Managing Different Sources of Uncertainty in a BDI Framework in a Principled Way with Tractable Fragments

Kim Bauters K.BAUTERS@QUB.AC.UK
Kevin McAreavey KEVIN.MCAREAVEY@QUB.AC.UK
School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Computer Science Building, 18 Malone Road, BT9 5BN Belfast, United Kingdom

Weiru Liu WL14805@BRISTOL.AC.UK
Merchant Venturers School of Engineering, University of Bristol, 75 Woodland Road, BS8 1UB Bristol, United Kingdom

Jun Hong JUN.HONG@UWE.AC.UK
Department of Computer Science and Creative Technologies, University of the West of England, Coldharbour Lane, BS16 1QY Bristol, United Kingdom

Lluís Godo GODO@IIIA.CSIC.ES
Carles Sierra SIERRA@IIIA.CSIC.ES
Institut d'Investigació en Intel·ligència Artificial, Consejo Superior de Investigaciones Científicos, Campus de la UAB, E-08193 Bellaterra, Spain

Abstract

The Belief-Desire-Intention (BDI) architecture is a practical approach for modelling large-scale intelligent systems. In the BDI setting, a complex system is represented as a network of interacting agents – or components – each one modelled based on its beliefs, desires and intentions. However, current BDI implementations are not well-suited for modelling more realistic intelligent systems which operate in environments pervaded by different types of uncertainty. Furthermore, existing approaches for dealing with uncertainty typically do not offer syntactical or tractable ways of reasoning about uncertainty. This complicates their integration with BDI implementations, which heavily rely on fast and reactive decisions. In this paper, we advance the state-of-the-art *w.r.t.* handling different types of uncertainty in BDI agents. The contributions of this paper are, *first*, a new way of modelling the beliefs of an agent as a set of epistemic states. Each epistemic state can use a distinct underlying uncertainty theory and revision strategy, and commensurability between epistemic states is achieved through a stratification approach. *Second*, we present a novel syntactic approach to revising beliefs given unreliable input. We prove that this syntactic approach agrees with the semantic definition, and we identify expressive fragments that are particularly useful for resource-bounded agents. *Third*, we introduce full operational semantics that extend CAN, a popular semantics for BDI, to establish how reasoning about uncertainty can be tightly integrated into the BDI framework. *Fourth*, we provide comprehensive experimental results to highlight the usefulness and feasibility of our approach, and explain how the generic epistemic state can be instantiated into various representations.

1. Introduction

In large-scale intelligent systems, dynamic knowledge and beliefs play a crucial role. Architectures such as the Belief-Desire-Intention architecture (BDI) (?) allow these notions to

be explicitly modelled by decomposing a complex intelligent system into a set of simpler autonomous and interacting agents. The *beliefs* model the agent's understanding of the environment, the *desires* are those states that the agent wishes to bring about, and the *intentions* are the desires the agent has chosen to act upon. Over the years, many agent-based programming languages based on the BDI architecture have been proposed in the literature, including PRS (?), AgentSpeak (?), and 2APL (?). These languages have been used with some success to model for example modern SCADA (Supervisory Control and Data Acquisition) systems (?, ?).

An important challenge is that current BDI implementations are not well-suited to model the next generation of SCADA systems as they cannot model (or reason about) uncertain information. In realistic settings, however, the beliefs of an agent tend to be uncertain (e.g. due to sensor noise, incomplete information *etc.*). This problem is further aggravated due to the typical computational complexity of uncertainty theories. While BDI implementations rely on reactive behaviour, most theories of uncertainty do not take tractability into account. Therefore, there is a mismatch between theory and practice.

An additional complication in dealing with uncertainty is that an uncertain input can be treated in different ways: either the input acts as a constraint that must be satisfied after belief revision, or the input is treated as a new belief with an associated strength (?). The latter is representative of a multi-agent environment, where new information from various sources does not necessarily cancel out existing beliefs, but may strengthen or weaken them.[1] Frameworks for dealing with uncertainty in this way have been proposed (?) but rely on semantic belief change operators, which restricts their usefulness for practical applications due to their computational cost. Syntactical approaches to belief revision have also been suggested in the literature, but most deal with classical inputs and are based on the AGM style of revision, see (?, ?). Syntactic operators that are able to deal with iterated belief revision are far less common. One popular semantic approach to define such epistemic states are Ordinal Conditional Functions (OCF) (?). A syntactic representation for OCF, along with the conditions that such a representation has to satisfy, was presented in (?). However, since OCF are not built as a general framework, they can be more difficult to instantiate into other theories of uncertainty such as possibility theory (?) or probability theory (?).

To address these shortcomings, we present in this paper an extension of the CAN framework in which the uncertain beliefs of an agent can be accurately modelled irrespective of the underlying cause of uncertainty. These uncertain beliefs can in turn be used and combined to derive new conclusions, and can be cast into a syntactic and tractable framework when certain restrictions are imposed. For example, an agent can have beliefs about his office hours, the fuel left in his car, and whether it is raining outside. These three pieces of uncertain information are to some degree irrelevant to each other, e.g. whether it rains outside does not have an effect on the level of fuel in the car. Also, these beliefs can be represented by different uncertainty theories, each revised using different strategies. Regardless, the information will need to be combined (e.g. taking the bus when it is raining

1. Treating uncertain inputs as beliefs with associated strengths has some connections with belief merging. However, since this topic is out of the scope of this paper, we do not discuss this to keep the paper more compact. Interested readers can read more on this topic from e.g. (?).

because fuel in the car is low) and revised (e.g. I need to leave early when it is raining to avoid being stuck in traffic).
The contributions of the paper are as follows:

First, we introduce a technique for modelling the beliefs of an agent as a set of epistemic states. Each epistemic state represents part of the beliefs held by an agent, and each state can use a distinct underlying uncertainty theory (e.g. possibilities or infinitesimal probabilities) with its own revision strategy. A set of epistemic states, referred to as the Global Uncertain Belief set or GUB for short, then represents all beliefs of the agent. The GUB provides commensurability among the different epistemic states, and ensures that its local epistemic states are correctly revised when new information is made available (even though only some of that information may apply to any specific local epistemic state).

Second, we introduce a syntactic operator for revising with an uncertain input. This way, we can avoid the exponential space requirements associated with the semantic operator. We prove that the syntactic operator corresponds to the semantic operator defined in (?), and we illustrate how to instantiate it to a given theory of uncertainty. In particular, we instantiate it to both probability theory and possibility theory.

Third, we introduce a tractable syntactic approach for revising with an uncertain input. We do this by identifying a fragment of the language that allows for tractable revision with uncertain inputs. It turns out that this fragment is sufficiently expressive and agrees with the restrictions often imposed on languages such as AgentSpeak(L) (?), and even extends beyond it.

Fourth, all the aforementioned ideas are integrated into the BDI setting through an extension of the operational semantics of Can.

Fifth, we provide an experimental evaluation of our framework to demonstrate its feasibility and to highlight its usefulness.

The paper is structured as follows. Some necessary preliminary notions about epistemic states are reviewed in Section 2. In Section 3 we introduce a novel framework for modelling and revising uncertain beliefs, and reasoning about such beliefs. While at first we focus on a single source of uncertainty in this section, we also show how the environment can be factored into different components, each potentially represented using its own theory of uncertainty. A full syntactic treatment of the belief revision is presented in Section 4, and a tractable subset of the language is identified in Section 5. In Section 6 we discuss how the tractable approach can be instantiated into possibility theory and probability theory. For possibility theory in particular, we also give an account of the full syntactical language. Some required preliminaries on the Can semantics are provided in Section 7, after which full operational semantics to integrate BDI with reasoning under uncertainty are given.

This paper combines and extends our earlier work (?, ?). Proofs for all propositions are produced in this paper, and many proofs have been rewritten to improve readability.

This extended version furthermore introduces the actual instantiations to probability and possibility theory, and provides their tractable syntactic counterparts. This paper also introduces the full syntactic version of the possibility theory instantiation. Finally, the paper offers an intricate scenario evaluation, which features our implementation of these ideas and supports the applicability of our proposed framework.

2. Background on Ma & Liu's Epistemic States

We start from a *finite set of atoms* $\mathcal{A}t$. We use *Lit* to denote the *set of literals* that can be constructed from $\mathcal{A}t$, i.e. $Lit = \{a \mid a \in \mathcal{A}t\} \cup \{\neg a \mid a \in \mathcal{A}t\}$. For a literal $l \in Lit$ we use l^* to denote the underlying atom, i.e. $l^* = a$ when $l = a$ or $l = \neg a$. The *language* $\mathcal{L}$ constructed over $\mathcal{A}t$ is defined in Backus-Naur Form (BNF) as $\varphi ::= a \mid \neg a \mid (\varphi_1 \wedge \varphi_2) \mid (\varphi_1 \vee \varphi_2)$, i.e. all formulas are in Negation Normal Form (NNF) which is used for its syntactic convenience (indeed, any arbitrary propositional formula can be efficiently converted into an equivalent proposition in NNF). For a formula $\varphi \in \mathcal{L}$, we use $lit(\varphi)$ to denote the set of literals in φ. A *possible world* ω, or *interpretation*, is a function that maps $\mathcal{A}t$ onto $\{0, 1\}$. The *set of all possible worlds* is denoted by Ω. Hence for e.g. $\mathcal{A}t = \{a, b\}$ we have $\Omega = \{\{a, b\}, \{a, \neg b\}\{\neg a, b\}, \{\neg a, \neg b\}\}$. We use the notation $\overline{\omega}$ to denote the conjunction of literals that the possible world ω makes true, e.g. $\overline{\omega} = a \wedge \neg b$ for $\omega = \{a, \neg b\}$. A possible world ω is a *model* of a propositional formula φ iff the possible world ω makes φ true, denoted as $\omega \models \varphi$. The *set of all models* of φ is denoted as $Mod(\varphi)$.

We are now ready to look at the definition of an epistemic state.

Definition 1. *(from (?)) Let* Ω *be the set of possible worlds. An* epistemic state Φ *is a mapping* $\Phi : \Omega \to \mathbb{Z} \cup \{-\infty, +\infty\}$.

Throughout the paper we denote epistemic states using capital Greek letters. An epistemic state Φ is used to represent the mental state of an agent, where the value $\Phi(\omega)$ represents the degree of belief in a possible world ω and is called the *weight* of ω. When $\Phi(\omega) = \infty$ (resp. $-\infty$) the agent believes ω to be fully plausible (resp. not at all plausible) while $\Phi(\omega) = 0$ indicates that the agent is totally ignorant about ω. For $\omega, \omega' \in \Omega$ and $\Phi(\omega) > \Phi(\omega')$ the intuition is that ω is more plausible than ω'.

It is important to clarify that the definition of an epistemic state given in Definition 1 allows for the construction of a general framework for dealing with uncertain beliefs. Indeed, this definition does not impose any restrictions on the values associated with the possible worlds. Other representations for epistemic states, which attach more specific meaning to the values, have been shown to be equivalent to the one from Definition 1. Specifically, Definition 1 induces an Ordinal Conditional Function (OCF) (?, ?)[2], which in turn can be transformed into other representations, e.g. those based on infinitesimal probabilities (?) and possibility theory (?). The representation from Definition 1 can thus be *instantiated* using any of the other representations to best suit the nature of the uncertainty and we will consider the specifics of some such instantiations in Section 6. Furthermore, an epistemic state as in Definition 1 is often easier to work with as it relies on integers (and not e.g. ordinal numbers), and does not need a normalisation step (e.g. as needed in OCF).

2. In addition, in (?) it has been shown that revising an epistemic state with an uncertain input is equivalent to combining the two corresponding OCFs using the combination operator suggested in (?).

New information, which we want to incorporate into our existing beliefs, consists of a proposition $\varphi \in \mathcal{L}$ and an associated weight $m \in (\mathbb{Z} \cup \{-\infty, +\infty\})$. This new information, or *input* (φ, m), is represented as an epistemic state Φ_{in} such that $\Phi_{in}(\omega) = m$ when $\omega \models \varphi$ and $\Phi_{in}(\omega) = 0$ otherwise. Such an epistemic state is also often referred to as a *simple epistemic state*, as it only encodes information from a single piece of input. Other than that, a simple epistemic state does not differ in any way from other epistemic states and the notation Φ_{in} is therefore only used as a visual aid. The operator introduced in (?) to revise an epistemic state Φ by Φ', denoted as $\Phi \circ \Phi'$, is defined as $\forall \omega \in \Omega, (\Phi \circ \Phi')(\omega) = \Phi(\omega) + \Phi'(\omega)$ with $+$ the addition operator [3]. Since an input corresponds to a simple epistemic state, we often simply write $\Phi \circ (\varphi, m)$. We will also use $\Phi \circ I$ with $I = \langle i_1, ..., i_n \rangle$ a sequence of inputs to denote $\Phi \circ i_1 \circ ... \circ i_n$. Notice that unlike AGM-style revision we have that the $\circ$ revision operator is both commutative and associative, which are desirable properties when dealing with revision based on uncertain inputs.

Example 1. *Let $\mathcal{A}t = \{a, b, c\}$. Consider the epistemic state Φ such that $\Phi(\{a, b, c\}) = \Phi(\{a, \neg b, c\}) = \Phi(\{a, b, \neg c\}) = \Phi(\{a, \neg b, \neg c\}) = 3$ and $\Phi(\omega) = 0$ for all other possible worlds ω. Intuitively, this models an agent that believes 'a' to be more plausible than '$\neg$a'. Indeed, exactly those possible worlds that model 'a' have a higher weight than the others. We say that the agent believes 'a' with a strength of 3 and is ignorant about the other literals in Lit. Now consider the input $(c, 2)$. This input corresponds to the simple epistemic state Φ' for which $\Phi'(\{a, b, c\}) = \Phi'(\{a, \neg b, c\}) = \Phi'(\{\neg a, b, c\}) = \Phi'(\{\neg a, \neg b, c\}) = 2$ while for all other worlds ω we have that $\Phi'(\omega) = 0$. The result of revising Φ given the input, denoted as $\Psi = \Phi \circ (c, 2)$, is given by Ψ such that:*

$$\begin{aligned} \Psi(\{a, b, c\}) &= 5 & \Psi(\{\neg a, b, c\}) &= 2 \\ \Psi(\{a, \neg b, c\}) &= 5 & \Psi(\{\neg a, \neg b, c\}) &= 2 \\ \Psi(\{a, b, \neg c\}) &= 3 & \Psi(\{\neg a, b, \neg c\}) &= 0 \\ \Psi(\{a, \neg b, \neg c\}) &= 3 & \Psi(\{\neg a, \neg b, \neg c\}) &= 0 \end{aligned}$$

In other words: the agent most strongly believes that both 'a' and 'c' are true in the real world, as expected, while still being ignorant as to whether 'b' is true or false.

The belief set, i.e. the sentences that an agent is committed to believe, is defined as the set that has all the most plausible worlds as its models. To define this set, we first recall the notion of a preorder. A *preorder* $\leq$ on a set A is a reflexive and transitive relation over $A \times A$. We say that $\leq$ is *total* iff for all $a, b \in A$ we have that either $a \leq b$ or $b \leq a$. Then:

Definition 2. *(from (?), Definition 4) Let Φ be an epistemic state. The belief set of Φ is $Bel(\Phi) = \{\varphi \in \mathcal{L} \mid \omega \models \varphi \text{ for all } \omega \in \min(\Omega, \leq)\}$. Here $\leq$ is a total preorder relation over Ω such that $\omega \leq \omega'$ iff $\Phi(\omega) \geq \Phi(\omega')$ and $\min(\Omega, \leq)$ denotes the set of minimal elements of Ω according to $\leq$.* [4]

3. Since $-\infty$ and $+\infty$ denote falsehood and truth, respectively, revising a world $\Phi(\omega) = \infty$ with $-\infty$ (and vice versa) is considered an inconsistency and not supported.
4. Preorder on models are also closely related to the notion of faithful assignment introduced by Katsuno & Mendelzon in (?) to characterise revision operators obeying AGM postulates.

While $Bel(\Phi)$ is defined above as a set of propositions, we can equivalently say that $Bel(\Phi)$ is the strongest (i.e. having the least number of models) proposition φ such that $Mod(\varphi) = \min(\Omega, \leq)$. This proposition φ is, of course, only unique up to logical equivalence.

Example 2. *Consider Ψ from Example 1. The models with the highest weight are given by $\min(\Omega, \leq) = \{\{a, b, c\}, \{a, \neg b, c\}\}$ and thus $\{a \wedge b \wedge c, a \wedge \neg b \wedge c\} \subseteq Bel(\Psi)$ or, equivalently, $Bel(\Psi) = a \wedge c$. We can easily verify that the agent believes that 'a' must be true, since $a \wedge c \models a$. Similarly, the agent does not believe that 'b' must also be true since $a \wedge c \not\models a \wedge b$. This is as expected, given that the agent is ignorant about the actual truth value of 'b'.*

3. Modelling and Reasoning about Uncertain Beliefs

In this section we expand on the classical idea of an epistemic state Φ by also taking those possible worlds into account that are not considered in the belief set $Bel(\Phi)$. The possible worlds not considered in $Bel(\Phi)$ constitute the uncertain information, i.e. they define the preferences the agent has over the outcomes that are currently not believed to be true. To reason about these beliefs, we introduce a new language $\mathcal{L}_{\geq}^{\mathcal{A}t}$ that expands upon $\mathcal{L}$. A well-formed formula $\varphi \in \mathcal{L}_{\geq}^{\mathcal{A}t}$ over $\mathcal{A}t$ is defined in BNF as:

$$\psi ::= a \mid \neg a \mid \psi_1 \wedge \psi_2 \mid \psi_1 \vee \psi_2$$
$$\varphi ::= a \mid \neg a \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \psi_1 \geq \psi_2 \mid \psi_1 > \psi_2 \mid \mathit{not}\ \psi$$

with $a \in \mathcal{A}t$. Whenever $\mathcal{A}t$ is clear from the context, we also simply write $\mathcal{L}_{\geq}$. The intuition of $(\psi_1 > \psi_2)$ is that ψ_1 is strictly more strongly believed (has a higher weight) than ψ_2, whereas *not* ψ reads as "ψ is assumed not to hold", i.e. negation-as-failure. Throughout the paper we also use $\neg\varphi$ with $\varphi \in \mathcal{L}_{\geq}$ as a shorthand for the NNF (Negation Normal Form) notation of φ. In particular, formulas of the form $\neg(\psi_1 > \psi_2)$ are rewritten as $\psi_2 \geq \psi_1$, $\neg(\psi_1 \geq \psi_2)$ is rewritten as $\psi_2 > \psi_1$, and $\neg\psi$ is rewritten as $\psi > \neg\psi$ (see Definition 3). Notice that the NNF of any formula φ is again an expression in $\mathcal{L}_{\geq}$.

The semantics of the language $\mathcal{L}_{\geq}$ are defined in terms of a mapping λ, which maps arbitrary formulas $\varphi \in \mathcal{L}_{\geq}$ onto $\mathbb{Z} \cup \{-\infty, +\infty\}$. Intuitively, the value $\lambda(\varphi)$ associated with the formula φ reflects how strongly the agent believes φ to be true. When the formula φ is a propositional statement, i.e. $\varphi \in \mathcal{L}$, this can be directly determined by finding the models of φ and assigning to $\lambda(\varphi)$ the highest weight, i.e. $\lambda(\varphi) = \max\{\Phi(\omega) \mid \omega \models \varphi\}$. In general, however, the formula is not a propositional statement (i.e. $\varphi \in \mathcal{L}_{\geq}$, but not $\varphi \in \mathcal{L}$) and we need to pare it down until we can determine its λ-value directly. We have:

Definition 3. *Let $\varphi \in \mathcal{L}_{\geq}$. Whenever $\varphi \in \mathcal{L}$ we define $\lambda(\varphi) = \max\{\Phi(\omega) \mid \omega \models \varphi\}$ with $\max(\emptyset) = -\infty$. Otherwise, we define $\lambda(\varphi) = \lambda(pare(\varphi))$ with pare defined as:*

$$pare(\varphi \oplus \psi) = check(\varphi) \oplus check(\psi) \qquad \oplus \in \{\vee, \wedge\}$$

$$pare(\varphi \geq \psi) = \begin{cases} \top & \text{if } \lambda(\neg\varphi) \leq \lambda(\neg\psi) \\ \bot & \text{otherwise} \end{cases}$$

$$pare(\varphi > \psi) = \begin{cases} \top & \text{if } \lambda(\neg\varphi) < \lambda(\neg\psi) \\ \bot & \text{otherwise} \end{cases}$$

$$pare(not\ \varphi) = \begin{cases} \top & \text{if } \varphi \in \mathcal{L} \text{ and } \lambda(\neg\varphi) \geq \lambda(\varphi) \\ \bot & \text{otherwise} \end{cases}$$

$$check(\varphi) = \begin{cases} \varphi & \text{if } \varphi \in \mathcal{L} \\ pare(\varphi) & \text{otherwise} \end{cases}$$

with $\top$ (resp. $\bot$) a tautology of $\mathcal{L}$ (resp. a contradiction of $\mathcal{L}$).

For the operators $\wedge$ and $\vee$ we thus verify whether the operands are expressions in the language $\mathcal{L}$. Otherwise, we need to further pare it down to a propositional formula. When the operator is $>$, we define it as an ordering with an expression such as $\varphi > \psi$ read as *"φ is more certain than ψ"* or, alternatively, *"we have more reason to believe $\neg\psi$ than $\neg\varphi$"* (and equivalently for $\geq$).[5] When the operator is *not*, we verify whether the classical negation of the formula is more strongly believed than the formula itself.

A close relation exists between λ and possibility measures (?) for propositional statements $\varphi, \psi \in \mathcal{L}$. From this relationship, some interesting properties can readily be derived: $\lambda(\varphi \vee \psi) = \max(\lambda(\varphi), \lambda(\psi))$, $\lambda(\varphi \wedge \psi) \geq \min(\lambda(\varphi), \lambda(\psi))$, and $\lambda(\bot) = -\infty$. If we use $\max_\Phi = \max\{\Phi(\omega) \mid \omega \in \Omega\}$ to denote the weight associated with the possible world(s) with the strongest belief, we also have that $\lambda(\top) = \max_\Phi$,[6] and $\max(\lambda(\varphi), \lambda(\neg\varphi)) = \max_\Phi$. As we will see in Section 6.2, this close relation between the λ-value and possibility measures will also allow us to instantiate an epistemic state as a possibility distribution in a rather straightforward way. Also notice that, due to the lack of normalisation, $\lambda(\top)$ can have different values in different epistemic states.

It should be noted at this stage that negation-as-failure and qualitative operators such as $\geq$ only make sense when each operand φ is a classical formula, i.e. $\varphi \in \mathcal{L}$. Formulas where this is not the case are not allowed in $\mathcal{L}_{\geq}$, and default to $\bot$ when evaluated semantically.

We now give some examples of the λ-value of well-formed formulas $\varphi \in \mathcal{L}_{\geq}$.

Example 3. *Consider λ of Ψ from Example 1 where $\max_\Psi = 5$. We have:*

$$\lambda(a \wedge c) = \max_\Psi \qquad \lambda(\neg a \wedge c) = 2 \qquad \lambda(a \wedge \neg a) = -\infty$$
$$\lambda\big((\neg a \wedge c) > (\neg a \wedge \neg c)\big) = \lambda(\top) = \max_\Psi \qquad \lambda(c \geq \neg c) = \max_\Psi \qquad \lambda(b > \neg b) = -\infty$$

For example, $\lambda(a \wedge c) = \max_\Psi$ since $\Psi(\{a, b, c\}) = \max_\Psi$ and $\{a, b, c\} \models a \wedge c$. An expression such as $(\neg a \wedge c) > (\neg a \wedge \neg c)$, which is also believed to be true, states that the agent, even if a would be false, still believes c over $\neg c$.

5. In terms of possibility theory: we want $N(\varphi) \geq N(\psi)$, i.e. $\Pi(\neg\varphi) \leq \Pi(\neg\psi)$.
6. Importantly, $\max_\Phi$ corresponds with a possibility of 1 as this is the λ-value at which the agent believes formulas to be true. Hence, $\top$ always evaluates to true as it always has $\lambda(\top) = \max_\Phi$.

Using the λ-mapping we next define when a formula φ is entailed:

Definition 4. *Let Φ be an epistemic state and φ a formula in $\mathcal{L}_{\geq}$. We say that φ is entailed by Φ, written as $\Phi \models \varphi$, if and only if $\lambda(\varphi) > \lambda(\neg\varphi)$.*

Note that for a proposition ψ simply requiring that $\lambda(\psi) = \max_\Phi$ is not enough to imply that $\Phi \models \psi$. Indeed, for $b \in \mathcal{A}t$ we could have that $\lambda(b) = \lambda(\neg b) = \max_\Phi$, which occurs when we are ignorant about the value of b. As such, we need to ensure that both expressions are mapped onto strictly distinct values. Only this notion of entailment (assuming $\psi \in \mathcal{L}$) corresponds exactly to those formulas that can be derived from the belief base $Bel(\Phi)$.

Proposition 1. *Let $\varphi \in \mathcal{L}$ be a propositional formula, Φ be an epistemic state with domain Ω and λ be the mapping over Ω as defined in Definition 3. We have $\Phi \models \varphi$ iff for all $\omega \in \Omega$ such that $\lambda(\overline{\omega}) = \max_\Phi$ we have $\omega \models \varphi$, i.e. $Bel(\Phi) \models \varphi$.*

Proof. $[(\Phi \models \varphi) \Rightarrow (Bel(\Phi) \models \varphi)]$ We know from Definition 4 that $\Phi \models \varphi$ implies that $\lambda(\varphi) > \lambda(\neg\varphi)$, or, due to Definition 3, that $\max\{\Phi(\omega) \mid \omega \models \varphi\} > \max\{\Phi(\omega) \mid \omega \models \neg\varphi\}$. It readily follows that for all ω such that $\lambda(\overline{\omega}) = \max_\Phi$ we must have that $\omega \models \varphi$. Indeed, assume that $\lambda(\overline{\omega'}) = \max_\Phi$ and $\omega' \not\models \varphi$, i.e. $\omega' \models \neg\varphi$. This implies that $\lambda(\varphi) \leq \lambda(\neg\varphi)$ and thus $\Phi \not\models \varphi$, a contradiction.
$[(\Phi \models \varphi) \Leftarrow (Bel(\Phi) \models \varphi)]$ $Bel(\Phi) \models \varphi$ is equivalent to stating that for all $\omega \in \Omega$ such that $\lambda(\overline{\omega}) = \max_\Phi$ we have that $\omega \models \varphi$. Assume that $\Phi \not\models \varphi$, i.e. $\lambda(\varphi) \leq \lambda(\neg\varphi)$. This implies that there must be an ω' such that $\lambda(\overline{\omega'}) = \max_\Phi$ for which $\omega' \models \neg\varphi$, which is a contradiction. □

Extending to multiple epistemic states

Usually an agent will have a number of epistemic states to represent its beliefs. This can be for complexity purposes, i.e. when certain subsets of beliefs do not influence each other they can be kept in separate epistemic states to reduce the (exponential) size of each epistemic state involved. It can also be for practical purposes, where the agent wants to use a different theory of uncertainty and/or different revision rules for every epistemic state. To accommodate multiple epistemic states, we introduce the concept of a global uncertain belief set, or GUB for short.

Definition 5. *A* GUB *denoted as $\mathcal{G}$ is a set $\{\Phi_1, ..., \Phi_n\}$ where each Φ_i is an epistemic state over the domain $A_i \subseteq \mathcal{A}t$ such that $\{A_1, ..., A_n\}$ is a partition of $\mathcal{A}t$.*

Each local (or *isolated*) epistemic state Φ_i thus models beliefs that are semantically related, e.g. the colour of a traffic light or the current battery level, and that are governed by the same form of uncertainty. A GUB groups a set of such local epistemic states, and is therefore a representation of the overall beliefs of an agent. However, a GUB is not itself an epistemic state. This has a number of benefits and consequences. First, it simplifies the exponential representation of an epistemic state by partitioning the beliefs. Second, it allows for a general framework where each local epistemic state can use a different representation, as we will see in more detail later on. Third, it does not include a revision strategy (as each local epistemic state can have a distinct revision strategy).

Despite these differences, we can use a GUB to determine if a context φ – the pre-condition for a plan in a BDI setting – is true according to the agent's collective beliefs. Intuitively, φ can be evaluated directly if it applies to a single local epistemic state Φ_i, i.e. we can verify whether $\Phi_i \models \varphi$. Otherwise, we need to break φ apart up to the point where we can evaluate it directly. Whether or not this is possible depends on the connective. An expression can trivially be split when the connective is either $\wedge$ or $\vee$ as these connectives allow both operands to be evaluated independently. However, for the connectives $\geq$ or $>$ an evaluation is only possible when both operands are from the same local epistemic state. Indeed, in general, stratifications of different formulas in different local epistemic states are incomparable due to the varying underlying structures. To make this explicit, we define the language to be used on the GUB level as $\mathcal{L}_\mathcal{G}$. We have that every $\varphi \in \mathcal{L}_{\geq}^{A_i}$ over A_i with $i \in [1, k]$ is also a well-formed formula in $\mathcal{L}_\mathcal{G}$. Furthermore, for $\varphi_1, \varphi_2 \in \mathcal{L}_\mathcal{G}$ we also have that $\varphi_1 \wedge \varphi_2$ and $\varphi_1 \vee \varphi_2$ are valid formulas in $\mathcal{L}_\mathcal{G}$.

We now formalise the evaluation of a formula $\varphi \in \mathcal{L}_\mathcal{G}$ in the context of a GUB. A formula φ is broken apart by simplifying it, which returns the evaluation of φ by evaluating the operands (or it returns $\bot$ if the connective is $\geq$ or $>$ and both operands are incomparable). We can then define $val_{GUB}(\varphi)$ as[7]:

$$val_{GUB}(\varphi) = \begin{cases} \top & \text{if } \varphi \in \mathcal{L}_{\geq}^{A_i}, \Phi_i \models \varphi \\ \bot & \text{if } \varphi \in \mathcal{L}_{\geq}^{A_i}, \Phi_i \not\models \varphi \\ simplify(\varphi) & \text{otherwise} \end{cases}$$

$$simplify(\varphi \otimes \psi) = val_{GUB}(\varphi) \otimes val_{GUB}(\psi) \qquad \otimes \in \{\wedge, \vee\}$$

Definition 6. *Let $\mathcal{G}$ be a GUB and φ be a formula in $\mathcal{L}_\mathcal{G}$. We say that φ is entailed by $\mathcal{G}$, written as $\mathcal{G} \models \varphi$, if and only if $val_{GUB}(\varphi) \equiv \top$.*

Finally, we look at the revision of a GUB $\mathcal{G}$ with an input, or uncertain belief, $b = (\varphi, m)$ and $\varphi \in \mathcal{L}$, denoted as $\mathcal{G} \circ b$.[8] While the notation suggests that $\mathcal{G}$ is treated as an epistemic state, which would be theoretically feasible by regarding it as the Cartesian product of its local epistemic states, such a transformation would be overly computationally expensive. Instead, we define the revision process of a GUB as a marginalisation of the propositional formula φ before revising the desired local epistemic state(s). Intuitively, the marginalisation involves splitting off those models of the input b relevant to a given local epistemic state. The resulting proposition is by definition a proposition in the correct language, thus allowing us to revise the local epistemic state Φ_i directly. Each Φ_i can then use the revision rules that are relevant for the chosen representation of the epistemic state.

Definition 7. *Let $\mathcal{G}$ be a GUB, $b = (\varphi, m)$ an input, and $A_{in} = \{l^* \mid l \in lit(\varphi)\}$ (i.e. the set of atoms used in φ). For every local epistemic state $\Phi_i \in \mathcal{G}$ we define $refine(b, \Phi_i)$ as:*

$$refine(b, \Phi_i) = \begin{cases} forget(b, \Phi_i) & \text{if } A_{in} \cap A_i \neq \emptyset \\ \langle\rangle & \text{otherwise} \end{cases}$$

7. Notice that the semantics will evaluate some formulas that are syntactically invalid. In all of these cases, however, the formulas will evaluate to $\bot$.
8. There may very well be a difference of magnitudes between the weights used in the different local epistemic states of a GUB. To accommodate for this, we simply need to adopt the revision strategy associated with each local epistemic state to correctly scale the input weight.

where $\mathit{forget}(b, \Phi_i) = \langle (\overline{\alpha}, m) \mid \omega \in \mathit{Mod}(\varphi),\ \alpha = \omega \cap \mathit{lit}(A_i) \rangle$ *is a sequence of inputs.*

Whenever the set of atoms A_i used in the epistemic state Φ_i and the set of atoms A_{in} used in the input b do not overlap, we have that $\mathit{refine}(b, \Phi_i) = \langle\rangle$. In other words: the input b does not affect Φ_i. However, when $A_{in} \cap A_i \neq \emptyset$, i.e. when some of the atoms in the input are found in Φ_i, then (b, m) is broken up into a *sequence of inputs* that apply to Φ_i. To clarify this, consider the following example:

Example 4. *Let* $\mathcal{A}t = \{c, d, e\}$, $\mathcal{G} = \{\Phi_1, \Phi_2\}$, *and* $A_1 = \{c\}$, $A_2 = \{d, e\}$. *Consider the input* $b = ((c \wedge d) \vee e, m)$, *which is a propositional formula over the set of atoms* $\mathcal{A}t$. *We have that* $\mathit{Mod}(\varphi) = \{\{c, d, e\}, \{c, d, \neg e\}, \{c, \neg d, e\}, \{\neg c, d, e\}, \{\neg c, \neg d, e\}\}$. *Then:*

$$\mathit{forget}(b, A_1) = \langle (c, m), (c, m), (c, m), (\neg c, m), (\neg c, m) \rangle$$
$$\mathit{forget}(b, A_2) = \langle (d \wedge e, m), (d \wedge \neg e, m), (\neg d \wedge e, m), (d \wedge e, m), (\neg d \wedge e, m) \rangle$$

Importantly, as can be seen in the example, *forget* derives the sequence of inputs relevant to a local epistemic state Φ_i based on the models of the propositional formula φ, which ensures that the principle of the *irrelevance of syntax* is fulfilled.

Once an input b is broken down into a sequence of inputs $\mathit{refine}(b, \Phi_i)$ for each epistemic state Φ_i, this sequence can be used to directly update the local epistemic state Φ_i. Indeed, each input $(\overline{\alpha'}, m)$ in the sequence $\mathit{refine}(b, \Phi_i)$ corresponds to a simple epistemic state from (?), i.e. to an epistemic state Φ_{in} with the domain 2^{A_i} such that $\Phi_{in}(\omega) = \mu$ iff $\omega \models \alpha'$ and $\Phi_{in}(\omega) = 0$ otherwise. Since any epistemic state Φ can be revised by a simple epistemic state Φ' with the same domain Ω, the revision becomes an iterated revision of every $\Phi_i \in \mathcal{G}$ using the corresponding simple epistemic states from $\mathit{refine}(b, \Phi_i)$.

Definition 8. *Let* $\mathcal{G}$ *be a GUB and* b *an input. We have* $\mathcal{G} \circ b = \{\Phi_i \circ \mathit{refine}(b, \Phi_i) \mid \Phi_i \in \mathcal{G}\}$ *with* $\circ$ *a revision operator associated with* Φ_i.

The final output of this iterated revision is unique regardless of the order in which we revise Φ_i with simple epistemic states Φ_{in} in $\mathit{forget}(b, \Phi_i)$ based on postulates B5 and B6 in (?) (i.e. weights are cumulative and the order of revising does not affect the result).[9] Thus, more accurately, $\mathit{forget}(\cdot, \cdot)$ and $\mathit{refine}(\cdot, \cdot)$ are multisets instead of sequences. A special case is furthermore when $A_{in} \subseteq A_i$, in which case the refine-and-forget strategy is equivalent to $\Phi_i \circ b$, as desired. A visual summary of revising a GUB is given in Figure 1.

4. Full Syntactic Approach to Model/Revise with Uncertain Inputs

So far we have focussed on the semantic representation of beliefs as epistemic states, and how we can reason about uncertain beliefs based on these epistemic states. Of course, for practical purposes, the exponential representation of an epistemic state is prohibitive (i.e. EXPSPACE). In this section we therefore develop a general syntactic approach to belief change with uncertain inputs, agreeing with the semantic notions, but better suited for practical applications (the solution we will propose is an NP-complete approach). Two competing sources of complexity will play an important role in the development of our syntactic approach. On the one hand, an agent will need to revise its beliefs when new information

9. Postulate B5 states $\Phi \circ (\varphi, m) \circ (\varphi, n) = \Phi \circ (\varphi, m+n)$, and B6 that $\Phi \circ (\varphi, m) \circ (\psi, n) = \Phi \circ (\psi, n) \circ (\varphi, m)$.

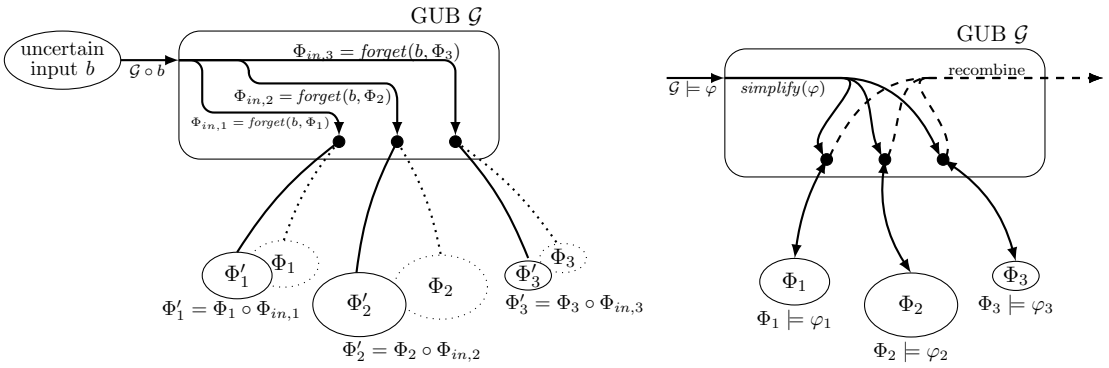

Figure 1: A visual representation of revising (left)/entailment (right) in a typical GUB.

is made available. On the other hand, an agent needs efficient ways of deciding belief entailment, i.e. whether $\Phi \models \varphi$ with $\varphi \in \mathcal{L}_{\geq}$.

The first step is to decide on a syntactic representation of the beliefs currently held by an agent. A common way of syntactically representing an epistemic state Φ is by means of a finite set of weighted formulas (ψ, m) with m the weight of formula ψ, see e.g. (?, ?). However, such a representation would make it complex to verify if a belief is entailed when considering uncertain input. Indeed, the weight of a possible world ω in the semantic representation might be dependent on the weight of some or all of the formulas in the syntactic representation. Instead, we propose a syntactic representation that is closer to the semantic one by requiring that no two weighted formulas are pairwise satisfiable:

Definition 9. *A* weighted belief base $\mathcal{B}$ *is a set of formulas of the form* (ψ, m) *with* $\psi \in \mathcal{L}$ *and* $m \in \mathbb{Z}$ *so that there does not exist* $(\psi_i, m_i), (\psi_j, m_j) \in \mathcal{B}$ *for which* $\psi_i \wedge \psi_j$ *is satisfiable.*

On a semantic level, this definition enforces that $Mod(\psi_i) \cap Mod(\psi_j) = \emptyset$, i.e. none of the formulas in $\mathcal{B}$ have models in common. While this representation might at first appear restrictive, it is important to note that on the semantic level the possible worlds adhere to this exact same restriction. Intuitively, in a weighted belief base $\mathcal{B}$ each formula $(\psi, m) \in \mathcal{B}$ corresponds to the set of possible worlds $Mod(\psi)$, all having the weight m.

Example 5. *Consider the epistemic states* Φ *and* Ψ *from Example 1. We can compactly represent* Φ *using the weighted belief base* $\{(a, 3)\}$*. Similarly, we can represent* Ψ *using the weighted belief base* $\{(a \wedge c, 5), (\neg a \wedge c, 2), (a \wedge \neg c, 3)\}$*.*

We now formalise the intuition from the previous example and define how a weighted belief base $\mathcal{B}$ can be used to retrieve the corresponding semantic representation:

Definition 10. *Let* $\mathcal{B}$ *be a weighted belief base. The epistemic state* $\Phi_{\mathcal{B}}$ *defined as*

$$\Phi_{\mathcal{B}}(\omega) = \begin{cases} m & \text{if there exists a } (\psi, m) \in \mathcal{B} \text{ such that } \omega \models \psi \\ 0 & \text{otherwise} \end{cases}$$

is the (semantic) epistemic state induced by $\mathcal{B}$*.*

The above definition formalises the intuition that every formula (φ, m) in $\mathcal{B}$ corresponds to a set of models $Mod(\varphi)$ such that for every $\omega \in Mod(\varphi)$ we have that $\Phi_{\mathcal{B}}(\omega) = m$. Hence, as desired, a weighted belief base is a compact representation of a semantic epistemic state. Furthermore, every epistemic state can be represented as a compact weighted belief base. Indeed, for $\omega \in \Omega$ we can define the equivalence class $[\omega] = \{\omega' \in \Omega \mid \Phi(\omega) = \Phi(\omega')\}$, i.e. all possible worlds with the same weight. For each equivalence class $[\omega]$ of Φ we then have that $(\varphi_\omega, \Phi(\omega)) \in \mathcal{B}$ with φ_ω a proposition such that $Mod(\varphi_\omega) = [\omega]$. So, φ_ω is a proposition that has as its models exactly those possible worlds that are in the equivalence class $[\omega]$, i.e. models with the same weight. It then readily follows from Definition 10 that $\Phi = \Phi_{\mathcal{B}}$.

Using a weighted belief base, it is straightforward to determine the weight associated with any arbitrary formula:

Proposition 2. *Let $\mathcal{B}$ be a weighted belief base and $\varphi \in \mathcal{L}$ a formula. We have that $\lambda(\varphi) = m_i \neq 0$ iff there exists a $(\psi_i, m_i) \in \mathcal{B}$ such that $\psi_i \wedge \varphi$ is satisfiable and there does not exist a $(\psi_j, m_j) \in \mathcal{B}$ with $m_j > m_i$ such that $\psi_j \wedge \varphi$ is satisfiable.*

Proof. This readily follows from the Definition 3 stating that $\lambda(\varphi) = \max_{\omega \models \varphi} \Phi_{\mathcal{B}}(\omega)$ and from Definition 10. Indeed, since every formula in $\mathcal{B}$ is a compact representation of a set of possible worlds ω and since none of the classical formulas in $\mathcal{B}$ share models, the definition of $\lambda(\varphi)$ reduces to finding the formula (ψ_i, m_i) in $\mathcal{B}$ with the highest weight m_i such that it shares models with φ, i.e. such that $\omega \models \varphi$ or $\varphi \wedge \psi_i$ is satisfiable. Otherwise, from Definition 10, we know that $\lambda(\varphi) = 0$. □

Proposition 3. *Let $\mathcal{B}$ be a weighted belief base, $\varphi \in \mathcal{L}$ a formula and $m \in \mathbb{Z}$ a weight. Determining whether $\lambda(\varphi) = m$ is* NP*-complete.*

Proof. It readily follows that this decision problem is in NP due to Proposition 2, i.e. verifying whether $\lambda(\varphi) = m$ requires $|\mathcal{B}|$ satisfiability checks, which in itself is an NP-complete problem. To prove NP-hardness, we reduce the satisfiability problem, i.e. verifying whether a given formula ψ is satisfiable, to the problem of verifiying whether $\lambda(\varphi) = m$. Without loss of generality, we can assume that ψ is in NNF. Let $\mathcal{B} = \{(\psi, 1)\}$ and $\varphi = a^\dagger \vee \neg a^\dagger$ with $a^\dagger$ a fresh atom, i.e. φ is a tautology. We have that ψ is satisfiable iff $\lambda(\varphi) = 1$. Indeed, we know from Proposition 2 that determining $\lambda(\varphi) = 1$ is equivalent to verifying that $\psi \wedge \varphi$ is satisfiable or, equivalently, that ψ is satisfiable since φ is a tautology. □

So far we have seen that a weighted belief base offers a practical representation of an epistemic state. While the representation appears restrictive at first, it has the same expressive power as an epistemic state. Furthermore, any epistemic state can be represented as a weighted belief base and each weighted belief base corresponds to exactly one epistemic state. Furthermore, a weighted belief base allows for a convenient way of determining the weight associated with any arbitrary formula, a problem shown to be NP-complete.

Clearly, a weighted belief base is a good representation for belief entailment. Indeed, information can efficiently be entailed from it (i.e. both entailment from a weighted belief base and a classical propositional belief base sit on the same level of the polynomial hierarchy), and a weighted belief base allows us to represent an epistemic state while avoiding its exponential space requirements.

We furthermore need to verify that a weighted belief base is a good representation for belief revision. In particular, we now need to verify whether there exists an actual syntactic revision operator that can transform a weighted belief base into another set of formulas given an arbitrary input. This new set of formulas then has to:

R1 agree with the definition of a weighted belief base (see Lemma 1); and

R2 correspond with belief change on a semantic level (see Proposition 4).

Before defining the syntactic revision operator, let $\mathcal{B}^*$ for a weighted belief base $\mathcal{B}$ denote the underlying classical set of formulas, i.e. $\mathcal{B}^* = \{\psi_i \mid (\psi_i, m_i) \in \mathcal{B}\}$ is the set of formulas stripped of their associated weights. This allows us to come to the following definition.

Definition 11. *Let $\mathcal{B}$ be a weighted belief base and (φ, μ) a non-trivial input, i.e. $\mu \neq 0$. The syntactic revision $\circ_s$ is given by $\mathcal{B} \circ_s (\varphi, \mu) = \mathcal{B}' \cup \left(\left(\varphi \wedge \neg \bigvee(\mathcal{B}^*_\varphi)\right), \mu\right)$ with $\mathcal{B}'$ given by:*

$$\mathcal{B}' = \{(\psi \wedge \varphi, m + \mu), (\psi \wedge \neg\varphi, m) \mid (\psi, m) \in \mathcal{B}_\varphi\} \ \cup \ \mathcal{B} \setminus \mathcal{B}_\varphi$$

for $\mathcal{B}_\varphi = \{(\psi, m) \mid (\psi, m) \in \mathcal{B}$ and $\psi \wedge \varphi$ is consistent$\}$. Whenever $\psi \wedge \neg\varphi$ is inconsistent we simply omit it from $\mathcal{B}'$ as it conveys no information.

Note that $(\psi \wedge \varphi)$ describes the models shared between ψ and φ, $(\psi \wedge \neg\varphi)$ are those models in ψ but not in φ and $(\varphi \wedge \neg \bigvee(\mathcal{B}^*_\varphi))$ are those models in φ that are not yet in $\mathcal{B}^*_\varphi$ (with $\mathcal{B}_\varphi$ as defined in Definition 11, and $\mathcal{B}^*$ as defined right before Definition 11). This makes the intuition of the syntactic operator clear: we change the weight of formulas in $\mathcal{B}$ with shared models, we leave the formulas that do not share models untouched, and we introduce a new formula with weight μ that encompasses those models that did not yet occur in $\mathcal{B}$.

Example 6. *Let $\mathcal{B}_0 = \{\}$ be an empty weighted belief base. We subsequently revise $\mathcal{B}_0$ with $(a, 3)$ and $(c, 2)$. We trivially have that $\mathcal{B}_1 = \mathcal{B}_0 \circ_s (a, 3) = \{(a, 3)\}$ since $\mathcal{B}_0$ is empty. We find that $\mathcal{B}_2 = \mathcal{B}_1 \circ_s (c, 2) = \mathcal{B}'_1 \cup (\varphi \wedge \neg \bigvee(\mathcal{B}^*_{1,\varphi})) = \{(a \wedge c, 5), (a \wedge \neg c, 3)\} \cup \{(c \wedge \neg a, 2)\}$ $= \{(a \wedge c, 5), (a \wedge \neg c, 3), (c \wedge \neg a, 2)\}$. Using Definition 10 we can verify that $Bel(\mathcal{B}_2) = a \wedge c$. Notice that the inputs coincide with Example 1, as does the resulting belief set. Furthermore, the results coincide with the intuition expressed in Example 5.*

It is easy to see that the syntactic operator only grows $\mathcal{B}$ in polynomial space. Indeed, after each revision with an input the number of new weighted formulas is bounded by $\mathcal{O}(2 \cdot |\mathcal{B}| + 1)$ with $|\mathcal{B}|$ the number of weighted formulas in $\mathcal{B}$. Indeed, we know from Definition 11 that the resulting weighted belief base $\mathcal{B} \circ_s (\varphi, \mu)$ will contain at most two pairs for every weighted belief (ψ, m) in the original belief base $\mathcal{B}$, i.e. $(\psi \wedge \varphi, m + \mu)$ and $(\psi \wedge \varphi, m)$. Furthermore, again from Definition 11, we know that every time we revise a weighted belief base with a given input we may add one additional pair to the resulting weighted belief base, i.e. $\mathcal{B} \circ_s (\varphi, \mu)$, i.e. $\left(\varphi \wedge \neg \bigvee(\mathcal{B}^*_\varphi)\right)$.

Interestingly, a weighted belief base never contains more formulas than possible worlds, i.e. $|\mathcal{B}| \leq |\Omega|$.[10] This follows by definition due to the pairwise inconsistency. Indeed, since the revision operator $\circ_s$ transforms weighted belief bases into weighted belief bases, even for small $\mathcal{A}t$ we never have more formulas in the syntactic representation than possible worlds.

10. Indeed, a weighted belief base essentially assign weights to members of a partition of Ω.

This is particularly interesting for relative small epistemic states, where the number of possible worlds is quite limited. Even in those cases, the syntactic representation is an efficient one.

We now verify **R1**, which we set out earlier: the syntactic revision needs to ensure that the set of classical formulas after revision remain pairwise inconsistent, i.e. the result is a weighted belief base.

Lemma 1. *Let $\mathcal{B}$ be a weighted belief base, (φ, m) an input, and $\mathcal{B}'' = \mathcal{B} \circ_s (\psi, m)$. We have that for all $(\psi_i'', m_i''), (\psi_j'', m_j'') \in \mathcal{B}''$ it holds that $\psi_i'' \wedge \psi_j''$ is not satisfiable, i.e. ψ_i'' and ψ_j'' are pairwise inconsistent.*

Proof. We start with some basic properties that will be used throughout this proof: for a satisfiable arbitrary propositional formula ρ we have that $Mod(\rho) \neq \emptyset$, $Mod(\rho) \cap Mod(\neg\rho) = \emptyset$ (i.e. a formula and its negation do not share models), and $Mod(\rho) \cup Mod(\neg\rho) = \Omega$ (i.e. models of a formula and its negation span all possible worlds).

We first show that modifying the existing weighted formulas in $\mathcal{B}$ does not violate their pairwise unsatisfiability. By definition, we know that all formulas $(\psi, m) \in \mathcal{B}_\varphi$, with $\mathcal{B}_\varphi$ as defined in Definition 11, are consistent with φ. Alternatively, we can say that $\psi \wedge \varphi$ is consistent, or, that $Mod(\psi) \cap Mod(\varphi) \neq \emptyset$. Since $Mod(\psi) \subseteq \Omega$ with $\Omega = (Mod(\varphi) \cup Mod(\neg\varphi))$ and since $Mod(\varphi) \cap Mod(\neg\varphi) = \emptyset$ we can rewrite $Mod(\psi)$ as $Mod(\psi \wedge \varphi) \cup Mod(\psi \wedge \neg\varphi)$ with $Mod(\psi \wedge \varphi) \cap Mod(\psi \wedge \neg\varphi) = \emptyset$. Hence, we know that the conjunction of $(\psi \wedge \varphi)$ and $(\psi \wedge \neg\varphi)$ is not satisfiable. Indeed, we just concluded that these formulas do not share models. In addition, for every weighted formula $(\psi', m) \in \mathcal{B} \setminus \mathcal{B}_\varphi$ the condition holds by assumption of $\mathcal{B}$ and since $Mod(\psi') \cap Mod(\varphi) = \emptyset$. We have thus shown that all formulas in $\mathcal{B}'$, with $\mathcal{B}'$ as defined in Definition 11, are indeed pairwise unsatisfiable.

It remains to be checked whether $((\varphi \wedge \neg \bigvee(\mathcal{B}_\varphi^*)), m)$ upholds the condition. Since $Mod(\rho) \cap Mod(\neg\rho) = \emptyset$ and $Mod(\rho) \cup Mod(\neg\rho) = \Omega$ with $\rho = \bigvee(\mathcal{B}_\varphi^*)$ we know that we can rewrite $Mod(\varphi)$ as $Mod(\varphi \wedge \bigvee(\mathcal{B}_\varphi^*)) \cup Mod(\varphi \wedge \neg \bigvee(\mathcal{B}_\varphi^*))$ where all models in $Mod(\varphi \wedge \bigvee(\mathcal{B}_\varphi^*))$ are considered in the case discussed above and, trivially, $Mod(\varphi \wedge \neg \bigvee(\mathcal{B}_\varphi^*)) \cap Mod(\psi) = \emptyset$ for all $(\psi, m) \in \mathcal{B}$ since no models are shared with the negation of a formula. □

The verification of **R2** requires us to show that the revision result obtained using the syntactic revision operator corresponds to the result obtained using the semantic revision operator. By relying on Lemma 1, which shows that none of the weighted formulas in a belief base $\mathcal{B}$ share any models, we can simplify this requirement. Indeed, it suffices to verify that the weight associated with each formula in the syntactic approach corresponds exactly with the weight associated with its models (i.e. possible worlds) in the semantic approach. In other words, we need to verify that $\Phi(\omega) = m$ with $\omega \in Mod(\psi)$ iff $(\psi, m) \in \mathcal{B}$ and otherwise $\Phi(\omega) = 0$. Similar as to the semantic case, we will use the notation $\mathcal{B} \circ_s I$ with $I = \langle i_1, ..., i_n \rangle$ to denote the revision with a sequence of inputs, i.e. $\mathcal{B} \circ_s i_1 \circ_s ... \circ_s i_n$.

Proposition 4. *Let I be a finite sequence of inputs. Let Φ_0 be the epistemic state such that $\forall\omega \in \Omega$ we have that $\Phi_0(\omega) = 0$. Let $\mathcal{B}_0 = \{\}$ and let $\Phi_n = \Phi_0 \circ I$ and $\mathcal{B}_n = \mathcal{B}_0 \circ_s I$. We have that $(\psi, m) \in \mathcal{B}_n$ iff $\Phi_n(\omega) = m$ for every $\omega \in Mod(\psi)$.*

Proof. We prove this by induction on the number of inputs in I. The base cases are trivial. When $I = \langle\rangle$ there is nothing to do and the proposition holds vacantly. When I contains

only a single element, i.e. $I = \langle(\psi_1, m_1)\rangle$, then we trivially obtain that $\mathcal{B}_1 = \{(\psi_1, m_1)\}$ while $\Phi_1(\omega) = m_1$ iff $\omega \models \psi_1$ and $\Phi_1(\omega) = 0$ otherwise. This is exactly the epistemic state induced by $\mathcal{B}_1$ according to Definition 10.

Now assume that for a sequence of inputs I of size n with $\mathcal{B}_n = \mathcal{B}_0 \circ_s I$ and $\Phi_n = \Phi_0 \circ I$ we have that $\Phi_n(\omega) = m$ with $\omega \in Mod(\psi)$ iff $(\psi, m) \in \mathcal{B}_n$. Equivalently, we thus have that $\forall\omega \in \Omega \cdot \Phi_n(\omega) = \Psi_n(\omega)$ with Ψ_n the epistemic state induced by $\mathcal{B}_n$ in Definition 10. We show that this equivalence is upheld after revising with the $(n+1)$-*th* input $in = (\varphi_{n+1}, \mu_{n+1})$ where $\Phi_{n+1} = \Phi_n \circ in$ and $\mathcal{B}_{n+1} = \mathcal{B}_n \circ_s in$.

By definition of a simple epistemic state and the semantic revision operator we know that $\Phi_{n+1}(\omega) = \Phi_n(\omega)$ for all ω such that $\omega \not\models \varphi_{n+1}$. Similarly, in $\mathcal{B}_{n+1}$ we have the formulas $\mathcal{B}_n \setminus (\mathcal{B}_n)_{\varphi_{n+1}}$ and the formulas $(\psi \wedge \neg\varphi_{n+1}, m)$ for those $(\psi, m) \in \mathcal{B}_n$ such that $\psi \wedge \varphi_{n+1}$ is consistent. In other words: the weight of formulas that do not have models in common with φ_{n+1} as well as the weight of the models of $\psi \wedge \neg\varphi_{n+1}$ (those models of ψ that are not models of φ_{n+1}) is unchanged.

For all ω such that $\omega \models \varphi_{n+1}$ we know that $\Phi_{n+1}(\omega) = \Phi_n(\omega) + \mu_{n+1}$. Similarly, in $\mathcal{B}_{n+1}$ we have the formulas $(\psi \wedge \varphi_{n+1}, m + \mu_{n+1})$ for those $(\psi, m) \in \mathcal{B}_n$ such that $\psi \wedge \varphi_{n+1}$ is consistent. However, there may be models in $Mod(\varphi_{n+1})$ that are not yet in $\mathcal{B}_n$. Indeed, these models are exactly the models of the formula $\varphi_{n+1} \wedge \neg \bigvee((\mathcal{B}_n)^*_\varphi)$ to which we assign the weight μ_{n+1}. Once again, it is easy to verify from Definition 10 that we thus find $\forall\omega \in \Omega \cdot \Phi_{n+1}(\omega) = \Psi_{n+1}(\omega)$ with Ψ_{n+1} the epistemic state induced by $\mathcal{B}_{n+1}$. □

We have thus shown that the syntactic revision operator introduced in Definition 11 behaves as expected. Indeed, given an input (ψ, m) it transforms an existing weighted belief base $\mathcal{B}$ into the weighted belief base $\mathcal{B}''$, such that $\Phi \circ (\psi, m) = \Phi_{\mathcal{B}''}$ with $\mathcal{B}'' = \mathcal{B} \circ_s (\psi, m)$. Given that the syntactic revision operator relies on satisfiability checks, NP complexity is trivially derived. As such, we have attained our goal of balancing the complexity of both belief entailment and belief revision. This shows that a weighted belief base is a good representation for belief revision. Indeed, a successively updated weighted belief base only grows polynomially in size, and never grows larger than the corresponding epistemic state.

Up until now we only considered formulas of the form $\varphi \in \mathcal{L}$. It is, however, straightforward to extend our approach to formulas in the language $\mathcal{L}_{\geq}$. Indeed, since we know how to determine the weight associated with a formula $\varphi \in \mathcal{L}$, we can readily apply $pare(\varphi \geq \psi)$ from Definition 3. Verifying whether $\varphi \geq \psi$ holds only requires $|\mathcal{B}| + 1$ satisfiability checks when $\mathcal{B}$ is sorted based on the weight of its formulas. Once we find the highest weight μ for φ, we only need to verify if $\psi \wedge \psi'$ is satisfiable for $(\psi', \mu') \in \mathcal{B}$ with $\mu' \leq \mu$. Reasoning about the relationship of the plausibility of two formulas $\varphi \in \mathcal{L}$ can thus be done as efficiently as determining the weight of a formula $\varphi \in \mathcal{L}$.

5. Tractable Syntactic Approach to Model/Revise with Uncertain Inputs

In the previous section we introduced the first syntactic operator capable of dealing with arbitrary unreliable inputs. From a BDI perspective, however, the NP-hardness of the syntactic operator might still turn out to be prohibitive. This is particularly relevant to resource-bounded agents or agents for which reactiveness is of paramount importance. For this reason, we now develop a tractable approach to belief change with unreliable

inputs. As we will see, this tractable approach is surprisingly powerful, and extends upon the language allowed in AgentSpeak(L) (?). Of course, tractability does come at the cost of additional restrictions over the full syntactic version. One very common restriction in the literature, which we will also adopt, is to only consider literals as inputs (e.g. (?)). For our purposes, this implies that we only keep track of the weights $\overset{+}{\mu}$ and $\overline{\mu}$ associated with each atom $a \in \mathcal{A}t$, with $\overset{+}{\mu}$ and $\overline{\mu}$ the weight of resp. a and $\neg a$.

Definition 12. *A* compact epistemic state $\mathcal{W}$ *is a mapping* $\mathcal{W} : \mathcal{A}t \to (\mathbb{Z} \cup \{-\infty, +\infty\})^2$ *such that* $\mathcal{W}(a) = (\overset{+}{\mu}, \overline{\mu})$*, i.e. the weights associated with resp. a and* $\neg a$*.*

The epistemic state $\Phi_{\mathcal{W}}$ associated with a compact epistemic state $\mathcal{W}$ is defined next. We denote the weight of a literal $l \in Lit$ given a compact epistemic state $\mathcal{W}$ as $w_{\mathcal{W}}(l) = \overset{+}{\mu}$ if $l = a$ and $w_{\mathcal{W}}(l) = \overline{\mu}$ if $l = \neg a$ with $\mathcal{W}(a) = (\overset{+}{\mu}, \overline{\mu})$.

Definition 13. *Let* $\mathcal{W}$ *be a compact epistemic state. We have that* $\Phi_{\mathcal{W}}$*, defined as*

$$\Phi_{\mathcal{W}}(\omega) = \sum_{\omega \models l} w_{\mathcal{W}}(l)$$

is the corresponding (semantic) epistemic state.

This definition is similar to Definition 10 from the previous section, where we now only use a set of literals (which can share models) as a compact representation of an epistemic state. Given the way we defined a compact epistemic state, and since we only allow (weighted) literals as input, a tractable belief change operator only has to change the weight of the literal given in the input.

Definition 14. *Let* $\mathcal{W}$ *be a compact epistemic state. Let* (l, μ) *be an input with* $l \in Lit$ *and* $\mu \in \mathbb{Z} \cup \{-\infty, +\infty\}$*. We define* $\mathcal{W}' = \mathcal{W} \circ_t (l, \mu)$ *as:*

$$\mathcal{W}'(a) = \begin{cases} (\overset{+}{\mu} + \mu, \overline{\mu}) & \text{if } l = a \\ (\overset{+}{\mu}, \overline{\mu} + \mu) & \text{if } l = \neg a \\ \mathcal{W}(a) & \text{otherwise} \end{cases}$$

with $\mathcal{W}(a) = (\overset{+}{\mu}, \overline{\mu})$*.*

Proposition 5. *Revising a compact epistemic state can be implemented using an algorithm with* $\mathcal{O}(\log_2 |\mathcal{A}t|)$ *complexity.*

Proof. A compact epistemic state can be implemented as a sorted map with each element being a pair (a, V). The atom 'a' is used as the key and V is a pair of values. Belief revision involves a binary search over the keys requiring at most $\log_2(|\mathcal{A}t|)$ steps and a constant time revision – which is a simple addition – of the respective value. □

Example 7. *Consider an agent who has received no prior input, i.e.* $\mathcal{W}(a) = (0, 0)$ *for every* $a \in \mathcal{A}t$*. Assume we change* $\mathcal{W}$ *with the inputs* $\langle(\neg c, 2), (a, 4), (b, -3), (a, 1), (c, 2), (\neg a, 4)\rangle$*. We obtain* $\mathcal{W}'$ *with:*

$$\mathcal{W}'(a) = (5, 4) \qquad \mathcal{W}'(b) = (-3, 0) \qquad \mathcal{W}'(c) = (2, 2).$$

Now assume that $\mathcal{W}'' = \mathcal{W}' \circ_t (\neg b, 4)$. *From Definition 14 we know that we only have to change the ordered pair associated with* b, *i.e. we have* $\mathcal{W}''(b) = (-3, 4)$, $\mathcal{W}''(a) = (5, 4)$, *and* $\mathcal{W}''(c) = (2, 2)$. *The epistemic state* $\Phi_{\mathcal{W}}$ *corresponding with* $\mathcal{W}''$ *according to Definition 13:*

$$\begin{aligned}
\Phi_{\mathcal{W}}(\{a, b, c\}) &= 4 & \Phi_{\mathcal{W}}(\{\neg a, b, c\}) &= 3 \\
\Phi_{\mathcal{W}}(\{a, b, \neg c\}) &= 4 & \Phi_{\mathcal{W}}(\{\neg a, b, \neg c\}) &= 3 \\
\Phi_{\mathcal{W}}(\{a, \neg b, c\}) &= 11 & \Phi_{\mathcal{W}}(\{\neg a, \neg b, c\}) &= 10 \\
\Phi_{\mathcal{W}}(\{a, \neg b, \neg c\}) &= 11 & \Phi_{\mathcal{W}}(\{\neg a, \neg b, \neg c\}) &= 10
\end{aligned}$$

We now prove the correctness of the operator $\circ_t$ introduced in Definition 14. As before, we use $\mathcal{B} \circ_t I$ with $I = \langle i_1, ..., i_n \rangle$ a sequence of inputs, each of the form (l, μ) with $l \in Lit$ and $\mu \in \mathbb{Z}$, to denote $\mathcal{B} \circ_t i_1 \circ_t ... \circ_t i_n$.

Proposition 6. *Let* I *be a finite sequence of inputs. Let* Φ_0 *be an epistemic state such that* $\forall \omega \in \Omega$ *we have that* $\Phi_0(\omega) = 0$. *Let* $\mathcal{W}_0$ *be a compact epistemic state with* $\mathcal{W}_0(a) = (0, 0)$ *for all* $a \in \mathcal{A}t$ *and let* $\Phi_n = \Phi_0 \circ I$ *and* $\mathcal{W}_n = \mathcal{W}_0 \circ_t I$. *We have that* $\Phi_n(\omega) = \Phi_{\mathcal{W}_n}(\omega)$ *for all* $\omega \in \Omega$ *with* $\Phi_{\mathcal{W}_n}$ *as defined in Definition 13.*

Proof. We prove this by induction on the number of inputs in I. When $I = \langle\rangle$ the proposition holds trivially. When $I = \langle (l_1, \mu_1) \rangle$ we have $\Phi_1(\omega) = \mu_1$ iff $\omega \models l_1$ and $\Phi_1(\omega) = 0$ otherwise. This corresponds exactly with $\Phi_{\mathcal{W}_1}$ since $w_{\mathcal{W}_1}(l_1) = \mu_1$ and $w_{\mathcal{W}_1}(l) = 0$ for all other $l \in Lit$. Now assume that $\Phi_{n-1}(\omega) = \Phi_{\mathcal{W}_{n-1}}(\omega)$ for I a sequence of $n-1$ inputs. We show that the equivalence is upheld after revising with the n-th input $in = (l_n, \mu_n)$.

Note that $\Phi_{n-1}(\omega) \neq \Phi_n(\omega)$ only if $\omega \models l_n$. In particular, we have that $\Phi_n(\omega) = \Phi_{n-1}(\omega) + \mu_n$. Similarly, $\Phi_{\mathcal{W}_{n-1}}(\omega) \neq \Phi_{\mathcal{W}_n}(\omega)$ only if $\omega \models l_n$. We have that $w_{\mathcal{W}_{n-1}} = w_{\mathcal{W}_n} + \mu_n$ from Definition 14 and thus, due to Definition 13, that $\Phi_{\mathcal{W}_{n-1}}(\omega) = \Phi_{\mathcal{W}_n}(\omega) + \mu_n$. Hence $\Phi_n(\omega) = \Phi_{\mathcal{W}_n}(\omega)$ due to the induction hypotheses and since both are modified with the weight μ_n only if $\omega \models l_n$. □

Clearly, restricting ourselves to literals as inputs makes it possible to use a simpler representation that allows for a very efficient belief change operator; yet it is proven to be equivalent to the revision of corresponding epistemic states. That is, this simpler form of revision does not lose the generality of belief modelling and revision of this new framework.

Of course, belief revision is only part of the problem, and we also require efficient techniques for belief entailment. Defining the belief set can easily be done. We have $Bel(\mathcal{W}) = \bigwedge \{l \in Lit \mid w_{\mathcal{W}}(l) > w_{\mathcal{W}}(\neg l)\}$ with $\mathcal{W}$ a compact epistemic state. It is then straightforward to verify, for an arbitrary compact epistemic state $\mathcal{W}$, that $Bel(\mathcal{W}) = Bel(\Phi_{\mathcal{W}})$ with $\Phi_{\mathcal{W}}$ the epistemic state induced by $\mathcal{W}$. Furthermore, since $Bel(\mathcal{W})$ is a conjunction of literals, evaluating whether a formula $\varphi \in \mathcal{L}$ is true according to the belief set of the agent simply consists of verifying whether φ is true after replacing all occurrences of literals from $Bel(\mathcal{W})$ in φ by $\top$ and all others by $\bot$.

While determining whether a formula is true or false given $Bel(\mathcal{W})$ is straightforward, the problem of reasoning about the uncertainty in a compact epistemic state – i.e. determining the λ-value of formulas – is more involved. To do so efficiently, we need to restrict the language of the formulas. In particular, we will look at a fragment of the language for which it is easy to determine those literals for which the weight is known, called the *bounded*

literals. We will clarify the intuition by reconsidering Example 7 where $\mathcal{A}t = \{a, b, c\}$. To determine the λ-value of the formula $\neg a \wedge c$, we are forced to use the weights associated with $\neg a$ and c, i.e. the set of bounded literals is $\{\neg a, c\}$, while 'b' is *unbounded*. It readily follows that the possible worlds satisfying the formula $\neg a \wedge c$ are $\{\neg a, b, c\}$ and $\{\neg a, \neg b, c\}$. The weight of the formula is then the highest weight associated with either of these possible worlds. From Definition 13 we know that $\Phi(\{\neg a, b, c\}) = 4 + (-3) + 2 = 3$ whereas $\Phi(\{\neg a, \neg b, c\}) = 4 + 4 + 2 = 10$. In other words: to determine the weight of the formula we have to use the weight associated with the bounded literals (i.e. $\neg a$ and c) while we can freely take the maximum weight associated with either the positive or negative atom 'b', since 'b' is unbounded. Thus, $\lambda(\neg a \wedge c) = 4 + \max\{-3, 4\} + 2 = 10$. As long as the fragment of the language we consider makes it easy and unambiguous to determine the set of bounded literals, determining the λ-value of a formula is a tractable problem.

We define the fragment $\mathcal{L}_t \subseteq \mathcal{L}$ in BNF as:[11]

$$\begin{aligned} \mathit{disj} &::= a \mid \neg a \mid \mathit{disj}_1 \vee \mathit{disj}_2 \\ \mathit{conj} &::= a \mid \neg a \mid \mathit{conj}_1 \wedge \mathit{conj}_2 \\ \varphi &::= a \mid \neg a \mid \mathit{disj} \wedge \mathit{conj} \mid \varphi_1 \vee \varphi_2 \end{aligned}$$

Intuitively, the language $\mathcal{L}_t$ ensures that whenever a conjunction occurs, one of the branches will be composed of only conjunctions while the other branch will only contain disjunctions. Notice that the fragment $\mathcal{L}_t$ is quite expressive as it is a superset of all DNF formulas.

We now define the weight associated with a formula in $\mathcal{L}_t$. To simplify this definition, we introduce the new notation $T_{\mathcal{W}} = \sum_{a \in \mathcal{A}t} \max \mathcal{W}(a)$, i.e. the total of all maximum weights associated with each atom (or: the maximum weight when no literals are bounded). Importantly, the value $T_{\mathcal{W}}$ can be computed as a byproduct of belief change. Indeed, for $\mathcal{W}' = \mathcal{W} \circ_t (l, \mu)$ we have that $T_{\mathcal{W}'} = T_{\mathcal{W}} - \max \mathcal{W}(l^*) + \max \mathcal{W}'(l^*)$.

Example 8. *Consider $\mathcal{W}'$ from Example 7, i.e. $\mathcal{W}'(a) = (5, 4)$, $\mathcal{W}'(b) = (-3, 0)$, and $\mathcal{W}'(c) = (2, 2)$. We have that $T_{\mathcal{W}'} = 7$. Furthermore, consider $\mathcal{W}'' = \mathcal{W}' \circ_t (\neg b, 4)$ from Example 7. We have that $\max \mathcal{W}'(b^*) = 0$, $\max \mathcal{W}''(b^*) = 4$ since $\mathcal{W}''(a) = (5, 4)$, $\mathcal{W}''(b) = (-3, 4)$, and $\mathcal{W}''(c) = (2, 2)$, and therefore that $T_{\mathcal{W}''} = 7 - 0 + 4 = 11$.*

The definition of the weight associated with a formula in $\mathcal{L}_t$ is then as follows:

Definition 15. *Let $\mathcal{W}$ be a compact epistemic state and $\varphi \in \mathcal{L}_t$. Let L be a set of literals. We recursively define $\lambda_t(\varphi, L)$ as:*

$$\begin{aligned} \lambda_t(\varphi_1 \vee \varphi_2, L) &= \max(\lambda_t(\varphi_1, L), \lambda_t(\varphi_2, L)) \\ \lambda_t(\mathit{disj} \wedge \mathit{conj}, L) &= \lambda_t(\mathit{disj}, L \cup \mathit{lit}(\mathit{conj})) \\ \lambda_t(l, L) &= \begin{cases} -\infty & \text{if } \mathit{inconsistent}(L \cup \{l\}) \\ \max_{T_{\mathcal{W}}}(l, L) & \text{otherwise} \end{cases} \end{aligned}$$

with $\mathit{inconsistent}(S)$ true whenever $\exists a \in \mathcal{A}t \cdot \{a, \neg a\} \subseteq S$, and we define $\max_{T_{\mathcal{W}}}(l, L) = T_{\mathcal{W}} - \sum_{l' \in L \cup \{l\}} \left| w_{\mathcal{W}}(l') - \max \mathcal{W}(l'^*) \right|$.

11. While the language does not explicitly allow formulas of the form $\mathit{conj} \wedge \mathit{disj}$ in φ, any such formula can trivially be written as $\mathit{disj} \wedge \mathit{conj}$ due to the commutativity of the conjunction.

The definition reflects the intuition we described earlier in the section. To evaluate a formula, we need to keep track of the bounded literals. A conjunction bounds variables (i.e. it expresses which literals must be true) while a disjunction takes the maximum of the values of both operands without altering the set of bounded literals. A formula that is (reduced to) just a literal l, is evaluated by considering the set $L \cup \{l\}$, i.e. all literals bounded so far including l. When $L \cup \{l\}$ is inconsistent the weight is $-\infty$. Otherwise, the weight is determined by starting from $T_{\mathcal{W}}$, i.e. the weight of an unbounded formula, and removing from it the maximum weight associated with its bounded literals. To obtain the λ-value of the formula we add back the correct, bounded, weight of each bounded literal.

Example 9. *Consider the formula $\varphi = (a \vee \neg c) \wedge (c \wedge b)$ and $\mathcal{W}''$ from Example 7. We have:*

$$\begin{aligned}
\lambda_t(\varphi, \emptyset) &= \lambda_t((a \vee \neg c) \wedge (c \wedge b), \emptyset) \\
&= \lambda_t(a \vee \neg c, \{b, c\}) \\
&= \max\{\lambda_t(a, \{b, c\}), \lambda_t(\neg c, \{b, c\})\} \\
&= \max\{\lambda_t(a, \{b, c\}), -\infty\} \\
&\quad \textit{(since } \{c, \neg c\} \subset (\{b, c\} \cup \{\neg c\})\textit{)} \\
&= \lambda_t(a, \{b, c\}) \\
&= T_{\mathcal{W}} - (|5 - 5| + |(-3) - 4| + |2 - 2|) \\
&= 11 - (0 + 7 + 0) = 4
\end{aligned}$$

Next, we prove the correspondence between $\lambda_t(\varphi, \emptyset)$ and $\lambda(\varphi)$:

Proposition 7. *Let $\mathcal{W}$ be a compact epistemic state and $\Phi_{\mathcal{W}}$ the epistemic state induced by $\mathcal{W}$. We have that $\lambda_t(\varphi, \emptyset) = \lambda(\varphi)$ with λ_t as in Definition 15.*

Proof. We first consider formulas without conjunction. Since the weight of a disjunction is the maximum of the weight of its constituents, we only need to verify that $\lambda_t(l, \emptyset) = \lambda(l)$. By definition, $T_{\mathcal{W}}$ is the highest weight associated with any (set of) possible world(s). Either l is entailed by a world with the highest weight, in which case $w_{\mathcal{W}}(l) \geq w_{\mathcal{W}}(\neg l)$, i.e. $T_{\mathcal{W}} - \max \mathcal{W}(l^*) + w_{\mathcal{W}}(l)$. Otherwise, $w_{\mathcal{W}}(l) < w_{\mathcal{W}}(\neg l)$ and the possible world ω such that $\omega \models l$ is the one that entails all other literals with highest associated weight. As such, the weight of ω is

$$\Big(\sum_{a \in \mathcal{A}t \setminus \{l^*\}} \max \mathcal{W}(a)\Big) + w_{\mathcal{W}}(l), \text{with} \sum_{a \in \mathcal{A}t \setminus \{l^*\}} \max \mathcal{W}(a) = T_{\mathcal{W}} - \max \mathcal{W}(l^*)$$

by definition of $T_{\mathcal{W}}$ and Definition 13.

We now consider the evaluation of a formula of the form $c \wedge d$ or, equivalently, $(\varphi_1 \wedge ... \wedge \varphi_n) \wedge (\psi_1 \vee ... \vee \psi_m)$. Using the distributive law, we can rewrite this as $(\varphi_1 \wedge ... \wedge \varphi_n \wedge \psi_1) \vee ... \vee (\varphi_1 \wedge ... \wedge \varphi_n \wedge \psi_m)$. We thus need to verify whether the weight of a formula of the form $\xi = (\varphi_1 \wedge ... \wedge \varphi_n \wedge \psi_i)$ is correctly determined. The possible world ω such that $\omega \models \xi$ with the highest associated weight is the one that entails all literals not found in ξ with the maximum associated weight. We have that the weight of ω is given by

$$\sum_{a \in \mathcal{A}t \setminus lit^*(\xi)} \max \mathcal{W}(a) + \sum_{l \in lit(\xi)} w_{\mathcal{W}}(l), \text{ with } \sum_{a \in \mathcal{A}t \setminus lit^*(\xi)} \max \mathcal{W}(a) = T_{\mathcal{W}} - \sum_{l \in lit(\xi)} \max \mathcal{W}(l^*)$$

where $lit^*(\xi) = \{l^* \mid l \in lit(\xi)\}$. Finally, we consider the situation where $lit(\xi)$ is inconsistent, i.e. there exists an $a \in \mathcal{A}t$ such that $\{a, \neg a\} \subseteq lit(\xi)$. We then have $\lambda(\xi) = -\infty$ and, correspondingly, Definition 15 returns $-\infty$. □

Proposition 8. *Computing the λ-value of a formula $\varphi \in \mathcal{L}_t$ using a compact epistemic state can be implemented using an algorithm with $\mathcal{O}(k \cdot \log_2 |\mathcal{A}t|)$ complexity where k is the number of literals in φ.*

Proof. An algorithm can straightforwardly be devised based on Definition 15 that traverses a given formula tree and collects the bounded literals in each conjunctive branch. Such a traversal is linear in the size of the formula. Once the set of bounded literals has been determined, the λ-value can be computed by retrieving the n distinct literals found in the bounded branch. Assuming that the compact epistemic state is encoded as sorted dictionary and a binary search algorithm is used, retrieving the value of each literal is $\mathcal{O}(\log_2 |\mathcal{A}t|)$. In the worst case, the value of all literals in the formula need to be determined. We thus need to retrieve at most k values where k is the number of literals in φ. □

Similar to the previous section, we can extend the language $\mathcal{L}_t$ to a language $\mathcal{L}_t^{\geq}$ for which it is easy to evaluate formulas of the form $\varphi \geq \psi$ or $\varphi > \psi$. We have seen that evaluating a formula $\varphi \in \mathcal{L}_t$ is tractable. Once we know the value $\lambda(\varphi)$ of a formula φ, we can readily apply $pare(\varphi \geq \psi)$ from Definition 3. As such, we define $\mathcal{L}_t^{\geq}$ in BNF as:

$$
\begin{aligned}
disj &::= a \mid \neg a \mid disj_1 \vee disj_2 \\
conj &::= a \mid \neg a \mid conj_1 \wedge conj_2 \\
\varphi &::= a \mid \neg a \mid disj \wedge conj \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \geq \varphi_2 \mid \varphi_1 > \varphi_2
\end{aligned}
$$

Example 10. *Consider $\mathcal{W}''$ from Example 7. We have that $c \geq a \wedge b$ since $\lambda(c) = 11$ and $\lambda(a \wedge b) = 4$. Similarly, we can verify that $(c > a) \vee (c > b)$ since $\lambda(a) = 11$ and $\lambda(b) = 4$, i.e. we have $\bot \vee \top$ or $\max\{\bot, \top\} = \top$.*

6. Instantiating Epistemic States

Until now, we have looked at a quite general framework for representing epistemic states. While this framework can be useful in some scenarios, it is more typical in real applications to use more specific representations such as possibility theory (?, ?) or probability theory (?, ?). In this section, we discuss in detail how the (compact representation of) epistemic states we have dealt with in the previous sections can be instantiated (in a broad sense), or perhaps better, *adapted*, into either of these frameworks. In particular, we will discuss instantiations of tractable syntactic versions of both possibility and probability theory. Furthermore, we will explain in detail how the syntactic approach from Section 4 can be applied to possibility theory to offer a full syntactical instantiation. In the probabilistic case the assumption of conditional independence, and the ensuing limitation that only literals are considered as inputs, collapses such a full syntactical instantiation into the tractable variant. Instantiations that allow for conditional dependence in the probabilistic case fall beyond the scope of this paper. For simplicity, we may define an instantiation with codomain C such that $C \neq \mathbb{Z} \cup \{-\infty, +\infty\}$. In this case, we assume the existence of a bijection

$f : C \to \mathbb{Z} \cup \{-\infty, +\infty\}$ by which we can obtain an instantiation satisfying Definition 1. For example, if an instantiation has the codomain $[0, 1]$, then f may be a simple linear mapping from $[0, 1]$ to $\mathbb{Z} \cup \{-\infty, +\infty\}$.

6.1 Instantiation with Probability Theory

Probability theory is a theory of uncertainty primarily used for dealing with stochastic information, often of a quantitative nature. In the context of a probabilistic modelling of beliefs, uncertainty is represented by (discrete) probability distributions P on the set of possible worlds, i.e. mappings of the form $P : \Omega \to [0, 1]$ such that $\sum_{\omega \in \Omega} P(\omega) = 1$. By convention, $P(\omega) = 1$ implies that ω is for certain the true state of the world, while $P(\omega) = 0$ implies that ω is certainly not the true state of the world. Ignorance is commonly modelled by the principle of insufficient reason with complete ignorance, i.e. complete ignorance is being represented as the uniform distribution $P(\omega) = 1/|\Omega|$ for all $\omega \in \Omega$.

A probability distribution P on Ω can be interpreted as an epistemic state similar to Definition 1, by replacing $\mathbb{Z} \cup \{-\infty, +\infty\}$ by the unit interval $[0, 1]$ and replacing the definition of the associated λ function in Definition 3 by stipulating $\lambda(\varphi) = \sum_{\omega \models \varphi} P(\omega)$, i.e. equating the λ values of a formula with its probability. Recall that the λ-value is used to offer semantics regardless of the underlying instantiation. Also, analogously to Definition 3, we see that a formula of the form $\varphi \geq \psi$ can be interpreted in the natural way as $P(\neg\psi) \geq P(\neg\varphi)$, or equivalently as $P(\varphi) \geq P(\psi)$. Similarly, we obtain for *not* φ that $P(\neg\varphi) > P(\varphi)$, i.e. it is more probable that φ is not true, or we have insufficient reasons to choose φ over $\neg\varphi$.

6.1.1 Tractable Instantiation

A tractable representation and revision strategy can be devised in analogy to Section 5. We start off with a similar restriction to only consider literals as input, as is common in tractable variants of belief revision. For the representation, we define what a compact epistemic state is, and its associated (full) epistemic state, as follows.

Definition 16. *A compact probabilistic epistemic state $\mathcal{W}^P$ is a mapping of the form $\mathcal{W}^P : \mathcal{A}t \to [0, 1]$. This extends to literals by defining $\mathcal{W}^P(\neg a) = 1 - \mathcal{W}^P(a)$ for every $a \in \mathcal{A}t$. The corresponding probabilistic epistemic state $\Phi_{\mathcal{W}^P}$ is then the probability distribution $\mathcal{W}^P : \Omega \to [0, 1]$ defined as $\Phi_{\mathcal{W}^P}(\omega) = \Pi_{\omega \models l} \mathcal{W}^P(l)$.*

The idea here is that a compact probabilistic epistemic state $\mathcal{W}^P$ assigns probability values to every atom a (and hence to their negations as $1 - \mathcal{W}^P(a)$), which extends to a probability distribution $\Phi_{\mathcal{W}^P}$ over the possible worlds assuming probabilistic independence of the atoms. The weight associated with each possible world is thusly given by the product of the weights associated with the literals made true by that possible world.

To revise a weighted probabilistic epistemic state we simply change it so that the probability of the input is enforced:

Definition 17. *Let $\mathcal{W}^P$ be a compact probabilistic epistemic state. Let (l, μ) be an input with $l \in Lit$. We define $\mathcal{W}^{P\prime} = \mathcal{W}^P \circ_t^P (l, \mu)$ as:*

$$\mathcal{W}^{P\prime}(a) = \begin{cases} \mu & \text{if } l = a \\ 1 - \mu & \text{if } l = \neg a \\ \mathcal{W}^P(a) & \text{otherwise} \end{cases}$$

Notably, we assume conditional independence throughout the revision process. This is important for tractability purposes, and puts the intuition close to that of simple Bayes. For entailment this is often sufficient, as we only care about whether an atom, or a formula, is preferred over its negation, i.e. we typically consider most probable explanation queries.

Example 11. *Let $\mathcal{W}^P(a) = \mathcal{W}^P(b) = \mathcal{W}^P(c) = 0.5$. Consider a sequence of inputs of the form $I = \langle (a, 0.6), (\neg b, 0.2), (c, 0.7) \rangle$. We have that $\mathcal{W}^{P\prime} = \mathcal{W}^P \circ_t^P I$ is given by $\mathcal{W}^{P\prime}(a) = 0.6, \mathcal{W}^{P\prime}(b) = 0.8, \mathcal{W}^{P\prime}(c) = 0.7$. If we subsequently revise $\mathcal{W}^P$ with $(\neg a, 0.6)$ to obtain $\mathcal{W}^{P\prime\prime}$ we get $\mathcal{W}^{P\prime\prime}(a) = 0.4, \mathcal{W}^{P\prime\prime}(b) = 0.8, \mathcal{W}^{P\prime\prime}(c) = 0.7$.*

As in Section 5, the belief set of a (compact) probabilistic epistemic state $\mathcal{W}^P$ can be defined in a similar way as:

$$Bel(\mathcal{W}^P) = \{\varphi \mid \lambda(\varphi) > \lambda(\neg\varphi)\} = \{\varphi \mid \lambda(\varphi) > 0.5\} \tag{1}$$

where $\lambda(\varphi) = \sum_{w \models \varphi} \Phi_{\mathcal{W}^P}(\omega)$. Hence, $\lambda(\varphi)$ is nothing but the probability of φ according to the probability distribution Φ_{W^P}. Do note that, contrary to Section 4 and 5, a belief set is not closed by (classical) deduction as it may well happen that $\lambda(\varphi) > 0.5$ and $\lambda(\psi) > 0.5$, but $\lambda(\varphi \wedge \psi) < 0.5$.

Whenever φ is a literal it is straightforward to check whether $\varphi \in Bel(\mathcal{W}^P)$, since it comes down to checking whether $\mathcal{W}^P(\varphi) > 0.5$. Determining the λ-values of other classical formulas can also still be straightforward, as long as we restrict the allowed language. In the case of a compact probabilistic epistemic state, we restrict ourselves to a language of DNF formulas. Such a restriction is in line with typical implementations of BDI agents, where the context of a rule – the component of which we need to determine its λ-value – is commonly a simple conjunction of literals that need to be true for the rule to be applicable (see Section 7). Namely, let φ be a DNF of the form $A_1 \vee ... \vee A_n$, where each A_i is a conjunction of literals $q_i^1 \wedge ... \wedge q_i^m$. First of all note that, by the very definition of $\Phi_{\mathcal{W}^P}$, the λ-value of a (non-contradictory) conjunction of literals is the product of the λ-values of the literals. The λ-value of the context can then be determined using the well-known inclusion-exclusion principle:

$$\lambda(A_1 \vee ... \vee A_n) = \sum_i \lambda(A_i) - \sum_{i,j} \lambda(A_i \wedge A_j) + \sum_{i,j,k} \lambda(A_i \wedge A_j \wedge A_k) \pm \dots$$

where the λ-value of any of these conjunctions $A_i \wedge A_j \wedge \dots$ is either 0 if it contains both an atom a and its negation $\neg a$, and otherwise is the product of λ-values of its literals.

Proposition 9. *Let $\mathcal{W}^P$ be a compact probabilistic epistemic state, and let $\Phi_{\mathcal{W}^P}$ be defined as $\Phi_{\mathcal{W}^P}(\omega) = \prod_{\omega \models l} w_{\mathcal{W}^P}(l)$. We have that $\lambda_t^P(\varphi) = \lambda(\varphi)$ for φ a classical formula.*

Proof. For formulas that only contain conjunction, the result trivially holds as $\lambda(\varphi \wedge \psi) = \lambda(\varphi)\lambda(\psi)$ or, equivalently, $P(\varphi \wedge \psi) = P(\varphi)P(\psi)$. This follows readily from the definition of $\Phi_{\mathcal{W}^P}$ as an epistemic state with an underlying probability distribution. Furthermore, the probability of formulas with multiple possible worlds is trivially the sum of the probabilities of the individual possible worlds. Since we defined it for a compact probabilistic epistemic state $\mathcal{W}^P$ as $\lambda(\varphi \wedge \psi) = \lambda(\varphi)\lambda(\psi)$ the correspondence clearly holds. When we also allow negation, the result still holds as $P(\neg\varphi)$ is identically defined in probability theory as $1 - P(\varphi)$. When also considering disjunctions such as $\varphi \vee \psi$, and since all formulas must be in DNF, we know that φ and ψ are either a literal or a conjunction of literals. The latter can be evaluated as we already discussed, and the former can directly be obtained through the compact probabilistic epistemic state. The λ-valuation of the disjunction $\lambda(\varphi \vee \psi)$ can then readily be obtained as $\lambda(\varphi) + \lambda(\psi) - \lambda(\varphi \wedge \psi)$, as per the definition. □

6.2 Instantiation with Possibility Theory

Possibility theory is a theory of uncertainty capable of dealing with incomplete information. It is defined in terms of a possibility distribution π which maps every possible world onto $[0, 1]$, i.e. it is a mapping $\pi : \Omega \to [0, 1]$. By convention, $\pi(\omega) = 0$ implies that the possible world ω is considered impossible, whereas $\pi(\omega) = 1$ means that none of the available information prevents ω from being the real world. Possibility degrees are mainly interpreted qualitatively: $\pi(\omega) > \pi(\omega')$ implies that ω is more possible than ω'. A possibility distribution is said to be normalised when $\exists\omega \in \Omega \cdot \pi(\omega) = 1$. Normalised possibility distributions are preferred, since a possibility distribution that is not normalised indicates the presence of conflicting information. A possibility distribution induces both a possibility and a necessity measure. A possibility measure is a mapping $\Pi : 2^\Omega \to [0, 1]$ defined as $\Pi(A) = \max\{\pi(\omega) \mid \omega \in A\}$. Equivalently, for a proposition φ such that $\mathit{Mod}(\varphi) = A$, we can write that $\Pi(\varphi) = \max\{\pi(\omega) \mid \omega \models \varphi\}$. A necessity measure is a mapping $N : 2^\Omega \to [0, 1]$ defined as $N(A) = 1 - \Pi(\Omega \setminus A)$. Equivalently, for a proposition φ we can write $N(\varphi) = 1 - \Pi(\neg\varphi)$. When the possibility distribution is normalised we have $\Pi(\Omega) = 1$ and $N(\emptyset) = 0$ (resp. $\Pi(\top) = 1$ and $N(\bot) = 0$). When a possibility distribution is subnormal, we have $\Pi(\Omega) < 1$ and $N(\emptyset) > 0$ (or $\Pi(\top) < 1$ and $N(\bot) > 0$). An important property of necessity measure is their min-decomposability w.r.t. conjunction: $N(\varphi \wedge \psi) = \min(N(\varphi), N(\psi))$. Dually, for possibility measures, we have that $\Pi(\varphi \vee \psi) = \max(\Pi(\varphi), \Pi(\psi))$. However, we only have that $N(\varphi \vee \psi) \geq \max(N(\varphi), N(\psi))$ and $\Pi(\varphi \wedge \psi) \leq \min(\Pi(\varphi), \Pi(\psi))$.

Similar to the way a probability distribution can be used to model an epistemic state, a possibility distribution can also be used to that effect. Given the definition of a λ-value in Definition 3, we readily have that $\lambda(\varphi) = \Pi(\varphi) = \max\{\pi(\omega) \mid \omega \models \varphi\}$ with $\max(\emptyset) = 0$. The intuition of $\lambda(\varphi)$ in a possibilistic setting is therefore to what degree φ is considered a possible outcome. As in the probabilistic setting we know that the λ-evaluations of classical formulas agree with the possibilistic semantics as they are determined directly on the level of the epistemic state. An expression of the form $\varphi \geq \psi$ is interpreted as $N(\varphi) \geq N(\psi)$ which corresponds with the intuition of $\geq$ as φ being at least as certain as ψ. Similarly, *not* φ is interpreted as $\Pi(\neg\varphi) \geq \Pi(\varphi)$, or, equivalently $N(\neg\varphi) \geq N(\varphi)$. This corresponds with the intuition of negation-as-failure: $\neg\varphi$ is at least as plausible as φ itself. On the

semantic level, revision of a possibility distribution π given an input (φ, m), denoted as $\pi' = \pi \circ^{\pi} (\varphi, m)$, is defined as follows for each $\omega \in \Omega$:

$$\pi'(\omega) = \begin{cases} \min(\pi(\omega), 1 - m) & \text{if } \omega \models \neg\varphi \\ \pi(\omega) & \text{otherwise} \end{cases}$$

Note how the probabilistic and possibilistic instantiations highlight the strengths of our framework. First, clearly, both model a different form of uncertainty. However, since both instantiations expose a λ-value, the semantics of the framework are still well-defined as it is agnostic to the underlying instantiation. Second, both instantiations consider different revision strategies. In the probabilistic setting, the revision strategy we implemented is that new information is adopted as-is. In the possibilistic setting, the revision strategy instead allows an inconsistent possibility distribution. As we will see in Section 8, this does not pose issues for our framework. **An agent is thus free to model its beliefs using different theories of uncertainty with distinct revision strategies.**

6.2.1 Full Syntactic Instantiation

On a syntactical level, a possibility distribution is commonly expressed as a necessity-valued knowledge base. However, unlike in the probabilistic setting, the syntactic approach devised in Section 4 can be adapted to the possibilistic setting with only minor modifications. Indeed, the only difference with Definition 9 is that we require $m \in [0, 1]$ to correspond with the codomain of a possibility distribution. Such a weighted (possiblistic) belief base is a syntactic representation of a possibility distribution, which can be retrieved by applying Definition 10. Furthermore, just as before, a weighted possibilistic belief base can be used to determine the possibility of any arbitrary formula.

Proposition 10. *Let $\mathcal{B}$ be a weighted possibilistic belief base such that $m \in [0, 1]$. Let $\varphi \in \mathcal{L}$ be a formula. We have that $\lambda(\varphi) = \Pi(\varphi) = m_i \neq 0$ iff there exists a $(\psi_i, m_i) \in \mathcal{B}$ such that $\psi_i \wedge \varphi$ is satisfiable and there does not exist a $(\psi_j, m_j) \in \mathcal{B}$ with $m_j > m_i$ such that $\psi_j \wedge \varphi$ is satisfiable. Otherwise, $\lambda(\varphi) = 1$.*

Proof. The proof is analogous to the proof of Proposition 2. The formulas in $\mathcal{B}$ do not share models, and a valid possibility distribution is recovered using Definition 10 since $m \in [0, 1]$. Finally, the neutral element is now 1. The result then readily follows. □

While a weighted possibilistic belief base clearly is a good representation for belief entailment, we also need to verify that it is a good representation for belief revision.

Care should be taken at this stage as input of the form (φ, μ) is interpreted in the more common necessity-valued way, i.e. as $N(\varphi) \geq \mu$, or, equivalently, as $\Pi(\neg\varphi) \leq 1 - \mu$. This is in line with possibilistic logic (?) and is reflected in the revision on the semantic level, but can cause some confusion at first. Specifically, a pair (φ, μ) is treated differently depending on where we find it; it is possibility-valued when it is a part of a weighted possibilistic belief base, whereas it is necessity-valued when it is given as an input.

Only minor changes are needed to Definition 11 to make it applicable to the possibilistic setting. Indeed, it suffices to take the minimum of the existing weight and the weight of the new input (both in possibilistic terms), rather than sum up the weights. Furthermore, as

discussed, we need to transform the necessity-valued input into a possibility-valued pair to revise the existing weighted possibilistic belief base. We thus have in the possibilistic setting that $\mathcal{B}' = \{(\psi \wedge \neg\varphi, \min(m, 1-\mu)), (\psi \wedge \varphi, m) \mid (\psi, m) \in \mathcal{B}_{\neg\varphi}\} \cup (\mathcal{B} \setminus \mathcal{B}_{\neg\varphi})$. As before, whenever $\psi \wedge \varphi$ is inconsistent we simply omit it from $\mathcal{B}'$ as it conveys no information.

Definition 18. *The syntactic revision of (the possibilistic) $\mathcal{B}$ with the input (φ, μ), denoted as $\mathcal{B} \circ_s^\pi (\varphi, \mu)$, is given by $\mathcal{B} \circ_s^\pi (\varphi, \mu) = \mathcal{B}' \cup \left(\left(\neg\varphi \wedge \neg \bigvee(\mathcal{B}^*_\varphi)\right), 1-\mu\right)$.*

Example 12. *Consider the inputs $N(a \wedge c) \geq 0.7$, $N(\neg a \vee \neg b) \geq 0.2$. The resulting possibility distribution is such that $\pi(\{a, \neg b, c\}) = 1$, $\pi(\{a, b, c\}) = 0.8$ and $\pi(\omega) = 0.3$ for all other possible worlds ω. When we furthermore impose that $N(b) \geq 0.4$ we have that $\pi(\{a, \neg b, c\}) = 0.6$. Starting from $\mathcal{B}_0 = \{\}$ we trivially obtain that $\mathcal{B}_1 = \mathcal{B}_0 \circ_s^\pi (a \wedge c, 0.7) = \{(\neg a \vee \neg c, 0.3)\}$. For $\mathcal{B}_2 = \mathcal{B}_1 \circ_s^\pi (\neg a \vee \neg b, 0.2)$ we find that $\mathcal{B}_2 = \{((\neg a \vee \neg b) \wedge (a \wedge b), 0.3), ((\neg a \vee \neg b) \wedge (\neg a \vee \neg b), 0.3), ((a \wedge b) \wedge (a \wedge c), 0.8)\}$. We can write this in a logically equivalent way as $\mathcal{B}_2 = \{(\neg a \vee \neg c, 0.3), (a \wedge b \wedge c, 0.8)\}$. In a similar way, we find that $\mathcal{B}_3 = \mathcal{B}_2 \circ_s^\pi (b, 0.4) = \{(\neg a \vee \neg c, 0.3), (a \wedge b \wedge c, 0.8), (a \wedge \neg b \wedge c, 0.6)\}$. It can easily be verified that $\Phi_{\mathcal{B}_3} = \pi$, i.e. the possibility distribution induced by $\mathcal{B}_3$ is identical to the possibility distribution π we computed on the semantic level at the start of the example.*

Clearly, the revision of a weighted possibilistic belief base produces a new weighted possibilistic belief base, i.e. it guarantees the mutual unsatisfiability of its formulas. The proof from Lemma 1 applies unchanged as the proof only relies on the modules of the formulas and not on the codomain used by any given weighted belief base.

Proposition 11. *Let I be a finite sequence of possibilistic inputs, i.e. formulas of the form (ψ, m) with $m \in [0, 1]$ and interpreted as $N(\psi) \geq m$. Let Φ_0 be the epistemic state such that $\forall\omega \in \Omega$ we have that $\Phi_0(\omega) = 1$, and let $\mathcal{B}_0 = \{\}$. Furthermore, we take $\Phi_n = \Phi_0 \circ I$ and $\mathcal{B}_n = \mathcal{B}_0 \circ_s^\pi I$. We then have that $(\psi, m) \in \mathcal{B}_n$ iff $\Phi_n(\omega) = m$ for every $\omega \in Mod(\psi)$.*

Proof. We prove this by induction on the number of inputs in I. Similar as in the proof of Proposition 4, the base cases for $\mathcal{B}_0$ and $\mathcal{B}_1$ are trivial. For the induction hypothesis, we assume that for a sequence of inputs I of size n with $\mathcal{B}_n = \mathcal{B}_0 \circ_s^\pi I$ and $\Phi_n = \Phi_0 \circ I$ we have that $\Phi_n(\omega) = m$ with $\omega \in Mod(\psi)$ iff $(\psi, m) \in \mathcal{B}_n$. Equivalently, we thus have that $\forall\omega \in \Omega \cdot \Phi_n(\omega) = \Psi_n(\omega)$ with Ψ_n the possibility distribution induced by $\mathcal{B}_n$ using Definition 10. We show that this equivalence is upheld after revising with the $(n+1)$-*th* input $in = (\varphi_{n+1}, \mu_{n+1})$ where $\Phi_{n+1} = \Phi_n \circ in$ and $\mathcal{B}_{n+1} = \mathcal{B}_n \circ_s^\pi in$.

The remainder of the proof is similar to the proof of Proposition 4. Indeed, by the interpretation of an input of the form $in = (\varphi_{n+1}, \mu_{n+1})$ as $N(\varphi_{n+1}) \geq \mu_{n+1}$ or, equivalently, $\Pi(\neg\varphi_{n+1}) \leq 1 - \mu_{n+1}$, we have that $\Phi_{n+1}(\omega) = \Phi_n(\omega)$ for all ω such that $\omega \models \varphi_{n+1}$ (i.e. $\omega \not\models \neg\varphi_{n+1}$). This is similar in how the syntactic revision does not change the weights of formulas that do not have models in common with $\neg\varphi_{n+1}$, nor does it change the weight of the models of $\psi \wedge \varphi_{n+1}$ (which are models of ψ that are not models of $\neg\varphi_{n+1}$).

Furthermore, for all ω such that $\omega \models \neg\varphi_{n+1}$ we know that $\Phi_{n+1}(\omega) = \min(\Phi_n(\omega), 1 - \mu_{n+1})$. Similarly, in $\mathcal{B}_{n+1}$ we have the formulas $(\psi \wedge \varphi_{n+1}, \min(m, 1-\mu_{n+1}))$ for those $(\psi, m) \in \mathcal{B}_n$ such that $\psi \wedge \neg\varphi_{n+1}$ is consistent. Finally, the models in $Mod(\neg\varphi_{n+1})$ but not in $\mathcal{B}_n$ are the models of the formula $\neg\varphi_{n+1} \wedge \neg \bigvee((\mathcal{B}_n)^*_\varphi)$ to which we assign the weight $\min(1, 1-\mu_{n+1}) = 1 - \mu_{n+1}$ since $0 \leq \mu_{n+1} \leq 1$. Once again, it is easy to verify from

Definition 10 that we thus find $\forall \omega \in \Omega \cdot \Phi_{n+1}(\omega) = \Psi_{n+1}(\omega)$ with Ψ_{n+1} the epistemic state induced by $\mathcal{B}_{n+1}$. □

Before concluding the full syntactic instantiation, we would like to note that the complexity results obtained in Section 4 for weighted belief bases trivially apply to weighted possibilistic belief bases as well. Indeed, only the codomain has changed in the possibilistic setting, and none of the proofs rely on the codomain for their correctness.

6.2.2 Tractable Instantiation

As in the probabilistic setting, a tractable revision strategy can be devised for possibility theory in which we use a variant of a compact epistemic state for representation. We have:

Definition 19. *A* compact (possibilistic) epistemic state $\mathcal{W}^\pi$ *is defined as a mapping* $\mathcal{W}^\pi : \mathcal{A}t \to [0,1] \times [0,1]$ *such that* $\mathcal{W}^\pi(a) = (\overset{+}{\mu}, \bar{\mu})$ *with* $\Pi(a) \leq \overset{+}{\mu}$ *and* $\Pi(\neg a) \leq \bar{\mu}$.[12]

Similar as in Section 5, we need to keep track of the possibility associated with both an atom and its negation. However, we no longer map these values onto $\mathbb{Z}$ but onto $[0,1]$ to match the codomain of a possibility distribution. The original possibility distribution associated with a compact possibilistic epistemic state can be obtained similarly as in Definition 13 where now $\Phi_{\mathcal{W}^P}(\omega) = \min_{\omega \models l} w_{\mathcal{W}^\pi}(l)$.

As in the probabilistic setting, the biggest change comes from the modified revision strategy. Surprisingly, in the possibilistic setting the revision strategy is close to the one we introduced in Section 5. Indeed, when dealing with a normalised underlying possibility distribution the only change that would be needed to Definition 14 is to replace the addition with a minimum, e.g. for $l = a$ we would associate with it the value $(\min(\overset{+}{\mu}, \mu), \bar{\mu})$. To also account for subnormal distributions we need to restrict the values in the tuples associated with any of the atoms to be restricted as soon as an inconsistency is detected. Luckily, detecting an inconsistency is straightforward as this implies that for a given atom a we have that $\mathcal{W}^\pi(a) = (m, n)$ with $m < 1$ and $n < 1$. We have:

Definition 20. *Let* $\mathcal{W}^\pi$ *be a compact possibilistic epistemic state. Let* (l, μ) *be an input with* $l \in Lit$*, denoting that* $N(l) \geq \mu$*. Let* $\alpha = \max(\overset{+}{\mu}, 1-\mu)$ *iff* $l = a$ *and* $\alpha = \max(1-\mu, \bar{\mu})$ *iff* $l = \neg a$*, with* $\mathcal{W}^\pi(a) = (\overset{+}{\mu}, \bar{\mu})$*. We then define* $\mathcal{W}^{\pi\prime} = \mathcal{W}^\pi \circ_t^\pi (l, \mu)$ *for every* $b \in \mathcal{A}t$ *as:*

$$\mathcal{W}^{\pi\prime}(b) = \begin{cases} (\overset{+}{\mu}, \min(\bar{\mu}, 1-\mu)) & \text{if } l = b \\ (\min(\overset{+}{\mu}, 1-\mu), \bar{\mu}) & \text{if } l = \neg b \\ (\min(\overset{+}{\mu}, \alpha), \min(\bar{\mu}, \alpha)) & \text{otherwise} \end{cases}$$

Notice here that α *denotes the inconsistency degree of the possibility distribution.*

Notice that since the input is always of the form (l, μ), we know that α is always well-defined as either $l = a$ or $l = \neg a$ with a an atom. When an inconsistency occurs, this is reflected in the possibility of every possible world. The last condition expressed for $\mathcal{W}^{\pi\prime}(a)$ ensures that the compact possibilistic epistemic state is changed accordingly, where the inconsistency degree affects all atoms.

12. Note that this definition allows for the representation of subnormal possibility distributions.

We now address the problem of belief entailment from a compact possibilistic epistemic state. On the face of it this seems to represent a considerable problem due to the decomposition rules in possibility theory. Indeed, in a compact possibilistic epistemic state we only store the possibility measure of an atom and its negation. General formulas therefore need to be decomposed into their constituents to determine whether or not they are entailed by the epistemic state. Since entailment is in turn defined as $\lambda(\varphi) > \lambda(\neg\varphi)$, i.e. it depends on both the original formula as well as its negation, it appears that we are not able to support either conjunction or disjunction in the language as in general $N(\varphi \vee \psi) \geq \max(N(\varphi), N(\psi))$. However, since we are only dealing with atomic inputs, it importantly holds that $N(\varphi \vee \psi) = \max(N(\varphi), N(\psi))$, which makes decomposition possible.

Proposition 12 (from (?))**.** *Let $\pi_x : X \to [0,1]$ and $\pi_y : Y \to [0,1]$ be two possibility distributions on two different universes X, Y. Let $\pi_{x,y} : X \times Y \to [0,1]$ be defined as $\pi_{x,y}(u,v) = \min(\pi_x(u), \pi_y(v))$ (equation 1.76), and let $N_{x,y}$ be the induced necessity measure on $X \times Y$. For every $A \subseteq X$ and $B \subseteq Y$, it then holds that $N_{x,y}(A^* \cup B^*) = \max(N_x(A), N_y(B))$, where A^* and B^* denote their corresponding cylindrical extensions on $X \times Y$ (i.e. $A^* = A \times Y$, $B^* = X \times B$) (equation 1.78).*

In the setting of this paper, the set of models of two different literals are of the same form and can be seen as mapping their respective literal to $\alpha \in [0,1]$, while being free in all other literals. Without loss of generality, they can be seen as both having the domain Ω. Furthermore, $\pi_{x,y}$ corresponds to the minimally specific possibility distribution satisfying the constraints imposed by the respective literals. So, the max-decomposability for necessity measures, and, equivalently, the min-decomposability for possibility measures always applies to revisions with literals. Indeed, in general it applies to the case of having disjunctions of literals that do not share any variables.

Determining the λ-values of arbitrary classical formulas is then straightforward.[13] Indeed, we have that $\lambda(\varphi \vee \psi) = \min(\lambda(\varphi), \lambda(\psi))$, $\lambda(\varphi \wedge \psi) = \max(\lambda(\varphi), \lambda(\psi))$, and $\lambda(\neg\varphi)$ can be directly evaluated from the compact possibilistic epistemic state if we assume – without loss of generality – that an arbitrary classical formula is in NNF. We use $\lambda_t^\pi(\varphi)$ to refer to the evaluation of the classical formula φ in this way. As in the probabilistic case, we can again extend the language to include formulas of the form $\varphi > \psi$ and $\varphi \geq \psi$, since the evaluation of such formulas relies on paring them down using Definition 3 to their classical components.

Proposition 13. *Let $\mathcal{W}^\pi$ be a compact possibilistic epistemic state and $\Phi_{\mathcal{W}}^\pi$ the epistemic state induced by $\mathcal{W}^\pi$. We have that $\lambda_t^\pi(\varphi) = \lambda(\varphi)$.*

Proof. Due to the definition of $\mathcal{W}^\pi$, which maintains possibilities only on an atomic level, we know from Definition 12 that $\Phi_{\mathcal{W}}^\pi$ is a possibility distribution such that $N(\varphi \vee \psi) = \max(\varphi, \psi)$ and, trivially, $N(\varphi \wedge \psi) = \min(\varphi, \psi)$. Equivalently, we have that $\Pi(\varphi \vee \psi) = \min(\varphi, \psi)$ and $\Pi(\varphi \vee \psi) = \max(\varphi, \psi)$. Since $\lambda(\varphi) = \Phi(\varphi)$ and since for $\mathcal{W}^\pi$ we have that $\lambda_t^\pi(\varphi)$ is defined as $\lambda_t^\pi(\varphi \vee \psi) = \min(\lambda_t^\pi(\varphi), \lambda_t^\pi(\psi))$, $\lambda_t^\pi(\varphi \wedge \psi) = \max(\lambda_t^\pi(\varphi), \lambda_t^\pi(\psi))$ the equivalence readily holds. Furthermore, since formulas are assumed to be in NNF, it readily follows that $\lambda(\varphi) = \lambda_t^\pi(\varphi)$ with $\varphi = a$ or $\varphi = \neg a$ since $\lambda(\varphi) = \Pi(\varphi)$ as before and $\lambda_t^\pi(\varphi) = w_{\Phi^\pi}(\varphi) = \Pi(\varphi)$ by Definition 19. □

13. Recall that λ-values match up with the notion of a possibility measure.

7. Operational Semantics of Uncertain Beliefs in a BDI context

So far we have seen that an agent can use an epistemic state to represent its uncertain beliefs. Different theories can be used to model the beliefs, and different revision strategies can be employed to revise those beliefs. Furthermore, different epistemic states can be grouped into a single GUB, either for complexity reasons or for the convenience of representing the beliefs using different theories of uncertainty. We have also seen that syntactic approaches exist that can efficiently perform both belief revision and belief entailment, and we even identified a tractable fragment of the input/language. However, we have not yet discussed how these components can be integrated into the BDI architecture. To that end, we present in this section full operation semantics that extend CAN, which highlight how these ideas can be tightly integrated into a BDI agent.

7.1 CAN background

The CAN language formalises the behaviour of a classical BDI agent, which is defined by a belief base $\mathcal{B}$ and a plan library Π. The only requirement imposed on a belief base of an agent is that it is a set of formulas over some logical language that supports entailment (i.e. $\mathcal{B} \models b$, b a belief), belief addition and belief deletion (resp. $\mathcal{B} \cup \{b\}$ and $\mathcal{B} \setminus \{b\}$). The plan library is a set of plans of the form $e : \psi \leftarrow P$ where e is an event, ψ is a context and P is a plan body. Events can either be external (i.e. from the environment in which the agent is operating) or internal (i.e. sub-goals that the agent itself tries to accomplish). Given a plan of the form $e : \psi \leftarrow P$, the plan body P is *applicable* to handle the event e when $\mathcal{B} \models \psi$, i.e. the context evaluates to true. The event and context differ in that the context is lazily evaluated; it is checked right before the execution of the plan body. The language of the plan body P is defined in Backus-Naur Form (BNF) as:

$$\begin{aligned} P ::= \mathit{nil} \mid +b \mid -b \mid \mathit{act} \mid ?\varphi \mid !e \mid P_1; P_2 \mid P_1 \parallel P_2 \mid \\ P_1 \triangleright P_2 \mid (|\Delta|) \mid \mathit{Goal}(\varphi_s, P, \varphi_f) \end{aligned}$$

with nil an empty or completed program, $+b$ and $-b$ belief addition and deletion, act a primitive action, $?\varphi$ a test to determine if φ is true in the belief base, and $!e$ a subgoal, i.e. an (internal) event. Actions, tests and subgoals can fail, e.g. when the preconditions are not met. Composition is possible through $P_1; P_2$ for sequencing, $P_1 \parallel P_2$ for parallelism (i.e. a non-deterministic ordering) and $P_1 \triangleright P_2$ to execute P_2 only on failure of P_1. $(|\Delta|)$ is used to denote a set of guarded plans, with Δ of the form $\psi_1 : P_1, ..., \psi_n : P_n$, which intuitively states that the plan body P_i is *guarded* by the context ψ_i, i.e. the context needs to be true to execute the plan body. This is convenient when multiple plans are applicable to handle an event e. While executing the first plan body (which may fail), the beliefs may change and some plan bodies may no longer be applicable, i.e. we need to retest ψ_i before executing the next plan body. The plan form $\mathit{Goal}(\varphi_s, P, \varphi_f)$ is a distinguishing feature of CAN that allows to model both declarative and procedural goals. It states that we should achieve the (declarative) goal φ_s using the (procedural) plan P, where the goal fails if φ_f becomes true during the execution.

With the syntax defined, we now look at the operational semantics of CAN which are defined in terms of configurations. A basic configuration is a tuple $\langle \mathcal{B}, \mathcal{A}, P \rangle$ with $\mathcal{B}$ a belief base, $\mathcal{A}$ the sequence of primitive actions that have been executed so far and P the remainder

of the plan body to be executed (i.e. the current intention). An agent (configuration) is a tuple $\langle \mathcal{N}, \mathcal{D}, \Pi, \mathcal{B}, \mathcal{A}, \Gamma \rangle$ with $\mathcal{N}$ the name of the agent, $\mathcal{D}$ the action description library, Π the plan library, Γ the set of current intentions of the agent and $\mathcal{B}$ and $\mathcal{A}$ as before. For each action *act* the action description library contains a rule of the form $act : \psi \leftarrow \varphi^-; \varphi^+$. We have that ψ is the precondition, while φ^- and φ^+ denote respectively a delete and add set of belief atoms, i.e. propositional atoms.

A *transition relation* $\longrightarrow$ on (both types of) configurations is defined by a set of derivation rules. A transition $C \longrightarrow C'$ denotes a single step execution from C yielding C'. We write $C \longrightarrow$ to state there exists a C' such that $C \longrightarrow C'$ and $C \not\longrightarrow$ otherwise. We use $\xrightarrow{*}$ to denote the transitive closure over $\longrightarrow$. A *derivation rule* consists of a (possibly empty) set of premises p_i and a single transition conclusion c. Such a derivation rule is denoted as

$$\frac{p_1 \qquad p_2 \qquad \cdots \qquad p_n}{c}\, l$$

with l a label attached to the derivation rule for easy reference. Transitions over basic configurations (resp. agent configurations) define what it means to execute a single intention (resp. the agent as a whole). For example, the transition for belief addition is:

$$\frac{}{\langle \mathcal{B}, \mathcal{A}, +b \rangle \longrightarrow \langle \mathcal{B} \cup \{b\}, \mathcal{A}, nil \rangle}\, +b$$

This states, intuitively, that when the next action in the plan body is belief addition ($+b$), we transition to a new configuration in which we add b to the belief base of the agent ($\mathcal{B} \cup \{b\}$) and the step is successfully completed ($+b$ is replaced by *nil*). A more complex derivation rule is the one for a primitive action:

$$\frac{(a : \psi \leftarrow \varphi^-; \varphi^+) \in \mathcal{D} \qquad a\theta = act \qquad \mathcal{B} \models \psi\theta}{\langle \mathcal{B}, \mathcal{A}, act \rangle \longrightarrow \langle (\mathcal{B} \setminus \varphi^-\theta) \cup \varphi^+\theta, \mathcal{A} \cdot act, nil \rangle}\, act$$

This rule states that when the unified precondition $\psi\theta$, with θ the unifier, is true in the belief base $\mathcal{B}$, the effect of the action is the application of the add and delete atom lists to the belief base. Furthermore, we also add this action to the list of actions we have executed so far and we replace *act* by *nil* to indicate the successful completion. We refer the reader to (?) for a full overview of the semantics of CAN.

7.2 Dealing with Uncertain Beliefs in a BDI Agent

Since a GUB acts as a set of formulas over the language $\mathcal{L}_{\geq}$, and it supports belief revision, it can conveniently take on the role of a belief base in CAN. This will be the foundation of our extension of CAN, which we will call CAN+. While mostly a straight-forward replacement, some modifications are needed to fully support the richer language $\mathcal{L}_{\geq}$ which allows for expressions that can reason about uncertainty. A first (conceptual) modification is that the context φ of a plan $e : \psi \leftarrow P$ is taken to be a sentence from the language $\mathcal{L}_{\geq}$, as is the formula φ of a test $?\varphi$. Next, we formally redefine the concept of configurations in CAN+. Rather than considering a belief base to model the knowledge, we instead use a GUB $\mathcal{G}$ to represent the uncertain beliefs of the agent:

Definition 21. *A* basic configuration *is a tuple* $\langle \mathcal{G}, \mathcal{A}, P \rangle$ *with* $\mathcal{G}$ *a GUB,* $\mathcal{A}$ *the list of executed actions and* P *a plan body being executed (i.e. the current intention). An* agent (configuration) *is a tuple* $\langle \mathcal{N}, \mathcal{D}, \Pi, \mathcal{G}, \mathcal{A}, \Gamma \rangle$ *with* $\mathcal{N}$ *the name of the agent,* $\mathcal{D}$ *the action description library which defines the primitive actions,* Π *the plan library,* Γ *the set of current intentions of the agent and* $\mathcal{G}$ *and* $\mathcal{A}$ *as before.*

With the configurations redefined we can extend the first set of rules from CAN, i.e. the rule for a test goal ($?\varphi$) and the rule for plan selection (*select*):

$$\frac{\mathcal{G} \models \varphi\theta}{\langle \mathcal{G}, \mathcal{A}, ?\varphi \rangle \longrightarrow \langle \mathcal{G}, \mathcal{A}, nil \rangle} \, ?\varphi$$

$$\frac{\psi_i : P_i \in \Delta \qquad \mathcal{G} \models \psi_i\theta}{\langle \mathcal{G}, \mathcal{A}, (|\Delta|) \rangle \longrightarrow \langle \mathcal{G}, \mathcal{A}, P_i\theta \triangleright (|\Delta \setminus \psi_i : P_i|) \rangle} \, select$$

We retain the notation as used in (?) to denote unification as e.g. $\varphi\theta$, i.e. variables are dealt with in the customary way. The modified rules make clear that verifying whether a belief or context holds is now achieved using the GUB. The language has been implicitly extended in both cases, since test goals and contexts can now include statements to reason over uncertain beliefs, i.e. $\varphi, \psi_i \in \mathcal{L}_{\geq}$.

So far we have looked at how to reason about the agent's (uncertain) beliefs, but we also want to revise these beliefs. Recall that a GUB $\mathcal{G}$ can be revised directly given an input $b = (\varphi, m)$ and denoted as $\mathcal{G} \circ (\varphi, m)$. Based on this notation, we introduce the $\circ b$ rule to CAN+ for belief change. The intuition of this new rule is clear; we want to change the beliefs encoded in the GUB with the uncertain belief b. We have:

$$\frac{}{\langle \mathcal{G}, \mathcal{A}, \circ b \rangle \longrightarrow \langle \mathcal{G} \circ b, \mathcal{A}, nil \rangle} \, \circ b$$

The rule for belief change can serve as a template to define the rules for classical belief addition $+\varphi$ and deletion $-\varphi$. Those rules would become:

$$\frac{}{\langle \mathcal{G}, \mathcal{A}, +\varphi \rangle \longrightarrow \langle \mathcal{G} \circ (\varphi, \max_{\mathcal{G}}), \mathcal{A}, nil \rangle} \, +\varphi$$

$$\frac{}{\langle \mathcal{G}, \mathcal{A}, -\varphi \rangle \longrightarrow \langle \mathcal{G} \circ (\varphi, \min_{\mathcal{G}}), \mathcal{A}, nil \rangle} \, -\varphi$$

with $\max_{\mathcal{G}} = \max \{\max_{\Phi_i} \mid \Phi_i \in \mathcal{G}\}$ and $\min_{\mathcal{G}}$ analogously defined. Notice that we transform the formula φ into an uncertain belief by assigning to it the weight $\max_{\mathcal{G}}$ ($\min_{\mathcal{G}}$). This ensures that φ will be true (false) after revision. We can also define belief addition and deletion as syntactic sugar on top of the belief change semantics. Indeed, a statement such as $+\varphi$ is nothing more than a shorthand for the statement $\circ(\varphi, \max_{\mathcal{G}})$. Similarly, $-\varphi$ can be considered a shorthand for $\circ(\varphi, \min_{\mathcal{G}})$. As we try to keep the semantics as concise as possible, we opt to define these operators in the latter way.

In conclusion, the new language for a plan body in CAN+ is given in BNF as:

$$P ::= nil \mid \circ b \mid act \mid ?\varphi \mid !e \mid P_1; P_2 \mid P_1 \parallel P_2 \mid$$
$$P_1 \triangleright P_2 \mid (|\Delta|) \mid Goal(\varphi_s, P, \varphi_f)$$

with b an uncertain belief and $\varphi, \varphi_s, \varphi_f \in \mathcal{L}_{\geq}$. We have modified the rules for $?\varphi$ and *select*, while dropping the rules for $+\varphi$ and $-\varphi$ and introducing a new rule for $\circ b$. The rules in CAN dealing with program flow do not require any changes and can be integrally applied to the CAN+ semantics. The rules on declarative goals do need to be modified, but in a straightforward way similar to $?\varphi$, i.e. we need to verify φ_s and φ_f against $\mathcal{G}$.

8. Qualitative Evaluation

In this section, we demonstrate the practical feasibility of our framework through the use of an implementation and illustrative scenario.

8.1 Implementation

In order to evaluate our framework, we have implemented the GUB model with three types of epistemic states: a probabilistic instantiation, a possibilistic instantiation and a ranking function instantiation. This was combined with a variant of the AgentSpeak interpreter (on which CAN is based) to form a full BDI implementation called TEAgentSpeak (Tractable Epistemic AgentSpeak).[14] This implementation supports traditional AgentSpeak programs with some minor changes to syntax, yet can model and revise uncertain beliefs and can reason about these uncertain beliefs during plan selection. In practice, our interpreter is very similar to the original definition of AgentSpeak, as well as the underlying AgentSpeak interpreter used in Jason (?). We can summarise our implemented AgentSpeak-style reasoning cycle as follows:

Planning stage:

Select event: At most one event is selected and removed from the event set E by an event selection function.

Select relevant plans: If an event is selected, then the interpreter iterates through the plan library and attempts to unify the triggering event of each plan with the selected event. The subset of plans in the plan library for which a unifier is found becomes the set of relevant plans for this event.

Select applicable plans: The interpreter iterates through the set of (partially instantiated) relevant plans and attempts to unify the context of each plan with the GUB. The subset of relevant plans for which a unifier is found becomes the set of applicable plans for this event.

Adopt intention: A single plan is selected from the set of (partially instantiated) applicable plans by a plan selection function. This plan is pushed to the intention contained in the selected event.[15] The intention is adopted by adding it to the intention set I.

Acting stage:

Select intention: At most one intention is selected and removed from the intention set I by an intention selection function.

14. `https://github.com/kevinmcareavey/teagentspeak`
15. Events generated externally to the agent will have an empty intention.

Execute step: If an intention is selected, then the next step in the body of the plan at the top of the intention is executed.

Conclude step: If the executed step involves a new subgoal then a new event containing the previously selected intention is added to the event set. Conversely, if the executed step involves belief revision then a new event containing an empty intention is added to the event set while the previously selected intention is added back to the intention set.

In procedural terms, we have not made any changes to the underlying AgentSpeak reasoning cycle. However, regarding belief revision and evaluation, there are a few notable differences:

(i) The belief base is a GUB rather than a set of belief atoms (as in AgentSpeak) or a set of belief literals (as in Jason).

(ii) Belief revision is supported while belief addition and belief deletion are not.

(iii) Plan contexts support any formula in the language $\mathcal{L}_t^{\geq}$ rather than being limited to a conjunction of literals (as in AgentSpeak) or to restricted classical formulae (as in recent versions of Jason). This allows plans to be selected based on the uncertainty associated with the beliefs of the agent.

Wherever possible, we have adhered to the design decisions made by the Jason developers. For example, we support different event, plan and intention selection functions but the default behaviour operates on a first-in first-out basis. The only syntactic differences between TEAgentSpeak and Jason are as follows:

(i) We allow plan contexts to be any formula in $\mathcal{L}_t^{\geq}$ by augmenting standard Jason syntax with φ `>` ψ and φ `>=` ψ to denote the formulas $\varphi > \psi$ and $\varphi \geq \psi$, respectively.

(ii) If $\mathcal{G}$ is the agent's GUB, then we use `*(`φ`,`μ`)` to denote the belief revision operation $\mathcal{G} \circ (\varphi, \mu)$ and the corresponding belief revision triggering event.

Due to these similarities, we will not elaborate on specific AgentSpeak implementation details and instead refer the reader to the comprehensive documentation on the Jason implementation of AgentSpeak (?)

The only remaining TEAgentSpeak-specific details to mention is how we model and revise a GUB and how we evaluate plan contexts. Firstly, by Definition 5, a GUB is just implemented as a set of epistemic states over disjoint sets of ground belief atoms. Secondly, the ranking, probabilistic and possibilistic epistemic state instantiations are all represented as described in Definitions 12, 16 and 19 and revised using Definitions 14, 17 and 20, respectively. Thirdly, plan contexts are implemented as binary expression trees which we evaluate using Definitions 3, 4 and 6 along with the λ values outlined in Propositions 7, 9 and 13, respectively. We omit a quantitative evaluation of these operations in this paper due to the tractable computational complexity proofs from Propositions 5 and 8. However, in the next section we will demonstrate the benefit of applying our framework to AgentSpeak by simulating a scenario involving uncertain information.

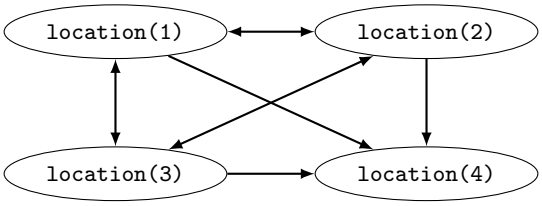

Figure 2: Possible sequences of actions in treasure hunt scenario.

8.2 Scenario

As part of a treasure hunt, a robot is required to find items of treasure and deliver them to a designated location. In the area, there are three locations in which treasure may be found: `location(1)`, `location(2)` and `location(3)`. The designated location for delivering treasure is `location(4)` and, once there, the robot will be retrieved, i.e. it is not possible for the robot to leave this location. A graphical illustration of all possible sequences of actions is shown in Figure 2. The robot itself is powered by a battery and if the remaining battery-life depletes before reaching `location(4)` then the robot will be irretrievable. Also, it is not possible to retrieve the robot from any other location and, for this reason, the robot will always make an attempt to reach `location(4)`. Thus, the robot must be careful to ensure that it can reach `location(4)` while still attempting to collect as much treasure as possible. In addition, the robot has some uncertain beliefs which it may consider in order to decide upon the best course of action. Firstly, a sensor in each location relays information to the robot about the location it is in. These sensors are *independent* of each other, i.e. they only report what they sense, without taking into account the information from the other sensors. Secondly, the robot has been informed that treasure may be found at `location(2)` and `location(3)`, but that `location(2)` is the more promising of the two. Thirdly, based on sensor information, the robot strongly believes that it is currently at `location(1)`. Fourthly, the robot has roughly estimated that it has sufficient battery to travel from `location(1)` to `location(2)`, or from `location(1)` to `location(3)`, before proceeding to `location(4)`. However, since the terrain is unfamiliar, the robot does not have enough information to speculate about other options.

$\mathcal{W}_i$	Instantiation	$b \in \mathcal{A}t_i$	$\mathcal{W}_i(b)$
$\mathcal{W}_1$	Possibility	`hasItems(location(1))`	(0.1, 1)
		`hasItems(location(2))`	(1, 0.2)
		`hasItems(location(3))`	(1, 0.3)
$\mathcal{W}_2$	Probability	`at(location(1))`	(0.9, 0.1)
		`at(location(2))`	(0.06, 0.94)
		`at(location(3))`	(0.03, 0.97)
		`at(location(4))`	(0.01, 0.99)
$\mathcal{W}_3$		`sufficientBattery(location(1),location(2),location(4))`	(1, 0)
		`sufficientBattery(location(1),location(3),location(4))`	(1, 0)
		`sufficientBattery(...,...,location(4))`	(0, 0)

Table 1: GUB definition and weights after initial belief revision.

```
1  // Initial beliefs.
2
3  *(∼hasItems(location(1)),0.9).
4  *(hasItems(location(2)),0.8).
5  *(hasItems(location(3)),0.7).
6  *(at(location(1)),0.9).
7  *(at(location(2)),0.06).
8  *(at(location(3)),0.03).
9  *(at(location(4)),0.01).
10 *(sufficientBattery(location(1),location(2),location(4)),1).
11 *(sufficientBattery(location(1),location(3),location(4)),1).
12
13 // Initial goals.
14
15 !completeMission.
16
17 // Plan library.
18
19 +!completeMission : at(location(X)) >= at(location(Y)) & at(location(X)) >= at(location(Z
       )) & at(location(X)) >= at(location(4)) & (hasItems(location(Y)) | hasItems(location(
       Z))) & hasItems(location(Y)) >= hasItems(location(Z)) & sufficientBattery(location(X)
       ,location(Y),location(4)) & X \== 4 & X \== Y & X \== Z & Y \== Z <- !proceed(
       location(X),location(Y)).
20
21 +!completeMission : at(location(X)) >= at(location(Y)) & at(location(X)) >= at(location(Z
       )) & at(location(X)) >= at(location(4)) & (hasItems(location(Y)) | hasItems(location(
       Z))) & hasItems(location(Y)) >= hasItems(location(Z)) & not sufficientBattery(
       location(X),location(Y),location(4)) & X \== 4 & X \== Y & X \== Z & Y \== Z <-
       broadcast(try(location(Y))); !proceed(location(X),location(4)).
22
23 +!completeMission : at(location(X)) >= at(location(Y)) & at(location(X)) >= at(location(Z
       )) & at(location(X)) >= at(location(4)) & ∼hasItems(location(Y)) & ∼hasItems(
       location(Z)) & X \== 4 & X \== Y & X \== Z & Y \== Z <- !proceed(location(X),location
       (4)).
24
25 +!completeMission : at(location(4)) >= at(location(X)) & at(location(4)) >= at(location(Y
       )) & at(location(4)) >= at(location(Z)) & X \== 4 & Y \== 4 & Z \== 4 & X \== Y & X
       \== Z & Y \== Z <- depositItems.
26
27 +!proceed(location(X),location(Y)) : true <- collectItems; *(∼hasItems(location(X)),1);
       *(hasItems(location(X)),0); !travel(location(X),location(Y)); !completeMission.
28
29 +!travel(location(X),location(Y)) : true <- *(at(location(X)),0.01); *(at(location(Y))
       ,0.99).
```

Listing 1: TEAgentSpeak program for robot in treasure hunt scenario.

To solve this problem, the robot's cognitive capabilities (which we refer to as the *agent*) have been implemented as an TEAgentSpeak program as shown in Listing 1. In this program, the agent's belief base is a GUB containing three epistemic states $\mathcal{W}_1, \ldots, \mathcal{W}_3$. The domain and underlying uncertainty representation of each epistemic state is described in Table 1. For example, the epistemic state $\mathcal{W}_1$ models the agent's beliefs about the possible location of treasure and, since this information is vague, is instantiated using possibility theory. Similarly, the epistemic state $\mathcal{W}_2$ models the agent's beliefs about its current location and, since this is based on sensor information, is instantiated using probability theory. Finally, the epistemic state $\mathcal{W}_3$ models the agent's beliefs about its battery-life and, since this information is vague and incomplete, is instantiated using a ranking function. For any instantiation $\mathcal{W}$ over $\mathcal{A}t$, then the initial (default) values for each $b \in \mathcal{A}t$ are:

$$\mathcal{W}(b) = \begin{cases} (1,1), & \text{if } \mathcal{W} \text{ is possibilistic,} \\ (0.5,0.5), & \text{if } \mathcal{W} \text{ is probabilistic,} \\ (0,0), & \text{if } \mathcal{W} \text{ is a ranking,} \end{cases}$$

where $\mathcal{W}(b) = (\overset{+}{\mu}, \overset{+}{\mu})$ denotes that $\overset{+}{\mu}$ is the λ value for b and that $\overset{-}{\mu}$ is the λ value for $\neg b$. In addition, given an epistemic state $\mathcal{W}$ and a formula φ in the language of $\mathcal{W}$, then recall that: $\lambda(\varphi) = \Pi(\varphi)$ if $\mathcal{W}$ is possibilistic (see Section 6.2); $\lambda(\varphi) = P(\varphi)$ if $\mathcal{W}$ is probabilistic (see Section 6.1); and $\lambda(\varphi)$ is taken from Definition 15 if $\mathcal{W}$ is a ranking function.

Prior to commencing the TEAgentSpeak reasoning cycle, it is possible to specify the agent's initial beliefs. This is performed on lines 3–11 of Listing 1. When the agent's GUB is revised to reflect these beliefs, we obtain the initial weights outlined in Table 1. The agent's initial goal is then described on line 15 of Listing 1. In this case, the achievement goal `!completeMission` is added to the agent's event set as an external event. At this point, the six plans in the agent's plan library on lines 19–29 are stored as plans $p_1, \ldots, p_6$ and the TEAgentSpeak reasoning cycle begins. A complete summary of the agent's reasoning (until both its event and intention sets are empty) is shown in Table 2. Informally, the agent receives the goal `!completeMission` as a new event and decides to travel from `location(1)` to `location(2)` using plan p_1 because it believes it is currently at `location(1)`, that it is more likely to find treasure at `location(2)` than `location(3)` and that it has sufficient battery to travel to `location(2)` while still being able to complete its journey to `location(4)`. At `location(2)`, the agent then decides to travel from `location(2)` to `location(4)` using plan p_2 because it believes it is currently at `location(2)` but does not believe it has sufficient battery to travel to `location(3)` while still being able to complete its journey to `location(4)`, even though it believes that it is likely to find more treasure at `location(3)`. Once at `location(4)`, the agent decides to complete its current planning phase using plan p_4. At this point, the agent waits for new events to act upon.

Obviously we are mainly interested in the effect of introducing a GUB to the AgentSpeak reasoning cycle. That is to say, we are mainly interested in the evaluation of plan contexts and the revision[16] of uncertain beliefs. We can describe these in detail by referring again to Listing 1 and Table 2. Notice that, for plans p_5 and p_6, the context is `true`. Thus, if p_5 and p_6 are relevant plans, then they will also be applicable plans since `true` trivially unifies with any GUB. Evaluating plan contexts in this scenario is only of interest when the set of applicable plans contains any of the plans $p_1, \ldots, p_4$.

16. Without loss of generality, we only consider belief revision triggered from within plan bodies.

Step	E	Relevant plans	Applicable plans	Adopt intention	I	Execute step	Generate event	I
1	$\{\underline{e_1}\}$	$\{\langle p_1, \emptyset\rangle, \langle p_2, \emptyset\rangle, \langle p_3, \emptyset\rangle, \langle p_4, \emptyset\rangle\}$	$\{\underline{\langle p_1, \{\texttt{X/1, Y/2, Z/3}\}\rangle}\}$	$i_1 = [\langle p_1, \{\texttt{X/1, Y/2, Z/3}\}\rangle]$	$\{\underline{i_1}\}$	`!proceed(location(1),location(2))`	$e_2 = \langle$`+!proceed(location(1),location(2))`$, i_1\rangle$	$\emptyset$
2	$\{\underline{e_2}\}$	$\{\langle p_5, \{\texttt{X/1, Y/2}\}\rangle\}$	$\{\underline{\langle p_5, \{\texttt{X/1, Y/2}\}\rangle}\}$	$i_2 = [\langle p_5, \{\texttt{X/1, Y/2}\}\rangle, \langle p_1, \{\texttt{X/1, Y/2, Z/3}\}\rangle]$	$\{\underline{i_2}\}$	`collectItems`	–	$\{i_2\}$
3	$\emptyset$	–	–	–	$\{\underline{i_2}\}$	`*(~hasItems(location(1)),1)`	$e_3 = \langle$`*(~hasItems(location(1)),1)`$, []\rangle$	$\{i_2\}$
4	$\{\underline{e_3}\}$	$\emptyset$	–	–	$\{\underline{i_2}\}$	`*(hasItems(location(1)),0)`	$e_4 = \langle$`*(hasItems(location(1)),0)`$, []\rangle$	$\{i_2\}$
5	$\{\underline{e_4}\}$	$\emptyset$	–	–	$\{\underline{i_2}\}$	`!travel(location(1),location(2))`	$e_5 = \langle$`+!travel(location(1),location(2))`$, i_2\rangle$	$\emptyset$
6	$\{\underline{e_5}\}$	$\{\langle p_6, \{\texttt{X/1, Y/2}\}\rangle\}$	$\{\underline{\langle p_6, \{\texttt{X/1, Y/2}\}\rangle}\}$	$i_3 = [\langle p_6, \{\texttt{X/1, Y/2}\}\rangle, \langle p_5, \{\texttt{X/1, Y/2}\}\rangle, \langle p_1, \{\texttt{X/1, Y/2, Z/3}\}\rangle]$	$\{\underline{i_3}\}$	`*(at(location(1)),0.01)`	$e_6 = \langle$`*(at(location(1)),0.01)`$, []\rangle$	$\{i_3\}$
7	$\{\underline{e_6}\}$	$\emptyset$	–	–	$\{\underline{i_3}\}$	`*(at(location(2)),0.99)`	$e_7 = \langle$`*(at(location(2)),0.99)`$, []\rangle$	$\{i_3\}$
8	$\{\underline{e_7}\}$	$\emptyset$	–	–	$\{\underline{i_3}\}$	`!completeMission`	$e_8 = \langle$`+!completeMission`$, i_3\rangle$	$\emptyset$
9	$\{\underline{e_8}\}$	$\{\langle p_1, \emptyset\rangle, \langle p_2, \emptyset\rangle, \langle p_3, \emptyset\rangle, \langle p_4, \emptyset\rangle\}$	$\{\underline{\langle p_2, \{\texttt{X/2, Y/3, Z/1}\}\rangle}\}$	$i_4 = [\langle p_2, \{\texttt{X/2, Y/3, Z/1}\}\rangle, \langle p_5, \{\texttt{X/1, Y/2}\}\rangle, \langle p_1, \{\texttt{X/1, Y/2, Z/3}\}\rangle]$	$\{\underline{i_4}\}$	`broadcast(try(location(3)))`	–	$\{i_4\}$
10	$\emptyset$	–	–	–	$\{\underline{i_4}\}$	`!proceed(location(2),location(4))`	$e_9 = \langle$`+!proceed(location(2),location(4))`$, i_4\rangle$	$\emptyset$
11	$\{\underline{e_9}\}$	$\{\langle p_5, \{\texttt{X/2, Y/4}\}\rangle\}$	$\{\underline{\langle p_5, \{\texttt{X/2, Y/4}\}\rangle}\}$	$i_5 = [\langle p_5, \{\texttt{X/2, Y/4}\}\rangle, \langle p_2, \{\texttt{X/2, Y/3, Z/1}\}\rangle, \langle p_5, \{\texttt{X/1, Y/2}\}\rangle, \langle p_1, \{\texttt{X/1, Y/2, Z/3}\}\rangle]$	$\{\underline{i_5}\}$	`collectItems`	–	$\{i_5\}$
12	$\emptyset$	–	–	–	$\{\underline{i_5}\}$	`*(~hasItems(location(2)),1)`	$e_{10} = \langle$`*(~hasItems(location(2)),1)`$, []\rangle$	$\{i_5\}$
13	$\{\underline{e_{10}}\}$	$\emptyset$	–	–	$\{\underline{i_5}\}$	`*(hasItems(location(2)),0)`	$e_{11} = \langle$`*(hasItems(location(2)),0)`$, []\rangle$	$\{i_5\}$
14	$\{\underline{e_{11}}\}$	$\emptyset$	–	–	$\{\underline{i_5}\}$	`!travel(location(2),location(4))`	$e_{12} = \langle$`+!travel(location(2),location(4))`$, i_5\rangle$	$\emptyset$
15	$\{\underline{e_{12}}\}$	$\{\langle p_6, \{\texttt{X/2, Y/4}\}\rangle\}$	$\{\underline{\langle p_6, \{\texttt{X/2, Y/4}\}\rangle}\}$	$i_6 = [\langle p_6, \{\texttt{X/2, Y/4}\}\rangle, \langle p_5, \{\texttt{X/2, Y/4}\}\rangle, \langle p_2, \{\texttt{X/2, Y/3, Z/1}\}\rangle, \langle p_5, \{\texttt{X/1, Y/2}\}\rangle, \langle p_1, \{\texttt{X/1, Y/2, Z/3}\}\rangle]$	$\{\underline{i_6}\}$	`*(at(location(2)),0.01)`	$e_{13} = \langle$`*(at(location(2)),0.01)`$, []\rangle$	$\{i_6\}$
16	$\{\underline{e_{13}}\}$	$\emptyset$	–	–	$\{\underline{i_6}\}$	`*(at(location(4)),0.99)`	$e_{14} = \langle$`*(at(location(4)),0.99)`$, []\rangle$	$\{i_6\}$
17	$\{\underline{e_{14}}\}$	$\emptyset$	–	–	$\{\underline{i_6}\}$	`!completeMission`	$e_{15} = \langle$`+!completeMission`$, i_6\rangle$	$\emptyset$
18	$\{\underline{e_{15}}\}$	$\{\langle p_1, \emptyset\rangle, \langle p_2, \emptyset\rangle, \langle p_3, \emptyset\rangle, \langle p_4, \emptyset\rangle\}$	$\{\underline{\langle p_4, \{\texttt{X/3, Y/1, Z/2}\}\rangle}\}$	$i_7 = [\langle p_4, \{\texttt{X/3, Y/1, Z/2}\}\rangle, \langle p_5, \{\texttt{X/2, Y/4}\}\rangle, \langle p_2, \{\texttt{X/2, Y/3, Z/1}\}\rangle, \langle p_5, \{\texttt{X/1, Y/2}\}\rangle, \langle p_1, \{\texttt{X/1, Y/2, Z/3}\}\rangle]$	$\{\underline{i_7}\}$	`depositItems`	–	$\emptyset$

Table 2: TEAgentSpeak reasoning cycle for Listing 1 where $[\langle p_1, \sigma_1\rangle, \ldots, \langle p_n, \sigma_n\rangle]$ denotes a stack of partially instantiated plans, such that $\langle p_1, \sigma_1\rangle$ is at the top of the stack, and $\underline{s_i}$ denotes that element s_i is selected and removed from the set $\{s_1, \ldots, s_n\}$ by the relevant selection function. Technically, each partially instantiated plan in an intention also has an index identifying the next step in the plan body to be executed, however we omit this index since it can be inferred from context. The initial event $e_1 = \langle$`!completeMission`$, []\rangle$ is defined on line 15 of Listing 1.

Also notice that each of these plans contain the subformula `X \== 4 & X \== Y & X \== Z & Y \== Z` which, combined with the rest of the formulas, restricts the possible unifiers to:

$$\sigma_1 = \{\texttt{X/1}, \texttt{Y/2}, \texttt{Z/3}\}, \quad \sigma_3 = \{\texttt{X/2}, \texttt{Y/1}, \texttt{Z/3}\}, \quad \sigma_5 = \{\texttt{X/3}, \texttt{Y/1}, \texttt{Z/2}\},$$
$$\sigma_2 = \{\texttt{X/1}, \texttt{Y/3}, \texttt{Z/2}\}, \quad \sigma_4 = \{\texttt{X/2}, \texttt{Y/3}, \texttt{Z/1}\}, \quad \sigma_6 = \{\texttt{X/3}, \texttt{Y/2}, \texttt{Z/1}\}.$$

We can thus omit this subformula from the rest of our discussion as we know that it will always be satisfied by each of these unifiers. To determine applicable plans we must substitute variables, apply Definition 6 to parse the context of each relevant plan, and determine λ values from the underlying epistemic states. We evaluate e.g. plan p_1 as:

$$\begin{aligned}
&P(\neg\texttt{at(location(X))}) \leq P(\neg\texttt{at(location(Y))}) \\
\wedge\ &P(\neg\texttt{at(location(X))}) \leq P(\neg\texttt{at(location(Z))}) \\
\wedge\ &P(\neg\texttt{at(location(X))}) \leq P(\neg\texttt{at(location(4))}) \\
\wedge\ &(\Pi(\texttt{hasItems(location(Y))}) > \Pi(\neg\texttt{hasItems(location(Y))}) \\
&\quad \vee \Pi(\texttt{hasItems(location(Z))}) > \Pi(\neg\texttt{hasItems(location(Z))})) \\
\wedge\ &\Pi(\neg\texttt{hasItems(location(Y))}) \leq \Pi(\neg\texttt{hasItems(location(Z))}) \\
\wedge\ &\lambda(\texttt{sufficientBattery(location(X),location(Y),location(4))}) \\
&\quad > \lambda(\neg\texttt{sufficientBattery(location(X),location(Y),location(4))})
\end{aligned}$$

Considering the six possible unifiers, we obtain the following results at step 1:

$$\begin{aligned}
p_1\sigma_1 &: (0.1 \leq 0.94) \wedge (0.1 \leq 0.97) \wedge (0.1 \leq 0.99) \wedge ((1 > 0.2) \vee (1 > 0.3)) \wedge (0.2 \leq 0.3) \wedge (1 > 0) &&\Leftrightarrow \top \\
p_1\sigma_2 &: (0.1 \leq 0.97) \wedge (0.1 \leq 0.94) \wedge (0.1 \leq 0.99) \wedge ((1 > 0.3) \vee (1 > 0.2)) \wedge (0.3 \leq 0.2) \wedge (1 > 0) &&\Leftrightarrow \bot \\
p_1\sigma_3 &: (0.94 \leq 0.1) \wedge (0.94 \leq 0.97) \wedge (0.94 \leq 0.99) \wedge ((0.1 > 1) \vee (1 > 0.3)) \wedge (1 \leq 0.3) \wedge (0 > 0) &&\Leftrightarrow \bot \\
p_1\sigma_4 &: (0.94 \leq 0.97) \wedge (0.94 \leq 0.1) \wedge (0.94 \leq 0.99) \wedge ((1 > 0.3) \vee (0.1 > 1)) \wedge (0.3 \leq 1) \wedge (0 > 0) &&\Leftrightarrow \bot \\
p_1\sigma_5 &: (0.97 \leq 0.1) \wedge (0.97 \leq 0.94) \wedge (0.97 \leq 0.99) \wedge ((0.1 > 1) \vee (1 > 0.2)) \wedge (1 \leq 0.2) \wedge (0 > 0) &&\Leftrightarrow \bot \\
p_1\sigma_6 &: (0.97 \leq 0.94) \wedge (0.97 \leq 0.1) \wedge (0.97 \leq 0.99) \wedge ((1 > 0.2) \vee (0.1 > 1)) \wedge (0.2 \leq 1) \wedge (0 > 0) &&\Leftrightarrow \bot
\end{aligned}$$

Thus p_1 is an applicable plan at step 1, with σ_1 the only valid unifier. Importantly, in TEAgentSpeak, we do not try to find all valid unifiers – rather we simply return the first valid unifier which is found. This approach to unification is consistent with logic programming convention and with Jason. In the same way, we can then determine the complete set of applicable plans for event e_1 at step 1 by evaluating the contexts of plans $p_2, \ldots, p_4$: in this case, we find that $p_1\sigma_1$ is the only applicable plan for e_1.

$b \in \mathcal{A}t$	Steps 1	4	7	8	13	16	17
`hasItems(location(1))`	(0.1, 1)	(0, 1)	–	–	(0, 0.2)	–	–
`hasItems(location(2))`	(1, 0.2)	–	–	–	(0, 0.2)	–	–
`hasItems(location(3))`	(1, 0.3)	–	–	–	(0.2, 0.2)	–	–
`at(location(1))`	(0.9, 0.1)	–	(0.01, 0.99)	–	–	–	–
`at(location(2))`	(0.06, 0.94)	–	–	(0.99, 0.01)	–	(0.01, 0.99)	–
`at(location(3))`	(0.03, 0.97)	–	–	–	–	–	–
`at(location(4))`	(0.01, 0.99)	–	–	–	–	–	(0.99, 0.01)

Table 3: Weights for belief atoms at the beginning of each step where omitted weights remain unchanged from the previous step. Weights for belief atoms in $\mathcal{A}t_3$ remain unchanged throughout execution and initial weights can be found in Table 1.

This type of evaluation must be repeated each time an event is selected for which relevant plans are found. For this reason, in order to explain the selection of applicable plans at each step, a complete listing of the weights modelled by the GUB throughout the reasoning cycle is provided in Table 3. For example, plan p_2 is selected at step 9 with unifier σ_4 because the agent now believes that it is at `location(2)` and that it is more likely that `location(3)` has items than `location(1)`, but the agent does not believe that they have sufficient battery to travel from `location(2)` to `location(3)` while still being able to reach `location(4)`. In addition, this table serves to summarize the effect of belief revision throughout execution as described in Definitions 14, 17 and 20. Notice that the ranking epistemic state $\mathcal{W}_3$ is unchanged throughout execution while the probabilistic epistemic state $\mathcal{W}_2$ only requires trivial changes. The more interesting change occurs in the possibilistic epistemic state $\mathcal{W}_1$ after the revision of $\sim$`hasItems(location(2))`, shown in step 13. In this case, the weights for all belief atoms in $\mathcal{A}t_1$ are revised so as to ensure consistency. Even with the simple types of uncertain information referred to in this scenario, TEAgentSpeak clearly demonstrates the benefits of applying our framework to practical BDI systems.

9. Related Work

The BDI framework (?) is used for modelling agents by expressing their Beliefs, Desires and Intentions. The Beliefs describe the knowledge of an agent, the desires express what the agent wants to bring about, and the intentions are those desires the agent has decided to act upon. The complex temporal modal logic used in the BDI framework, along with strong assumptions (e.g. unlimited resources), means that directly implementing the BDI framework has proven difficult. The AgentSpeak language (?), proposed by one of the original authors on BDI, resolved this shortcoming by instead introducing an abstract agent-based language. The language was, on the one hand, strongly related to the BDI theory but, on the other hand, easily implementable and based on existing attempts at implementing the BDI framework. Although AgentSpeak allowed a theoretical treatment of actual implementations, it did not yet allow for declarative goals. The Can language (?) in turn extended the AgentSpeak language with full operational semantics, including semantics for dealing with declarative goals. Such goals allow a considerable increase in flexibility. For example, plans can be stopped when the goal is reached instead of being blindly executed until the end. Declarative goals also open the door to the use of first-principles planners in Can, as the declarative goal describes the goal to reach rather than simply the steps to (try to) reach an implicit goal.

A shortcoming of the BDI framework in general, and languages such as AgentSpeak and Can in particular, is that they do not consider uncertainty. One of the first works to look at integrating uncertainty in a BDI context is the work on graded BDI (?). In the graded BDI setting it is assumed that the beliefs, desires and intentions have a degree of uncertainty. Furthermore, it was realised that different theories of uncertainty are needed to correctly model the different facets of uncertainty. The graded BDI system was further extended to incorporate norms (?), i.e. patterns of behaviour that should be adhered to in given circumstances. These norms are acquired and enforced in the same uncertain environment. To accommodate this, norms have an associated prominence to reflect their importance in the given uncertain environment. While the graded BDI framework is of a clear theoretical

interest, its usage of the same complex modal logic axiomatisation that made it hard to implement BDI have similarly prevented any direct implementations of the graded BDI framework. Early work also looked into the relationship between BDI and (PO)MDP (?). The authors found that BDI can be endowed with the ability to reason about uncertainty by linking it up with a corresponding POMDP-based method. The resulting hybrid BDI-POMDP framework outperforms both BDI and POMDP in that it can elegantly model team problems, even when faced with uncertainty. However, by design, this hybrid BDI-POMDP framework is limited to the modelling power and computational complexity of the POMDP component.

On the contrary, the work in (?) is one of the first to implement uncertain percepts in an AgentSpeak setting. It was not based on the graded BDI framework but approached the problem more pragmatically. The authors describe a classical AgentSpeak agent, along with Markov Decision Process (MDP) models for some aspects of the problem domain. As needed, these MDP models can be used to reason about the uncertain aspects of particular areas of the domain. A difficulty with this approach lies in the duplication of information, as the knowledge encoded in the MDPs often overlaps with the knowledge encoded in the AgentSpeak agent. In addition, there is no real integration between AgentSpeak and MDPs, which makes the framework particularly difficult to extend. In (?), one of the papers extended in this work, we proposed the first approach in which the ideas of graded beliefs are adopted. This is accomplished by allowing more fine-grained control over the beliefs by dividing those beliefs into isolated parts, each with their own representation and revision strategies. In spirit, this work is very close to CanPlan (?). As the beliefs of an agent are modelled as epistemic states in this framework, it allows for a very tight integration between the modelling of the uncertainty about the beliefs, and the actual reasoning capabilities of the agent itself.

Some work has been done on plan selection under uncertainty. In (?, ?) the authors address the limitation that the context of a plan is a Boolean formula that has to be specified at design time. Instead, they propose methods for a BDI system to learn the probability of success for a plan execution based on previous experience using a decision tree model. While the authors take uncertainty into account in the form of a probability of success, the work is limited when it comes to reasoning about uncertainty as the framework is restricted to reasoning about plan success. In (?) a framework is presented for plan selection in probabilistic BDI agents. Such agents can reason about the cost of plan execution as well as the success chance of plan execution. They do so by introducing a plan selection strategy that is able to choose a subset of plans to maximise the maximum number of goals that can be achieved while ensuring that all intentions are consistent and that given resource bounds are respected. Our work is distinct in that we focus on how to model uncertain beliefs and how such beliefs can trigger plans. As such, all these works address different problems and enrich each other (through learning, revising intentions, modelling belief uncertainty) rather than compete with each other.

The idea of a GUB – a set of local epistemic states – bears some resemblance with concepts introduced in other works. In (?) the author introduced a logic of viewpoints, where a formula is not simply true in a given world but is dependent on the viewpoint of that world, i.e. how the world is conceptualised. In a sense, the local epistemic states can be seen as viewpoints, although they are different in that they are used to model different

types of uncertainty and to model different *parts* of the world rather than different *ways of conceptualising the world*. Another related idea is that of multi-context systems (?, ?). In a multi-context system, each context can employ its own logic and information is shared between contexts using so-called bridge rules. There is a resemblance between local epistemic states and contexts, and the λ-evaluation proposed in this paper could be said to perform a similar role as bridge rules (although the λ-evaluation is used to extract information, not to pass information). The concept of multiple contexts was used in graded BDI (?), which we discussed earlier.

The epistemic states used in the current paper to model the uncertain beliefs are often only defined on a semantic level. Many interesting bodies of work tackle the problem of epistemic state revision, given a wide variety of different inputs (for an overview we refer the reader to e.g. (?)). However, a direct implementation of these semantic definitions is often (too) computationally expensive. Syntactic operators for revision with either classical or uncertain inputs have been considered in the literature, where most deal with classical inputs and are based on the AGM style of revision, e.g. (?, ?). Syntactic operators that are able to deal with iterated belief revision are far less common, and are usually only defined on the (semantic) level of epistemic states, for which we can use Ordinal Conditional Functions (OCF) (?) or, for example, the representation we used in this paper based on (?). A syntactic representation for OCF (?), along with the conditions that such a representation has to satisfy, was presented in (?).

A syntactic revision operator in the setting of possibility theory was later presented in (?) that could also deal with uncertain input. This operator makes use of the ability to transform a possibility distribution into an OCF and vice versa, effectively developing a revision operator for both frameworks based on the earlier work. However, this approach treats uncertain input as a form of conditioning, where the resulting beliefs have to conclude the formula with exactly the given degree of uncertainty. They do not interpret uncertain inputs as in this paper, i.e. as information that strengthens or weakens the beliefs that the agent currently holds. Postulates for how to reasonably treat uncertain input as unreliable information were only presented later in the literature (?).

Interesting work in the cross-section with BDI has been carried out in (?), where the authors develop a tractable form of belief revision by devising a cross-over between AGM style revision and reason-maintenance style belief revision (?). In particular, in a BDI setting where beliefs are modelled by literals (and plans take their usual form of rules) it can be shown that the operator satisfies most of the AGM postulates. Still, this approach can only deal with classical input. A framework for BDI agents dealing with uncertain input has been presented in (?), where the authors develop a theoretical framework based on possibility theory where both beliefs and desires are represented as possibility distributions. Confusingly, (?) used the term belief change that was also used in (?), but both frameworks are distinct. In (?) they then develop a way to select the best set of goals to be adapted depending on the consistency of these goals, which in turn depends on the uncertain beliefs of the agent. Their work was extended in (?) where they developed a syntactic approach for their framework, highlighting the practical feasibility. However, as discussed, their work is based on the notion of interpreting uncertain input in the sense of conditioning, where the beliefs of the agent need to exactly reflect the input uncertainty.

10. Conclusion

As human beings, we can be seen as prime examples of autonomous agents. We are capable of seamlessly making snapshot decisions, persisting in our attempts to achieve our desires, and reacting in intelligent ways in a real world pervaded by uncertainty. There is a significant challenge in artificial intelligence to develop similarly capable intelligent and autonomous agents, along with the tools to define them. The BDI framework, as well as AgentSpeak, have clearly provided significant advances in this ongoing quest. Languages such as AgentSpeak, and their derivatives such as Jason and 3APL, have greatly simplified how we can express the complicated behaviour of an agent. Through their underpinnings with the BDI framework the resulting agents already express a number of the desired properties. Indeed, the BDI framework allows the design of reactive agents that can adopt intentions which they potentially pursue in a relentless fashion. In this work, we have further advanced the state-of-the-art in the BDI domain by designing a framework that allows different types of uncertainty to be easily modelled, that allows an agent to effectively reason about – and react to – this uncertainty, and that offers a computationally efficient way of modelling and reasoning about this uncertainty. To the best of our knowledge this paper is the first to introduce a syntactic approach to belief change for dealing with unreliable input. It is also the first in which the beliefs of an agent are expressed as different epistemic states, each capable of using different uncertainty representations and revision mechanisms. We have extended on our previous work by demonstrating how these epistemic states can be initialised to actual theories of uncertainty such as possibility theory and probability theory, and we have developed the machinery to offer both fully syntactic revisions as well as tractable revisions. Finally, to highlight the capabilities of our new framework we have created and evaluated an intricate scenario to demonstrate how these new capabilities can be readily modelled and easily put into practice. We hope our work will inspire some future research in this direction. We also anticipate to see the use of our framework in real-world applications.

Acknowledgements

We would like to thank the reviewers of this paper for their invaluable contributions, their helpful comments, and the many hours they must have spent to read the paper in such detail. Without their help the paper would not have been as good as it is now.

Prof. Dr. Weiru Liu and Dr. Jun Hong completed most of the work on this paper while affiliated with Queen's University Belfast (QUB). This work has been partially funded by EPSRC PACES project (Ref: EP/J012149/1).