Reasoning with conditional ceteris paribus preference statements. In: Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence, pp. 71–80. Morgan Kaufmann Publishers Inc. (1999)
19. Brafman, R., Domshlak, C.: Preference handling-an introductory tutorial. AI magazine **30**(1), 58 (2009)
20. Brafman, R.I., Domshlak, C.: Tcp-nets for preference-based product configuration. In: Proceedings of the Forth Workshop on Configuration (in ECAI-02), pp. 101–106 (2002)
21. Caminada, M., Amgoud, L.: On the evaluation of argumentation formalisms. Artificial Intelligence **171**(5), 286–310 (2007)
22. Châtel, P., Malenfant, J., Truck, I.: Qos-based late-binding of service invocations in adaptive business processes. In: Web Services (ICWS), 2010 IEEE International Conference on, pp. 227–234. IEEE (2010)
23. Châtel, P., Truck, I., Malenfant, J.: A linguistic approach for non-functional constraints in a semantic soa environment. In: 8th International FLINS Conference on Computational Intelligence in Decision and Control (FLINS08), pp. 889–894 (2008)
24. Chesñevar, C.I., Maguitman, A.G., Loui, R.P.: Logical models of argument. ACM Computing Surveys (CSUR) **32**(4), 337–383 (2000)
25. Dix, J., Parsons, S., Prakken, H., Simari, G.: Research challenges for argumentation. Computer Science-Research and Development **23**(1), 27–34 (2009)
26. Ferrando, S.P., Onaindia, E.: Defeasible argumentation for multi-agent planning in ambient intelligence applications. In: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1, pp. 509–516. International Foundation for Autonomous Agents and Multiagent Systems (2012)
27. Gallier, J.H.: Logic for Computer Science: Foundations of Automatic Theorem Proving. Wiley (1987)
28. Goldsmith, J., Junker, U.: Preference handling for artificial intelligence. AI Magazine **29**(4), 9 (2009)
29. Gonzales, C., Perny, P.: Gai networks for decision making under certainty. In: IJCAI05–Workshop on Advances in Preference Handling. Citeseer (2005)
30. Hamblin, C.L.: Instants and intervals. In: The Study of Time, pp. 324–331. Springer (1972)
31. Homola, M., Patkos, T., Flouris, G., Šefránek, J., Šimko, A., Frtús, J., Zografistou, D., Baláž, M.: Resolving conflicts in knowledge for ambient intelligence. The Knowledge Engineering Review **30**(05), 455–513 (2015)
32. Huyck, C., Augusto, J., Gao, X., Botía, J.A.: Advancing ambient assisted living with caution. In: International Conference on Information and Communication Technologies for Ageing Well and e-Health, pp. 19–32. Springer (2015)
33. Lukasiewicz, T.: Introducing ontological cp- nets (2012)
34. Madrid, N.M., Fernández, J.M., Seepold, R., Augusto, J.: Sensors for ambient assisted living (aal) and smart homes
35. Muñoz, A., Augusto, J.C., Villa, A., Botía, J.A.: Design and evaluation of an ambient assisted living system based on an argumentative multi-agent system. Personal and Ubiquitous Computing **15**(4), 377–387 (2011)
36. Muñoz, A., Botía, J.A.: Developing an intelligent parking management application based on multi-agent systems and semantic web technologies. In: Hybrid Artificial Intelligence Systems, pp. 64–72. Springer (2010)
37. Muñoz, A., Botía, J.A., Augusto, J.C.: Using argumentation to understand ambiguous situations in intelligent environments. In: Ambient Intelligence Perspectives II - Selected Papers from the Second International Ambient Intelligence Forum 2009, Hradec Králové, Czech Republic, 16-17 September 2009, pp. 35–42 (2009)
38. Munoz, A., Botía, J.A., Augusto, J.C.: Intelligent decision-making for a smart home environment with multiple occupants. In: Computational Intelligence in Complex Decision Systems, pp. 325–371. Springer (2010)
39. Pigozzi, G., Tsoukiàs, A., Viappiani, P.: Preferences in artificial intelligence. Annals of Mathematics and Artificial Intelligence pp. 1–41 (2014)
40. Prakken, H.: A logical framework for modelling legal argument. In: Proceedings of the 4th international conference on Artificial intelligence and law, pp. 1–9. ACM (1993)
41. Prakken, H., Vreeswijk, G.: Logics for defeasible argumentation. In: Handbook of philosophical logic, pp. 219–318. Springer (2001)
42. Santhanam, G.R., Basu, S., Honavar, V.: Tcp- compose–a tcp-net based algorithm for efficient composition of web services using qualitative preferences. In: Service-Oriented Computing–ICSOC 2008, pp. 453–467. Springer (2008)
43. Simari, G.R., Loui, R.P.: A mathematical treatment of defeasible reasoning and its implementation. Artif. Intell. **53**(2-3), 125–157 (1992)
44. Stary, C., Pasztor, A.: Luisa logic for task-oriented user interface specification. International journal of intelligent systems **10**(2), 201–231 (1995)
45. Truck, I., Schmid, W.: A new proposal to represent the linguistic conditional preference networks. In: Intelligent Systems and Knowledge Engineering (ISKE), 2015 10th International Conference on, pp. 514–520. IEEE (2015)
46. Wang, H., Shao, S., Zhou, X., Wan, C., Bouguettaya, A.: Web service selection with incomplete or inconsistent user preferences. In: Service-Oriented Computing, pp. 83–98. Springer (2009)
47. Zadeh, L.A.: The concept of a linguistic variable and its application to approximate reasoningi. Information sciences **8**(3), 199–249 (1975)
48. Zhang, S., Mouhoub, M., Sadaoui, S.: Integrating tcp-nets and csps: The constrained tcp-net (ctcp-net) model. In: Current Approaches in Applied Artificial Intelligence, pp. 201–211. Springer (2015)