Belief Operations for Motivated BDI Agents

Patrick Krümpelmann Matthias Thimm
Manuela Ritterskamp Gabriele Kern-Isberner

Information Engineering Group, Department of Computer Science, Dortmund University of Technology

ABSTRACT

The beliefs of an agent reflecting her subjective view of the world constitute one of the main components of a BDI agent. In order to incorporate new information coming from other agents, or to adjust to changes in the environment, the agent has to carry out belief change operations while taking metalogical information on time and reliabilities into account. In this paper, we describe a framework for belief operations within a BDI agent, sketching the interactions of beliefs with desires and intentions, respectively. Furthermore, we illustrate how motivations and know-how come into play in our agent model of this framework. We focus on the presentation of a complex setting for belief change that makes use of techniques both from merging and update, and provides a BDI agent with advanced reasoning capabilities. Extended logic programs under the answer set semantics will serve as the basic knowledge representation formalism.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Intelligent agents, Multiagent systems

General Terms

Design, Theory, Human Factors

Keywords

Multiagent System, BDI, Belief Revision, Motivation

1. INTRODUCTION

For realising agents, the BDI model [16] has become a leading paradigm of the field. This model distinguishes between *Beliefs*, *Desires*, and *Intentions* as the main components of an agent's mind, the interactions of which determine her behaviour. Roughly, *Beliefs* comprise (plausible) knowledge the agent has concerning the current situation and how the world works in general, *Desires* encode what she wishes to achieve and hence represent possible goals, whereas *Intentions* focus on the next actions the agent should undertake to achieve the current goal. The role of *Beliefs* in this scenario is to provide the agent with useful information to evaluate the current situation and to find reasonable and effective ways to achieve her goals.

In this paper we introduce the KiMAS framework and sketch the complete model of a KiMAS agent. The KiMAS project *(Knowledge in Multiagent Systems)* [11] is a multiagent system, that focuses on the representation of beliefs and on methods to process information exchanged among the agents. It has been implemented in a student project group using the Jadex framework [1] and DLV [3] as the underlying answer set reasoner. The model of a KiMAS agent bases on the BDI model. In addition it explicitly differentiates between the logical belief about the world and the beliefs about how specific goals can be achieved (*know-how*), as the two types of belief influence the dynamics of the internal state of the agent differently. Furthermore, we introduce motivations to emphasise the autonomy and personality of the agents and to model the effect of their character on the selection of new goals.

We focus the representation of the KiMAS framework on the *Beliefs* component which may be quite complex in itself. Besides the representation of generic and evidential beliefs, it must also dispose of methods to process information, be it for inference, updating, or merging beliefs. However, the agent should not believe everything she is told, the reliability of information depends on the reliability of the source that it came from. Moreover, in a multiagent environment, pieces of information coming from different agents can be conflicting, so the agent should be provided with strategies to resolve such conflicts.

In this paper, we present a complex picture of belief operations, taking prior beliefs, new information, and metalogical information about time and credibility into account. We distinguish between belief bases, epistemic states, and belief sets, and realise dynamic belief changes via a combination of base merging and inductive inference by the aid of an update mechanism. Extended logic programs will serve as the basic medium to represent beliefs, and belief sets, as collections of most plausible beliefs, will be obtained via answer set semantics. This epistemic component will endow our agent with advanced reasoning capabilities.

The rest of this paper is organised as follows: In Sec. 2 we start with some preliminaries for extended logic programs under the answer set semantics. Sec. 3 covers the epistemic capabilities of the agents and introduces an exemplary scenario to be modeled using our multiagent framework. In Sec. 4 we describe the environment and the communication within the system. In Sec. 5 we introduce the extensions we made to the BDI model and the operations used to alter

Cite as: Belief Operations for Motivated BDI Agents, Patrick Krümpelmann, Matthias Thimm, Manuela Ritterskamp, Gabriele Kern-Isberner, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.),May,12-16.,2008,Estoril,Portugal,pp. 421-428.

the internal state of the agent. Sec. 6 focuses on operations for revising the belief base of agents, particularly on one of our implemented belief base operators. Then we conclude in Sec. 7 with perspectives for further work.

2. PRELIMINARIES

We are working with extended logic programs under the answer set semantics [6] which are capable of dealing with incomplete information in open environments. An extended logic program consists of rules over a set of atoms Σ using strong negation $\neg$ and default negation not. A literal L can be an atom A or a negated atom $\neg A$. The complement of a literal L is denoted by $\neg L$ and is A if $L = \neg A$ and $\neg A$ if $L = A$. The set of literals is denoted by $\mathcal{L}$.

A rule r is written as $H(r) \leftarrow \mathcal{B}$ where the head of the rule $H(r)$ is either empty or consists of a literal L and the body $\mathcal{B}$ is a set of literals $\{L_0, \ldots, L_m, \mathsf{not}\ L_{m+1}, \ldots, \mathsf{not}\ L_n\}$. The body consists of a set of default negated literals denoted by $\mathcal{B}^- = \{\mathsf{not}\ L_{m+1}, \ldots, \mathsf{not}\ L_n\}$ and a set of not default negated literals $\mathcal{B}^+ = \{L_0, \ldots, L_m\}$. Given this we can write a rule as $H(r) \leftarrow \mathcal{B}^+, \mathcal{B}^-$. If $\mathcal{B} = \emptyset$ we call r a fact. If $H(r) = \emptyset$ we call r a constraint.

A set of literals which is consistent, i. e. it does not contain complementary literals L and $\neg L$, is called an interpretation I. A literal L is *true* in I ($I \models L$) iff $L \in I$ and *false* otherwise. The body $\mathcal{B}$ of a given rule r is *true* in I iff each $L \in \mathcal{B}^+$ is *true* in I and each $L \in \mathcal{B}^-$ is *false* in I. A rule r is *true* in I ($I \models r$) iff $H(r)$ is *true* in I whenever $\mathcal{B}$ is *true* in I. An interpretation is a model of a program P ($I \models P$) if $I \models r$ for all $r \in P$.

The reduct P^S of a program P relative to a set S of literals is defined as: $P^S := \{H(r) \leftarrow \mathcal{B}^+(r) | r \in P, \mathcal{B}^-(r) \cap S = \emptyset\}$ An answer set of a program P is an interpretation I which is a minimal model of P^I. In our framework an agents epistemic state is represented by an extended logic program P.

3. AGENTS AND BELIEFS

3.1 The epistemic capabilities of agents

We assume our agents to be equipped with an epistemic state in which both factual and conditional knowledge can be represented and which is provided with a basic inference mechanism that allows the agent to draw logical and plausible conclusions. From a conceptual point of view, the distinction between evidential and generic beliefs is crucial. Whereas evidential knowledge or belief is often volatile and subject to changes, generic beliefs are usually deeper epistemically entrenched and can be applied to different situations. Mostly, generic beliefs have the form of rules that establish meaningful relations between isolated pieces of information and hence determine the intellectual capability of an agent.

Moreover, in order to judge information more adequately and to solve conflicts between contradictory pieces of information, some metalogical formalism to allow for qualitative or quantitative comparisons must be provided. In belief revision theory, often degrees of epistemic entrenchment or degrees of disbelief [5] are used. For the evaluation of incoming information, the agent will find it quite useful to estimate the degree of *reliability* of the corresponding source which will be the environment or another agent. We will usually assume that the agents can not be completely sure about the truth of information they get. Other agents may be dishonest or simply stupid, observations from the environment may be fallacious. Nevertheless, each agent (as in reality) should be capable of judging incoming information with regard to credibility, combining her currently held beliefs with the content and some meta-information about the corresponding message.

The basic structure that we will use here to implement an agent's epistemic state is given by extended logic programs under the answer set semantics [6]. Together with metalogical information on both time and credibility and a change mechanism, this framework will prove to be powerful enough to satisfy the requirements sketched above. To comply with the terminology of belief revision, logic programs will serve as basic ingredients for belief bases. A metalogical formalism will be provided by ranks of credibility that can be assigned to both rules and facts and turn the logic program into a ranked knowledge base. Any incoming information will also be assigned a rank in dependence of the reliability of its source. An update mechanism which is based on the ideas of [4] will make use of these ranks to incorporate the new information. To be more precise, updating in this scenario will mean to compare ranks and give way to the most credible information which may yield quite a complex restructuring of the logic program. In this way, all available information is compiled into the resulting update logic program, which will be used as an epistemic state. From this, we obtain belief sets as representation of the most plausible beliefs via (credulous) answer set inference.

The formal picture can be described as follows. Our update mechanism $\diamond$ takes any sequence of logic programs and yields a logic program, which represents as much information of the sequence as possible and solves conflicts by taking the order of the sequence into consideration:

$$\diamond : (P_1, \ldots, P_n) \mapsto P^*$$

We stick to the term "update" here, although this operation may also be looked upon as a kind of inductive reasoning for ranked knowledge bases. The update operator is the most important element within our framework of belief operations, as it tightens up loosely linked belief base informations to return an epistemic state P^*. From this, the belief set $Bel(P^*)$ is computed by (credulous) answer set semantics as a set of literals. Please note that we give up the demand for deductive closure here which is usually imposed on belief sets.

However, the result of update depends crucially on the input sequence of logic programs. Basically, this sequence is built up from the agent's initial beliefs and from new pieces of information that the agent receives over time. Prior belief bases (represented as ranked logic program bases) and new information are merged via a non-prioritized base revision operation which yields a posterior ranked program base. In this way, ideas from both revision and update formalisms are used and interlaced in our approach, based on the unifying belief change framework developed in [8].

We postpone further explanations and discussions to Section 6. First, we will present an example that will illustrate ideas and techniques and continue with a general representation of our multiagent system.

3.2 The Case of Evelyn

For our implemented system we used an adaptation of a criminal story by Agatha Christie as a complex scenario.

The following typical fragment will be used to illustrate our multiagent system throughout this paper:

> Evelyn was murdered on Friday either by Bob (her husband) or by Carl (her ex-lover). On Saturday the police officer Alice is investigating this crime and with her are Bob, Carl and Dave, Evelyns brother, who lives in the same house as Evelyn and Bob and serves as a witness throughout the investigation.

This scenario yields to a multiagent system which consists of four agents: *Alice*, *Bob*, *Carl*, and *Dave*. In this system, the agent *Alice* tries to determine who is the murderer of *Evelyn*. All agents have specific beliefs, which will influence their behaviour in the system. The initial beliefs of the agents are informally given by

- *Alice's* beliefs:
  - Between Tuesday and Thursday *Bob* found out, that *Evelyn* had an affair with *Carl*.
  - If someone threatens someone else to kill him and this one got killed during the next few days and the first one has no alibi for the time of crime, then the first one is the murderer.
  - *Evelyn* was jogging on Wednesday and on Friday.
- *Bob's* beliefs:
  - *Evelyn* told *Bob* on Thursday, that she had an affair with *Carl* but quit the relationship.
  - *Bob* was on a work trip the whole Wednesday.
  - *Bob* was on a work trip the whole Friday.
- *Carl's* beliefs:
  - *Carl* murdered *Evelyn* on Friday.
  - *Bob* was on a work trip the whole Wednesday.
  - *Carl* had a quarrel with *Evelyn* on Wednesday. He was angry, because she broke up with him.
- *Dave's* beliefs:
  - On Wednesday *Dave* overheard a quarrel between *Evelyn* and (apparently) her husband. During this quarrel (apparently) *Bob* told *Evelyn*, that he hates her and wishes she would be dead.
  - Each time *Bob* is on a work trip, *Evelyn* goes jogging that day in the morning (and only then).
  - A quarrel between two persons is a disharmony between them.
  - On Wednesday *Dave* saw *Carl* in the house.

Figure 1 shows (informally) an exemplary dialog between the participating agents that might happen in this scenario (agent names are abbreviated by their first letters). This dialog will be used in the upcoming sections to illustrate important properties of our multiagent system.

We do not give a full representation of the beliefs as an answer set program but exemplary for some fragments.

Example 1. *The initial (logical) beliefs of* Dave *can be represented as:*

$$quarrel(bob, evelyn). \quad threatened(bob, evelyn).$$
$$jogging(evelyn, DAY) \leftarrow worktrip(bob, DAY).$$
$$worktrip(bob, DAY) \leftarrow jogging(evelyn, DAY).$$
$$disharmony(X, Y) \leftarrow quarrel(X, Y).$$

A asks C:	"Did you murder Evelyn?"
C answers A:	"No."
A asks B:	"Did you murder Evelyn?"
B answers A:	"No."
A asks D:	"Did you notice any disharmony between Evelyn and Bob or Evelyn and Carl?"
D answers A:	"I overheard a quarrel on Wednesday, where Bob wished Evelyn was dead." *(Alice now thinks, that Bob is the murderer)*
A asks B	"Where were you on Friday?"
B answers A	"I was on a work trip."
A asks D	"Can you confirm this statement?"
D answers A	"I could, if you can tell me, if Evelyn was jogging on Friday."
A answers D	"Evelyn was jogging on Friday."
D answers A	"Then I can confirm this statement." *(Bob has an alibi)*
A asks B	"Where were you on Wednesday?"
B answers A	"I was on a work trip."
A asks D	"Can you confirm this statement?"
D answers A	"I could, if you can tell me, if Evelyn was jogging on Wednesday."
A answers D	"Evelyn was jogging on Wednesday."
D answers A	"Then I can confirm this statement."
A asks C	"Did you meet Evelyn on Wednesday?"
C answers A	"No."
A asks D	"Is that true?"
D answers A	"No." *(Alice infers that Carl was the one who threatened Evelyn and therefore he is the murderer)*

Figure 1: Exemplary dialog in the *Evelyn*-scenario

4. COMMUNICATION/INTERACTION

Every action an agent performs in a given world situation results in a specific event in this situation. As possibly many different agents may be present at this event, they each perceive it in a different way. In our system this is realised using a special abstract entity as the environment, which monitors the world evolvement and notifies present agents on occurring events. In general agents may perform many different kinds of actions. As we focus our work in this paper on the belief component, we do not emphasise the description of a situation calculus. Therefore we restrain the set of possible actions to message actions, where one agent wants to communicate an information to one or more other agents. Our communication protocol is loosely based on KQML[2] and thus follows the directives of *speech act theory* [12]. Message actions result in one or more messages which are sent to the present agents.

Definition 1 (Message). *A* Message *is a quadruple (S, R, T, C), where S is the sender, R is the set of receivers, T is the type and C is the content of the message. The set of all possible messages is denoted by $\mathcal{M}$.*

In our system, the content of a message is represented as an extended logic program.

Once an agent performs a message action, the resulting event is interpreted by the environment and translated into one or more messages to specific agents. This way the eavesdropping of messages without notice of the participating agents is made possible. Furthermore the environment can

alter the content of the message if appropriate, e. g removing the sender of the message if the communication channel is somehow distorted.

Although *speech act theory* describes many types of messages, we only use two: *query* (q) and *general message* (g). If an agent poses a query to another agent, she expects an answer from her which must be of type *general message*. In general the type *general message* is used to exchange information between agents.

We distinguish three subtypes of queries:

1. *instantiate*-queries: the querying agent wants the answering agent to instantiate a given non-fully-grounded predicate, e. g. the query of *Alice* to *Bob*, in which *Alice* wants to know where *Bob* was on Friday can be represented as $location(bob, X, friday)$ and then the answer of *Bob* would be $location(bob, worktrip, friday)$.
2. *element-info*-queries: the querying agent wants to obtain some information about an individual or an object, e. g. *Alice* asks *Dave* what he knows about *Bob*. In this case the query would only contain the constant "*bob*" and *Dave* could answer with $honest(bob)$ or $loves(bob, evelyn)$ or both.
3. *yes/no*-queries: the querying agents wants to know if the other agent believes the given ground instance of a specific predicate, e. g. *Alice* asks *Carl* if he murdered *Evelyn*, i. e. she asks $murderer(carl, evelyn)$, and he returns "*no*".

Example 2. *Consider the* Evelyn*-scenario of Section 3.* Dave *overheard a quarrel between* Evelyn *and apparently* Bob. *But the actual initiator of the message act was* Carl. *The message action of* Carl *contained the information, that he hates* Evelyn *(*$hate(carl, evelyn)$*). Then the environment sends the message* $(carl, \{evelyn\}, g, hate(carl, evelyn))$ *to* Evelyn*, because she was near* Carl *when he performed this action.* Dave *was in a room next to the room of the message action, so the communication channel has been distorted. The environment attends this distortion by altering the message that is sent to* Dave. *Let* x *denote an unknown constant, then* Dave *would receive the message* $(x, \{evelyn\}, g, hate(x, evelyn))$. *Internally* Dave *would assume that* x *must be "bob" (perhaps because he thinks* Bob *is at home) and appropriately update his beliefs.*

5. AGENT MODEL

Our agent model is inspired by the commonly known BDI-model, see [16] for an overview. The BDI-model divides an agent's interior state into three components: *Beliefs*, *Desires* and *Intentions*. We denote *Des* as the set of all desires or possible goals an agent can have where a desire or a possible goal is an atom that an agent wants to become true in her beliefs. An agent maintains a subset $D \subseteq Des$ of all possible goals, e. g. agent *Alice* in the running example (informally) has desires $D = \{murder\_case\_solved, being\_healthy\}$. Every agent has at each moment one selected goal, which he currently pursuits, denoted $selected(D)$. Thus agent *Alice* has $selected(D) = murder\_case\_solved$. Similary an agent maintains a set of abstract intentions $I \subseteq Int$, where *Int* denotes the set of all possible intentions. Intentions describe the aims of the agent he currently pursuits in order to fulfill her selected goal. They are represented as atoms as well but are easier to fulfill than desires. Everytime an agent selects a new goal to pursuit, this goal also becomes the next intention. In general at any time the set I correlates directly to the current pursuited goal $selected(D)$ and contains the next subgoals the agent wants to become true in order to fulfill $selected(D)$. Some intentions can be directly fulfilled by performing an atomic action. These intentions are called *atomic intentions*. The set I is often seen as a stack [7], where the top element might be removed by an atomic action or be split up into more concrete intentions and stacked upon I. When the stack is empty, the selected goal is fulfilled.

An agent in our framework interacts with her fellow agents by means of messages as defined in Definition 1. In messages our agents exchange extended logic programs. When interacting they also keep track of the temporal order in which they receive messages and internally build up *information objects*.

Definition 2 (Information Object). *An* information object *is a tuple* $I = (P, S, T)$*, where* P *is an extended logic program,* S *is the source of* P *and* T *represents the point in time* P *has been received.*

These information objects are then used to represent an agents belief base.

Definition 3 (KiMAS belief base). *A* KiMAS belief base *of an agent* A *is a set of information objects* $KB^A = \{I_0^A, \ldots, I_n^A\}$ *containing all information objects maintained from received messages and the initial beliefs represented by a special information object* $I_0^A = (P_{ini}^A, A, t_0)$*. Let* $\mathcal{K}$ *be the set of all* KiMAS *belief bases.*

We further extend the agent model by introducing *know-how* [15, 14] and *motivation* [9, 10] as explicit components in the representation of an agent.

Definition 4 (KiMAS agent). *A* KiMAS agent *is a tuple* $A = (KB, K_h, D, I, C, M, \varphi)$*, where* $KB \in \mathcal{K}$ *is the belief base of the agent,* K_h *is the know-how of the agent,* $D \subseteq Des$*,* $I \subseteq Int$*,* C *is a set of atomic actions the agent can perform,* M *is the (basic) motivation of the agent and* φ *is a functional component.*

The functional component of a KiMAS agent describes the dynamics of the behaviour of the agent, e. g. belief revision or goal generation. In the following subsections we further discuss know-how, motivation and the functional component of a KiMAS agent.

5.1 Know-How

While in the BDI-model the whole belief of an agent is capsulated in one component, we explicitly distinguish between the logical belief, also called know-that, and know-how [14].

In [14] know-that is described as the belief an agent has about the world, the current situation and her factual knowledge. Besides that, a reasonable agent also has beliefs about how to act to reach specific goals. For example, agent *Alice* in our running example knows, that in order to determine the murderer of *Evelyn* she first has to determine the suspects and find some evidence. Without some structural knowledge on goals and subgoals an agent can not relate her actions to her goals and therefore can not act goal-oriented.

While in [14] know-how is discussed under a very theoretical point of view, we take here a more pragmatical view on

know-how, as we see it as a tool for an agent that gives him the capability of fundamental planning routines. Know-how structures intentions into hierarchies which helps an agent to split up abstract intentions into more concrete intentions, which can be fulfilled more easily, down to atomic intentions, which can be fulfilled by an atomic action. For the next definition let S^* denote all ordered sequences of elements of the finite set S.

DEFINITION 5 (KNOW-HOW). *Know-how K_h is a set of pairs (α, β), s. t. $\alpha \in Int$ and $\beta \in Int^*$. Let $\mathcal{KH}$ denote the set of all possible know-how structures.*

EXAMPLE 3. *The intention murder_case_solved of* Alice *could be split up into less abstract intentions*

$$\beta = (suspects\_found, evidences\_found).$$

Thus $(murder\_case\_solved, \beta)$ might be in the know-how of Alice.

Furthermore, there may be other pairs with the intention *solve_murder_case* as the first component in the know-how of *Alice*, which gives her the possibility to choose from different alternatives to pursue her goals. Moreover, as the intentions *suspects_found* and *evidences_found* are probably not atomic intentions, these should be further split up by appropriate pairs in the know-how of *Alice*.

Know-how in our model is not an active component of an agent but a data structure. It is used by specific functions of an agent to deliberate what to do next. These functions will be presented in Section 5.3.

5.2 Motivation

The desires *Des* of autonomous agents correspond to the set of possible goals. An agent maintains a subset $D \subseteq Des$, but she might not be able to aim at several of these desires simultaneously. Thus a mechanism is needed to decide, which $d \in D$ will be taken into consideration next. In this context we introduce motivations [9, 10]. The possible motivations *Mot* are non-derivative components that characterise the personality of an autonomous agent and provide the agent with a higher-level control.

These components, e. g. greed or altruism, do not specify a state of affairs to be achieved and can hardly be described in logical terms. Therefore they are not equal to the notion of goals in the classical sense of artificial intelligence. Instead motivations provide reasons for a goal, which could be having someone else's money or being generous. More precisely, in our system a motivation M induces a total preorder $\leq_M$ on the set of possible goals, which is a total, transitive and reflexive relation. Let d_1, $d_2 \in D$ be two desires, then $d_1 \leq_M d_2$ iff the motivation M for d_1 is at least as strong as for d_2. Thus the set of the agent's desires is partitioned into several sets $D = (D_0^M, \ldots, D_k^M)$ with

$$d_1 \in D_i^M \wedge d_2 \in D_j^M \wedge\ i \leq j \ \Leftrightarrow\ d_1 \leq_M d_2$$

The next goal, $selected(D)$, will be a randomly selected $g \in D_0^M$, as the set D_0^M contains the desires which are motivated the most.

To simplify matters we let every agent be driven by only one motivation M, which forms the personality of the agent. In the case of *Alice* her motivation M is the sense of justice:

EXAMPLE 4. *Let $M = sense\_of\_justice$ be the basic motivation of Alice and let murder_case_solved and being_healthy be her only desires. The set of desires will be partitioned into $D_0^M = \{murder\_case\_solved\}$ and $D_1^M = \{being\_healthy\}$, as Alice's motivation provides a reason to solve the case while it does not amplify the volition to eat healthy food. Thus murder_case_solved will be selected as her next goal.*

5.3 The functional component

Given an initial internal state of a KiMAS agent, namely KB, K_h, D, I, C, M, the agent acts and evolves in her environment and thus her internal state changes over time. The procedures that determine how an agent uses her current state to deliberate her next actions are described by her *functional component*. The *functional component* of a KiMAS agent is a set of functions, which revises her internal state in a particular situation or outputs an action that the agent wants to perform.

DEFINITION 6 (FUNCTIONAL COMPONENT). *We call a tuple $\varphi = (\alpha_{bel}, \alpha_{goal}, \alpha_{subgoal}, \alpha_{action})$ a* functional component, *if α_{bel} is a belief operation, α_{goal} is a goal generation operation, $\alpha_{subgoal}$ is a subgoal generation operation and α_{action} is an action selection operation.*

At first an agent has to handle incoming messages by revising her beliefs of the world accordingly. A belief operation is a function, that revises (in an abstract manner) the logical belief base of an agent appropriately to a newly given evidence, i. e. a newly received message.

DEFINITION 7 (BELIEF BASE OPERATION). *A* belief operation *is a function $\mathcal{K} \times \mathcal{M} \to \mathcal{K}$.*

A belief base operation incorporates a newly given evidence into an agents belief base. In our approach this is done by just merging the new information into the belief base and let an epistemic belief operation do the actual belief operation, e. g. revision or update. In the upcoming section we describe our solution of an epistemic belief operator, which follows the approach of causal rejection [4].

The main purpose of an agent is to act in her environment. As our agents are goal-driven, they have to act according to their current goals and intentions. Suppose an agent has no intentions at a given time ($I = \emptyset$). This means, that the agent does not currently pursue any goal and therefore there is no need to act in any way. In this case the agent has to select a goal from her desires D and make it her next intention. As described in Subsection 5.2 the motivation M of the agent is used for the determination of a new selected goal. A goal generation operation is a function, which uses an agent's motivation to generate the newly selected goal, which is put onto her intentions stack. It is also influenced by the current belief of an agent, as some goals might only be triggered, if some outer circumstances hold.

DEFINITION 8 (GOAL GENERATION). *A* goal generation operation *is a function $\mathcal{K} \times Mot \times \mathfrak{P}(Des) \to Int$.*

Intentions can be fulfilled directly by an atomic action or must be split up in less abstract intentions by using the agent's know-how. So if the intentions of an agent are not empty ($I \neq \emptyset$) but there is no intention directly executable by an atomic action, then a less abstract subgoal has to be generated. A subgoal generation operation uses the agents know-how to split up the current pursuited goal or the next intentions into more concrete intentions.

DEFINITION 9 (SUBGOAL GENERATION). *A* subgoal generation operation *is a function* $\mathcal{K} \times \mathcal{KH} \times Int \rightarrow \mathfrak{P}(Int)$.

Intentions are split up recursively down to atomic intentions. If at least one intention is atomic, then the agent can select the appropriate atomic action, execute it and remove the intention from I. If in one situation more than one action is possible, then an agent has to select one of these actions first. An action-selection operation chooses one action to perform from a set of actions. We abstract here from any preferences that the agent might have in order to decide which action to perform.

DEFINITION 10 (ACTION SELECTION). *An* action selection operation *is a function* $\mathfrak{P}(Int) \times \mathfrak{P}(C) \times \mathcal{K} \rightarrow C$.

Figure 2 shows a graphical representation of a KiMAS agent. In this representation the functional dependencies are represented by dashed lines and action flow is represented by solid lines.

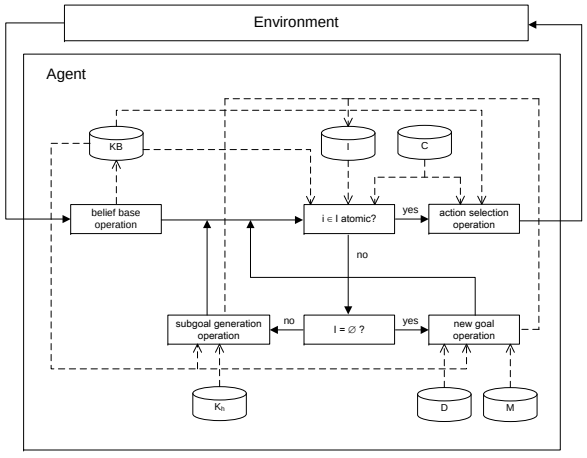

Figure 2: A graphical representation of a KiMAS-Agent

EXAMPLE 5. *In our example consider the situation, in which* Alice *asks* Dave *if he could confirm that* Bob *was on a work trip on Friday. At first* Dave *updates his beliefs KB with the information, that* Alice *wants to know the truth value of* $worktrip(bob, friday)$. *As* Dave *does not pursue any intentions at this time, he has to generate a new goal in order to react to this change of beliefs accordingly. The basic motivation of* Dave *here is* $family\_bond$ *and this motivation leads to the goal* $sisters\_murderer\_determined$. *Based on this goal and the information that providing* Alice *with the needed knowledge would help to fulfill that goal,* Dave *needs to know, whether* $jogging(evelyn, friday)$ *is true (because then he can apply the rule* $worktrip(bob, DAY) \leftarrow jogging(evelyn, DAY)$*) in order to answer* Alice*'s query. Observe that* Dave *could also just answer unknown (a valid answer in this situation) instead of a useful answer, but this is not compatible with* Dave*'s motivation. So* Dave *brings up the new intentions* $was\_evelyn\_jogging\_on\_friday$ *and* $answer\_to\_alice$. *The first intention can be fulfilled by asking* Alice *about the truth value of* $jogging(evelyn, friday)$. *After getting the answer true from* Alice *and integrating* $jogging(evelyn, friday)$ *into his beliefs* Dave *now believes that* $worktrip(bob, friday)$ *is true. Now he can directly fulfill the intention* $answer\_to\_alice$ *by confirming the original statement.*

6. BELIEF OPERATIONS

The epistemic capabilities of the agents in our framework were introduced in Section 3.1. We will now describe the actual belief operations. Each agent has her initial beliefs in form of evidential knowledge and generic beliefs which are deeper entrenched than knowledge acquainted from other agents. The agent has to deal with messages received from other agents and from the environment itself. As stated, agents are not necessarily honest nor impeccable and thus an agent has to be careful when dealing with information obtained from such, more or less credible, sources.

Our agents obtain information by receiving and processing messages, which contain the sender of the message. Thus, for each piece of information the agents know its origin and can use this to evaluate the truthfulness of this particular information.

DEFINITION 11 (ASSESSMENT). *An* assessment c *is a function* $c : \mathcal{A} \rightarrow \{0, \ldots, N\}$ *assigning a credibility value from a linearly ordered scale to an agent.*

An assessment is used to represent the credibilities an agent gives to other agents known to her. These credibilities are kept in a separate structure to allow for a dynamic change of credibilities within time.

The belief bases of our agents are structured as stated in Definition 3. If an agent wants to perform reasoning an epistemic operator is applied to the belief base to get her current epistemic state.

DEFINITION 12 (EPISTEMIC OPERATOR). *A function* $\beta : \mathcal{K} \rightarrow \mathcal{P}$ *which generates an extended logic program from a KiMAS belief base is called an* epistemic operator.

The resulting epistemic state of the epistemic operator is given to an answer set reasoner e. g. DLV [3] or smodels [13] which computes answer sets representing the belief sets of the agent.

The belief base structure is based on information objects, which include meta information about time and source of information. The sources of information again are connected to current credibility values by means of an assessment. This structure enables epistemic operators to include credibility and temporal information into the reasoning process. Our framework can deal with a variety of epistemic operators potentially varying between different agents.

In the following we will describe the epistemic operator we developed and implemented. It is based on and extending [4], utilizing the structure and possibilities of our framework. We are working on a KiMAS belief base $KB^A = \{I_0^A, \ldots, I_n^A\}$ which consists of a set of extended logic programs with meta information source s and time of reception t over the alphabet $\mathcal{L}$. We assume, that only one piece of information can be received at each point in time. Our goal is to compile this set into one single, consistent extended logic program $P_\diamond$. In the agents assessment a credibility is assigned to each source known to her. A default credibility is assigned to unknown sources. Initial information or information coming directly from the perception of the environment can be modeled by

giving it the maximum credibility. These credibilities are, augmented by the time information, used to give priorities to the programs received from other agents. Instead of giving priorities to rules in a static manner like other approaches following the causal rejection principle we prioritize literals and take the priorities of the body literals of a rule into consideration to determine the priority of the associated head literal, giving a more truthful priority to inferred literals. The priority of the head literal is based on the body literals which originate directly from sources with varying credibility, or are inferred within the program themselves. We use a cautious reasoning approach giving the head literal the minimum of the body literal priorities. However, we have to take the priority of the respective rule into consideration. We do this by not allowing the head literal to be prioritized higher than the rule is prioritized itself. At last we have to deal with the fact, that the same instances of literals can be inferred in different ways or can come from different sources and hence exist parallel with different priorities. To take this into consideration in the determination of the head literal priority we use the maximal priority that we have for a given literal. We accomplish this priority selection and propagation solely using aggregate functions available in answer set solvers but do not rely on them.

We will denote the set of programs of KB^A by $\mathbf{P}$. The original alphabet $\mathcal{L}$ of $\mathbf{P}$ is extended to $\mathcal{L}'$ by adding several new predicates and atoms, which we will introduce now. For each literal $L(\vec{x_L})$ occurring in $\mathbf{P}$ a prioritized version $\hat{L}(\vec{x}_L, \mu)$ is added. Here, the variable μ represents the priority of the considered instance of the literal L. Let $\mathcal{H}(\mathbf{P})$ denote the set of literals occurring in the head of a rule r in $\mathbf{P}$. Predicates named $rej_L(\vec{x}_L, \mu)$ are introduced for each literal $L \in \mathcal{H}(\mathbf{P})$. $\mathcal{B}(r)$ is the set of literals in the body of the rule r. Let the priority of program P_i be given by $\mu(P_i)$.

In a preprocessing step we sort the set $\mathbf{P}$ lexicographically in s and t of KB^A respectively. Then we give priorities to the individual programs according to this ordering obtaining a sequence $(P_0, \ldots, P_n)$ of logic programs. Now we are able to describe the construction of the program $P_\diamond = P_0 \diamond \ldots \diamond P_n$ in four steps by generating the following rules:

1. for each rule $r \in P_i, 0 \leq i \leq n$ with head $H(\vec{x}_H)$:

$$\begin{aligned} \hat{H}(\vec{x}_H, \mu) &\leftarrow \mathcal{B}(r), \mathsf{min}(MaxB(\vec{x}_B, r)) = \mu. \\ MaxB(\vec{x}_B, r) &= \{\mu_{max}(B(\vec{x}_B))| \\ &\quad \hat{B}(\vec{x}_B, \mu') \in \mathcal{B}^+(r)\} \cup \{\mu(P_i)\} \\ \mu_{max}(B(\vec{x}_B)) &= \mathsf{max}\{\mu' | \hat{B}(\vec{x}_B, \mu') \in \mathcal{H}(\mathbf{P})\} \end{aligned}$$

2. for each literal $L \in \mathcal{H}(\mathbf{P})$:

$$L(\vec{x}_L) \leftarrow \hat{L}(\vec{x}_L, \mu), \mathsf{not}\ rej_L(\vec{x}_L, \mu).$$

3. for each literal $L \in \mathcal{H}(\mathbf{P})$:

$$rej_L(\vec{x}_L, \mu) \leftarrow \hat{L}(\vec{x}_L, \mu), \neg\hat{L}(\vec{x}_L, \mu'), \mu < \mu'.$$

4. all constraints from $P_i, 0 \leq i \leq n$

In the first construction step the original rules from $\mathbf{P}$ are adopted. Their literals are replaced by their corresponding prioritized versions and the rules are extended in a way to support the propagation of priorities. To determine the priority that should be propagated to the head literal the minimum of the maximum priorities of the body literals is used. A literal can be inferred in different rules with different priorities. That is why we consider the maximum priority of each literal. Only body literals are considered which are not default negated. This is done as the priority of a default negated literal should not give us any information about the priority we should assign to the head literal.

The second construction step connects the prioritized to the unprioritized layer of literals. A literal L holds if a prioritized version of it holds and is not blocked due to conflicts.

The third construction step introduces the rej predicates which ensure that no prioritized literal is connected to its unprioritized equivalent if this would lead to an inconsistency. A literal is blocked if it is conflicting with a higher prioritized instance of itself. Through this, more reliable or more current information dominates less reliable or older information.

The constructed program $P_\diamond$ is given to an answer set reasoner to compute answer sets for it. A resulting answer set S' for the program $P_\diamond$ is defined over the alphabet $\mathcal{L}'$ and has to be projected to the original alphabet $\mathcal{L}$.

DEFINITION 13. *S is an answer set of $\boldsymbol{P} = (P_1, \ldots P_n)$ iff $S = S' \cap \mathcal{L}$ for an answer set S' of $P_\diamond$*

We illustrate this mechanism in our running example.

EXAMPLE 6. *Consider the following belief base of* Alice*:*

$$\begin{aligned} KB^{Alice} = \{(\{&murdered(evelyn). \\ &suspect(X) \leftarrow threatened(X,Y), murdered(Y). \\ &threatened(X, evelyn) \leftarrow met(X, evelyn, wed). \\ &murderer(X) \leftarrow suspect(X), \mathsf{not}\ alibi(X). \\ &\}, Alice, t_0), \\ &(\{\neg murderer(carl, evelyn)\}, Carl, t_1), \\ &(\{\neg murderer(bob, evelyn)\}, Bob, t_2), \\ &(\{alibi(bob)\}, Dave, t_3), \\ &(\{met(carl, evelyn, wed)\}, Dave, t_4), \\ &(\{\neg met(carl, evelyn, wed)\}, Carl, t_5), \} \end{aligned}$$

Looking at the belief base it is obvious that there are several conflicts. In addition to the belief base Alice *has an assessment function $c : \mathcal{A} \rightarrow \{0, \ldots, 10\}$ (the interval might be any other limited interval).*

$$c(alice) = 10, c(carl) = 3, c(bob) = 3, c(dave) = 6$$

As Alice *is self-confident she assigns the maximum credibility to herself, her initial beliefs respectively, and only to herself.* Carl *and* Bob *are suspects and get small credibility.* Dave*, as* Evelyn*'s brother, is more credible.*

Starting at this state the epistemic operator generates the epistemic state. As described earlier the programs are ordered lexicographically in their assigned credibility and time. In the example this leads to the following order:

$$\begin{aligned} P_1 &: \{\neg murderer(carl, evelyn).\} \\ P_2 &: \{\neg murderer(bob, evelyn).\} \\ P_3 &: \{\neg met(carl, evelyn, wed).\} \\ P_4 &: \{alibi(bob).\} \\ P_5 &: \{met(carl, evelyn, wed).\} \\ P_6 &: \{murdered(evelyn).\} \\ P_7 &: \{suspect(X) \leftarrow threatened(X,Y), murdered(Y). \\ &\quad threatened(X, evelyn) \leftarrow met(X, evelyn, wed). \\ &\quad murderer(X) \leftarrow suspect(X), \mathsf{not}\ alibi(X).\} \end{aligned}$$

Note that we separated Alices initial beliefs into factual and rule based knowledge. From these rules the program $P_\diamond$ is generated as described above and is given to an answer set solver. The resulting answer set is the following:

$$\begin{aligned} S' \;=\; \{&\neg murderer(carl,1), \neg murderer(bob,2),\\ &\neg met(carl,evelyn,wed,3), alibi(bob,4),\\ &met(carl,evelyn,wed,5), murdered(evelyn,6),\\ &suspect(carl,5), threatened(carl,evelyn,5),\\ &murderer(carl,5), murderer(carl), alibi(bob),\\ &met(carl,evelyn,wed), \neg murderer(bob),\\ &murdered(evelyn), suspect(carl),\\ &threatened(carl,evelyn), rej_{\neg murderer}(carl,1),\\ &rej_{\neg met}(carl,evelyn,wed,3)\} \end{aligned}$$

Here we can see how the solving of conflicts on the prioritised predicates worked. According to Definition 13 the projected answer set looks like this:

$$\begin{aligned} S \;=\; \{&murderer(carl), alibi(bob), suspect(carl),\\ &met(carl,evelyn,wed), \neg murderer(bob),\\ &murdered(evelyn), threatened(carl,evelyn)\} \end{aligned}$$

Given this answer set Alice *beliefs that* Carl *was the one who met* Evelyn *on Wednesday and thus the one who threatened* Evelyn. *Given this and the fact, that* Bob *has an alibi she concludes, as desired, that* Carl *is the murderer of* Evelyn.

7. CONCLUSION AND FUTURE WORK

The adequate processing of information received from the environment is a principal prerequisite for intelligent behaviour. However, the agent should not simply believe whatever she is told, but should be able to judge each new piece of information by its importance and reliability.

In this paper, we presented belief change operations within a BDI agent in a multiagent setting. From the perspective of each agent, the other agents are assigned degrees of reliability which determine the credibility of their utterances. In combination with a time stamp, the credibility is used on a meta-level to generate ranked knowledge bases as belief bases for each agent. From this belief base, an epistemic state is computed by update techniques for logic programming, and answer sets are taken to represent most plausible beliefs. In this way, a complete information processing mechanism is described, taking prior beliefs of the agent as well as content and context of new information into account. We also implemented a motivation for each agent via preference relations on possible goals, helping her to pursue a personal line of goals. All this is embedded into the BDI model, and we made interactions between the different components explicit by specifying a functional component.

As part of our current work, we plan to extend the described belief component in order to realise substantially different belief change operations; for instance, the agents should be able to distinguish between revision and update. We will also investigate the formal properties of our epistemic operator according to well-known postulates for belief change and show the advantages of the propagation of priorities in comparison to other known mechanisms. As agents may have more than one motivation, we will study possible interactions of different motivations within one and the same agent, applying techniques from preference fusion.

Acknowledgments We are thankful to the anonymous reviewers for their helpful comments and to the students of the Project Group 491 at the Dortmund University of Technology for their inspiring discussions.

8. REFERENCES

[1] L. Braubach, A. Pokahr, and W. Lamersdorf. Jadex: A BDI agent system combining middleware and reasoning. In R. Unland, M. Calisti, and M. Klusch, editors, *Software Agent-Based Applications, Platforms and Development Kits*. Birkhäuser Book, 2005.

[2] H. Chalupsky, T. Finin, R. Fritzson, D. McKay, S. Shapiro, and G. Weiderhold. An overview of KQML. Technical report, KQML Advis. Group, 1992.

[3] T. Eiter, W. Faber, C. Koch, N. Leone, and G. Pfeifer. DLV - A system for declarative problem solving. In C. Baral and M. Truszczynski, editors, *Proc. of the 8th Int. Workshop on Non-Monotonic Reasoning*, 2000.

[4] T. Eiter, M. Fink, G. Sabbatini, and H. Tompits. On properties of update sequences based on causal rejection. *Theory Pract. Log. Program.*, 2(6), 2002.

[5] P. Gärdenfors and H. Rott. Belief revision. In D. M. Gabbay, C. J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*. Oxford University Press, 1995.

[6] M. Gelfond and N. Leone. Logic programming and knowledge representation — The A-Prolog perspective. *Artif. Intelligence*, 138(1–2):3–38, 2002.

[7] F. Ingrand, M. Georgeff, and A. Rao. An architecture for real-time reasoning and system control. *IEEE Expert*, 7 (6), 1992.

[8] G. Kern-Isberner. *Conditionals in nonmonotonic reasoning and belief revision*. Springer, Lecture Notes in Artificial Intelligence LNAI 2087, 2001.

[9] M. Luck and M. d'Inverno. Motivated behaviour for goal adoption. In Zhang and Lukose, editors, *Proc. of the 4th Australian Workshop on Distributed Artificial Intelligence*, pages 58–73. Springer, 1998.

[10] T. J. Norman and D. Long. Goal creation in motivated agents. In M. Wooldridge and N. R. Jennings, editors, *Intelligent Agents: Theories, Architectures, and Languages*, pages 277–290. Springer, 1995.

[11] Projektgruppe 491. Endbericht der Projektgruppe 491. Technical report, University of Dortmund, 2007.

[12] J. R. Searle. *Speech Acts: An Essay in the Philosophy of Language*. Cambrige University Press, 1969.

[13] P. Simons, I. Niemelä, and T. Soininen. Extending and implementing the stable model semantics. *Artificial Intelligence*, 134(1-2):181–234, 2002.

[14] M. Singh. Know-how. In A. Rao and M. Wooldridge, editors, *Foundations of Rational Agency, Applied Logic Series*, pages 105–132. Kluwer, 1999.

[15] M. Singh, A. Rao, and M. Georgeff. Formal methods in DAI: Logic-based representation and reasoning. In G. Weiss, editor, *Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence*, pages 331–376. MIT Press, Cambridge, Massachusetts, 1999.

[16] G. Weiss, editor. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1999.