Methods for Model-Based Reasoning within Agent-Based Ambient Intelligence Applications*

Tibor Bosse, Fiemke Both, Charlotte Gerritsen, Mark Hoogendoorn, and Jan Treur

Vrije Universiteit Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
{tbosse, fboth, cg, mhoogen, treur}@few.vu.nl
http://www.few.vu.nl/~{tbosse, fboth, cg, mhoogen, treur}

Abstract. Within agent-based Ambient Intelligence applications agents react to humans based on information obtained by sensoring and their knowledge about human functioning. Appropriate types of reactions depend on the extent to which an agent understands the human and is able to interpret the available information (which is often incomplete, and hence multi-interpretable) in order to create a more complete internal image of the environment, including humans. Such an understanding requires that the agent has knowledge to a certain depth about the human's physiological and mental processes in the form of an explicitly represented model of the causal and dynamic relations describing these processes. In addition, given such a model representation, the agent needs reasoning methods to derive conclusions from the model and interpret the (partial) information available by sensoring. This paper presents the development of a toolbox that can be used by a modeller to design Ambient Intelligence applications. This toolbox contains a number of model-based reasoning methods and approaches to control such reasoning methods. Formal specifications in an executable temporal format are offered, which allows for simulation of reasoning processes and automated verification of the resulting reasoning traces in a dedicated software environment. A number of such simulation experiments and their formal analysis are described. The main contribution of this paper is that the reasoning methods in the toolbox have the possibility to reason using both quantitative and qualitative aspects in combination with a temporal dimension, and the possibility to perform focused reasoning based upon certain heuristic information.

Keywords: agent-based Ambient Intelligence applications, model-based reasoning, default logic, simulation, formal analysis.

1 Introduction

The relatively new field of Ambient Intelligence aims to develop and research intelligent environments with the goal to support people with their everyday life activities and tasks by means of electronic devices that are aware of humans and their environment; cf. [1, 2, 47]. For example, our car may monitor us and warn us when we are not fit enough to drive (this example is studied in case study 1A in Section 5.1.1). Similarly, the workspace of a technical worker may monitor the person's stress level and provide support in case it is too high (see case study 1B in Section 5.1.2). As another example, an elderly person may wear a device that monitors his or her well-being and generates an action when a dangerous situation is noticed (see case study 2A in Section 5.2.1).

* The basic reasoning approaches are based on [17]; the reasoning with incomplete information approach is based on [9].

Such applications can be based on possibilities to acquire sensor information about humans and their functioning, but more substantial applications depend on the availability of adequate knowledge for analysis of information about human functioning. If knowledge about human functioning is explicitly represented in the form of computational models in agents within an Ambient Intelligence application, these agents can show more understanding, and (re)act accordingly by performing actions in a knowledgeable manner to improve a person's wellbeing and performance. In recent years, human-directed scientific areas such as cognitive science, psychology, neuroscience and biomedical sciences have made substantial progress in providing an increased insight in the various physical and mental aspects involved in human functioning. Although much work still remains to be done, dynamic models have been developed and formalised for a variety of such aspects and the way in which humans (try to) manage or regulate them. From a biomedical angle, examples of such aspects are (management of) heart functioning, diabetes, eating regulation disorders, and HIV-infection; e.g., [10, 29]. From a psychological and social angle, examples are emotion regulation, attention regulation, addiction management, trust management, stress management, and criminal behaviour management; e.g., [122, 18, 30]. These models can be the basis for dedicated model-based reasoning methods that allow an agent in an Ambient Intelligence application to derive relevant conclusions from these models and available sensor information.

To give a concrete example, an intelligent Ambient Agent can use a domain model of diabetes to monitor and predict the blood levels of glucose and insuline in a patient by reasoning about this model. In such a case, the domain model itself could contain knowledge about the relevant physical processes (e.g., the speed of glucose metabolism), and the ambient agent could use model-based reasoning techniques on top of this domain model, in order to estimate the patient's current and future state over time. This information can then be used by the agent to provide support. Similarly, explicit reasoning about a domain model of visual attention would make it possible to predict attention levels of a technical operator during his/her duties (e.g., monitoring surveillance videos). When the attention level for an important aspect of a task threatens to decrease below a certain threshold, an ambient agent can decide to intervene.

This paper presents the development of a toolbox containing a variety of reasoning methods to support a designer of Ambient Intelligence applications. This toolbox has the following features:

- The reasoning methods are given at a conceptual formal specification level in an executable temporal logical format
- Both reasoning methods and approaches to (meta-level) control of the reasoning methods are specified in a unified manner
- Both numerical and logical aspects can be modelled in a hybrid manner
- It provides a unified variety of available and possible reasoning methods
- It also includes abduction and default reasoning methods to address incomplete information

Reasoning methods from the literature that can be included are, for example, assumption-based reasoning methods such as presented in [22,34, 35]. Given any candidate set of assumptions, as in [22, 34, 35] such reasoning methods can be applied to derive consequences including predicted observations that can be evaluated against actual observation information that is available or can be made available. However, in addition, the reasoning methods offered here can be used to generate such candidate sets of assumptions from a (possibly incomplete) initially given set of observations, so that a two-pass combination of reasoning processes occurs: first a reasoning process from observations to possible causes to determine a candidate set of assumptions, and next from these possible causes to (predicted) observations, in order to evaluate the candidate set. Moreover, as a further addition, both types of reasoning processes can be controlled by explicitly specified control methods.

In some further detail, in the current paper two classes of reasoning methods are distinguished that can be used to design agents that have knowledge about human behaviours and states over time in the form of explicitly represented models of the causal and dynamical relations involved:

(1) *Basic reasoning approaches to reason about human behaviours*

Hereby, existing model-based reasoning methods are taken as a basis to enable a matching of certain observed sensor information about a human with explicit models about the human process under analysis, and reason through these models (using both forward and backward reasoning techniques) in order to derive beliefs about other states in the process. The types of reasoning methods addressed cover a variety of phenomena such as causal and numerical simulation, qualitative reasoning and simulation, abductive reasoning [4, 38, 22, 33, 42], and explanation generation. The main contribution of this paper is that these reasoning methods are extended with several aspects, namely the possibility to reason using both quantitative and qualitative aspects in combination with a temporal dimension, and to perform focused reasoning based upon certain heuristic information. This extensive reasoning framework provides a solid basis for conceptual and detailed design of model-based agents in an Ambient Intelligence application that need such capabilities.

(2) *Reasoning based on incomplete information*

One additional element that needs to be addressed is the potential problem that information obtained via sensors about the human and the environment is often incomplete. Therefore, applications that require a high level of context awareness (see also [48, 49, 50]) depend on the availability of methods to analyse such incomplete information. Even when incomplete sensor information is refined on the basis of such models to create a more complete internal image of the environment's and human's state, still this may result in partial information that can be interpreted in different manners. Reactions of agents then depend on the extent to which they are able to handle the available multi-interpretable information. To do this, the agent needs a reasoning method to generate one or more possible interpretations, which cannot easily be done with the more simple reasoning techniques mentioned under (1). Techniques from the area of nonmonotonic logic can provide adequate analysis tools for reasoning processes concerning partial information. Within nonmonotonic logic approaches it is possible to formalise reasoning processes that deal with multiple possible outcomes, which can be used to model different possibilities of interpretation; see [29] for a similar perspective on the application of nonmonotonic logic tools. Thus, the second type of reasoning techniques introduced is based on generic model-based default reasoning. Hereby, standard default reasoning techniques are taken as a basis, and these are extended in this paper to enable the exploitation of the available causal model and the use of software tools to determine the different default extensions that form the possible interpretations, given the sensor information and the causal model. Moreover, by formally specifying the default rules in an executable temporal format (another contribution of this paper), and using formally specified heuristic knowledge for the control of the reasoning, explicit default reasoning processes can be generated.

In order to evaluate whether the reasoning techniques can indeed work for the domain of human-aware Ambient Intelligence applications, they have been applied to a variety of cases, ranging from support for car drivers, support in demanding circumstances, elderly care, and street crime.

This paper is organized as follows. Section 2 describes the formal modelling approach that is used throughout this paper. Next, in Section 3 and 4 the reasoning methods themselves are presented. Section 3 addresses (both uncontrolled and controlled variants of) the basic methods for belief generation, and Section 4 explains how default logic can be used for belief generation with incomplete information. Section 5 illustrates how these reasoning methods can be used, by performing simulation experiments in four example case studies: two for the basic reasoning methods, and two for the reasoning methods with incomplete information. Section 6 verifies a number of basic properties for model-based reasoning methods within agents against the simulation traces from Section 5. Related work is described in Section 7 and Section 8 concludes the paper with a discussion.

2 Modelling Approach

This section introduces the formal modelling approach that is used throughout this paper to express the reasoning methods. First, Section 2.1 explains the methodology of our approach. Section 2.2 briefly describes the Temporal Trace Language (TTL) for specification of dynamic properties (and its executable sublanguage LEADSTO), and Based on this language, Section 2.3 briefly explains how reasoning methods are formalised in this paper.

2.1 Methodology

In this section, a methodology to develop intelligent human-aware systems is presented. Here, human-aware is defined as being able to analyse and estimate what is going on in the human's mind (a form of mindreading) and in his or her body (a form of bodyreading). Input for these processes are observed information about the human's state over time, and dynamic models for the human's physical and mental processes. The methodology has been developed especially for agents that need to reason about the state of a human and that need to be human-aware to do the task the agent is designed for. It may be suitable for other types of agents that interact with humans, but that was not the aim for this research. Agents that communicate with humans without the need for awareness of the human may be better off using a less complex methodology. In particular, the type of awareness meant here refers to building images of (mental and physical) states of humans by reasoning based on acquired sensor data. This is an aspect that distinguishes Ambient Intelligence applications from other computer-based support systems.

The case studies in Section 5 include models that relate to drug intake, stress levels, blood pressure and the level of testosterone. A dynamic model of mental processes (sometimes called a Theory of Mind; e.g., [8]) may cover, for example, emotion, attention, intention, and belief.

The dynamic models of the domain can be integrated into an Ambient Intelligent application to create a human-aware application. By incorporating these domain models into an agent model, the intelligent agent gets an understanding of the processes in the agents' surrounding environment, which is a solid basis for knowledgeable intelligent behavior. Three different ways to integrate domain models within agent models can be distinguished. A most simple way is to use a domain model that specifically models human behavior directly as an agent model to simulate human behavior. This is shown in the right part of Figure 1. An example of this situation is to develop a domain model about emotion generation, and to incorporate this model within a virtual agent (e.g., in a serious game application), thereby enabling the agent to show emotions. For ambient agent models, domain models can be integrated within their agent models in two different ways, in order to obtain one or more (sub)models; see Figure 1. Here the solid lined arrows indicate information exchange between processes (data flow) and the dashed lined arrows the integration process of the domain models within the agent models. An analysis model performs analysis of the human's states and processes by reasoning based on observations (possibly using specific sensors) and the domain model. Returning to the example of the emotion model, an analysis model for this domain would be able to reason about a human's state of emotion, e.g., based on observations about this person's facial expression. Finally, a support model generates support for the human by reasoning based on the domain model. A support model can use information from the analysis and human agent models to reason about support possibilities. For the emotion case, a support model could for instance reason about support measures to bring a human user's emotion in a desired state.

In this article, the focus is on reasoning techniques which can be used primarily in analysis models. In the next subsections, a formal language is introduced to represent such reasoning techniques. The techniques themselves are introduced in Section 3 (basic methods) and Section 4 (methods to deal with incomplete information).

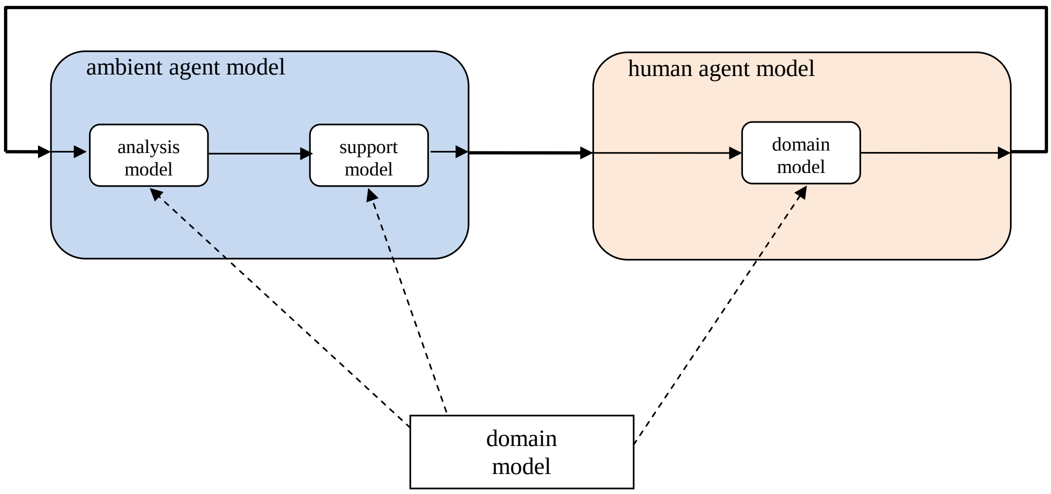

Figure 1. Overview of the multi-agent system architecture.

2.2 The Temporal Trace Language TTL

In order to execute and verify human-like ambience models, the expressive language TTL is used [14]. This predicate logical language supports formal specification and analysis of dynamic properties, covering both qualitative and quantitative aspects. By using the language TTL to develop the reasoning methods, the qualitative and quantitative aspects of TTL can be considered in the models and the reasoning about the models. This is one of the contributions of this paper. TTL is built on atoms referring to states, time points and traces.

A *state* of a process for (state) ontology Ont is an assignment of truth values to the set of ground atoms in the ontology. The set of all possible states for ontology Ont is denoted by STATES(Ont). To describe sequences of states, a fixed *time frame* T is assumed which is linearly ordered[†]. A *trace* γ over state ontology Ont and time frame T is a mapping $\gamma : T \rightarrow$ STATES(Ont), i.e., a sequence of states γ_t ($t \in T$) in STATES(Ont). The set of *dynamic properties* DYNPROP(Ont) is the set of temporal statements that can be formulated with respect to traces based on the state ontology Ont in the following manner. Given a trace γ over state ontology Ont, the state in γ at time point t is denoted by state(γ, t). These states can be related to state properties via the formally defined satisfaction relation |=, comparable to the Holds-predicate in the Situation Calculus: state(γ, t) |= p denotes that state property p holds in trace γ at time t. Based on these statements, dynamic properties can be formulated in a sorted first-order predicate logic, using quantifiers over time and traces and the usual first-order logical connectives such as $\neg, \wedge, \vee, \Rightarrow, \forall, \exists$. A special software environment has been developed for TTL, featuring both a Property Editor for building and editing TTL properties and a Checking Tool that enables formal verification of such properties against a set of (simulated or empirical) traces. TTL has been used to describe and verify a number of basic properties that may hold for the reasoning methods, see Section 6.

TTL has some similarities with situation calculus [46] and event calculus [36], which are two well-known formalisms for representing and reasoning about temporal domains. However, a number of important syntactic and semantic distinctions exist between TTL and both calculi. In particular, the central notion of the situation calculus - a situation - has different semantics than the notion of a state in TTL. That is, by a situation is understood a history or a finite sequence of actions, whereas a state in TTL is associated with the assignment of truth values to all state properties (a "snapshot" of the world). Moreover, in contrast to the situation calculus, where transitions between situations are described by actions, in TTL actions are in fact properties of states.
Although a time line has been recently introduced to the situation calculus [46], still only a single path (a temporal line) in the tree of situations can be explicitly encoded in the formulae. In contrast, TTL provides more expressivity by allowing explicit references to different temporally ordered sequences of states (traces) in dynamic properties (e.g., the trust monotonicity property).

In contrast to event calculus, TTL does not employ the mechanism of events that initiate and terminate fluents. Events in TTL are considered to be functions of the external world that can change states of components, according to specified properties of a system. Furthermore, similarly to the situation calculus, also in event calculus only one time line is considered.

Executable Format. To specify simulation models and to execute these models, the language LEADSTO [15], an executable sublanguage of TTL, is used. The basic building blocks of this language are causal relations of the format $\alpha \twoheadrightarrow_{e, f, g, h} \beta$, which means:

if	state property α holds for a certain time interval with duration g,
then	after some delay (between e and f) state property β will hold
	for a certain time interval of length h.

where α and β are state properties of the form 'conjunction of literals' (where a literal is an atom or the negation of an atom), and e, f, g, h non-negative real numbers.

[†] Note that the fact that this time frame is linearly ordered does not imply that the modeller needs to specify the temporal dependencies between any two events in the system. Thus, also systems of which the events produce a partial order can be modelled.

Because of the executable nature of LEADSTO and the possibility to describe direct temporal relations, this language has been used in Section 5 for the case studies.

2.3 Temporal Specification of Reasoning Methods

In this paper a dynamic perspective on reasoning is taken. In practical reasoning situations usually different lines of reasoning can be generated, each leading to a distinct set of conclusions. In logic semantics is usually expressed in terms of models that represent descriptions of conclusions about the world and in terms of entailment relations based on a specific class of this type of models. In the (sound) classical case each line (trace) of reasoning leads to a set of conclusions that are true in all of these models: each reasoning trace fits to each model. However, for non-classical reasoning methods the picture is different. For example, in default reasoning or abductive reasoning methods a variety of mutually contradictory conclusion sets may be possible. It depends on the chosen line of reasoning which one of these sets fits.

The general idea underlying the approach followed here is that a particular reasoning trace can be formalised by a sequence of *information states* M_0, M_1, ... Here any M_t is a description of the (partial) information that has been derived up to time point t. From a dynamic perspective, an inference step, performed in time duration D is viewed as a transition $M_t \rightarrow M_{t+D}$ of a current information state M_t to a next information state M_{t+D}. Such a transition is usually described by application of a deduction rule or proof rule, which in the dynamic perspective on reasoning gets a temporal aspect. A particular reasoning line is formalised by a sequence $(M_t)_{t \in T}$ of subsequent information states labelled by elements of a flow of time T, which may be discrete, based on natural numbers, or continuous, based on real numbers.

An information state can be formalised by a set of statements, or as a three-valued (false, true, undefined) truth assignment to ground atoms, i.e., a partial model. In the latter case, which is followed here, a sequence of such information states or reasoning trace can be interpreted as a partial temporal model. A transition relating a next information state to a current one can be formalised by temporal formulae the partial temporal model has to satisfy. For example, a modus ponens deduction rule can be specified in temporal format as:

$$\text{derived(I)} \wedge \text{derived(implies(I, J))} \twoheadrightarrow \text{derived(J)}$$

So, inference rules are translated into temporal rules thus obtaining a temporal theory describing the reasoning behaviour. Each possible reasoning trace can be described by a linear time model of this theory (in temporal partial logic).

In this paper, this dynamic perspective on reasoning is applied in combination with facts that are labelled with temporal information, and models based on causal or temporal relationships that relate such facts. To express the information involved in an agent's internal reasoning processes, the ontology shown in Table 1 is used which consists of the most prominent constructs for an agent within the area of Ambient Intelligence.

Table 1. Generic Ontology used within the Agent Model

Predicate	**Description**
belief(I:INFO_EL)	information I is believed
world_fact(I:INFO_EL)	I is a world fact
has_effect(A:ACTION, I:INFO_EL)	action A has effect I
Function to INFO_EL	**Description**
leads_to_after(I:INFO_EL, J:INFO_EL, D:REAL)	state property I leads to state property J after duration D
at(I:INFO_EL, T:TIME)	state property I holds at time T

As an example belief(leads_to_after(I:INFO_EL, J:INFO_EL, D:REAL)) is an expression based on this ontology which represents that the agent has the knowledge that state property I leads to state property J with a certain time delay specified by D. An example of a kind of dynamic modus ponens rule can be specified as

$$\text{belief(at(I, T))} \wedge \text{belief(leads\_to\_after(I, J, D))} \twoheadrightarrow \text{belief(at(J, T+D))}$$

This temporal rule states that if it is believed (by the agent) that I holds at T and that I leads to J after duration D, then it will be believed that J holds at T + D. This representation format will be used to formalise this and other types of model-based reasoning methods, as will be shown more extensively in Sections 3 and 4.

2.4 About Verification and Validation using TTL

In addition to a dedicated editor, the TTL software environment includes a TTL checker. This is a verification tool that takes TTL formulae and a set of traces, and automatically checks wether the formulae hold in these traces. As time plays an important role in TTL-formulae, special attention is given to continuous and discrete time range variables. Because of the finite variability property of TTL traces (i.e., only a finite number of state changes occur between any two time points), it is possible to partition the time range into a minimum set of intervals within which all atoms occurring in the property are constant in all traces. Quantification over continuous or discrete time variables is replaced by quantification over this finite set of time intervals. In order to increase the efficiency of verification, the TTL formula that needs to be checked is compiled into a Prolog clause. Compilation is obtained by mapping conjunctions, disjunctions and negations of TTL formulae to their Prolog equivalents, and by transforming universal quantification into existential quantification. Thereafter, if this Prolog clause succeeds, the corresponding TTL formula holds with respect to all traces under consideration. The complexity of the algorithm has an upper bound in the order of the product of the sizes of the ranges of all quantified variables. However, if a variable occurs in a holds atom, the contribution of that variable is no longer its range size, but the number of times that the holds atom pattern occurs (with different instantiations) in trace(s) under consideration. The contribution of an isolated time variable is the number of time intervals into which the traces under consideration are divided. Specific optimisations make it possible to check realistic dynamic properties with reasonable performance. Verification time is polynomial in the number of isolated time range variables occurring in the formula under verification. For more details, see [14] and [51].

In a real-world context an important part of a validation process is tuning of the model's parameters to a situation at hand, for example, a person's characteristics. Automated parameter tuning methods are available in the literature (e.g., [52]) and have been succesfully applied in AmI applications; see, for example, [10, 31, 32].

3 Basic Methods for Model-Based Reasoning

Below, in Section 3.1 a number of basic model-based reasoning methods for generation of beliefs are presented. Next, Section 3.2 presents a model and specification format to control the reasoning.

3.1 Model-Based Reasoning Methods for Belief Generation

Two types of reasoning methods to generate beliefs can be distinguished, that in applications can be used in combination:

- *Forward reasoning methods for belief generation*
  These are reasoning methods that follow the direction of time and causality, deriving from beliefs about properties (causes) at certain time points, new beliefs about properties (effects) at later time points.
- *Backward reasoning methods for belief generation*
  These are reasoning methods that follow the opposite direction of time and causality, deriving from beliefs about properties (effects) at certain time points, new beliefs about properties (potential causes) at earlier time points.

In comparison to assumption-based reasoning methods such as presented in [22, 34, 35], for any given candidate set of assumptions, as in [22, 34, 35] *forward* reasoning methods can be applied to derive consequences arriving at predicted observations to be evaluated against actual observation information. In contrast, the *backward* reasoning methods offered here can be used to generate such candidate sets of assumptions from an initially given set of observations. The overall reasoning process has two passes: first a *backward reasoning* pass from observations to possible causes to determine a candidate set of assumptions, and next a *forward reasoning* pass from these possible causes to (predicted) observations, in order to evaluate the candidate set. Both types of reasoning processes can be controlled by explicitly specified control methods, as discussed in Section 3.2.

In Section 3.1.1 the forward reasoning methods for belief generation are discussed, and in Section 3.1.2 the backward reasoning methods.

3.1.1 Forward reasoning methods for belief generation

Forward reasoning methods are often used to make predictions on future states, or on making an estimation of the current state based on information acquired in the past. The first reasoning method is one that occurs in the literature in many variants, in different contexts and under different names, varying from, for example, computational (numerical) simulation based on difference or differential equations, qualitative simulation, causal reasoning, execution of executable temporal logic formulae, and forward chaining in rule-based reasoning, to generation of traces by transition systems and finite automata. The basic specification of this reasoning method can be expressed as follows.

Belief Generation based on Positive Forward Simulation

If it is believed that I holds at T and that I leads to J after duration D, then it is believed that J holds after D.

∀I,J:INFO_EL ∀D:REAL ∀T:TIME

belief(at(I, T)) ∧ belief(leads_to_after(I, J, D)) ↠ belief(at(J, T+D))

If it is believed that I1 holds at T and that I2 holds at T, then it is believed that I1 and I2 holds at T.

belief(at(I1,T)) ∧ belief(at(I2, T)) ↠ belief(at(and(I1, I2), T))

Note that, if the initial beliefs are assumed correct, belief correctness holds for leads to beliefs, and positive forward correctness of leads to relationships holds, then all beliefs generated in this way are correct. A second way of belief generation by forward simulation addresses the propagation of negations. This is expressed as follows.

Belief Generation based on Single Source Negative Forward Simulation

If it is believed that I does not hold at T and that I leads to J after duration D, then it is believed that J does not hold after D.

$\forall$I,J:INFO_EL $\forall$D:REAL $\forall$T:TIME

belief(at(not(I), T)) $\wedge$ belief(leads_to_after(I, J, D)) $\twoheadrightarrow$ belief(at(not(J), T+D))

If it is believed that I1 (resp. I2) does not hold at T, then it is believed that I1 and I2 does not hold at T.

belief((at(not(I1),T))) $\twoheadrightarrow$ belief(at(not(and(I1, I2)), T))

belief(at(not(I2),T)) $\twoheadrightarrow$ belief(at(not(and(I1, I2)), T))

Note that this only provides correct beliefs when the initial beliefs are assumed correct, the property of 'belief correctness' (see Section 6.1) holds for leads to beliefs, and 'single source negative forward correctness' (see Section 6.2) holds for the leads to relationships.

Belief Generation based on Multiple Source Negative Forward Simulation

If for any J and time T, for every I that is believed to lead to J after some duration D, it is believed that I does not hold before duration D, then it is believed that J does not hold.

$\forall$I,J:INFO_EL $\forall$D:REAL $\forall$T:TIME

$\forall$I, D [belief(leads_to_after(I, J, D)) $\rightarrow$ belief(at(not(I), t-D)] $\twoheadrightarrow$ belief(at(not(J), T))

If it is believed that I1 (resp. I2) does not hold at T, then it is believed that I1 and I2 does not hold at T.

belief(at(not(I1),T)) $\twoheadrightarrow$ belief(at(not(and(I1, I2)), T))

belief(at(not(I2),T)) $\twoheadrightarrow$ belief(at(not(and(I1, I2)), T))

This provides correct beliefs when the initial beliefs are assumed correct, 'belief correctness' holds for leads to beliefs, and 'multiple source negative forward correctness' (see Section 6.2) holds for the leads to relationships.

3.1.2 Backward reasoning methods for belief generation

The basic specification of a backward reasoning method is specified as follows.

Belief Generation based on Modus Tollens Inverse Simulation

If it is believed that J does not hold at T and that I leads to J after duration D, then it is believed that I does not hold before duration D.

$\forall$I,J:INFO_EL $\forall$D:REAL $\forall$T:TIME

belief(at(not(J), T)) $\wedge$ belief(leads_to_after(I, J, D)) $\twoheadrightarrow$ belief(at(not(I), T-D))

If it is believed that not I1 and I2 holds at T and that I2 (resp. I1) holds at T, then it is believed that I1 (resp. I2) does not hold at T.

belief(at(not(and(I1, I2), T)) $\wedge$ belief(at(I2, T)) $\twoheadrightarrow$ belief(at(not(I1), T))

belief(at(not(and(I1, I2), T)) $\wedge$ belief(at(I1, T)) $\twoheadrightarrow$ belief(at(not(I2), T))

Belief Generation based on Simple Abduction

If it is believed that J holds at T and that I leads to J after duration D, then it is believed that I holds before duration D.

$\forall$I,J:INFO_EL $\forall$D:REAL $\forall$T:TIME

belief(at(J, T)) $\wedge$ belief(leads_to_after(I, J, D)) $\twoheadrightarrow$ belief(at(I, T-D))

If it is believed that I1 and I2 holds at T, then it is believed that I1 holds at T and that I2 holds at T.

belief(at(and(I1, I2), T)) $\twoheadrightarrow$ belief(at(I1,T)) $\wedge$ belief(at(I2, T))

As another option, an abductive causal reasoning method can be internally represented in a simplified form as follows.

Belief Generation based on Multiple Effect Abduction

If for any I and time T, for every J for which it is believed that I leads to J after some duration D, it is believed that J holds after duration D, then it is believed that I holds at T.

$\forall$I:INFO_EL $\forall$T:TIME

∀J [belief(leads_to_after(I, J, D)) → belief(at(J, T+D))] →→ belief(at(I, T))

If it is believed that I1 and I2 holds at T, then it is believed that I1 holds at T and that I2 holds at T.

belief(at(and(I1, I2), T)) →→ belief(at(I1,T)) ∧ belief(at(I2, T))

Belief Generation based on Context-Supported Abduction

If it is believed that J holds at T and that I2 holds at T and that I1 and I2 leads to J after duration D, then it is believed that I1 holds before duration D.

∀I,J:INFO_EL ∀D:REAL ∀T:TIME

belief(at(J, T)) ∧ belief(at(I2, T-D)) ∧ belief(leads_to_after(and(I1, I2), J, D)) →→ belief(at(I1, T-D))

If it is believed that I1 and I2 holds at T, then it is believed that I1 holds at T and that I2 holds at T.

belief(at(and(I1, I2), T)) →→ belief(at(I1,T)) ∧ belief(at(I2, T))

3.2 Controlling Belief Generation

An uncontrolled belief generation approach may easily lead to a combinatorial explosion of generated beliefs, for example, based on all conjunctions that can be formed. Therefore, a controlled approach where selection is done in some stage of the process can be more effective, this is therefore proposed in this paper. Often more specific knowledge is available based on which belief generation can leave some (or most) of the possible beliefs that can be generated out of consideration. To incorporate such selections, the following three approaches are possible: *selection afterwards overall, selection afterwards step by step, selection before*. Each of these options is discussed in more detail. Furthermore, it is discussed what selection criteria can be used to make such a selection. Below specification of selection criteria is generic in the sense that it leaves them abstract. They can be specified by the developer, but it is also possible that the developer specifies a large set of possible criteria from which a user can choose a small subset, and possibly adapt this set during the process.

3.2.1 Belief Generation Selection

Selection Afterwards Overall

In this approach first (candidate) beliefs are generated in an uncontrolled manner, and after that a selection process is performed based on some selection criterion. Two examples, one for a forward belief generation form and one for a backward belief generation form are as follows.

Controlled Belief Generation based on Positive Forward Simulation by Selection Afterwards Overall

If it is believed that I holds at T and that I leads to J after duration D, then it is believed that J holds after D.

∀I,J:INFO_EL ∀D:REAL ∀T:TIME

belief(at(I, T)) ∧ belief(leads_to_after(I, J, D)) →→ belief(at(J, T+D))

If it is believed that I1 holds at T and that I2 holds at T, then it is believed that I1 and I2 holds at T.

belief(at(I1,T)) ∧ belief(at(I2, T)) →→ belief(at(and(I1, I2), T))

If I is a belief and selection criterion s is fulfilled, then I is a selected belief.

belief(at(I, T)) ∧ s →→ selected_belief(at(I, T))

Controlled Belief Generation based on Multiple Effect Abduction by Selection Afterwards Overall

If for any I and time T, for every J for which it is believed that I leads to J after some duration D, it is believed that J holds after duration D, then it is believed that I holds at T.

∀I:INFO_EL ∀T:TIME

$\forall J$ [belief(leads_to_after(I, J, D)) $\rightarrow$ belief(at(J, T+D))] $\twoheadrightarrow$ belief(at(I, T))
If it is believed that I1 and I2 holds at T, then it is believed that I1 holds at T and that I2 holds at T.
belief(at(and(I1, I2), T)) $\twoheadrightarrow$ belief(at(I1,T)) $\wedge$ belief(at(I2, T))
If I is a belief and selection criterion s is fulfilled, then I is a selected belief.
belief(at(I, T)) $\wedge$ s $\twoheadrightarrow$ selected_belief(at(I, T))

This approach to control can only be applied when the number of beliefs that is generated in an uncontrolled manner is small. Otherwise more local approaches are better candidates to consider.

Selection Afterwards Step by Step

The step by step variant of selection afterwards performs the selection immediately after a belief has been generated. By such a local selection it is achieved that beliefs that are not selected can not be used in further belief generation processes, thus limiting these processes. The approach uses the temporal selection rule given above together with a slightly adapted form of specification to generate beliefs. Again two examples, one for a forward belief generation form and one for a backward belief generation form are as follows.

Controlled Belief Generation based on Positive Forward Simulation by Selection Afterwards Step by Step

If it is believed that I holds at T and that I leads to J after duration D, then it is believed that J holds after D.
$\forall$I,J:INFO_EL $\forall$D:REAL $\forall$T:TIME
selected_belief(at(I, T)) $\wedge$ belief(leads_to_after(I, J, D)) $\twoheadrightarrow$ belief(at(J, T+D))
If it is believed that I1 holds at T and that I2 holds at T, then it is believed that I1 and I2 holds at T.
selected_belief(at(I1,T)) $\wedge$ selected_belief(at(I2, T)) $\twoheadrightarrow$ belief(at(and(I1, I2), T))
If I is a belief and selection criterion s is fulfilled, then I is a selected belief.
belief(at(I, T)) $\wedge$ s $\twoheadrightarrow$ selected_belief(at(I, T))

Controlled Belief Generation based on Multiple Effect Abduction by Selection Afterwards Step by Step

If for any I and time T, for every J for which it is believed that I leads to J after some duration D, it is believed that J holds after duration D, then it is believed that I holds at T.
$\forall$I:INFO_EL $\forall$T:TIME
$\forall J$ [belief(leads_to_after(I, J, D)) $\rightarrow$ selected_belief(at(J, T+D))] $\twoheadrightarrow$ belief(at(I, T))
If it is believed that I1 and I2 holds at T, then it is believed that I1 holds at T and that I2 holds at T.
selected_belief(at(and(I1, I2), T)) $\twoheadrightarrow$ belief(at(I1,T)) $\wedge$ belief(at(I2, T))
If I is a belief and selection criterion s is fulfilled, then I is a selected belief.
belief(at(I, T)) $\wedge$ s $\twoheadrightarrow$ selected_belief(at(I, T))

This selection approach may be much more efficient than the approach based on selection afterwards overall, because the selection is made after every step. In most cases, this will lead to a smaller number of beliefs,

Selection Before

The approach of selection afterwards step by step can be slightly modified by not selecting the belief just after its generation, but just before. This allows for a still more economic process of focus generation. Again two examples, one for a forward belief generation form and one for a backward belief generation form are as follows.

Controlled Belief Generation based on Positive Forward Simulation by Selection Before

If the belief that I holds at T was selected and it is believed that I leads to J after duration D, and selection criterion s1 holds, then the belief that J holds after D is selected.

∀I,J:INFO_EL ∀D:REAL ∀T:TIME

selected_belief(at(I, T)) ∧ belief(leads_to_after(I, J, D)) ∧ s1 →→ selected_belief(at(J, T+D))

If the beliefs that I1 holds at T and that I2 holds at T were selected, and selection criterion s2 holds, then the conjunction of I1 and I2 at T is a selected belief.

selected_belief(at(I1,T)) ∧ selected_belief(at(I2, T)) ∧ s2 →→ selected_belief(at(and(I1, I2), T))

Controlled Belief Generation based on Multiple Effect Abduction by Selection Before

If for any I and time T, for every J for which it is believed that I leads to J after some duration D, the belief that J holds after duration D was selected, and selection criterion s1 holds, then it the belief that I holds at T is a selected belief.

∀I:INFO_EL ∀T:TIME

∀J [belief(leads_to_after(I, J, D)) → selected_belief(at(J, T+D))] ∧ s1 →→ selected_belief(at(I, T))

If the beliefs that I1 and I2 holds at T were selected, and selection criterion s2 holds then the belief that I1 holds at T is a selected belief.

selected_belief(at(and(I1, I2), T)) ∧ s2 →→ selected_belief(at(I1,T))

If the beliefs that I1 and I2 holds at T were selected, and selection criterion s2 holds then the belief that I2 holds at T is a selected belief

selected_belief(at(and(I1, I2), T)) ∧ s3 →→ selected_belief(at(I2, T))

3.2.2 Selection Criteria in Reasoning Methods for Belief Generation

Selection criteria needed for controlled belief generation can be specified in different manners. A simple manner is by assuming that the agent has knowledge which beliefs are relevant, expressed by a predicate in_focus. For example, the agent may need to answer a question about wether a specific belief or group of beliefs are true. It is beyond the scope of this paper whether such foci may be static or dynamic and how they can be determined by an agent. For cases that such general focus information is not available, the selection criteria can be specified in different ways.

If the assumption is made that the agent has knowledge about relevant beliefs, then any selection criterion s can be expressed as in_focus(I), where I is the property for which a belief is considered. The general idea is that if a belief can be generated, it is selected (only) when it is in focus. For example, for the two methods for selection afterwards, the temporal rule will be expressed as:

belief(at(I, T)) ∧ in_focus(I) →→ selected_belief(at(I, T))

For the method based on selection before, based on focus information the temporal rules will be expressed for the forward example by:

∀I,J:INFO_EL ∀D:REAL ∀T:TIME

selected_belief(at(I, T)) ∧ belief(leads_to_after(I, J, D)) ∧ in_focus(J) →→ selected_belief(at(J, T+D))

selected_belief(at(I1,T)) ∧ selected_belief(at(I2, T)) ∧ in_focus(and(I1, I2))

→→ selected_belief(at(and(I1, I2), T))

For the backward example of the method based on selection before, the temporal rules will be expressed by:

∀I:INFO_EL ∀T:TIME

∀J [belief(leads_to_after(I, J, D)) → selected_belief(at(J, T+D))] ∧ in_focus(I)

→→ selected_belief(at(I, T))

selected_belief(at(and(I1, I2), T)) ∧ in_focus(I1) →→ selected_belief(at(I1,T))

selected_belief(at(and(I1, I2), T)) ∧ in_focus(I2) →→ selected_belief(at(I2, T))

4 Model-Based Reasoning with Incomplete Information

In order to perform reasoning with incomplete information, Section 4.1 presents some formalisms for multiple interpretation of information. Section 4.2 addresses representation in default logic, Section 4.3 addresses model-based refinement of partial information, and Section 4.4 addresses control of the default reasoning.

4.1 Multiple Interpretation

Reasoning to obtain an interpretation of partial information can be formalised at an abstract generic level as follows. A particular interpretation for a given set of formulae considered as input information for the reasoning, is formalised as another set of formulae, that in one way or the other is derivable from the input information (output of the reasoning towards an interpretation). In general there are multiple possible outcomes. The collection of all possible interpretations derivable from a given set of formulae as input information (i.e., the output of the reasoning towards an interpretation) is formalised as a collection of different sets of formulae. A formalisation describing the relation between such input and output information is described at an abstract level by a multi-interpretation operator.

The input information is described by propositional formulae in a language L_1. An interpretation is a set of propositional formulae, based on a language L_2.

a) A *multi-interpretation operator* MI with input language L_1 and output language L_2 is a function $MI : \mathcal{P}(L_1) \rightarrow \mathcal{P}(\mathcal{P}(L_2))$ that assigns to each set of input facts in L_1 a set of sets of formulae in L_2.

b) A multi-interpretation operator MI is *non-inclusive* if for all $X \subseteq L_1$ and $S, T \in MI(X)$, if $S \subseteq T$ then $S = T$.

c) If $L_1 \subseteq L_2$, then a multi-interpretation operator MI is *conservative* if for all $X \subseteq L_1$, $T \in MI(X)$ it holds $X \subseteq T$.

The condition of non-inclusiveness guarantees a relative maximality of the possible interpretations. Note that when MI(X) has exactly one element, this means that the set $X \subseteq L_1$ has a unique interpretation under MI. The notion of multi-interpretation operator is a generalisation of the notion of a nonmonotonic belief set operator, as introduced in [24]. The generalisation was introduced and applied to approximate classification in [27]. A reasoner may explore a number of possible interpretations, but often, at some point in time a reasoner will focus on one (or possibly a small subset) of the interpretations. This selection process is formalised as follows (see [27]).

a) A *selection operator* s is a function $s : \mathcal{P}(\mathcal{P}(L)) \rightarrow \mathcal{P}(\mathcal{P}(L))$ that assigns to each nonempty set of interpretations a nonempty subset: for all A with $\phi \neq A \subseteq \mathcal{P}(L)$ it holds $\phi \neq s(A) \subseteq A$. A selection operator s is *single-valued* if for all non-empty A the set s(A) contains exactly one element.

b) A *selective interpretation operator* for the multi-interpretation operator MI is a function $C : \mathcal{P}(L_1) \rightarrow \mathcal{P}(L_2)$ that assigns one interpretation to each set of initial facts: for all $X \subseteq L_1$ it holds $C(X) \in MI(X)$.

4.2 Representation in Default Logic

The *representation problem* for a nonmonotonic logic is the question whether a given set of possible outcomes of a reasoning process can be represented by a theory in this logic. More specifically, representation theory indicates what are criteria for a set of possible outcomes, for example, given by a collection of deductively closed sets of formulae, so that this collection can occur as the set of outcomes for a theory in this nonmonotonic logic. In [39] the representation problem is solved for default logic, for the finite case. Given this context, in the current section Default Logic is chosen to represent interpretation processes. For the empirical material analysed, default theories have been specified such that their extensions are the possible interpretations.

A *default theory* is a pair $\langle D, W\rangle$. Here W is a finite set of logical formulae (called the background theory) that formalise the facts that are known for sure, and D is a set of default rules. A default rule has the form: α: β / γ. Here α is the precondition, it has to be satisfied before considering to believe the conclusion γ, where the β, called the justification, has to be consistent with the derived information and W. As a result γ might be believed and more default rules can be applied. However, the end result (when no more default rules can be applied) still has to be consistent with the justifications of all applied default rules. *Normal default theories* are based on defaults of the form α: β / β. In the approach *supernormal* default rules will be used: normal default rules where α is trivial: true. Such supernormal rules are denoted by β / β or : β / β; they are also called prerequisite-free normal defaults. For more details on Default Logic, such as the notion of extension, see, e.g., [40, 45].

4.3 Default logic for model-based refinement of partial information

The *causal theory* CT of the agent consists of a number of statements a $\rightarrow$ b for each causal relation from a to b, with a and b atoms. Sometimes some facts to indicate that some atoms exclude each other (for example, $\neg$(has_value(temperature, high) $\wedge$ has_value(temperature, low) assuming that temperature can only be high or low), or that at least one of a set of atoms is true, (for example: has_value(pulse, high) $\vee$ has_value(pulse, normal) $\vee$ has_value(pulse, low)) are included in this set. A set of literals S is *coherent* with CT if S $\cup$ CT is consistent. The set S is called a *maximal coherent* set for CT if it is coherent, and for all sets T coherent with CT with S $\subseteq$ T it holds S = T. Let X be a set of formulae. The multi-interpretation operator $MI_{CT}(X)$ is defined by

$$MI_{CT}(X) = \{ Cn(X \cup CT \cup S) \mid S \text{ maximal coherent with CT} \}$$

This operator defines the set of all complete refinements of X which are coherent with the causal model for the partial information the agent may have at some point in time (indicated by set of literals X). This operator has been defined above in an abstract manner, and only indicates the possible outcomes of a reasoning process, not the steps of the reasoning process itself. A next step is to obtain a representation of this operator in a well-known formalism such as default logic. Based on this default logic representation, reasoning processes can be defined that can be performed to obtain one or more of the interpretations.

The following Default Theory $\Delta_{CT}(X) = \langle W, D\rangle$ can be used to represent the multi-interpretation operator MI_{CT} (notice that this is a supernormal default theory); see also [24 below], Theorem 5.1:

$$W = CT \cup X$$
$$D = \{ (\text{true: a / a}) \mid \text{a literal for an atom occurring in CT} \}$$

Here a literal is an atom or a negation of an atom. That this default theory represents MI_{CT} means that for any set X indicating partial information the set of interpretations defined by $MI_{CT}(X)$ can be obtained as the set of all extensions of the default theory $\Delta_{CT}(X)$. This representation allows to determine the interpretations by using known methods and tools to determine the extensions of a default theory. One of these methods is worked out in a tool called Smodels, based on answer set programming; cf. [25]. Another method to determine the extensions of a default theory is by controlled or prioritised default reasoning. This method will be explained in the next subsection.

4.4 Controlled Default Reasoning

As discussed earlier, to formalise one reasoning trace in a multiple interpretation situation, a certain selection has to be made, based on control knowledge that serves as a parameter for the interpretation to be achieved. Variants of Default Logic in which this can be expressed are Constructive Default Logic [54] and Prioritized Default Logic [19, 20]. A *Prioritized Default Theory* is a triple $\langle D,W,<\rangle$, where $\langle D,W\rangle$ is a Default Theory and < is a strict partial order on D. *Constructive Default Logic*, see [54], is a Default Logic in which selection functions are used to control the reasoning process. Selection functions take the set of consequents of possibly applicable defaults and select one or a subset of them. A selection function can represent one of the different ways to reason from the same set of defaults, and thus serves as a parameter for different reasoning traces (achieving different interpretations). This knowledge determines a selection operator (see Section 4.1).

The generic simulation model for default reasoning described below is an executable temporal logical formalisation of Constructive Default Logic, based on the temporal perspective on default and nonmonotonic reasoning as developed in [26]. The input of the model is (1) a set of normal default rules, (2) initial information, and (3) knowledge about the selection of conclusions of possibly applicable rules. The output is a trace which describes the dynamics of the reasoning process over time. Globally, the model can be described by a generate-select mechanism: first all possible (default) assumptions (i.e., candidate conclusions) are generated, then one conclusion is selected, based on selection knowledge. Such selection knowledge could, e.g., also reflect the probability of particular occurrences. After selection, the reasoning process is repeated. In the LEADSTO language, the generic default reasoning model can be described by the following local dynamic properties (LPs):

LP1 Candidate Generation using Supernormal Rules

If the agent has a supernormal default rule that allows it to assume x, and it does not have any information about the truth of x yet, then x will be considered a possible assumption.

∀x:info_element
default_rule(x, x) ∧ not belief(x) ∧ not belief(not(x)) ↠ possible_belief(x)

If the agent has a supernormal default rule that allows it to assume not(x), and it does not have any information about the truth of x yet, then x will be considered a possible assumption.

∀x:info_element
default_rule(not(x), not(x)) ∧ not belief(x) ∧ not belief(not(x)) ↠ possible_belief(not(x))

LP2 Candidate Comparison

If a possible belief x has a certain priority p1, and a possible belief y has a priority p2, and p1 > p2, then y is an exceeded possible belief.

∀x, y:info_element, p1, p2:real
possible_belief(x) ∧ possible_belief(y) ∧ has_priority(x, p1) ∧ has_priority(y, p2) ∧ p1 > p2
↠ exceeded_belief(y)

If a possible belief not(x) has a certain priority p1, and a possible belief y has a priority p2, and p1 > p2, then y is an exceeded possible belief.

∀x, y:info_element, p1, p2:real
possible_belief(not(x)) ∧ possible_belief(y) ∧ has_priority(not(x), p1) ∧ has_priority(y, p2) ∧ p1 > p2
↠ exceeded_belief(y)

If a possible belief x has a certain priority p1, and a possible belief not(y) has a priority p2, and p1 > p2, then not(y) is an exceeded possible belief.

∀x, y:info_element, p1, p2:real
possible_belief(x) ∧ possible_belief(y) ∧ has_priority(x, p1) ∧ has_priority(not(y), p2) ∧ p1 > p2
↠ exceeded_belief(not(y))

If a possible belief not(x) has a certain priority p1, and a possible belief not(y) has a priority p2, and p1 > p2, then not(y) is an exceeded possible belief.

∀x, y:info_element, p1, p2:real
possible_belief(not(x)) ∧ possible_belief(not(y)) ∧ has_priority(not(x), p1) ∧ has_priority(not(y), p2) ∧ p1 > p2
↠ exceeded_belief(not(y))

LP3 Candidate Selection
If x is a possible belief, and it is not exceeded by any other belief, then it will be derived
∀x:info_element
possible_belief(x) ∧ ¬exceeded_belief(x) ↠ belief(x)
If not(x) is a possible belief, and it is not exceeded by any other belief, then it will be derived
∀x:info_element
possible_belief(not(x)) ∧ ¬exceeded_belief(not(x)) ↠ belief(not(x))

LP4 Persistence
If x is derived, then this will remain derived.
∀x:info_element
belief(x) ↠ belief(x)
If not(x) is derived, then this will remain derived.
∀x:info_element
belief(not(x)) ↠ belief(not(x))

By these temporal rules the following global reasoning pattern is modelled:

```
while there is a default d that is applicable to T
  generate the consequence possible belief of d
while there is a possible belief
  find the best belief b based on the priorities
  add the best belief b to T
  add all negations of values inconsistent with belief b to T
```

A default rule is applicable if the negation of the justification and the justification itself do not exist within the information state derived. After all possible beliefs are generated the best belief is selected based on priority. The belief with the highest priority is derived, and reasoning rules from the background knowledge can be applied. Next all negations of values inconsistent with the new belief are derived. This also has the effect that no inconsistent beliefs will be derived because those default rules to generate them do not apply anymore. All extensions of the default theory can be found by varying different settings of priority numbers.

5 Case studies

This section illustrates by means of small simulation experiments in Ambient Intelligence contexts how the developed toolbox with reasoning methods introduced in the previous sections can be used in a concrete situation. Section 5.1 addresses two case studies illustrating the basic reasoning methods (introduced in Section 3), and Section 5.2 addresses two case studies illustrating the reasoning methods with incomplete information (introduced in Section 4).

5.1 Basic Reasoning Methods

This section illustrates for a number of the reasoning methods (and their control) presented in Section 3 how they can be used within agents in an Ambient Intelligence application that perform model-based reasoning. This is done by means of two example case studies, each involving an Ambient Intelligence system that uses a causal dynamic model to represent the behaviour of a human, and uses the reasoning methods to determine the state of the human in a particular situation. Modelling human functioning provides more information than pure input-output relations (see e.g. the area of cognitive modelling [5]). Section 5.1.1 focuses on a system that monitors the state of car drivers in order to avoid unsafe driving. Section 5.1.2 addresses an ergonomic system that monitors the stress level of office employees. Both case studies have been formalised and, using the LEADSTO simulation software [15], have been used to generate a number of simulation traces. For each model one example simulation trace is shown. More simulation traces can be found in the Appendix on http://www.cs.vu.nl/~mhoogen/reasoning/appendix-rm-ami.pdf.

5.1.1 Case Study 1A - Car Driver

The example model used as an illustration in this section is inspired by a system designed by Toyota which monitors drivers in order to avoid unsafe driving. The system can basically measure drug level in the sweat of a driver (e.g., via a sensor in the steering wheel, or in an ankle belt), and monitor steering operations and the gaze of the driver. Note that the system is still in the experimental phase. The model used in this paper describes how a high drug intake leads to a high drug level in the blood and this leads to physiological and behavioural consequences: (1) physiological: a high drug level (or a substance relating to the drug) in the sweat, (2) behavioural: abnormal steering operation and an unfocused gaze. The dynamical model is represented within the agent by the following beliefs (where D is an arbitrary time delay):

belief(leads_to_after(drug_intake_high, drug_in_blood_high, D)
belief(leads_to_after(drug_in_blood_high, drug_in_sweat_high, D)
belief(leads_to_after(drug_in_blood_high, abnormal_steering_operation, D)
belief(leads_to_after(drug_in_blood_high, unfocused_gaze, D)

Figure 2 shows this dynamical model in a graphical form.

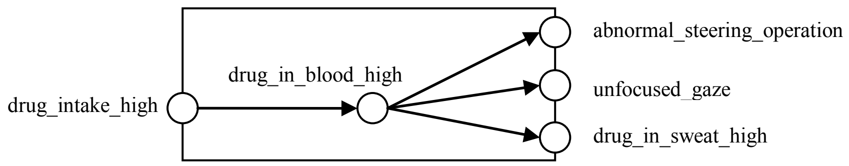

Figure 2. Graphical representation of the dynamical model

By applying the different reasoning methods specified in Section 3, the state of the driver and the expected consequences can be derived. In the simulations below the controlled belief generation method has been used based on selection before beliefs are generated; every temporal rule requires that certain selection criteria are met and that the belief to be derived is in focus. In the following simulations, for the sake of simplicity all information is desired, therefore all derivable beliefs are in focus. The selection criteria involve knowledge about the number of effects and sources that are required to draw conclusions. The knowledge used in this model is the following.

sufficient_evidence_for(and(abnormal_steering_operation, unfocused_gaze), drug_in_blood_high)
sufficient_evidence_for(drug_in_sweat_high, drug_in_blood_high)
sufficient_evidence_for(drug_in_blood_high, drug_intake_high)
in_focus(drug_intake_high); in_focus(drug_in_blood_high); in_focus(drug_in_sweat_high);
in_focus(abnormal_steering_operation); in_focus(unfocused_gaze)

Here, the predicate sufficient_evidence_for(P, Q) represents the belief that expression P is sufficient evidence for the system to derive Q. An example simulation trace is shown in

Figure 3. In the figure, the left side shows the atoms that occur during the simulation, whereas the right side represents a time line where a grey box indicates an atom is true at that time point, and a light box indicates false. In this trace, it is known (by observation) that the driver is steering abnormally and that the driver's gaze is unfocused. Since these two beliefs are sufficient evidence for a high drug level in the blood, using the reasoning method Belief Generation based on Multiple Effect Abduction, at(drug_in_blood_high, 1) becomes a selected belief at time point 3. Given this derived belief, the belief can be deduced that the drug level in the sweat of the driver is high, using Positive Forward Simulation. At the same time (time point 4), the reasoning method Simple Abduction determines the belief that the drug intake of the driver must have been high.

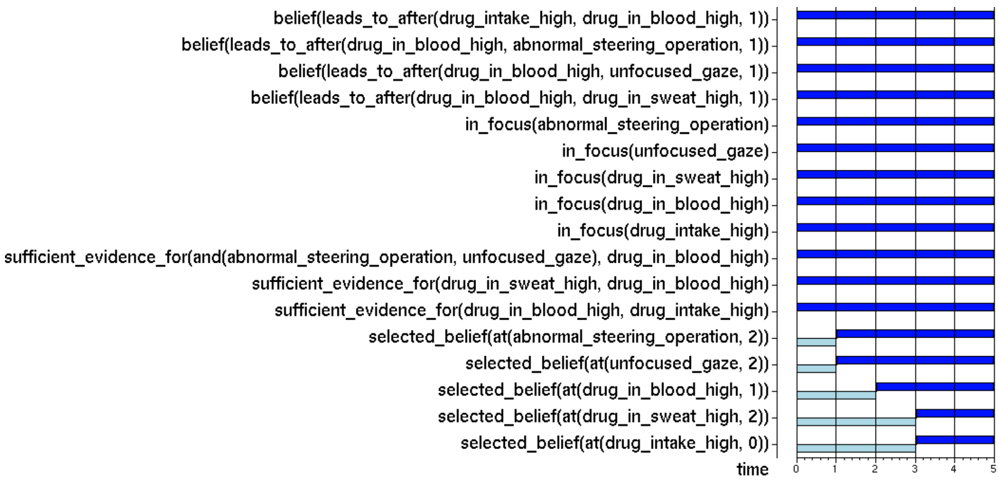

Figure 3. Simulation Trace: abnormal steering and unfocused gaze detected

5.1.2 Case Study 1B - Stress and Workload

The example model used in this section (which is a simplified version of the model presented in [10]) is inspired by ergonomic systems that monitor the activities of office employees in their workspace, e.g., in order to avoid CANS (Complaints of Arm, Neck, Shoulder; for example, WorkPace, see [58]). Such systems may measure various types of information. In this section, three types of measurable (sensor) information are taken into account, namely *actions* (e.g., mouse clicks or key strokes), *biological aspects* (e.g., heart beat, temperature, or skin conductivity), and *activities* (e.g., incoming e-mails, telephone calls, or electronic agenda items). The model considered here describes how (the observation of) a certain activity can lead to a high level of stress and this leads to biological/physiological and behavioural consequences: (1) biological: called here 'high biological aspect' (e.g., increased heart rate) (2) behavioural: changed action (e.g., high number of keystrokes per second). The dynamical model is represented within the agent by the following beliefs:

belief(leads_to_after(activity, observes(activity), D))
belief(leads_to_after(observes(activity), preparedness_to_act, D))
belief(leads_to_after(observes(activity), stress(high), D))
belief(leads_to_after(preparedness_to_act, stress(high), D))
belief(leads_to_after(stress(high), preparedness_to_act, D))
belief(leads_to_after(preparedness_to_act, action, D))
belief(leads_to_after(stress(high), biological_aspect, D))

Figure 4 shows this dynamical model in a graphical form.

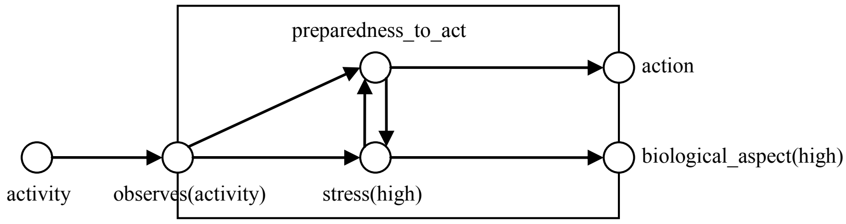

Figure 4. Graphical representation of the dynamical model

Similar to Section 5.1.1, by applying the different reasoning methods specified earlier, the expected consequences for the state of the human and can be derived. A number of simulation traces have been generated, each with different settings for the selection criteria:, for example

sufficient_evidence_for(biological_aspect(high), stress(high)) or in_focus(activity).
sufficient_evidence_for(biological_aspect(high), stress(high)),
sufficient_evidence_for(observes(activity), activity),
sufficient_evidence_for(preparedness_to_act, stress(high)),
sufficient_evidence_for (preparedness_to_act, observes(activity)),
sufficient_evidence_for(stress(high), preparedness_to_act),
sufficient_evidence_for(stress(high), observes(activity)),
sufficient_evidence_for(action, preparedness_to_act),

in_focus(action); in_focus(biological_aspect(high); in_focus(stress(high));
in_focus(observes(activity)); in_focus(activity)

In other words, by selecting different combinations of these criteria, different reasoning steps will be performed. The user is able to select these different combinations and to attribute values to the properties. Notice that the model considered here contains a cycle (see Figure 4). Therefore it is possible to derive an infinite number of beliefs for different time points. For example, if at(preparedness_to_act, 8) is believed, then by simple Positive Forward Simulation also at(stress(high), 9) would be derived, after which at(preparedness_to_act, 10) would be derived, and so on. However, it is not conceptually realistic, nor desirable that an agent attempts to derive beliefs about time points very far in the future. Therefore, by means of the in_focus predicate, an indication of a focus time interval has been specified, for example by statements like in_focus(at(preparedness_to_act, 8)). Note that the values used are for modelling purposes only. For real applications real and more detailed numbers are used.

An example simulation trace is shown in Figure 5. This trace uses as foci all possible information between time point 0 and 10. These foci have been derived using the following rule: in_focus(I) $\wedge$ $0 \leq T \leq 10$ $\twoheadrightarrow$ in_focus(at(I, T)). The only initially available knowledge that is present in this trace is at(action, 5). In the first step, at(preparedness_to_act, 4) is derived using Simple Abduction. Next, using the same method, at(observes_activity, 3) and at(stress(high), 3) are derived and with Positive Forward Simulation at(stress(high), 5).

As shown in the figure, this process continues by performing both Positive Forward Simulation and Simple Abduction several times, eventually leading to all possible derivable information between time point 0 and 10.

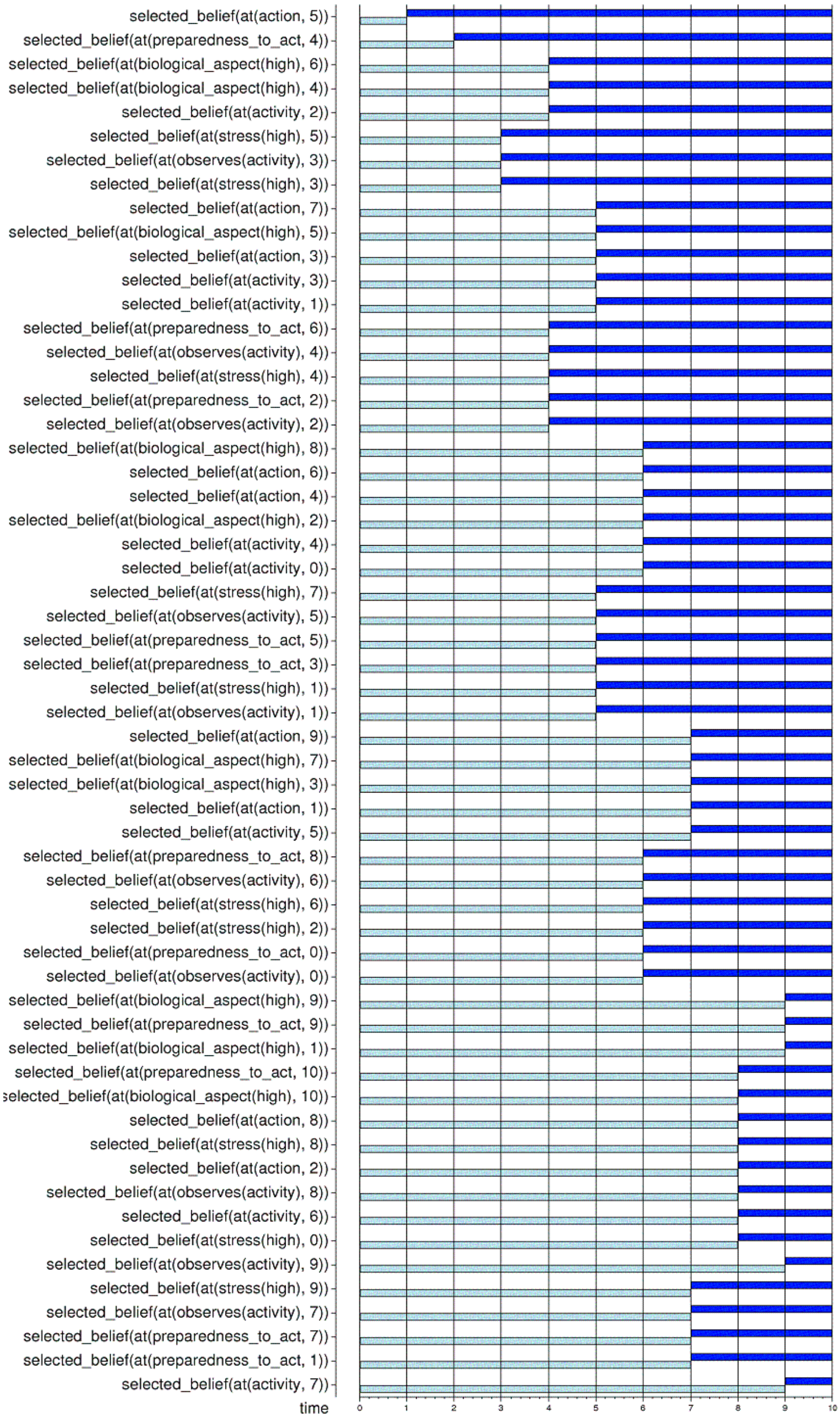

Figure 5. Simulation Trace: Employee performs a series of stress-inducing activities

5.2 Methods for Reasoning with Incomplete Information

This section illustrates for a number of the reasoning methods (and their control) from the developed toolbox presented in Section 4 how they can be used within agents that perform model-based reasoning in an Ambient Intelligence application in case there is incomplete information. Section 5.2.1 focuses on a case study about an intelligent wristband for elderly. Section 5.2.2 addresses a case of a pervasive system that measures criminal activities on the street.

5.2.1 Case Study 2A - Wristband for Elderly

As a case study, the reasoning concerning conditions that occur amongst elderly people is used. Figure 6 shows a simplified causal model for such conditions. On the left hand side five conditions are shown: awake, asleep, syncope (fainted), myocardial infarction (heart attack) and cardiac arrest. The output of the model consists of symptoms that can be measured with a wristband, which are pulse, blood pressure and body temperature. Such a causal model can help in finding out the current condition of an elderly person based on sensory information from the wristband.

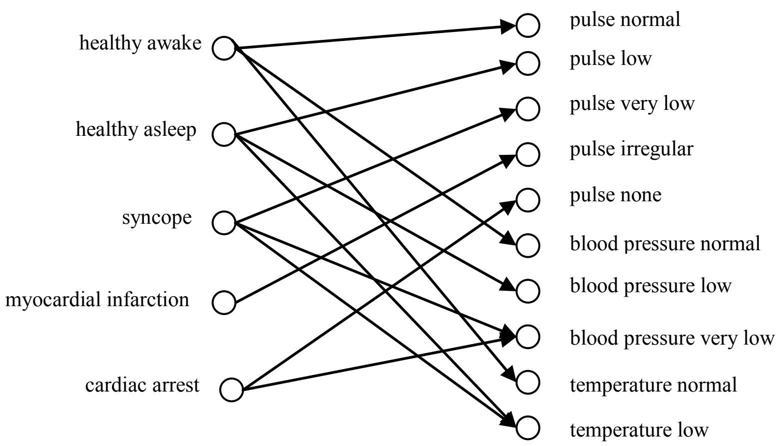

Figure 6. Causal model for the condition of an elderly person

In order to represent this knowledge, the following default theory has been specified. First, the causal background theory (W = CT) is defined, based on the causal graph shown in Figure 6. Furthermore, inconsistent values are defined for the various facets (i.e. pulse, temperature, blood pressure, and condition), for example:

```
inconsistent_values(pulse, normal, low)
inconsistent_values(condition, healthy_awake, healthy_asleep)
```

If an attribute has a certain value and this value is inconsistent with another value, then this other value is not the case.

$$\text{has\_value}(y, x1) \wedge \text{inconsistent\_values}(y, x1, x2) \rightarrow \neg\, \text{has\_value}(y, x2)$$

Besides the background theory, also the default theory Δ_{CT} has been generated from this causal theory CT. The default rules for the atoms are simply as follows:

```
has_value(condition, healthy_awake) / has_value(condition, healthy_awake)
has_value(condition, healthy_asleep) / has_value(condition, healthy_asleep)
has_value(condition, syncope) / has_value(condition, syncope)
has_value(condition, myocardial_infarction) / has_value(condition, myocardial_infarction)
has_value(condition, cardiac_arrest) / has_value(condition, cardiac_arrest)
has_value(pulse, normal) / has_value(pulse, normal)
has_value(pulse, low) / has_value(pulse, low)
has_value(pulse, very_low) / has_value(pulse, very_low)
has_value(pulse, irregular) / has_value(pulse, irregular)
has_value(pulse, none) / has_value(pulse, none)
has_value(blood_pressure, normal) / has_value(blood_pressure, normal)
has_value(blood_pressure, low) / has_value(blood_pressure, low)
has_value(blood_pressure, very_low) / has_value(blood_pressure, very_low)
has_value(temperature, normal) / has_value(temperature, normal)
has_value(temperature, low) / has_value(temperature, low)
```

Besides these default rules, similar defaults for the negations of these atoms are included. Using a system called Smodels [43], the extensions for the default theory specified can be calculated. Using the theory above, 30 extensions result. Hereby, in 19 out of 30 cases neither of the 5 conditions holds (i.e. awake, asleep, syncope, myocardial infarction and cardiac arrest). However, by adding strict rules which express that at least one of the conditions holds, only 11 extensions are found. The extensions that follow after adding these strict rules are shown in Table 2.

Table 2. All extensions of the default theory

#	Condition	Values
1	healthy_awake	has_value(pulse, normal) has_value(blood_pressure, normal) has_value(temperature, normal)
2	healthy_asleep	has_value(pulse, low) has_value(blood_pressure, low) has_value(temperature, low)
3	syncope	has_value(pulse, very_low) has_value(blood_pressure, very_low) has_value(temperature, low)
4	myocardial_infarction	has_value(pulse, irregular) has_value(blood_pressure, normal) has_value(temperature, normal)
5	myocardial_infarction	has_value(pulse, irregular) has_value(blood_pressure, low) has_value(temperature, normal)
6	myocardial_infarction	has_value(pulse, irregular) has_value(blood_pressure, very_low) has_value(temperature, normal)
7	myocardial_infarction	has_value(pulse, irregular) has_value(blood_pressure, normal) has_value(temperature, low)
8	myocardial_infarction	has_value(pulse, irregular) has_value(blood_pressure, low) has_value(temperature, low)
9	myocardial_infarction	has_value(pulse, irregular) has_value(blood_pressure, very_low) has_value(temperature, low)
10	cardiac_arrest	has_value(pulse, none) has_value(blood_pressure, very_low) has_value(temperature, normal)
11	cardiac_arrest	has_value(pulse, none) has_value(blood_pressure, very_low) has_value(temperature, low)

Partial information X may be given that includes the information that the person has a normal temperature. Such a set X can be added to the background theory W. Table 3 shows the extensions resulting when the following facts are added to W:

X = { has_value(temperature, normal), has_value(pulse, irregular) }

Table 3. All extensions given the changed background theory

#	Condition	Values
1	myocardial_infarction	has_value(pulse, irregular) has_value(blood_pressure, normal) has_value(temperature, normal)
2	myocardial_infarction	has_value(pulse, irregular) has_value(blood_pressure, low) has_value(temperature, normal)
3	myocardial_infarction	has_value(pulse, irregular) has_value(blood_pressure, very_low) has_value(temperature, normal)

Finally, Table 4 shows the extensions when the following set X is added to W:

X = {has_value(temperature, normal) , has_value(pulse, normal) , has_value(blood_pressure, normal)}

Table 4. All extensions of the default theory

#	Condition	Values
1	healthy_awake	has_value(pulse, normal) has_value(blood_pressure, normal) has_value(temperature, normal)

Using Smodels basically means that the model serves as input for the software tool, and the tool delivers the extensions as an output. However, it does not provide any detail about the reasoning process followed to come to such an extension. Therefore, the reasoning process towards such an interpretation is not very insightful and explainable. To solve this problem, a simulation of the process has been performed, using the method described in Section 4.4. The result of the simulation is shown in Figure 7.

In Table 2, showing all 11 extensions in the last column, the information elements that are given a high priority in order to obtain the extension are shown. Note that only the positive literals (atoms) are shown. The example simulation trace in Figure 7 shows how extension 9 is found with high priorities for information elements myocardial_infarction, has_value(blood_pressure, very_ low) and has_value(temperature, low). The condition myocardial is introduced as a possible belief by a default rule shown in the first line of the figure. Because this condition has the highest priority, the belief belief(has_value(condition, myocardial_infarction)) is derived. At the next time point, beliefs on other values that are implied by the new belief are derived: the other conditions can be ruled out because they are inconsistent with the current belief about the condition myocardial infarction. In this example trace only the possible beliefs of not myocardial infarction and healthy sleep are shown. Then, at time point 5, the model reasons that the pulse must be irregular. Since the pulse must be irregular, the other values for pulse are ruled out at time point 6. A default rule introduces the possible belief that the blood pressure is very low. Since the priority level is not exceeded by another possible belief, belief(has_value(blood_pressure, very_low)), is derived. Next, the other values for blood pressure can be ruled out (time point 8). At time point 10 the belief with the highest priority – the belief's priority is no longer exceeded - is derived (belief(has_value(temperature, low)) and the inconsistent values of that belief are ruled out (belief(has_value(temperature, low)).

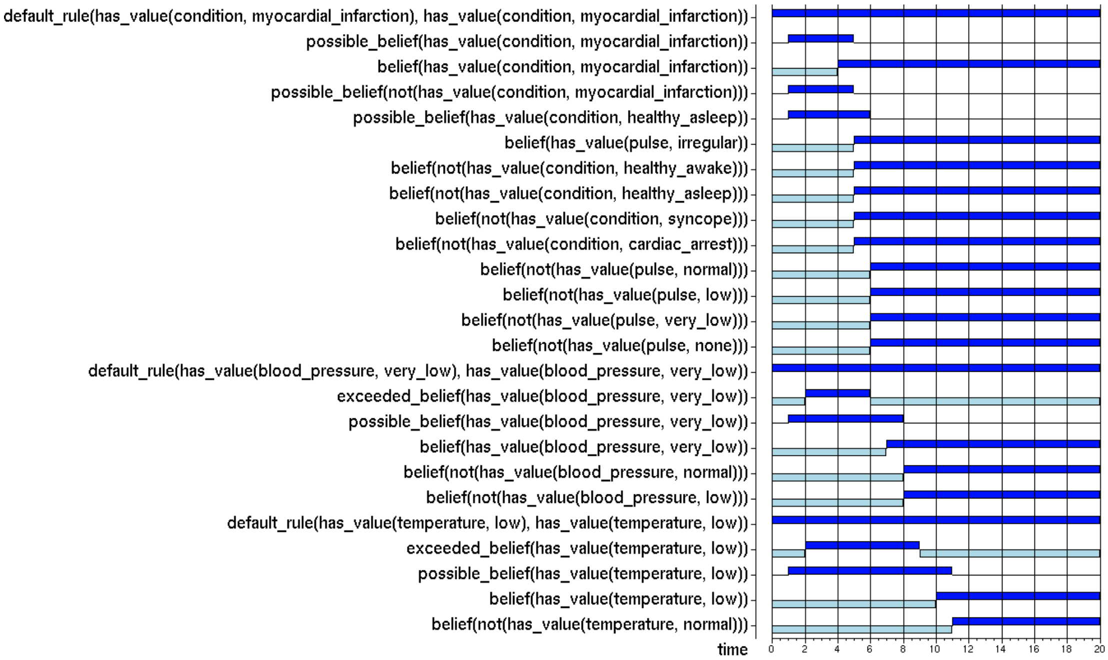

Figure 7. Simulation trace of extension 9.

5.2.2 Case Study 2B - Street Crime

In this case study, a system is used that can help the police solve a crime using Ambient Intelligence facilities. A Dutch company (Sound Intelligence) developed microphones that can distinguish aggressive sounds. Consider the situation in which these microphones are distributed at crucial points in the city, similar to surveillance cameras. Furthermore, suppose in this scenario that for some persons ankle bracelets are used as a form of punishment, which can measure the level of ethanol in the person's perspiration, and indicate their position.

In this example scenario, someone is beaten up nearby a microphone. The microphone picks up the sound of the fight and records this. After an investigation, the police have three suspects. The first suspect is known to have a high level of testosterone, which often leads to aggressive behaviour. The second suspect is someone who is sensitive for alcohol (causing aggression) and wears an ankle bracelet that measures the level of ethanol in his system. He has been seen in a nearby cafe. The third suspect is diagnosed with Intermittent Explosive Disorder (IED), which is a disorder that can lead to a terrible outburst of rage after an unpleasant or stressful meeting. Witnesses saw suspect 2 in the company of someone else.

Figure 8 shows a causal model that is used for this situation that can help the police officers to figure out what information is missing and help them to plan their strategy. For example, did suspect 2 have a conflict with the person he was with? Did suspect 3 drink alcohol? Is it needed to administer testosterone tests with the subjects? Aggressive sounds are caused by persons that are aggressive, according to the model. Three possible causes for this aggressiveness are considered, as can be seen in Figure 8: someone can have a high level of testosterone, someone can just have been in a situation of conflict or someone can have a high level of alcohol.

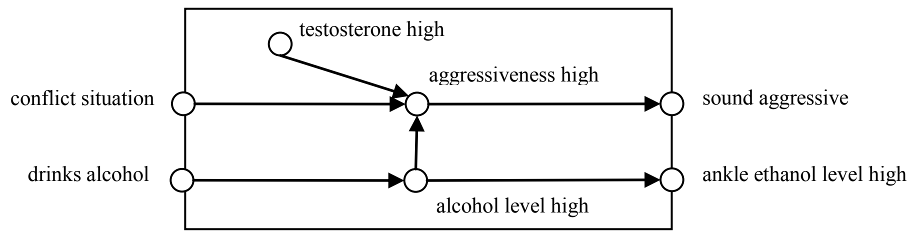

Figure 8. Causal model for the crime case

Similar to the Elderly Wristband, the default theory Δ_{CT} for the crime case has been generated from the causal model:

has_value(situation, conflict) / has_value(situation, conflict)
has_value(situation, drinks_alcohol) / has_value(situation, drinks_alcohol)
has_value(testosterone, high) / has_value(testosterone, high)
has_value(sounds, aggressive) / has_value(sounds, aggressive)
has_value(ankle_ethanol_level, high) / has_value(ankle_ethanol_level, high)
has_value(aggressiveness, high) / has_value(aggressiveness, high)
has_value(alcohol_level, high) / has_value(alcohol_level, high)
not(has_value(situation, conflict) / not(has_value(situation, conflict))
not(has_value(situation, drinks_alcohol) / not(has_value(situation, drinks_alcohol))
not(has_value(testosterone, high) / not(has_value(testosterone, high))
not(has_value(sounds, aggressive) / not(has_value(sounds, aggressive))
not(has_value(ankle_ethanol_level, high) / not(has_value(ankle_ethanol_level, high))
not(has_value(aggressiveness, high) / not(has_value(aggressiveness, high))
not(has_value(alcohol_level, high) / not(has_value(alcohol_level, high))

Furthermore, aggressive sound has been observed, therefore the following fact is added to W:

X = {has_value(sound, aggressive)}

The resulting number of extensions is 18. Hereby however, the reasoning has not been performed using a closed world assumption, whereby values can only occur in case they result from a known causal relation or in case they are input variables (i.e. the situation). In order to perform reasoning with such a closed world assumption, the following rules have been added. First, a rule expressing that in case there is only one source from which a value can be derived, then this source should have the appropriate value (in this case, this holds for all variables except for aggressiveness).

has_value(X1, Y1) $\wedge$ leads_to(has_value(X2, Y2), has_value(X1, Y1)) $\wedge$ X1 $\neq$ aggressiveness $\rightarrow$ has_value(X2, Y2)

For the aggressiveness a different set of rules is used, since only one out of three conditions needs to hold. An example of one instance of such a rule is the following:

has_value(aggressivness,high)$\wedge$not(has_value(testosterone,high) $\wedge$ not(has_value(situation,conflict) $\rightarrow$ has_value(alcohol_level, high)

Given that these rules are added, 7 extensions result using Smodels as shown in Table 5. Note that sound is not shown since that is fixed in advance already. The last column shows to which suspect this extension is applicable. Hereby the suspect with high testosterone is marked with 1, the oversensitive alcohol suspect with 2, and the IED suspect with 3.

Table 5. Extensions given that aggressive sound has been observed

#	Situation	Testosterone	Aggressiveness	Alcohol level	Ankle Ethanol level	Suspect
1	¬conflict ¬drinks_alcohol	high	high	¬high	¬high	1
2	conflict ¬drinks_alcohol	high	high	¬high	¬high	1
3	conflict ¬drinks_alcohol	¬high	high	¬high	¬high	3
4	conflict drinks_alcohol	high	high	high	high	1
5	conflict drinks_alcohol	¬high	high	high	high	2, 3
6	¬conflict drinks_alcohol	¬high	high	high	high	2
7	¬conflict drinks_alcohol	high	high	high	high	1

The simulation trace in Figure 9 shows how extension 7 is found using the method introduced in Section 4.4, with high priorities for information elements has_value(situation, drinks_alcohol) and has_value(sound, aggressive).

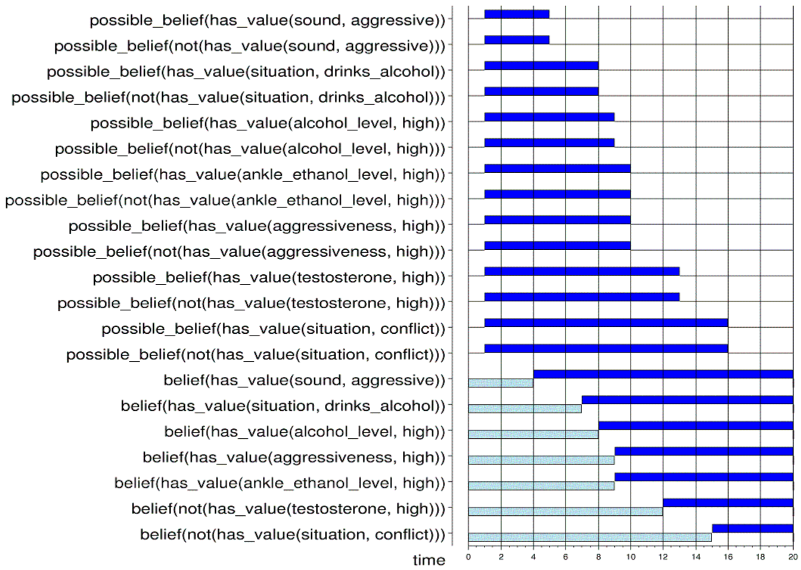

Figure 9. Simulation trace of the derivation of extension 7.

The condition belief(has_value(sound, aggressive)) has the highest priority, because this information is given in the scenario. Because it has the highest priority, this condition is the first to be derived (at time point 4). The condition with the second highest priority is derived at time point 7 (belief(has_value(situation, drinks_alcohol). This belief leads to the following beliefs: belief(has_value(alcohol_level, high)) at time point 8, belief(has_value(aggressiveness, high)) and belief(has_value(ankle_ethanol_level, high) at time point 9. At time point 12 the belief belief(not(has_value(testosterone, high))) becomes true and at time point 15 the belief belief(not(has_value(situation, conflict))) becomes true.

6 Formal Analysis of Dynamic Properties

This section provides a number of basic properties that may hold for model-based reasoning methods within agents in an Ambient Intelligence application. These properties can be checked against the traces generated in the case studies to verify that the methods indeed function appropriately. Section 6.1 addresses properties of world facts and beliefs; Section 6.2 addresses properties of LEADSTO relations. Section 6.3 addresses analysis of dynamic properties in terms of interlevel relationships and presents the result of the verification of properties against the traces of the case studies as presented in Section 5.

6.1 Properties of world facts and beliefs

The following basic assumptions concerning two-valued world facts may hold:

Consistency of world facts
In any state, it never happens that a world fact and its negation both hold.
not [state(γ, t) |= world_fact(I) & state(γ, t) |= world_fact(not(I))]

Completeness of world facts
In any state, for any world fact it holds or its negation holds.
state(γ, t) |= world_fact(I) | state(γ, t) |= world_fact(not(I))

Consistency and completeness of world facts
In any state, for any world fact it holds if and only if its negation does not hold
state(γ, t) |= world_fact(I) ⇔ not state(γ, t) |= world_fact(not(I))

Belief consistency
In any state, it never happens that a fact and its negation are both believed.
not [state(γ, t) |= belief(I) & state(γ, t) |= belief(not(I))]

Belief correctness
In any state, when a fact is believed it holds as a world fact.
state(γ, t) |= belief(at(I, t')) ⇒ state(γ, t') |= world_fact(I)

Belief persistence
In any state, if a fact is believed, it will be believed at any later time point, unless its negation is believed at that time point.
∀t, t'≥t [state(γ, t) |= belief(I) & not state(γ, t') |= belief(not(I)) ⇒ state(γ, t') |= belief(I)]
∀t, t'≥t [state(γ, t) |= belief(not(I)) & not state(γ, t') |= belief(I) ⇒ state(γ, t') |= belief(not(I))]

Belief completeness
For any state, any fact is believed or its negation is believed.
state(γ, t) |= belief(I) | state(γ, t) |= belief(not(I))

Belief coverage
In any state, any true world fact is believed.
state(γ, t) |= world_fact(I) ⇒ state(γ, t) |= belief(I)

In the general form, where a universal quantifier is assumed over I, belief completeness and belief coverage will usually not hold. However, it may hold for a specific class of information I. For example, sometimes it is assumed that the agent has complete beliefs about leads to relationships.

6.2 Properties of leads to relationships

The leads_to_after relationship expresses the conceptual core of a wide class of dynamic modelling concepts that occur in the literature in different contexts and under different names; see also [16]. Examples of such dynamical modelling concepts are, computational numerical modelling by difference or differential equations, qualitative dynamic modelling, causal relationships, temporal logic specifications, rule-based representations, Petri net representations, transition systems and finite automata. Often, either explicitly or implicitly the general assumption is made that when facts are true in the world, the facts to which they lead are also true in the world. This property is expressed as follows, also formulated by contraposition into a logically equivalent one:

Positive forward correctness
If a world fact I holds in a state and it leads to another world fact J after duration D, then in the state after duration D this J will hold
state(γ, t) |= world_fact(I) & state(γ, t) |= world_fact(leads_to_after(I, J, D)) ⇒
state(γ, t+D) |= world_fact(J)

Negative backward correctness
If a world fact J does not hold in a state and another world fact I leads to J after duration D, then in the state before duration D this I will not hold
state(γ, t) |= world_fact(not(J)) & state(γ, t) |= world_fact(leads_to_after(I, J, D)) ⇒
state(γ, t-D) |= world_fact(not(I))

Sometimes, also the more specific assumption is made that a world fact can be true *only* when a world fact preceding it via a leads to relation is true. This assumption can be seen as a temporal variant of a Closed World Assumption.

Negative forward correctness (single source)
If a world fact I doers not hold in a state and it leads to another world fact J after duration D, then in the state after duration D this J will not hold
state(γ, t) |= world_fact(not(I)) & state(γ, t) |= world_fact(leads_to_after(I, J, D)) ⇒
state(γ, t+D) |= world_fact(not(J))

Positive backward correctness (single source)
If a world fact J holds in a state and another world fact I leads to J after duration D, then in the state before duration D this I will hold
state(γ, t) |= world_fact(J) & state(γ, t) |= world_fact(leads_to_after(I, J, D)) ⇒
state(γ, t-D) |= world_fact(I)

The latter property can be formulated by contraposition into a logically equivalent property of the former one. These properties play a role in abductive reasoning methods, and automated explanation generation (in particular for why-explanations: answers on questions such as 'Why does J hold?'). The latter two properties may not be fulfilled in cases that two (or multiple) non-equivalent world facts I1 and I2 exist that each lead to a world fact J. If I1 holds, and it leads to the truth of J, then it may well be the case that I2 was never true. A more complete property to cover such cases is the following.

Negative forward correctness (multiple sources)
If for a world fact J, for every world fact I which leads to J after a duration D it does not hold in the state before duration D, then in the state after duration D this J will not hold
∀I, D [state(γ, t-D) |= world_fact(leads_to_after(I, J, D)) ⇒ state(γ, t-D) |= world_fact(not(I))]
⇒ state(γ, t) |= world_fact(not(J))

Positive backward correctness (multiple sources)
If a world fact J holds in a state, then there exists a world fact I which leads to J after a duration D which holds in the state before duration D.
state(γ, t) |= world_fact(J)
⇒ ∃I, D [state(γ, t-D) |= world_fact(leads_to_after(I, J, D)) & state(γ, t-D) |= world_fact(I)]

To obtain a logical foundation for a temporal variant of the Closed World Assumption in such situations in the context of executable temporal logic, in [26] the notion of temporal completion was introduced, as a temporal variant of Clark's completion in logic programming.

6.3 Interlevel Relationships

This section shows how it can be verified that the reasoning methods introduced in Section 3 and 4 (and simulation traces generated on the basis of these methods) satisfy certain basic properties as introduced above. This is done by establishing logical (inter-level) relationships between a *global property* (GP) of reasoning methods on the one hand, and the basic reasoning steps (or *local properties*, LP's) on the other hand, in such a way that the combination of reasoning steps (logically) entails the global property. In order to establish such inter-level relationships, also certain *intermediate properties* (IP's) are constructed, which can be used as intermediate steps in the proof. Here, the focus is on one particular property from Section 6.1, namely the Belief Correctness property. This global property for belief generation is expressed below in GP1 and states that all beliefs should be correct. This should hold for all reasoning intervals within the trace (i.e. starting at an observation interval, and the reasoning period thereafter without new observation input). Note that all variables γ that are not explicitly declared are assumed to be universally quantified. Moreover, E is assumed to be the duration of a reasoning step.

GP1 (Belief Correctness)
For all time points t1 and t2 later than t1 whereby at t1 an observations is receivedare observed, and between t1 and t2 no new observations are received, GP1(t1, t2) holds.
GP1 ≡
∀t1, t2 ≥ t1
[state(γ, t1) |= observation_interval &
¬state(γ, t2) |= observation_interval &
∀t' < t2 & t' > t1 [state(γ, t2) |= ¬observation_interval]]
⇒ GP1(t1, t2)

The specification of the global property for an interval is expressed below.

GP1(t1, t2) (Belief Correctness from t1 to t2)
Everything that is believed to hold at T at time point t' between t1 and t2, indeed holds at that time point T.
GP1(t1, t2) ≡
∀I, T, t' ≥ t1 & t' ≤ t2 state(γ, t') |= belief(at(I, T)) ⇒ state(γ, T) |= world_fact(I)

In order to prove that property GP1 indeed holds, a proof by means of induction is used. The basis step of this proof is specified in property LP1, whereby the beliefs during the observation interval need to be correct.

LP1(t) (Belief Correctness Induction Basis)
If time point t is part of the observation interval, then everything that at time point t is believed to hold at time point T, indeed holds at time point T.
LP1(t) ≡
state(γ, t) |= observation_interval ⇒
[∀I,T state(γ, t) |= belief(at(I, T)) ⇒ state(γ, T) |= world_fact(I)]

Furthermore, the induction step includes that if the global property holds from a time point t to the same time point, then the property should also hold between t and t + E.

IP1 (Belief Correctness Induction Step)
For all time points t, if GP1(t, t) holds, then also GP1(t, t+E) holds.
IP1 ≡
∀t GP1(t, t) ⇒ GP1(t, t+E)

In order to prove that this induction step indeed holds, the following three properties are specified: IP2, LP2, and LP3. First of all, the *grounding* of the belief generation (IP2) which states that for all beliefs that have not been generated since the last observation interval, they should either have been derived by means of forward reasoning, or by means of abduction.

IP2 (Belief Generation Grounding)
For all time points t+E, if information element J is believed to hold at time point T and J was not believed during the last observation interval, then either this was derived by applying a forward leadsto rule, or by means of abduction.
IP2 ≡
$\forall$t,t0,J,T
[state(γ, t) |= belief(at(J, T)) & last_observation_interval(t, t0) & $\neg$state(γ, t0) |= belief(at(J, T))
$\Rightarrow$ $\exists$I,t2, D
[state(γ, t2) |= belief(at(I, T-D)) & state(γ, t2) |= belief(leads_to_after(I, J, D)) |
state(γ, t2) |= belief(at(I, T+D)) & state(γ, t2) |= belief(leads_to_after(J, I, D))]

Property LP2 expresses the correctness of the model believed, that should correspond with the model present in the world.

LP2 (Model Representation Correctness)
For all time points t, if it is believed that I leads to J after duration D, then I indeed leads to J after duration D.
LP2 ≡
$\forall$t,I,J,D
state(γ, t) |= belief(leads_to_after(I, J, D)) $\Rightarrow$ state(γ, t) |= world_fact(leads_to_after(I, J, D))

The correctness of the derivations within the world is expressed in LP3.

LP3 (Positive Forward Correctness)
For all time points t, if information element I holds and I leads to J after duration D, then at time point t+D information element J holds.
LP3 ≡
$\forall$t,I,J,T,D
state(γ, t) |= world_fact(I) & state(γ, t) |= world_fact(leads_to_after(I, J, D)) $\Rightarrow$
state(γ, t+D) |= world_fact(J)

The final properties specified (LP4 and LP5) are used to ground property IP2. LP4 expresses that if a certain belief concerning an information element holds, and from this belief another belief concerning an information element can be derived, then this is the case at some time point t2.

LP4 (Belief Generation based on Positive Forward Simulation)
For all time points t, if information element I is believed to hold at time point T and it is believed that I leads to J after duration D, then there exists a time point t2 information element J is believed to hold at time point T+D.
LP4 ≡
$\forall$t1,t2,I,J,T,D
state(γ, t1) |= belief(at(I, T)) & state(γ, t1) |= belief(leads_to_after(I, J, D)) $\Rightarrow$
state(γ, t2) |= belief(at(J, T+D))

Property LP5 specifies how beliefs can be generated based on abduction.

LP5 (Belief Generation based on Abduction)
For all time points t, if information element J is believed to hold at time point T and it is believed that I leads to J after duration D, then there exists a time point t2 information element I is believed to hold at time point T-D.
LP4 ≡
$\forall$t1,t2,I,J,T,D
state(γ, t1) |= belief(at(J, T)) & state(γ, t1) |= belief(leads_to_after(I, J, D)) $\Rightarrow$
state(γ, t2) |= belief(at(I, T-D))

Figure 10 depicts the relations between the various properties by means of an AND tree. Here, if a certain property is connected to properties at a lower level, this indicates that the properties at the lower level together logically imply the higher level property. Note: LP4G and LP5G are the *grounding*[‡] variant of LP4 and LP5 respectively, which is why they are depicted in grey.

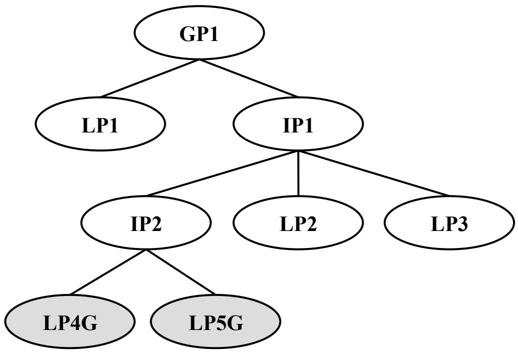

Figure 10. Proof of GP1 depicted by means of an AND tree

Figure 10 shows that global property GP1 can be related (by logical relations, as often used in mathematical proof) to a set of local properties (LPs) of the reasoning methods put forward in Section 3 and 4. Note that it is not claimed here that GP1 holds for all reasoning methods, but that it holds for those methods that satisfy the lower level properties (LP1, LP4G, LP5G, LP2, and LP3). Such inter-level relations can be useful for *diagnosis of dysfunctioning* of a reasoning process. For example, suppose for a given reasoning trace (obtained either by simulation, such as in Section 5, or by other means, e.g. based on empirical material of an existing Ambient Intelligence system) that the dynamic property GP1 does not hold, i.e., not all beliefs are correct. Given the AND-tree structure in Figure 10, at least one of the children nodes of GP1 will not hold, which means that either LP1 or IP1 will not hold. Suppose by further checking it is found that IP1 does not hold. Then the diagnostic process can be continued by focusing on this property. It follows that either IP2, LP2, or LP3 does not hold. This process can be continued until the cause of the error is localised.

The process mentioned above is based on the assumption that it is possible to (automatically) check any property against a trace. To this end, the TTL Checker Tool [15] can be used (and has indeed been used). For the traces presented in Section 5 all properties shown in Figure 10 were checked, and turned out to hold.

7 Related Work

Various approaches in the literature present generic (agent-based) frameworks for Ambient Intelligence applications. For example, [37] presents a framework that utilizes mobile agents for ambient intelligence in a distributed ubiquitous environment. In this work, however, the focus is not on methods to reason about the dynamics of human-related processes, but on the architecture and the mathematical formulation (using π-Calculus) that can be used for evaluation and verification. Similarly, in [21, 57], a multi agent-based framework for a typical ambient intelligence "space" is proposed. It provides a hierarchical

‡ The grounding variant of an executable property states that there is no other property with the same consequent. For example, the grounding variant of $A \Rightarrow B$ states that there is no other property with B in its consequent, thus $B \Rightarrow A$ can be derived.

system model for an ambient intelligence space, a model of the middleware and of the physical structure of the application. The main difference between these approaches is that the current paper makes use of formal methods for reasoning about the dynamics of human-related processes. In that sense, it has some similarities with the work presented in [41]. There, a framework is presented to enable home healthcare. The framework enables the observation of patients' clinical data via wearable devices, and of movements via sensor networks. Based on these types of information, habits and actions are derived by means of logic programming techniques.

Moreover, the presented paper obviously took some inspiration from early approaches for model-based reasoning (which originally were not developed specifically for Ambient Intelligence applications), such as the general abductive reasoning framework [22] In this framework, integrity constraints can be specified (see e.g. [3, 23]). Such constraints can also be specified using the approach specified in this paper, namely by incorporating these by means of the focus mechanism specified in Section 3. Note that the notion of a focus is not only meant to avoid integrity constraints not being satisfied, but is also meant as a way to direct the reasoning process in an appropriate and efficient way.

The important issue of how the models of the humans can be created has been investigated in the literature as well. Ballim and Wilks [7] present approaches to generate beliefs based upon stereotypes and perturbation. In this paper however, the model of the human is assumed to be known, but could for instance be based upon approaches such as introduced by Ballim and Wilks. In [56] Wilks and Hartley introduce a combination of various systems they have developed in order to generate beliefs and explanations for particular sets of data, which is also done in this paper. The explicit incorporation of time within the reasoning process is however not part of their approach. Additionally, the approach is more focused on finding the actual models to explain particular phenomena whereas the approach presented in this paper assumes a complete set of causal knowledge in the world already and reasons about these causal chains.

Furthermore, in [6] temporal reasoning is combined with an Active Database (ADB) for the detection of complex events in Smart Homes. The focus of that research is the combination of ADB and temporal reasoning. There is no selection mechanism in that paper as in the current work: the focus mechanism. Another example of temporal reasoning in Ambient Intelligence, [53], developed a multi-agent system based on a knowledge-goal-plan (KGP) agent for transparent communication between users and an Ambient Intelligence device. They have based their reasoning model on well-known reasoning techniques such as Abductive Logic Programming and Logic Programming with Priorities. In the current work however, the focus is on developing the underlying reasoning methods that are useful in Ambient Intelligence applications.

Another formalism for handling causal or temporal reasoning within Ambient Intelligence is proposed in [28]. The application of nonmonotonic logic as put forward in this paper adds the possibility to specify human like reasoning in a natural way, possibly even resulting in multiple stable sets that can be the outcome of such a reasoning process.

Finally, Bayesian network-based solutions (introduced by [44]) offer the possibility to deal with uncertainty and temporality [55] when reasoning about (causal) models. However, our approaches deal with temporality in a human-like manner, in contrast with the methods described in [55]. In addition, Bayesian network solutions in general do not offer a selection mechanism like our focus mechanism. This offers a computational advantage because some (or most) of the possible beliefs are not considered.

8 Discussion

The main assumption behind the current paper is that agents in an Ambient Intelligence application will be able to provide more efficient, personalised support when they have knowledge about human behaviours and states over time. In order to endow them with such knowledge, it is useful when they possess explicitly represented causal and dynamical models about the human's processes, and use them in their model-based reasoning processes (e.g., [4, 38, 42]). Next, once an agent has such a model, a number of logical reasoning methods can be based on such a model, and formally specified as part of the agent design, as shown in this paper.

The main contribution of this paper is the development of a unified toolbox containing a variety of reasoning methods to support a designer of Ambient Intelligence applications. The reasoning methods are given at a conceptual formal specification level in a hybrid executable temporal logical format. In addition, to the reasoning methods themselves, also approaches to (meta-level) control of the reasoning methods are specified in a unified manner. It provides a variety of available and possible reasoning methods, including abduction and default reasoning methods to address incomplete information.

Two main classes of reasoning approaches were presented on a more detailed level: (1) basic model-based reasoning methods based upon existing methods, but extended with explicit temporal reasoning as well as the ability to reason about both quantitative and qualitative aspects, and perform controlled focused reasoning; (2) default reasoning approaches which are able to cope with incomplete information and advanced selection strategies whereby again standard approaches have been used, extended with temporal reasoning and advanced selection methods. Again, both qualitative and quantitative aspects can be expressed in the language and reasoning approach used. Model-based default refinement can be useful to obtain (on top of sensor information) a high level of context awareness; see also [48, 49, 50]. In several simulation experiments, example reasoning patterns were shown based on these techniques, thus showing reusability of the agent design. These simulation traces have been formally analysed and verified by checking a number of global dynamic properties.

Based on the different reasoning methods presented in this paper, the question may arise when to select which method. In fact, in some cases the answer to this question may be easier than one would expect. The two basic model-based reasoning methods described in Section 3, namely forward and backward reasoning, can be selected automatically based on the information that is available. For example, if information about the node at the right-hand side of a model is observed, only backward reasoning rules can be applied and if information about a node in the middle is available, rules from both methods can be applied. The choice for controlled default reasoning (Section 4) can be made when there is partial information available and more possible conclusions are desired.

Although the proposed reasoning methods have been applied successfully in four case studies, the examples addressed were modelled at an abstract, conceptual level. In future work and work in progress, more complex and realistic case studies will be performed. For example, in [13] a more elaborate agent model for assessment of driving behaviour has been developed. In addition, case study 1B is also being studied more thoroughly in [10], in which a complex model for a person's "functional state" is presented, which estimates a person's state of experienced pressure and exhaustion based on various elements, including environmental task demands and personality characteristics. Finally, recently, an EU project called ICT4depression started in which a component-based system is developed that supports people with a depression during therapy. In this system, different domain, analysis and support models are being developed that use the methods proposed in the current paper to monitor and support patients. These models will be much more complex and will contain much more temporally dependent relations. Another challenge in the ICT4depression project is dealing with the complexity of live-fed databases containing information about medication intake and self-reports about the patients well-being. By abstracting this information before the Ambient Agents start analyzing it, the complexity of the reasoning process will be reduced. In all of these case studies, the possibilities to incorporate the proposed reasoning methods in real artefacts in the environment are being explored. A specific question that will be addressed in the future is to what extent the reasoning methods are able to deal with dynamic learning of new knowledge.

References

1. Aarts, E., Collier, R., van Loenen, E., and Ruyter, B. de (eds.) (2003). *Ambient Intelligence. Proc. of the First European Symposium, EUSAI 2003*. Lecture Notes in Computer Science, vol. 2875. Springer Verlag, 2003, pp. 432.
2. Aarts, E., Harwig, R., and Schuurmans, M. (2001), Ambient Intelligence. In: P. Denning (ed.), *The Invisible Future*, McGraw Hill, New York, pp. 235-250.
3. Abdennadher, S., and Christiansen, H. (2000). An Experimental CLP Platform for Integrity Constraints and Abduction, Fourth International Conference on Flexible Query Answering Systems, FQAS 2000, Warsaw, Poland.
4. Anderson, D.R., 2008. Model Based Inference in the Life Sciences: A Primer on Evidence. Springer, New York.
5. Anderson, J.R., and Lebiere, C. (1998). *The atomic components of thought*. Lawrence Erlbaum Associates, Mahwah, NJ.
6. Augusto, J., and Nugent, C. (2004). The use of temporal reasoning and management of complex events in smart homes. In de Mantaras, R.L., Saitta, L., eds.: *Proceedings of European Conference on Artificial Intelligence (ECAI 2004)*, IOS Press (Amsterdam, The Netherlands) pp. 778–782. August, 22-27.
7. Ballim, A., and Wilks, Y., Beliefs, Stereotypes and Dynamic Agent Modeling. *User Modeling and User-Adapted Interaction*, 1:33-65, 1991.
8. Baron-Cohen, S. (1995). *Mindblindness*. MIT Press.
9. Bosse, T., Both, F., Gerritsen, C., Hoogendoorn, M., and Treur, J., Model-Based Reasoning Methods within an Ambient Intelligent Agent Model. In: *Proceedings of the First International Workshop on Human Aspects in Ambient Intelligence*. Published in: M. Mühlhäuser, A. Ferscha, and E. Aitenbichler (eds.), *Constructing Ambient Intelligence: AmI-07 Workshops Proceedings*. Communications in Computer and Information Science (CCIS), vol. 11, Springer Verlag, 2008, pp. 352-370.
10. Bosse, T., Both, F., Hoogendoorn, M., Jaffry, S.W., Lambalgen, R. van, Oorburg, R., Sharpanskykh, R., Treur, J., and Vos, M. de, Design and Validation of a Model for a Human's Functional State and Performance. *International Journal of Modeling, Simulation, and Scientific Computing*, vol. 2, 2011, in press. Extension of: Bosse, T., Both, F., Lambalgen, R. van, and Treur, J., An Agent Model for a Human's Functional State and Performance. In: *Proceedings of the 8th IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT'08*. IEEE Computer Society Press, 2008, pp. 302-307.
11. Bosse, T., Delfos, M.F., Jonker, C.M., and Treur, J. (2006). Analysis of Adaptive Dynamical Systems for Eating Regulation Disorders. *Simulation Journal (Transactions of the Society for Modelling and Simulation)*, vol. 82, 2006, pp. 159-171.
12. Bosse, T., Gerritsen, C., and Treur, J. (2007). Integration of Biological, Psychological and Social Aspects in Agent-Based Simulation of a Violent Psychopath. In: Shi, Y., Albada, G.D. van, Dongarra, J., and Sloot, P.M.A. (eds.), *Computational Science II, Proceedings of the Seventh International Conference on Computational Science, ICCS'07, Part II*. Lecture Notes in Computer Science, vol. 4488. Springer Verlag, 2007, pp. 888-895.

13. Bosse, T., Hoogendoorn, M., Klein, M.C.A., and Treur, J., A Component-Based Agent Model for Assessment of Driving Behaviour. In: Sandnes, F.E., Burgess, M., and Rong, C. (eds.), *Proceedings of the Fifth International Conference on Ubiquitous Intelligence and Computing*, UIC'08. Lecture Notes in Computer Science, vol. 5061, Springer Verlag, 2008, pp. 229-243.
14. Bosse, T., Jonker, C.M., Meij, L. van der, Sharpanskykh, A., and Treur, J., (2009). Specification and Verification of Dynamics in Agent Models. *International Journal of Cooperative Information Systems*, vol. 18, 2009, pp. 167 - 193. Shorter version in: Nishida, T. et al. (*eds.), Proceedings of the Sixth International Conference on Intelligent Agent Technology, IAT'06*. IEEE Computer Society Press, 2006, pp. 247-254.
15. Bosse, T., Jonker, C.M., Meij, L. van der, and Treur, J. (2007). A Language and Environment for Analysis of Dynamics *by Simulation. International Journal of Artificial Intelligence Tools*, vol. 16, 2007, pp. 435-464.
16. Bosse, T., and Treur, J., (2008). A Philosophical Foundation for Unification of Dynamic Modelling Methods Based on Higher-Order Potentialities and their Reducers. *Advances in Complex Systems Journal*, vol. 11, 2008, pp. 831 - 860. Shorter version in: M.M. Veloso (ed.), *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence, IJCAI'07*. AAAI Press, 2007, pp. 262-267.
17. Both. F., Gerritsen, C., Hoogendoorn, M., and Treur, J., Model-Based Default Refinement of Partial Information within an Ambient Agent. In: *Proceedings of the First International Workshop on Artificial Intelligence Methods for Ambient Intelligence*. Published in: M. Mühlhäuser, A. Ferscha, and E. Aitenbichler (eds.), *Constructing Ambient Intelligence: AmI-07 Workshops Proceedings*. Communications in Computer and Information Science (CCIS), vol. 11, Springer Verlag, 2008, pp. 34-43.
18. Brazier, F.M.T., Jonker, C.M., and Treur, J. (2000). Compositional Design and Reuse of a Generic Agent Model. *Applied Artificial Intelligence Journal,* vol. 14, 2000, pp. 491-538.
19. Brewka, G. (1994), Adding priorities and specificity to default logic. In: MacNish, C., Pereira, L., and Pearce, D., (eds.), *Proc. of JELIA'94, LNAI*, vol. 838. Springer Verlag, pp. 247-260.
20. Brewka, G., and Eiter, T. (1999), Prioritizing Default Logic: Abridged Report. In *Festschrift on the occasion of Prof.Dr. W. Bibel's 60th birthday*. Kluwer.
21. Chen, R., Hou, Y., Huang, Z., Zhang, Y., Li, H. (2007) Framework for Local Ambient Intelligence Space: The AmI-Space Project. In: *Proceedings of the Computer Software and Applications Conference, 2007 (COMPSAC 2007).* Volume 2, 24-27 July 2007, 95 – 100.
22. Coombs, M.J., and Hartley, R. T. (1987). The MGR algorithm and its application to the generation of explanations for novel events. *International Journal of Man-Machine Studies*, 27, 679-708.
23. Endriss, U., Mancarella, P., Sadri, F., Terreni, G. and Toni, F. (2004). Abductive logic programming with ciff: Implementation and applications. In Proceedings *of the Convegno Italiano di Logica Computazionale CILC-2004*. University of Parma.
24. Engelfriet, J., Herre, H., and Treur, J. (1998), Nonmonotonic Reasoning with Multiple Belief Sets, *Annals of Mathematics and Artificial Intelligence,* vol. 24, pp. 225-248.
25. Engelfriet, J., Jonker, C.M., and Treur, J. (2002). Compositional Verification of Multi-Agent Systems in Temporal Multi-Epistemic Logic. *Journal of Logic, Language and Information,* vol. 11, 2002, pp. 195-225.
26. Engelfriet, J., and Treur, J. (1998), An Interpretation of Default Logic in Minimal Temporal Epistemic Logic. *Journal of Logic, Language and Information*, vol. 7, pp. 369-388.
27. Engelfriet, J., and Treur, J. (2003), Multi-Interpretation Operators and Approximate Classification. *Int. Journal of Approximate Reasoning*, vol. 32, pp. 43-61.
28. Galton, A., Causal Reasoning for Alert Generation in Smart Homes. In: J. C. Augusto and C. D. Nugent (eds.), *Designing Smart Homes*, Springer-Verlag LNAI 4008, 2006, pp.57-70.
29. Green D. J. (2005). Realtime Compliance Management Using a Wireless Realtime Pillbottle - A Report on the Pilot Study of SIMPILL. In: *Proc. of the International Conference for eHealth, Telemedicine and Health, Med-e-Tel'05*, 2005, Luxemburg.
30. Gross, J.J. (ed.) (2007). *Handbook of emotion regulation*. New York: Guilford Press.
31. Hoogendoorn, M., Jaffry, S.W., and Maanen, P.P. van., Validation and Verification of Agent Models for Trust: Independent compared to Relative Trust. Proceedings of the 5th IFIP WG

11.11 International Conference on Trust Management (TM'11), Springer Verlag, 2011, to appear.
32. Hoogendoorn, M., Jaffry, S.W., Maanen, P.P. van, and Treur, J., Modeling and Validation of Biased Human Trust. In: Boissier, O., et al. (eds.), *Proceedings of the 11th IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT'11*. IEEE Computer Society Press, 2011, to appear.
33. Josephson, J.R., and Josephson, S.G. (eds.) (1996), Abductive Inference: Computation, Philosophy, Technology. New York: Cambridge University Press.
34. de Kleer, J. (1986) An assumption-based TMS, *Artificial Intelligence* **28** (2) pp. 127-162
35. de Kleer, J. (1986) Problem solving with the ATMS, *Artificial Intelligence* **28** (2) pp. 197-224
36. Kowalski, R. and Sergot, M. (1986). A logic-based calculus of events. *New Generation Computing*, 4: 67-95.
37. Lee, Y., Khorasani, E. S., Rahimi, S., and Gupta, B. (2008). A generic mobile agent framework for ambient intelligence. In *Proceedings of the 2008 ACM Symposium on Applied Computing* (Fortaleza, Ceara, Brazil, March 16 - 20, 2008). SAC '08. ACM, New York, NY, 1866-1871.
38. Magnani, L., Carnielli, W., Pizzi, C., Carnielli, W.A., Claudio, P. (eds.), Model-Based Reasoning in Science and Technology, Abduction, Logic, and Computational Discovery. Studies in Computational Intelligence, vol. 314. Springer Verlag, 2010
39. Marek, V.W., Treur, J., and Truszczynski, M. (1997), Representation Theory for Default Logic. *Annals of Mathematics and AI*, vol. 21, pp. 343-358.
40. Marek, V.W., and Truszczynski, M. (1993), *Nonmonotonic Logics*. Springer Verlag.
41. Mileo, A., Merico, D., and Bisiani, R. (2007). CyberCare: Reasoning about Patient's Profile in Home Healthcare. In: Proceedings of the Symposium on "Artificial Societies for Ambient Intelligence", ASAmI'07, 2007.
42. Nersessian, N.J. & Patton, C.: "Model-based reasoning in interdisciplinary engineering" in *Handbook of the Philosophy of Technology and Engineering Sciences*, A. Meijers, ed. (Amsterdam: Elsevier, in press, 2009), pp. 687-718.
43. Niemelä, I., Simons, P., and Syrjänen, T. (2000). Smodels: a system for answer set programming. In: *Proceedings of the 8th International Workshop on Non-Monotonic Reasoning*, Breckenridge, Colorado, USA, April 2000.
44. Pearl, J. (1985). Bayesian Networks: A Model of Self-Activated Memory for Evidential Reasoning (UCLA Technical Report CSD-850017). *Proceedings of the 7th Conference of the Cognitive Science Society*, University of California, Irvine, CA. pp. 329–334.
45. Reiter, R. (1980) A logic for default reasoning. *Artificial Intelligence*, 13:81-132.
46. Reiter, R. (2001). Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems. Cambridge MA: MIT Press.
47. Riva, G., F. Vatalaro, F. Davide, M. Alcañiz (eds.) (2005). *Ambient Intelligence*. IOS Press, 2005.
48. Schmidt, A., Interactive Context-Aware Systems - Interacting with Ambient Intelligence. In: G. Riva, F. Vatalaro, F. Davide, , M. Alcañiz (eds.), *Ambient Intelligence*. IOS Press, 2005, pp. 159-178.
49. Schmidt, A., Beigl, M., and Gellersen, H.W. (1999), There is more to Context than Location. *Computers & Graphics Journal*, vol. 23, 19, pp.893-902.
50. Schmidt, A., Kortuem, G., Morse, D., and Dey, A. (eds.), Situated Interaction and Context-Aware Computing. *Personal and Ubiquitous Computing*, vol. 5(1), 2001, pp. 1-81.
51. Sharpanskykh, A., and Treur, J., A Temporal Trace Language for Formal Modelling and Analysis of Agent Systems. In: Dastani, M., Hindriks, K.V., and Meyer, J.J.Ch. (eds.), *Specification and Verification of Multi-Agent Systems*. Springer Verlag, 2010, pp. 317-352.
52. Sorenson, H.W. (1980). *Parameter Estimation: Principles and Problems*. Marcel Dekker, Inc., New York.
53. Stathis, K., and Toni, F. (2004). Ambient Intelligence using KGP Agents. *Lecture Notes in Artificial Intelligence*, 3295:351–362.
54. Tan, Y.H., and Treur, J. (1992) Constructive Default Logic and the Control of Defeasible Reasoning. In: B. Neumann (ed.), *Proc. ECAI'92*. Wiley and Sons, 1992, pp. 299-303.
55. Tawfik, A.Y. and E. Neufeld. Temporal reasoning and Bayesian networks. *Computational Intelligence*, 16(3):349–377, 2000.

56. Wilks, Y., and Hartley, R., Belief Ascription and Model Generative Reasoning: joining two paradigms to a robust parser of messages. In: Proceedings of the Workshop on Speech and Natural Language, pp. 219-240, 1989.
57. Zhang, Y., Hou, Y., Huang, Z., Li, H., Chen, R. (2006). A Context-Aware AmI System Based on MAS Model, ,pp.703-706, *2006 International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP'06)*.
58. http://workpace.com/