CAN(PLAN)+: Extending the Operational Semantics of the BDI Architecture to deal with Uncertain Information

Kim Bauters and **Weiru Liu** and **Jun Hong**
Queen's University Belfast
Belfast, United Kingdom

Carles Sierra and **Lluís Godo**
IIIA, CSIC, Bellaterra, Spain
Queen's University Belfast, Belfast, United Kingdom

Abstract

The BDI architecture, where agents are modelled based on their beliefs, desires and intentions, provides a practical approach to develop large scale systems. However, it is not well suited to model complex Supervisory Control And Data Acquisition (SCADA) systems pervaded by uncertainty. In this paper we address this issue by extending the operational semantics of CAN(PLAN) into CAN(PLAN)+. We start by modelling the beliefs of an agent as a set of epistemic states where each state, possibly using a different representation, models part of the agent's beliefs. These epistemic states are stratified to make them commensurable and to reason about the uncertain beliefs of the agent. The syntax and semantics of a BDI agent are extended accordingly and we identify fragments with computationally efficient semantics. Finally, we examine how primitive actions are affected by uncertainty and we define an appropriate form of lookahead planning.

1 INTRODUCTION

SCADA (Supervisory Control And Data Acquisition) systems are known for their large scale processes in a wide variety of domains, including production processes [Zhi et al., 2000] and energy and transportation systems [Boyer, 2009]. One way of modelling such systems is by means of the BDI architecture [Bratman, 1987] which allows us to decompose a complex system into a set of autonomous and interacting agents, where an agent is defined by its (B)eliefs, (D)esires and (I)ntentions. Agent-based programming languages based on the BDI framework have been proposed [Ingrand et al., 1992, Rao, 1996, Dastani, 2008] and have been used to some extent to model SCADA systems (e.g. [McArthur et al., 2007]).

However, current BDI implementations are not well-suited to model the next generation of complex SCADA systems. The reason for this is two-fold. On the one hand, current BDI implementations are not able to deal with uncertainty associated with the beliefs of an agent (e.g. due to noisy sensing) or the uncertain effects of actions (e.g. due to actuator malfunctions). This limits the ability of a BDI agent to react in a satisfactory way in an uncertain environment. On the other hand, and closely related, is that most BDI implementations do not provide any mechanisms for lookahead planning to guide (part of) the BDI execution in this uncertain setting.

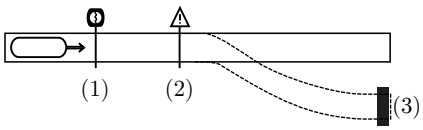

Figure 1: Scenario for a train agent with an unreliable signal (1), a dangerous junction (2) and a goal station (3).

To illustrate these issues, consider the running example in Figure 1. A train agent is moving along a track with a signal (1). The signal, which is green or orange, informs the agent if it violates the safe distance (with uncertainty due to e.g. mist, conflicting signals ...). Once the agent has passed the signal, the agent decides on how to approach the junction (2). The speed of the train is not known exactly, yet the agent needs to decide whether it wants to keep speeding (as it is running late) or slow down (resp. 75% and 50% chance of reaching the junction in time). Once at the junction, the action to take the junction only has a 30% chance of succeeding when speeding (e.g. due to derailment). Otherwise, the junction can safely be taken. For simplicity, the station is reached on time only when the junction is safely taken. Clearly, an agent should be able to reason about the uncertainty and be able to plan ahead, e.g. foresee that slowing down is the best action.

Not a lot of work in the literature on BDI tackled the problem of representing, and reasoning about, uncertain information. A notable exception is the recent work in [Chen et al., 2013], which incorporates uncertain perceptions in the epistemic state of an agent after which

it is mapped to a classical belief base, thus ignoring the other information. The work on graded BDI systems [Casali et al., 2005, Casali et al., 2011] similarly discusses how uncertainty can pervade the beliefs, desires and intentions. However, the graded BDI framework is mainly of theoretical interest and has not led to actual implementations, contrary to how AgentSpeak and CAN have helped to advance the state-of-the-art in BDI implementations.

Planning in a BDI agent, where the agent reflects on its actions before executing them, has been considered in numerous works. While the BDI model does not prevent planning, most BDI implementations resort to simple plan selection strategies to avoid the computational cost associated with declarative planning. This prevents them from acting optimally when needed, e.g. when important resources are consumed during the execution of plans. A formal approach to planning in BDI, called CANPLAN, was presented in [Sardiña et al., 2006]. CANPLAN is based on the Conceptual Agent Notation (CAN) [Winikoff et al., 2002], a high-level agent language in the spirit of BDI [Rao and Georgeff, 1991]. It is closely related to AgentSpeak [Rao, 1996] but allows for declarative goals alongside procedural steps (i.e. we can state *what* we want to achieve, not just *how* to achieve it). CANPLAN extends this work by introducing a $Plan(\cdot)$ action, making planning on-demand an integral part of the BDI framework. Nevertheless, none of the approaches to BDI address the issues that arise when dealing with actions with uncertain effects, or uncertain beliefs in general.

In this paper we propose the CAN+ and CANPLAN+ frameworks, which extend CAN and CANPLAN to provide formal approaches for dealing with uncertain beliefs and (planning for) actions with uncertain effects. The beliefs of an agent are modelled as a set of epistemic states, with each *local* epistemic state representing a distinct part of the beliefs held by the agent. Each epistemic state can deal with a different form of uncertainty (e.g. possibilities or infinitesimal probabilities) and includes its own revision strategy. Such a set of local epistemic states will be called a *Global* Uncertain Belief set (GUB) and allows the agent to reason about different forms of uncertainty in a uniform way, as long as these can be expressed using epistemic states that are equivalent to Definition 1. This is achieved in two steps. Firstly, a stratification of each local epistemic state allows for commensurability, along with the ability to reason about the uncertain beliefs. In other words: it enables an agent to reason about those beliefs it currently does not assume to be true (in the sense of beliefs in classical logics). Nevertheless, an agent commonly still considers some outcomes to be more plausible than others. The agent thus gains the ability to reflect on its own uncertainty. Secondly, an agent will be able to revise a GUB directly, with the GUB ensuring that only the information relevant to a specific local epistemic state is used to revise it. This idea of a GUB will be introduced in the CAN framework to obtain CAN+. CANPLAN+ further extends upon it by adding the ability to execute and plan for non-deterministic actions, all while dealing with uncertain beliefs.

The remainder of this paper is organised as follows. Some preliminary notions are discussed in Section 2. We explore how we can efficiently model and reason about uncertain beliefs in Section 3, where we introduce the idea of epistemic states and how they can be applied in a BDI setting. In Section 4 we extend CAN to enable us to deal with uncertain beliefs, while uncertain actions and planning under uncertainty are addressed in Section 5. Related work is discussed in Section 6 and conclusions are drawn in Section 7.

2 PRELIMINARIES

An agent in the BDI framework is defined by its beliefs, desires and intentions. The beliefs encode the agent's understanding of the environment, the desires are those goals that an agent would like to accomplish and the intentions those desires that the agent has chosen to act upon.

CAN, and its extension CANPLAN, formalise the behaviour of a classical BDI agent, which is defined by a belief base $\mathcal{B}$ and a plan library Π. The belief base of an agent 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 the 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). 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 used in the plan body P is defined in Backus-Naur Form (BNF) as:

$$P ::= nil \mid +b \mid -b \mid act \mid ?\phi \mid !e \mid P_1; P_2 \mid P_1 \parallel P_2 \mid P_1 \triangleright P_2 \mid (|\Delta|) \mid Goal(\phi_s, P, \phi_f) \mid Plan(P)$$

with nil an empty or completed program, $+b$ and $-b$ belief addition and deletion, act a primitive action, $?\phi$ a test for ϕ 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 $P2$ only on failure of $P1$. $(|\Delta|)$ 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. The plan form $Goal(\phi_s, P, \phi_f)$ 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. CANPLAN furthermore introduces the $Plan(P)$ construct, which is used for offline lookahead planning. This construct will be discussed in more detail in Section 5.

The operational semantics of CANPLAN 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 $act : \psi \leftarrow \phi^-; \phi^+$. 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

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

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 and a primitive action are:

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

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

where the latter states that when the unified precondition $\psi\theta$ 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. We refer the reader to [Sardiña and Padgham, 2011] for a full overview of the semantics of CANPLAN.

Finally, a *preorder* $\leq_A$ defined on any set A is a reflexive and transitive relation over $A \times A$. We say that $\leq_A$ is *total* iff for all $a, b \in A$ we have that either $a \leq_A b$ or $b \leq_A a$. A strict order $<_A$ and an indifference relation $=_A$ can conventionally be induced from $\leq_A$.

3 MODELLING AND REASONING ABOUT UNCERTAIN BELIEFS

As discussed in Section 2, a belief base in CAN is defined over a logic for which operations are available to add, delete and entail beliefs. This classical setting allows for an easy approach to belief revision. However in this paper we are concerned with the modelling of, and reasoning over, uncertain information. To deal with uncertainty we will need more elaborate ways to both represent the beliefs and to revise the beliefs when new information becomes available. To this end, we will use epistemic states instead of a belief base as in CAN.

3.1 MODELLING UNCERTAIN BELIEFS AS EPISTEMIC STATES

To define epistemic states, we first start by considering a finite set $\mathcal{A}t$ of propositional atoms. For a set of atoms $A \subseteq \mathcal{A}t$ we define the set of literals that can be constructed using the atoms in A as $lit(A) = \{a \mid a \in A\} \cup \{\neg a \mid a \in A\}$. A proposition ϕ is defined in BNF as $\phi ::= a \mid \neg a \mid (\phi_1 \wedge \phi_2) \mid (\phi_1 \vee \phi_2)$, i.e. all propositions are in Negation Normal Form (NNF). This does not affect the expressiveness of our language as arbitrary formulas can be efficiently converted into NNF. It will, however, make the definition of our semantics easier. We will denote this language as $\mathcal{L}$. Now we introduce the concept of an epistemic state:

Definition 1. *(from [Ma and Liu, 2011]) Let Ω be a set of possible worlds. An* epistemic state *is a mapping $\Phi : \Omega \to \mathbb{Z} \cup \{-\infty, +\infty\}$.*

An epistemic state will be used to represent the mental state of an agent, where the value $\Phi(\omega)$ associated with a possible world ω, called the *weight* of ω, is understood as the degree of belief in the possible world ω. Throughout the paper we will denote epistemic states using capital Greek letters. For $\omega, \omega' \in \Omega$ and $\Phi(\omega) > \Phi(\omega')$ the intuition is that ω is more plausible than ω'. Two epistemic states Φ and Ψ are *semantically equivalent* iff $\exists k \in \mathbb{Z} \cdot \forall \omega \in \Omega : \Phi(\omega) = \Psi(\omega) + k$, i.e. the value associated with the possible worlds only has a relative meaning. In the remainder of this paper we assume that epistemic states have 2^A as their domain with $A \subseteq \mathcal{A}t$. The *strength* of preference on a propositional formula ϕ is defined as $\Phi(\phi) - \Phi(\neg\phi)$ with $\Phi(\phi) = \max_{\omega \models \phi}(\Phi(\omega))$. We use $\max_\Phi$ to denote $\max_\Phi = \max_{\omega \in \Omega}(\Phi(\omega)) - \min_{\omega \in \Omega}(\Phi(\omega)) + 1$, i.e. the weight stronger than any of the strengths associated with information in Φ, and $\min_\Phi = 1 - \max_\Phi$.[1] The values $\max_\Phi$ and $\min_\Phi$ are only needed in Section 4 when considering belief additions and deletions as in CAN.

Before we provide an example, it is important to clarify that the definition of an epistemic state from Definition 1 allows for the construction of a general framework. Indeed, this definition does not impose any restrictions on the values associated with the possible worlds, other than that they are weights. As such, it is the

[1] These values only change when the epistemic state is revised and can be computed as a by-product of revision.

most general way in which we can talk about an epistemic state, regardless of the actual representation. 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. For example, Definition 1 induces an Ordinal Conditional Function (OCF) [Spohn, 1988][2], which in turn can be transformed into other representations, e.g. those based on infinitesimal probabilities [Darwiche and Goldszmidt, 1994] and possibility theory [Dubois and Prade, 1995]. The representation from Definition 1 can thus be *instantiated* using any of the other representations to best suit the nature of the uncertainty. After developing all main concepts, we will show in Section 4 that this will allow us to work with these different forms of uncertainty in a uniform way.

We now give an example of such an epistemic state.

Example 1. *Consider a signal that can be* (o)*range or* (g)*reen (but never both on). When the signal is orange it usually indicates that the agent is about to violate the safe distance* (sd)*. The agent believes that the light is green and that there is still a safe distance with the train in front. Even when the signal would turn out not to be green, the agent still believes that there would be a safe distance with the train in front of it. An epistemic state* Φ *could be:*

$$\begin{aligned} \Phi(\{o,g,sd\}) &= -\infty & \Phi(\{\neg o,g,sd\}) &= 10 \\ \Phi(\{o,g,\neg sd\}) &= -\infty & \Phi(\{\neg o,g,\neg sd\}) &= -2 \\ \Phi(\{o,\neg g,sd\}) &= 7 & \Phi(\{\neg o,\neg g,sd\}) &= 7 \\ \Phi(\{o,\neg g,\neg sd\}) &= 6 & \Phi(\{\neg o,\neg g,\neg sd\}) &= -2 \end{aligned}$$

where $\max_\Phi = +\infty$ *and* $\min_\Phi = -\infty$*. The weight associated with e.g.* $\{o,g,sd\}$ *means that we strongly disbelieve this possible world, while the weight associated with* $\{\neg o,g,sd\}$ *implies that we believe this possible world to be more plausible than any of the other possible worlds.*

The belief set, i.e. the sentences that an agent is committed to believe, is commonly defined as the set that has all the most plausible worlds as its models.

Definition 2. *(from [Ma and Liu, 2011]) Let* Φ *be an epistemic state. The* belief set *of* Φ*, denoted as* $Bel(\Phi)$*, is defined as* $Bel(\Phi) = \phi$ *with* ϕ *any propositional formula such that* $Mod(\phi) = \min(\Omega, \leq_\Phi)$*. Here* $Mod(\phi)$ *is the set of models of* ϕ *and* $\leq_\Phi$ *is a total preorder relation over* Ω *such that* $\omega \leq_\Phi \omega'$ *iff* $\Phi(\omega) \geq \Phi(\omega')$*.*

The model of the belief set thus only contains those possible worlds with the highest weight. In this paper we extend on this idea by also taking the other possible worlds into account. These other possible worlds constitute the uncertain information, i.e. they define the preferences the agent has over the outcomes that are currently not believed to be true.

[2]Compared to OCFs, the representation from Definition 1 avoids the normalisation step.

To clearly identify these preferences, irrespective of the actual values of these weights in different representations of the epistemic states, we consider a stratification of the set of possible worlds. The highest stratum (containing those possible worlds with the strongest associated belief) corresponds to the set of models of the belief set, i.e. that what the agent believes to be true. The other strata constitute the uncertain information, with information in a higher stratum believed/preferred over information in a lower stratum.

Definition 3. *Let* Φ *be an epistemic state. The stratification* λ *of the domain* Ω *from* Φ *induced by the total preorder relation* $\leq_\Phi$ *is defined as:*

$$\lambda(\omega) = \begin{cases} 1 & \omega \in \min(\Omega, \leq_\Phi) \\ n+1 & \omega \in \min(\Omega \setminus \{\omega \mid \lambda(\omega) \leq n\}, \leq_\Phi) \end{cases}$$

In Example 1 we obtain $\lambda(\{\neg o,g,sd\}) = 1$, $\lambda(\{o,\neg g,sd\}) = \lambda(\{\neg o,\neg g,sd\}) = 2$, etc. This idea of a stratification readily corresponds with the more common notion of a stratification over propositional formulas as any subset of possible worlds can trivially be represented by a single propositional formula.

Notice that the models of the belief set (see Definition 2) correspond to those possible worlds ω for which $\lambda(\omega) = 1$, i.e. information on all the other strata is ignored. Instead of simply ignoring this information, we want to make it possible for a BDI agent to reason about the preferences expressed throughout the stratification. To this end we extend the language $\mathcal{L}$ with the connectives $\geq$ and $>$. The intuition of $a \geq b$ (resp. $a > b$) is that the agent believes a to be at least as plausible as b (resp. a is strictly more plausible than b). These new connectives are taken to have the lowest precedence. The resulting language $\mathcal{L}_\geq$, or the *context language*, can be defined in BNF as:

$$\phi ::= a \mid \neg a \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \phi_1 \geq \phi_2 \mid \phi_1 > \phi_2$$

where formulas with connectives such as $\rightarrow$ and $\leftrightarrow$ can easily be transformed into logically equivalent statements in the language $\mathcal{L}_\geq$. Notice that this definition is the equivalent of NNF for propositional formulas. Any proposition using the connectives $\neg$, $\wedge$, $\vee$, $\geq$ and $>$ can be turned into an equivalent formula in $\mathcal{L}_\geq$ in the usual way and by rewriting $\neg(\psi_1 \geq \psi_2)$ as $(\psi_2 > \psi_1)$ and $\neg(\psi_1 > \psi_2)$ as $(\psi_2 \geq \psi_1)$. We assume that this has been done when needed throughout paper.

By extending the mapping λ we can define the semantics of $\mathcal{L}_\geq$ over arbitrary formulas. We have:

$$\lambda(\phi) = \begin{cases} \min\{\lambda(\omega) \mid \omega \models \phi\} & \text{if } \phi \in \mathcal{L} \\ \lambda(pare(\phi)) & \text{otherwise} \end{cases}$$

with $\min(\emptyset) = \infty$. Before defining the function $pare$, we point out that λ is closely related to a possibility measure [Dubois et al., 1994] for propositional

formulas $\phi, \psi \in \mathcal{L}$. We readily establish some interesting properties such as $\lambda(\phi \vee \psi) = \min(\lambda(\phi), \lambda(\psi))$, $\lambda(\phi \wedge \psi) \geq \max(\lambda(\phi), \lambda(\psi))$, $\lambda(\top) = 1$, $\lambda(\bot) = \infty$ and $\min(\lambda(\phi), \lambda(\neg\phi)) = 1$.

When ϕ is not a propositional statement (i.e. $\phi \notin \mathcal{L}$), we need to pare down the formula until the formula is a classical propositional statement. This is done by:

$$pare(\phi \wedge \psi) = check(\phi) \wedge check(\psi)$$
$$pare(\phi \vee \psi) = check(\phi) \vee check(\psi)$$
$$pare(\phi \geq \psi) = \begin{cases} \top & \text{if } \lambda(\neg\phi) \geq \lambda(\neg\psi) \\ \bot & \text{otherwise} \end{cases}$$
$$check(\phi) = \begin{cases} \phi & \text{if } \phi \in \mathcal{L} \\ pare(\phi) & \text{otherwise} \end{cases}$$

with $pare(\phi > \psi)$ equivalently defined as $pare(\phi \geq \psi)$. The intuition of paring down is straightforward: for each operand of the operators $\wedge$ and $\vee$ we verify whether it is an expression in the language $\mathcal{L}$ (for which the λ-value can readily be determined). Otherwise, we need to further pare it down. When the operator is $\geq$ or $>$, we define it as a plausibility ordering with an expression such as $\phi > \psi$ read as *"ϕ is more plausible than ψ"* or, alternatively, *"we have less reason to believe $\neg\phi$ than $\neg\psi$"*.[3] Such an expression can always be evaluated to true or false, i.e. $\top$ or $\bot$.

Finally, we can 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 $\Phi \models \phi$, if and only if $\lambda(\phi) < \lambda(\neg\phi)$.*

Example 2. *Consider λ of Φ from Example 1. We have:*

$$\lambda(g \wedge sd) = 1 \quad \lambda(o \wedge sd) = 2 \quad \lambda(g \wedge \neg g) = \infty$$
$$\lambda((o \vee g) > \neg sd) = 1 \quad \lambda(g \geq o) = 1 \quad \lambda(o \geq g) = \infty$$

For example, $\lambda(g \wedge sd) = 1$ since $\lambda\{\neg o, g, sd\} = 1$ and $\{\neg o, g, sd\} \models g \wedge sd$. An expression such as $(o \vee g) > \neg sd$, which is also believed to be true, states that the agent does not care about the colour of the light as long as the agent is not violating the safe distance.

Note that in Definition 4 it is insufficient to state that a formula is entailed when $\lambda(\psi) = 1$. Indeed, for $a \in \mathcal{A}t$ we can have that $\lambda(a) = \lambda(\neg a) = 1$, which occurs when we are ignorant about the value of a. As such, we need to ensure that both expressions are mapped onto strictly distinct strata. 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 $\phi \in \mathcal{L}$ be a propositional formula, Φ an epistemic state with domain Ω and λ the stratification of Ω. We have that $\Phi \models \phi$ iff for all $\omega \in \Omega$ such that $\lambda(\omega) = 1$ we have that $\omega \models \phi$, i.e. $Bel(\Phi) \models \phi$.*

[3] In terms of possibilistic theory: we want $N(\phi) \geq N(\psi)$, i.e. we want $\Pi(\neg\phi) \leq \Pi(\neg\psi)$ (with λ a reversed order).

3.2 SEMANTICS BASED ON LITERAL MAPPING

While the semantics we presented thus far allows us to reason about the uncertain information, they are computationally expensive for evaluating a context because they rely on an exponential structure. A tractable way to evaluate contexts can be obtained by restricting ourselves to a fragment of the language $\mathcal{L}_{\geq}$, allowing us to determine the truth of a context based on the λ-value associated with the constituent literals.

Example 3. *Consider the stratification λ of Φ from Example 1. We have $\lambda(o) = 2$, $\lambda(\neg o) = 1$, $\lambda(g) = 1$, $\lambda(\neg g) = 2$, $\lambda(sd) = 1$ and $\lambda(\neg sd) = 3$.*

Due to the way we defined λ over arbitrary formulas, we know that $\lambda(\phi \vee \psi)$ is decomposable, while $\lambda(\phi \wedge \psi)$ is not. Indeed, recall that we only have that $\lambda(\phi \wedge \psi) \geq \max(\lambda(\phi), \lambda(\psi))$. Therefore, we cannot allow $\wedge$ in our restricted language. Furthermore, it affects our ability to verify whether for a given expression ψ we have that $\lambda(\psi) < \lambda(\neg\psi)$. Indeed, we can only allow disjunction as part of an operand of the operators $\geq$ or $>$ as otherwise its negation would turn it into a conjunction, which we do not allow. As such, we obtain the fragment $L_{\geq}$, defined in BNF as:

$$d ::= a \mid \neg a \mid d_1 \vee d_2 \quad \phi ::= a \mid \neg a \mid d_1 \geq d_2 \mid d_1 > d_2$$

Contexts in this language can easily be evaluated, once we have the λ-values of the literals $lit(A)$:

$$\lambda(\phi \vee \psi) = \min(\lambda(\phi), \lambda(\psi))$$
$$\lambda(\phi \geq \psi) = \begin{cases} 1 & \lambda(\neg\phi) \geq \lambda(\neg\psi) \\ \infty & \text{otherwise} \end{cases}$$

and equivalently for $\lambda(\phi > \psi)$. As before, we say that ϕ is true iff $\lambda(\phi) < \lambda(\neg\phi)$. Even though enforcing tractability carries a penalty in terms of the expressive power, we still retain a language that takes advantage of the new connectives we have introduced, thus allowing us to reason over the plausibility of statements.

3.3 EFFICIENTLY MODELLING ISOLATED UNCERTAIN BELIEFS

As a final step in the representation of uncertain beliefs for a BDI agent, we introduce the concept of a global uncertain belief set (GUB) which applies to both Section 3.1 and 3.2.

Definition 5. *A* global uncertain belief set *$\mathcal{G}$ is a set $\{\Phi_1, ..., \Phi_n\}$ with each Φ_i 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 models beliefs that are semantically related, e.g. the colour of the signal or the condition of the track, and that are governed by the same

form of uncertainty. A GUB then groups a set of such local epistemic states. A GUB is therefore a representation of the overall beliefs of an agent, yet it differs in three significant ways from a global epistemic state. First, it avoids the exponential representation of a global epistemic state by partitioning the beliefs. Second, it allows for a general framework where each local epistemic state can use a different representation. Third, it does not include a revision strategy (as each local epistemic state can have a distinct revision strategy), i.e. it is not itself an epistemic state.

Despite these differences, we can use a GUB to determine if a context ϕ 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 $\Phi_i \models \phi$. Otherwise, we need to break ϕ apart up until the point where we can evaluate it directly. An expression can be split when the connective is either $\wedge$ or $\vee$. Since both operands will either be true or false such a decomposition is trivial. However, this also implies that we cannot decompose $\geq$ or $>$, since in this paper we require both operands to be 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. The problem of comparing the plausibilities of different local epistemic states is left for future work.

To formalise this intuition, we use $\mathcal{L}_{\geq}^{A_i}$ to denote the language $\mathcal{L}_{\geq}$ limited to atoms $a \in A_i$, i.e. the language corresponding to the epistemic state Φ_i. A formula ϕ is broken apart by ($simp$)lifying 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 $val_{GUB}(\phi)$ as:

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

$$simp(\phi \wedge \psi) = val_{GUB}(\phi) \wedge val_{GUB}(\psi)$$
$$simp(\phi \vee \psi) = val_{GUB}(\phi) \vee val_{GUB}(\psi)$$
$$simp(\phi \geq \psi) = \bot$$

and $simp(\phi > \psi)$ equivalently defined as $simp(\phi \geq \psi)$.

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

A visual representation of a GUB is given in Figure 2.

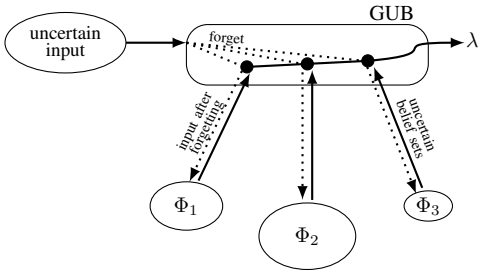

Figure 2: A GUB models the belief of an agent as a set of epistemic states Φ_i, each having its own representation (Definition 5). Commensurability is obtained by stratifying the possible worlds, with each stratum constituting an (uncertain) belief set (Definition 3). These stratifications can be combined to compute the λ-value of any arbitrary propositional formula. When new input is received, the local epistemic states are revised by ignoring (or *forget*ting) irrelevant information, as discussed in Section 4.

4 DEALING WITH UNCERTAIN BELIEFS IN A BDI AGENT

In the previous section we discussed how the beliefs of an agent can be represented as a set of local epistemic states. We also discussed how a GUB, and the underlying stratification of the local epistemic states, can be used to ensure commensurability. In this section, we extend CAN to CAN+ by adding to it a GUB to represent uncertain beliefs and by extending its syntax so that a CAN+ agent can reason about its uncertain beliefs. After Definition 8, we introduce how a GUB can be revised directly, thus allowing an agent to revise its beliefs irrespective of the various forms of uncertainty that govern those beliefs. In our extension CAN+, a context ψ is taken to be a sentence from the language $\mathcal{L}_{\geq}$. We assume the language for a plan body to be defined as in CAN, where we will gradually modify the language throughout this section. First though, we redefine the concept of configurations in CAN+. Rather than considering a belief base to model the knowledge, we will thus use a GUB to represent the uncertain beliefs of the agent. We have:

Definition 7. *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 (defined in Section 5), 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 ($?\phi$) and the rule for plan selection ($select$):

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

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

We retain the notation as used in [Sardiña et al., 2006] to denote unification as e.g. $\phi\theta$, 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 done against

the GUB. The language has implicitly been extended in both cases, since test goals and contexts can now include statements to reason over uncertain beliefs, i.e. $\phi, \psi_i \in \mathcal{L}_{\geq}$.

So far we have looked at how we can reason about the agent's (uncertain) beliefs, but we also want to revise these beliefs. When new input is presented (e.g. due to an internal belief change or the effects of an action), a naive approach would be to compute the global epistemic state as the Cartesian product of the local epistemic states, apply the input and then marginalise the outcome. However, such an approach is computationally too expensive. Instead, we will apply the input directly to the relevant epistemic states. First though, we define the notion of an *uncertain belief*.

Definition 8. *Let ϕ be a sentence in the language $\mathcal{L}^{A_{in}}$ with $A_{in} \subseteq A$. Let $\mu \in (\mathbb{Z} \cup \{-\infty, +\infty\})$. We say that $b = (\phi, \mu)$ is an* uncertain belief.

An input b, which is an uncertain belief, corresponds to a sequence of inputs $\mathit{refine}(b, \Phi_i)$ for any given local epistemic state $\Phi_i \in \mathcal{G}$. We have:

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

with $\mathit{forget}(b, \Phi_i)$ a sequence of inputs defined as $\langle (m', \mu) \mid b = (\phi, \mu), m \in \mathit{Mod}(\phi), m' = m \cap \mathit{lit}(A_i) \rangle$ and m' in (m', μ) treated as a conjunction of literals. When $A_{in} \subseteq A_i$ we could equivalently take $\mathit{forget}(b, \Phi_i) = \langle b \rangle$.

By $\mathcal{G} \circ b$ we denote that we want to revise the current beliefs of the agent with the input b, such that $\mathcal{G} \circ b = \{\Phi_i \circ \mathit{refine}(b, \Phi_i) \mid \forall \Phi_i \in \mathcal{G}\}$ with $\circ$ a revision operator. That is, revising a global uncertain belief set is taken as revising the local epistemic states with the given input. Each input (m', μ) in the sequence $\mathit{refine}(b, \Phi_i)$ corresponds to a simple epistemic state from [Ma and Liu, 2011], i.e. to an epistemic state Φ_{in} with the domain 2^{A_i} such that $\Phi_{in}(\omega) = \mu$ iff $\omega \models m'$ and $\Phi_{in}(\omega) = 0$ otherwise. An epistemic state Φ can be revised by a simple epistemic state Φ' with the same domain Ω, denoted as $\Phi \circ \Phi'$, as $\forall \omega \in \Omega, (\Phi \circ \Phi')(\omega) = \Phi(\omega) + \Phi'(\omega)$.[4] As such, when the input has been transformed to $\mathit{refine}(b, \Phi_i)$ for a given local epistemic state Φ_i, the revision is equivalent to iterated revision using the simple epistemic states in $\mathit{refine}(b, \Phi_i)$. 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 [Ma and Liu, 2011] (i.e. weights are cumulative and the order of updating does not affect the result).

Now we can 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:

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

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

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

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

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 $+\phi$ is nothing more than a shorthand for the statement $\circ(\phi, \max_{\mathcal{G}})$. Similarly, $-\phi$ can be considered a shorthand for $\circ(\phi, \min_{\mathcal{G}})$. As we try to keep the semantics as concise as possible, we opt to define these operators in the latter way. Such a choice will also need to be made in the next section, where we will directly present the approach based on syntactic sugar.

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

$$P ::= \mathit{nil} \mid \circ b \mid \mathit{act} \mid ?\phi \mid !e \mid P_1; P_2 \mid P_1 \parallel P_2 \mid P_1 \triangleright P_2 \mid (|\Delta|) \mid \mathit{Goal}(\phi_s, P, \phi_f)$$

with b an uncertain belief and $\phi, \phi_s, \phi_f \in \mathcal{L}_{\geq}$. We also modified the rules for $?\phi$ and select, while dropping the rules for $+\phi$ and $-\phi$ 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 $?\phi$, i.e. we need to verify ϕ_s and ϕ_f against $\mathcal{G}$.

5 DEALING WITH UNCERTAIN ACTIONS IN A BDI AGENT

The primitive actions of a BDI agent are affected by uncertainty in a variety of ways. Usually described in a STRIPS-like style such as $\mathit{act} : \psi \leftarrow \phi^-; \phi^+$, an action act can have uncertainty in the precondition ψ, uncertainty as to a specific effect (where the effect will change the epistemic state and possibly the belief set) or uncertainty as to the outcome of an action (with a probability for each outcome).

The first form of uncertainty is the easiest to incorporate. Similar to how the rule select for plan selection allows us to consider uncertain information, we can take $\psi \in \mathcal{L}_{\geq}$ and verify whether this context, or precondition, is satisfied.

Next, ϕ^- and ϕ^+ are usually taken to be *delete* and *add* lists of atoms. Nothing in the semantics for CAN+ prevents

[4] For other epistemic states these values can be extrapolated.

us from instead considering a list of uncertain beliefs ϕ^u as the results of an action. Not only does this considerably increases the expressive powers of action effects, but it also allows to define ϕ^- and ϕ^+ as special cases of ϕ^u with each being a list of propositions $\phi \in \mathcal{L}$ to which the weight $\min_\Phi$ and $\max_\Phi$ is assigned, respectively. As we did before, we assume hereafter that ϕ^- and ϕ^+ are forms of syntactic sugar for which we will not explicitly define the semantics.

Finally, the effects of an action may not be known in advance. This form of uncertainty has already been extensively considered in the literature, leading to variations of the STRIPS language that consider various outcomes with associated probabilities. Rather than a single outcome $\phi^-; \phi^+$ we consider a set of outcomes $\{\langle p_1, \phi_1^-, \phi_1^+\rangle, ..., \langle p_n, \phi_n^-, \phi_n^+\rangle\}$ with $\sum_{i=1}^n p_i = 1$.

By adopting a STRIPS-like probabilistic action library $\mathcal{D}$, populated by *probabilistic action description rules* – each representing a single independent action – we can model these three forms of uncertainty with rules of the form:

$$act : \psi_{act} \leftarrow \{\langle p_1, \phi_1^u\rangle, ..., \langle p_n, \phi_n^u\rangle\}$$

such that $p_i \geq 0$, $\sum_{i=1}^n p_i = 1$ with ψ_{act} an uncertain belief and ϕ_i^u a list of uncertain beliefs.

Example 4. *Consider the running example from the introduction and Figure 1. We can model the actions to slow down and to continue at the same speed as*

$$\begin{aligned} slow : true \leftarrow \{&\langle 0.4, [(junc, \max_{\mathcal{G}}), (sp, -20)]\rangle, \\ &\langle 0.6, [(late, \max_{\mathcal{G}}), (sp, -20)]\rangle\} \\ cont : sd \geq \neg sd \leftarrow \{&\langle 0.75, [(junc, \max_{\mathcal{G}})]\rangle, \\ &\langle 0.25, [(late, \max_{\mathcal{G}})]\rangle\} \end{aligned}$$

The first action can always be applied and has two outcomes. With 40% chance the junction is reached in time and with 60% the train is late. In both cases the (sp)eed is reduced. The second action encodes an agent in a hurry: the agent will not wait until there is a safe distance, i.e. the agent continues whenever he thinks it is at least more plausible that there is still a safe distance (or when the agent is ignorant and doesn't care).

While we already know how to correctly deal with uncertain beliefs, we do not yet have the machinery in the operational semantics to deal with probabilistic effects. To model a probabilistic action we use the notion of a probabilistic transition $C \longrightarrow_p C'$ where p represents the transition probability between the configurations C and C' [Di Pierro and Wiklicky, 1998]. Notice that all the transition rules used thus far are special cases of probabilistic transition rules where the probability of the transition is 1. As such we assume in CAN+ that all transition rules are probabilistic transition rules, where the probability is 1 unless explicitly specified. The act derivation rule can then be defined as:

$$\frac{(a : \psi \leftarrow effects) \in \mathcal{D} \qquad a\theta = act \qquad \mathcal{G} \models \psi\theta}{\langle \mathcal{G}, \mathcal{A}, act\rangle \longrightarrow_{p_i} \langle \mathcal{G} \circ \phi_i^u\theta, \mathcal{A} \cdot act, nil\rangle}\ act$$

with *effects* the set $\{\langle p_1, \phi_1^u\rangle, ..., \langle p_n, \phi_n^u\rangle\}$. As expected, the transition will depend on the probabilities of the different effects associated with the action act.

Thus far we have only discussed how CAN+, which extends CAN, adds the ability to model and reason about uncertain information. A parallel endeavour is to extend CANPLAN into CANPLAN+. The main difference between CAN and CANPLAN is the ability of the latter to perform lookahead planning by means of the $Plan(\cdot)$ action. A similar idea can be incorporated in CAN+ to arrive at CANPLAN+.

We know from the way we extended the act rule that, during the BDI execution, it is the probability of the transition that determines the effect of a primitive action. Furthermore, when a BDI agent tries to achieve some intention, this may involve the execution of a large number of plans. However, merely selecting the plan with the highest probability of reaching the next state without taking future actions into account may lead to poor performance. Indeed, this single step may not be on the same path that offers the highest overall chance of achieving our goal. Such issues can be addressed by using lookahead planning. During planning performed through the $Plan(\cdot)$ action we can take the probability of the different transitions into account and thus maximise the probability of achieving our intention.

To formalise this idea, we introduce the notion of maximising the overall transition probability. Intuitively, given two configurations C and C'', there may be more than one option such that $C \xrightarrow{\mathsf{plan}*} C''$. When a transition is labelled with 'plan' or 'bdi', the transition is resp. only valid in the planning context or during BDI execution. We want to take the transition $C \xrightarrow{\mathsf{plan}} C'$ such that C' is the next configuration on the path which offers us the highest overall chance of reaching our goal, which we will denote as $C \xrightarrow[C']{\mathsf{max}*} C''$.

Definition 9. *Let C and C'' be configurations such that $C \xrightarrow{\mathsf{plan}*} C''$. Furthermore, let p be such that $p = \prod_{i=0}^n p_i$ and $C \xrightarrow{\mathsf{plan}}_{p_1} C' \xrightarrow{\mathsf{plan}}_{p_2} \ldots \xrightarrow{\mathsf{plan}}_{p_n} C''$. We say that $C \xrightarrow[C']{\mathsf{max}*} C''$ when there does not exist a configuration D' that is different from C' in either its belief base, executed actions or plan body such that $C \xrightarrow{\mathsf{plan}}_{p'_1} D' \xrightarrow{\mathsf{plan}}_{p'_2} \ldots \xrightarrow{\mathsf{plan}}_{p'_m} C''$ and $p' > p$ with $p' = \prod_{i=0}^m p'_i$.*

In other words: C' is the next configuration on the most probable path to reach C''. Using Definition 9 we can extend the operational semantics of the $Plan(\cdot)$ construct to take into account that we are dealing with uncertain actions. In CANPLAN the rule for $Plan(\cdot)$ is as follows:

$$\frac{C \xrightarrow{\mathsf{plan}} C' \qquad C' \xrightarrow{\mathsf{plan}*} C''}{\langle \mathcal{B}, \mathcal{A}, Plan(P)\rangle \xrightarrow{\mathsf{bdi}} \langle \mathcal{B}', \mathcal{A}', Plan(P')\rangle}\ plan$$

with the configurations C, C' and C'' defined as $\langle \mathcal{B}, \mathcal{A}, P \rangle$, $\langle \mathcal{B}', \mathcal{A}', P' \rangle$ and $\langle \mathcal{B}'', \mathcal{A}'', nil \rangle$, respectively. Intuitively, this rule states that the next action to execute is the one that, according to our lookahead planning; will *eventually* lead us to achieving our goal.

The rule in CANPLAN+ for $Plan(\cdot)$, which not only ensures that we reach our goal but also maximises the chances of reaching our goal, is then defined as:

$$\frac{C \xrightarrow{\mathsf{plan}} C' \qquad C \xrightarrow[C']{\mathsf{max}*} C''}{\langle \mathcal{G}, \mathcal{A}, Plan(P) \rangle \xrightarrow{\mathsf{bdi}} \langle \mathcal{G}', \mathcal{A}', Plan(P') \rangle} \; plan$$

with C, C' and C'' defined as $\langle \mathcal{G}, \mathcal{A}, P \rangle$, $\langle \mathcal{G}', \mathcal{A}', P' \rangle$ and $\langle \mathcal{G}'', \mathcal{A}'', nil \rangle$, respectively.

Example 5. *Consider the running example from Figure 1. Assume we are at the decision point just after reaching the signal. An agent that plans ahead for the goal of reaching the station in time, will make the rational choice to slow down. If the agent did not perform lookahead planning and only looked at the highest chance to reach the junction, then continuing at the same speed would be preferred.*

6 RELATED WORK

The BDI framework [Rao and Georgeff, 1991] is notable for treating beliefs and intentions as two distinct ideas in an agent-based setting. However, due to the complex temporal modal logic being used and the assumption of unlimited resources there was a disconnect between the theory and implementations based on BDI. This problem was mostly resolved in [Rao, 1996] where an abstract agent-based language, called AgentSpeak, was proposed. This language was strongly related to the original BDI theory, while being easily implementable.

CAN [Winikoff et al., 2002] follows up on this approach of AgentSpeak and provides operational semantics for dealing with declarative goals. Such goals allow more flexibility, e.g. plans can be stopped when the goal is reached instead of being blindly executed until the end. Declarative goals also make it easier to define semantics for planning in a BDI setting. Most approaches on planning [de Silva and Padgham, 2005, Walczak et al., 2006, Meneguzzi et al., 2007] had been ad-hoc approaches without a semantical background for the integration of planning in BDI. Such a semantical background was provided in [Sardiña et al., 2006, Sardiña and Padgham, 2011] with the introduction of CANPLAN, an extension of CAN with a $Plan(\cdot)$ action that allows for offline lookahead planning.

Notable work on the integration of uncertainty in a BDI context has been done in the setting of graded BDI [Casali et al., 2005]. In graded BDI it is assumed that the beliefs, desires and intentions have a degree of uncertainty. While of theoretical interest, their framework uses a complex modal logic axiomatisation which makes it hard to implement the work directly. Later, in [Criado et al., 2014], 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 an uncertain environment. To accommodate this, norms have an associated salience to reflect their importance in the given uncertain environment.

Implementations that deal with uncertain percepts in a BDI setting [Chen et al., 2013] have not been based on the graded BDI framework but approached the problem more pragmatically. Our work extends upon the ideas of graded beliefs (i.e. what the agent knows), where we allow more fine-grained control by dividing the beliefs into isolated parts, each with their own representation and revision strategies. Contrary to graded BDI, our work has a vested interest in the feasibility of implementations while still providing strong theoretical underpinnings. In that sense, our work is close to the spirit of CANPLAN.

7 CONCLUSIONS

In this paper we showed how operational semantics for a BDI agent can be devised to deal with uncertain beliefs and actions affected by various forms of uncertainty. We introduced CANPLAN+, an extension of CANPLAN, in which we introduce a novel way of representing the agent's beliefs as a set of epistemic states. We furthermore introduced the idea of stratifying the domains of epistemic states. This allows an agent to reason about the plausibility of his beliefs within a local epistemic state and allows commensurability over the evaluation of these local results. As such, an agent can select more appropriate plans and can revise his current beliefs with uncertain information from the environment. In addition, it gives a BDI agent system designer the freedom to choose the best representation for the beliefs at hand. We also established a way to model actions triggered by uncertain beliefs, have uncertain effects and have effects that may introduce extra uncertainty into the beliefs. Finally, we extended the $Plan(\cdot)$ action from CANPLAN to allow a BDI agent to plan for the most optimal plan, i.e. with the highest chance of achieving the goal.

For future work, we plan to develop complete algorithms as well as approximate/tractable algorithms to use in BDI implementations that model and allow to reason about the uncertain beliefs of an agent. Moreover, we want to explore how existing planners under uncertainty can be extended to deal with the various forms of uncertainty faced in a SCADA environment.

Acknowledgements

This research is supported by the UK EPSRC project EP/J012149/1.

References

[Boyer, 2009] Boyer, S. (2009). *SCADA: Supervisory Control And Data Acquisition*. International Society of Automation.

[Bratman, 1987] Bratman, M. (1987). *Intention, Plans, and Practical Reason*. Harvard University Press.

[Casali et al., 2005] Casali, A., Godo, L., and Sierra, C. (2005). Graded BDI models for agent architectures. In *Proc. of CLIMA'04*, pages 126–143.

[Casali et al., 2011] Casali, A., Godo, L., and Sierra, C. (2011). A graded BDI agent model to represent and reason about preferences. *Artificial Intelligence*, 175(7–8):1468–1478.

[Chen et al., 2013] Chen, Y., Hong, J., Liu, W., Godo, L., Sierra, C., and Loughlin, M. (2013). Incorporating PGMs into a BDI architecture. In *Proc. of PRIMA'13*, pages 54–69.

[Criado et al., 2014] Criado, N., Argente, E., Noriega, P., and Botti, V. (2014). Reasoning about norms under uncertainty in dynamic environments. *International Journal of Approximate Reasoning*. In press.

[Darwiche and Goldszmidt, 1994] Darwiche, A. and Goldszmidt, M. (1994). On the relation between kappa calculus and probabilistic reasoning. In *Proc. of UAI'94*, pages 145–153.

[Dastani, 2008] Dastani, M. (2008). 2APL: a practical agent programming language. *Autonomous Agents and Multi-Agent Systems*, 16(3):214–248.

[de Silva and Padgham, 2005] de Silva, L. and Padgham, L. (2005). Planning on demand in BDI systems. In *Proc. of ICAPS'05*, pages 37–40.

[Di Pierro and Wiklicky, 1998] Di Pierro, A. and Wiklicky, H. (1998). An operational semantics for probabilistic concurrent constraint programming. In *Proc. of ICCL'98*, pages 174–183.

[Dubois et al., 1994] Dubois, D., Lang, J., and Prade, H. (1994). Possibilistic logic. *Handbook of Logic for Artificial Intelligence and Logic Programming*, 3(1):439–513.

[Dubois and Prade, 1995] Dubois, D. and Prade, H. (1995). Possibility theory as a basis for qualitative decision theory. In *Proc. of IJCAI'95*, pages 1924–1930.

[Ingrand et al., 1992] Ingrand, F., Georgeff, M., and Rao, A. (1992). An architecture for real-time reasoning and system control. *IEEE Expert: Intelligent Systems and Their Applications*, 7(6):34–44.

[Ma and Liu, 2011] Ma, J. and Liu, W. (2011). A framework for managing uncertain inputs: An axiomization of rewarding. *International Journal of Approximate Reasoning*, 52(7):917–934.

[McArthur et al., 2007] McArthur, S., Davidson, E., Catterson, V., Dimeas, A., Hatziargyriou, N., Ponci, F., and Funabashi, T. (2007). Multi-agent systems for power engineering applications – part i: Concepts, approaches, and technical challenges. *IEEE Transactions on Power Systems*, 22(4):1743–1752.

[Meneguzzi et al., 2007] Meneguzzi, F. R., Zorzo, A. F., da Costa Móra, M., and Luck, M. (2007). Incorporating planning into BDI systems. *Scalable Computing: Practice and Experience*, 8(1).

[Rao, 1996] Rao, A. (1996). Agentspeak(l): BDI agents speak out in a logical computable language. In *Proc. of MAAMAW'96*, pages 42–55.

[Rao and Georgeff, 1991] Rao, A. and Georgeff, M. (1991). Modeling rational agents within a BDI-architecture. In *Proc. of KR'91*, pages 473–484.

[Sardiña et al., 2006] Sardiña, S., de Silva, L., and Padgham, L. (2006). Hierarchical planning in BDI agent programming languages: a formal approach. In *Proc. of AAMAS'06*, pages 1001–1008.

[Sardiña and Padgham, 2011] Sardiña, S. and Padgham, L. (2011). A BDI agent programming language with failure handling, declarative goals, and planning. *Autonomous Agents and Multi-Agent Systems*, 23(1):18–70.

[Spohn, 1988] Spohn, W. (1988). Ordinal conditional functions: A dynamic theory of epistemic states. In *Causation, Coherence, and Concepts: Proceedings of Proceedings of the Irvine Conference on Probability and Causation*, pages 105–134.

[Walczak et al., 2006] Walczak, A., Braubach, L., Pokahr, A., and Lamersdorf, W. (2006). Augmenting BDI agents with deliberative planning techniques. In *Proc. of ProMAS'06*, pages 113–127.

[Winikoff et al., 2002] Winikoff, M., Padgham, L., Harland, J., and Thangarajah, J. (2002). Declarative & procedural goals in intelligent agent systems. In *Proc. of KR'02*, pages 470–481.

[Zhi et al., 2000] Zhi, L., Qin, J. S., Yu, T. B., Hu, Z. J., and Hao, Z. (2000). The study and realization of scada system in manufacturing enterprises. In *Proc. of WCICA'00*, volume 5, pages 3688–3692.