# AgentSpeak$^+$: AgentSpeak with Probabilistic Planning

Yingke Chen*, Kim Bauters*, Weiru Liu*, Jun Hong*, Kevin McAreavey*, Lluís Godo*† and Carles Sierra*†
*Queen's University Belfast (QUB), Belfast, United Kingdom
Email: {k.bauters, w.liu, j.hong, kevin.mcareavey}@qub.ac.uk
†IIIA, CSIC, Bellaterra, Spain
Email: {godo, sierra}@iiia.csic.es

***Abstract*—AgentSpeak is a logic-based programming language, based on the Belief-Desire-Intention (BDI) paradigm, suitable for building complex agent-based systems. To limit the computational complexity, agents in AgentSpeak rely on a plan library to reduce the planning problem to the much simpler problem of plan selection. However, such a plan library is often inadequate when an agent is situated in an uncertain environment. In this paper, we propose the AgentSpeak$^+$ framework, which extends AgentSpeak with a mechanism for probabilistic planning. The beliefs of an AgentSpeak$^+$ agent are represented using epistemic states to allow an agent to reason about its uncertain observations and the uncertain effects of its actions. Each epistemic state consists of a POMDP, used to encode the agent's knowledge of the environment, and its associated probability distribution (or belief state). In addition, the POMDP is used to select the optimal actions for achieving a given goal, even when facing uncertainty.**

## I. Introduction

Using the Belief-Desire-Intention (BDI) agent architecture [1], we can develop complex systems by treating the various system components as autonomous and interactive agents [2]. The beliefs determine the desires that are achievable, the desires are the goals an agent wants to achieve and the intentions are those desires the agent is acting upon. A number of successful agent-oriented programming languages have been developed based on this architecture, such as AgentSpeak [3] and Can [4]. Notable BDI implementations include, for example, JASON [5] and JADEX [6]. The benefits of the BDI model in scalability, autonomy and intelligence have been illustrated in various application domains such as control systems [2]. Key to the efficiency of BDI agents is the use of a set of pre-defined plans, which simplify the planning problem to an easier plan selection problem. However, obtaining a plan library that can cope with every possible situation requires adequate domain knowledge. This knowledge is not always available, particularly when dealing with uncertain situations. As such, when faced with uncertainty, an autonomous and intelligent agent should resort to other forms of planning to make rational decisions.

To illustrate the problems, consider the example shown in Figure 1. A truck needs to collect materials from three different factories, each producing a distinct type of material that may or may not be available (i.e. the environment is *stochastic*). The truck needs to collect all materials by visiting each factory while limiting costs (e.g. fuel). The truck agent is uncertain as to whether the material in a factory is ready to collect, but it can use previous experience to estimate a degree of belief. To further complicate the situation, the truck agent can only infer its location by observing nearby signposts (e.g. the agent is near a supermarket or a petrol station). Travelling between factories may also fail (i.e. non-deterministic actions).

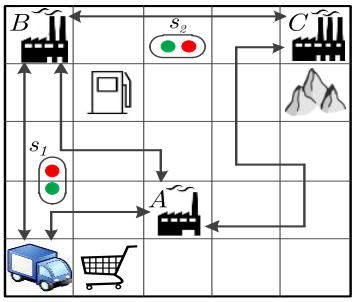


Fig. 1. The material collection scenario.

The large number of possibilities make a pre-defined plan library infeasible, even in this small example. We address these issues by combining AgentSpeak with Partially Observable Markov Decision Processes (POMDPs). POMDPs are a framework for probabilistic planning [7], and are often used as a decision theory model for agent decision making. Other frameworks, such as probabilistic graphplan [8], only consider the *uncertain effects* of actions. Similarly, the *partial observability* of the *stochastic* environment is not addressed by approaches such as probabilistic Hierarchical Task Networks (HTN) [9] and Markov Decision Processes (MDPs) [10]. As such, POMDPs seem to offer an elegant solution to deal with examples such as the one discussed above. In particular, when optimal solutions are required (e.g. the truck wants to collect as many materials subject to the fuel limit), POMDPs can be used to compute these solutions. However, even though efficient algorithms to compute the optimal policy (e.g., [11]) exist, POMDPs are still computationally intensive.

By integrating POMDPs into a BDI architecture, we retain the scalability of the BDI architecture, while adding to it the ability to model uncertainty and as well as on-demand access to the optimal actions (provided by the POMDP component). The framework we propose is called AgentSpeak$^+$. In this framework we introduce the concept of epistemic states [12] to the BDI framework. These epistemic states are used to model the beliefs of an agent about uncertain information, along with information on how these beliefs evolve over time. To achieve this, POMDPs are embedded into the agent's

epistemic states and are used to represent some aspects of the agent's domain knowledge. When needed, the optimal actions generated by these POMDPs can be fed into the agent's plan execution. Therefore, alongside the traditional trigger-response mechanism based on pre-defined plans in BDI, an AgentSpeak$^+$ agent also has the ability to deal with uncertainty and to take optimal actions when dealing with an uncertain and partially observable environment.

The remainder of the paper is organised as follows. Related work is discussed in Section II. Preliminary notions on AgentSpeak and POMDPs are mentioned in Section III. In Section IV we propose the AgentSpeak$^+$ architecture which integrates POMDPs into AgentSpeak. In Section V a scenario described using AgentSpeak$^+$ is discussed and in Section VI we conclude the paper.

## II. Related Work

There have been several approaches to modelling uncertainty in multi-agent systems. In [13] the degree of belief of a BDI agent is quantified using Dempster-Shafer theory, while in [14] a graded BDI approach is proposed that uses uncertain beliefs (as probabilities) and graded preferences (as expected utilities) to rank plans. However, the theoretical nature of both works have so far precluded the development of any practical implementations. In this paper, we instead propose an extension based on a widely used agent-oriented programming language. Our extension is based on epistemic states, similar to [15] where the concept of an epistemic state is introduced to model uncertain perceptions. Similar to their work, Boolean belief atoms (e.g. propositional statements) can be derived from epistemic states to support further reasoning. However, the work in [15] only focuses on MDPs, i.e. it cannot be used in the more natural setting of partially observable environments.

In [16] a theoretical comparison of POMDPs and the BDI architecture identifies a correspondence between desires and intentions on the one hand, and rewards and policies on the other. In addition, the performance and scalability of (PO)MDPs and the BDI architecture are compared in [17]. The authors of [17] conclude that while (PO)MDPs exhibit better performance when the domain size is small, they do not scale well since the state space grows exponentially. The BDI architecture (which uses a pre-defined plan library), however, has better scalability at the cost of optimality and can be successfully used to plan over significantly larger state spaces.

There has also been some work in the literature on hybrid BDI-POMDP approaches. For example, in [18] an algorithm was proposed to build AgentSpeak plans from optimal POMDP policies. However, most characteristics of the original BDI framework are not retained in such hybrid approaches. In contrast, our proposed approach embeds POMDPs in the traditional BDI agent framework. Normal BDI execution is used by default, with the POMDP component allowing an agent to generate new plans on-demand during execution. Extending the BDI architecture with more elaborate planning techniques has also been investigated in the literature. In [4], the authors present a formal framework to integrate planning into BDI, called CanPlan. Similarly, in [19], the authors integrate classical planning problems into the BDI interpreter to allow an agent to respond to unforeseen scenarios. However, neither approach considers the issues of uncertainty.

## III. Preliminaries

We start with some preliminaries on AgentSpeak and POMDP.

*1) AgentSpeak:* We first define how an agent program can be written. We use $\mathcal{S}$ to denote a finite set of symbols for predicates, actions, and constants, and $\mathcal{V}$ to denote a set of variables. Following convention, elements from $\mathcal{S}$ are written using lowercase letters and elements from $\mathcal{V}$ using uppercase letters. We use the standard first-order logic definition of a term and we use $\mathbf{t}$ as a compact notation for $t_1, \ldots, t_n$, i.e. a vector of terms. We have [3]:

*Definition* 1. If $b$ is a predicate symbol, and $\mathbf{t}$ are terms, then $b(\mathbf{t})$ is a *belief atom*. If $b(\mathbf{t})$ and $c(\mathbf{s})$ are belief atoms, then $b(\mathbf{t})$, $\neg b(\mathbf{t})$, and $b(\mathbf{t}) \wedge c(\mathbf{s})$ are *beliefs*.

*Definition* 2. If $g(\mathbf{t})$ is a belief atom, then $!g(\mathbf{t})$ and $?g(\mathbf{t})$ are goals with $!g(\mathbf{t})$ an *achievement goal* and $?g(\mathbf{t})$ a *test goal*.

*Definition* 3. If $p(\mathbf{t})$ is a belief atom or goal, then $+p(\mathbf{t})$ and $-p(\mathbf{t})$ are *triggering events* with $+$ and $-$ denoting the addition and deletion of a belief/goal, respectively.

*Definition* 4. If $a$ is an action symbol and $\mathbf{t}$ are terms, then $a(\mathbf{t})$ is an *action*.

*Definition* 5. If $e$ is a triggering event, $h_1, \ldots, h_m$ are beliefs and $q_1, \ldots, q_n$ are goals or actions, then $e : h_1 \wedge \ldots \wedge h_m \leftarrow q_1, \ldots, q_n$ is a *plan*. We refer to $h_1 \wedge \ldots \wedge h_m$ as the *context* of the plan and to $q_1, \ldots, q_n$ as the *plan body*.

On the semantic level, the state of an AgentSpeak agent $\mathbb{A}$ can be described by a tuple $\langle \mathsf{BB}, \mathsf{PLib}, \mathsf{E}, \mathsf{A}, \mathsf{I} \rangle$, with $\mathsf{BB}$ the belief base (treated as a set of ground belief atoms), $\mathsf{Plib}$ the plan library, $\mathsf{E}$ the event stack, $\mathsf{A}$ the action set and $\mathsf{I}$ the intention stack [3]. Intuitively, when an agent reacts to a new event $e$, it selects those plans that have $e$ as the triggering event. We say that a plan is *applicable* when the context of the plan evaluates to true according to the belief base of the agent. For a given event $e$ there may be many applicable plans, one of which is selected and added to the intention stack. The intention stack is a stack of those plans that are currently being executed, i.e. those desires that the agent has chosen to pursue. Each intention is executed by performing the actions in the plan body, which may in turn change the environment and/or the agent's beliefs. The execution of an intention may then also generate new internal events (or subgoals).

*2) POMDP:* Partially Observable Markov Decision Processes (POMDPs) have gained popularity as a computational model for solving probabilistic planning problems in a *partially observable* and *stochastic* environment (see e.g., [7]).

*Definition* 6. A POMDP $\mathcal{M}$ is a tuple $\mathcal{M} = \langle S, A, \Omega, R, T, O \rangle$ where $S$, $A$, and $\Omega$ are sets of states, actions, and observations, respectively. Furthermore, $R : S \times A \rightarrow \mathbb{R}$ is the reward function[1], $T : S \times A \rightarrow \Delta(S)$ is the transition function and $O : S \times A \rightarrow \Delta(\Omega)$ is the observation function. Here, $\Delta(\cdot)$ is the space of probability distributions.

Instead of knowing the current state exactly, there is a probability distribution over the state space $S$, called the *belief state*[2] $b(S)$, with $b(s)$ the probability that the current state is $s$. When $S$ is clear from the context, we simply write $b$ instead of $b(S)$.

[1] When depending on the resulting state, it is defined as $R : S \times A \times S \rightarrow \mathbb{R}$.

[2] Not to be confused with the *belief base* of an agent, which we see later.

In Definition 6, we have that the *Markovian* assumption is encoded in the transition function since the new state depends only on the previous state. Given the belief state $b_t$ at time $t$, after performing action $a$ and receiving observation $o$, the new belief state $b_{t+1}$ is obtained using the Bayesian rule:

$$b_{t+1}(s) = P(s \mid b_t, a, o) = \frac{O(o \mid s, a) \cdot \sum_{s' \in S} T(s \mid s', a) \cdot b_t(S = s')}{P(o \mid b_t, a)} \quad (1)$$

where $P(o \mid b_t, a)$ is a normalisation factor obtained by marginalising $s$ out as follows:

$$P(o \mid b_t, a) = \sum_{s \in S} O(o \mid s, a) \cdot \sum_{s' \in S} T(s \mid s', a) \cdot b_t(S = s').$$

The decision $a$ at horizon $t$ takes into account both the instant reward as well as all possible rewards in future decision horizons (of the POMDP execution). Given a POMDP $\mathcal{M}$, its policy $\pi : \mathfrak{B} \to A$ is a function from the space of belief states (denoted as $\mathfrak{B}$) to the set of actions. The policy provides the optimal action to perform for a given belief state at each decision horizon, i.e. it is the action that should be performed in the current belief state to maximise the expected reward.

*Definition* 7 (Optimal Policy). Given a POMDP $\mathcal{M}$ with the initial belief state $b_1$, $\pi^*$ is an optimal policy over the next $H$ decision horizons if it yields the highest cumulated expected reward value $V^*$:

$$V^*(b_1) = \sum_{t=1}^{H} \gamma^{t-1} \cdot R(s, \pi^*(b_t)) \cdot b_t(s)$$

where $b_t(s)$ is updated according to equation (1) and $\gamma \in (0, 1]$ is a discounting factor to ensure that future rewards are lower. Here $\pi^*(b_t)$ is the action determined by policy $\pi^*$ and the belief state $b_t$.

A probabilistic planning problem is then defined as the problem of finding the optimal actions for a given POMDP $\mathcal{M}$ and an initial belief state $b$ (where the optimal actions in a POMDP setting are described using a policy $\pi$).

## IV. Integration of AgentSpeak and POMDPs

We now discuss how AgentSpeak and POMDP can be integrated into a single framework. The resulting framework, called AgentSpeak$^+$, allows us to define agents that can perform on-demand planning based on the POMDP to provide optimal decisions in an uncertain environment. We start by introducing the concept of epistemic states to model the uncertain beliefs of an agent. We define an epistemic state as containing both a POMDP $\mathcal{M}$ and a belief state $b$, where the former encodes the agent's domain knowledge about the partially observable environment and the latter represents the current uncertain information about the states modelled in $\mathcal{M}$.

### A. Epistemic States

*Definition* 8 (Epistemic states). Let $\mathcal{M}$ be a POMDP which models the situated partially observable stochastic environment. By definition, $\mathcal{M}$ includes a set of states $S$. The epistemic state $\Phi$ over the state space $S$ is defined as $\Phi = \langle b, \mathcal{M} \rangle$, where $b : S \to [0, 1]$ is a probability distribution over $S$.

*Example* 1. Let $\Phi = \langle b, \mathcal{M} \rangle$ be an epistemic state. The state space $S$ in $\mathcal{M}$ is {locA, locB}, i.e. two possible locations. The current belief state is given by $b(\mathsf{locA}) = 0.6$ and $b(\mathsf{locB}) = 0.4$, which will change based on the actions performed by the agent (e.g. probably in locB if you move from locA) and its observations (e.g. probably in locA when you see a supermarket). The POMDP $\mathcal{M}$ encoding the relevant knowledge is graphically illustrated in Figure 2.

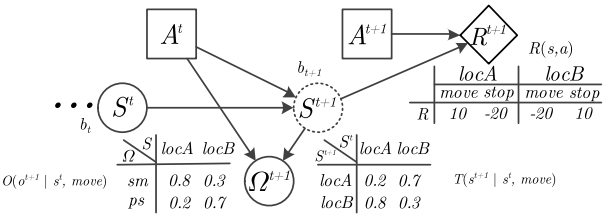


Fig. 2. Graphical representation of the POMDP $\mathcal{M}$. We have that $S$ = {locA, locB}, $\Omega$ = {(s)uper(m)arket, (p)etrol (s)tation} and $A$ = {move, stop}. The transition, observation and reward functions are shown as tables.

Each epistemic state contains all knowledge needed to plan over and reason about a well-defined subset of the environment. Given that each epistemic state has an optimal policy, this optimal policy intuitively encodes a subplan for this subset of the environment. Throughout the paper, we assume that a corresponding symbol $\Phi$ is available on the syntactic level as well, i.e. on the level of an AgentSpeak agent we are able to refer to a specific epistemic state. An executed action and a newly obtained observation are together taken as new input to revise the epistemic state using the belief updating process in Equation (1), where revision is defined as:

*Definition* 9 (Epistemic state revision). Let $\Phi = \langle b, \mathcal{M} \rangle$ be an epistemic state and $I = \langle a, o \rangle$ an input with $a \in A$ an action and $o \in \Omega$ an observation. The revision of $\Phi$ by $I$, denoted as $\Phi \circ I$, is defined as $\Phi \circ I = \langle b', \mathcal{M} \rangle$ with $\circ$ a revision operator. Particularly, $\circ$ is given by Equation (1). The result of revision is a new epistemic state, where the new belief state $b'$ is determined based on the old belief state $b$ and the input $I$.

*Example* 2 (Example 1 cont'd). Let $\Phi = \langle b, \mathcal{M} \rangle$ be the epistemic state from Example 1 with the belief state $b$. After performing action move, and receiving a new observation ps, the revised probability distribution over possible locations is $b'(\mathsf{locA}) = 0.16$ and $b'(\mathsf{locB}) = 0.84$.

It is important to note that revision will only revise the belief state, while keeping the corresponding POMDP unchanged (i.e. revision does not alter the domain knowledge of the agent in this case). When there are a sequence of inputs $I_1, \ldots, I_n$ the epistemic state is simply revised iteratively. Furthermore, we assume that an agent can have multiple epistemic states, each dealing with a localised and isolated part of the beliefs of the agent. For example, the availability of material at each factory is independent of the colour of the traffic light. Localised epistemic states allow an agent to revise a corresponding epistemic state given a new input without affecting other epistemic states[3]. This reflects, to some extent, the notion of *minimal change principle* in belief revision.

[3]For simplicity, we restrict ourselves in this paper to the case where each input is relevant to only one epistemic state.

We will also allow the belief state (i.e. the beliefs maintained by a POMDP) to be extracted from the agent's *belief base* (i.e. the component of an AgentSpeak agent where beliefs are stored). This is useful for designing an AgentSpeak$^+$ agent, as it allows the automatic extraction of the belief state of the POMDP from the AgentSpeak$^+$ program.

*Definition* 10 (Correlated belief atoms). Two belief atoms $h(x, m, \mathcal{M})$ and $h(x', m', \mathcal{M})$ are said to be correlated if $x$ and $x'$ are two states of variable $X_j$ (which is one of the variables in the joint state space $S$) defined in the POMDP $\mathcal{M}$, where $m$ and $m'$ are their corresponding probability values.

*Definition* 11 (Extraction). Let $\{h(x_i, m_i, \mathcal{M}) \mid i \in 1, \ldots, k\}$ be a set of exhaustively correlated belief atoms for variable $X_j$. The belief state $b(x_i) = m_i$ can be directly derived from this set iff $\sum_{i=1}^{k} m_i = 1$ and $X_j = \{x_1, \ldots, x_k\}$.

When the state space $S$ of a POMDP $\mathcal{M}$ has a set of variables $\{X_1, \ldots, X_n\}$, then the belief state $b(S)$ is the joint probability distribution obtained from $b(X_j)$. When the belief state cannot be extracted from an agent's initial beliefs we assume a default probability distribution, i.e. a uniform distribution, for the belief state. Finally, whenever an epistemic state is initiated or revised the belief base of the agent will be updated accordingly using the corresponding triggering events $-h(x_i, m_i, \mathcal{M})$ and $+h(x_i, m_i', \mathcal{M})$ in AgentSpeak.

*Example* 3 (Example 2 cont'd). The belief state $b$ can be modelled as the belief atoms $\mathsf{location}(\mathsf{locA}, 0.6, \mathcal{M})$ and $\mathsf{location}(\mathsf{locB}, 0.4, \mathcal{M})$. The revised belief state $b'$ is represented as $\mathsf{location}(\mathsf{locA}, 0.08, \mathcal{M})$ and $\mathsf{location}(\mathsf{locB}, 0.92, \mathcal{M})$.

### B. Probabilistic Planning

Now that we have defined an epistemic state (which can deal with uncertainty) and how to revise it, we look at how probabilistic planning can be integrated into AgentSpeak. The POMDPs we use in the epistemic state allow us to decide the optimal action at each decision horizon by taking into account the immediate expected reward and the future rewards. However, simply computing the optimal plan at each step would severely hamper the reactiveness of the AgentSpeak agent due to the computational cost. Instead, we introduce a new action to AgentSpeak, ProbPlan, which can be used in AgentSpeak plans to explicitly compute the optimal action to achieve a goal for a given epistemic state $\mathcal{M}$. This enables the agent to react optimally when needed, e.g. for when performing the wrong action likely carries a high penalty. When optimality is not required or when reactiveness is of primary importance, the agent can instead rely on the abstractness and high performance of the normal BDI plan selection strategy.

*Definition* 12 (Probabilistic planning action). Let ProbPlan be an ordinary AgentSpeak action symbol and $\Phi = \langle b, \mathcal{M} \rangle$ an epistemic state. We say that ProbPlan($\Phi$, $H$) is a *probabilistic planning action*, with $H$ the number of steps and corresponding rewards we should consider (i.e. $H$ is the horizon). The effect of executing ProbPlan($\Phi$, $H$) is that the probabilistic planning problem defined by a POMDP $\mathcal{M}$ with initial belief state $b$ and horizon $H$ is solved, after which the optimal action $a_i \in A$ is executed.

Importantly, the action set $A$ defined in a POMDP can contain both primitive actions and *compound* actions (i.e. subgoals), each representing different levels of planning granularity. In the latter case, a pre-defined plan in AgentSpeak will be triggered to pursue the goal corresponding to the optimal action $a_i$. This allows the optimal plan to be as specific as possible (to reach the goal without taking excess steps) while being as abstract as possible (to fully take effect of the domain knowledge encoded in the set of pre-defined plans). The effects of these compound actions can be computed either cautiously (i.e. only considering effects shared by all relevant plans) or bravely, which allows us to balance optimality and reactiveness for the given problem. In addition, it should be noted that while the result of ProbPlan($\Phi$, $H$) is an optimal action at the time of computation, there is no guarantee that this action will still be optimal during execution. Indeed, the optimal action/subgoal is not (by default) immediately executed and may be intertwined with the execution of other subgoals which alter the environment. The benefit of not enforcing this optimality but rather *trying* to be optimal is that we retain the reactiveness of BDI and are able to fully use the knowledge already encoded by the system developer in the subgoals.

For the running example, we consider a POMDP with the state space $\{O, A, B, C\}$, i.e. the origin location $O$ and three factories $A$, $B$ and $C$. In addition, there is an action set consisting of 9 subgoals: 3 subgoals to go from the origin to a factory; and 6 subgoals to go from one factory to another. In all cases, the subgoals consist of both going to the location as well as collecting the corresponding material. For example, we will use goOBcollect to denote the subgoal to move from the origin to factory B in order to collect material B.

*Definition* 13 (Probabilistic planning plan). A plan pl is called a *probabilistic planning plan* if it contains at least one probabilistic planning action ProbPlan in the plan body.

Due to the fact that each probabilistic planning problem defined on $\mathcal{M}$ always has a (not necessarily unique) optimal action, a probabilistic planning plan does not introduce infinite recursion. Furthermore, an optimisation can be applied to reduce the computational cost. Indeed, whenever the first optimal action is decided, a complete optimal policy over $H$ decision horizons has already been constructed as part of the probabilistic planning problem. Before deliberating over the next action, the epistemic state will be revised with the current action and the new observation as defined in Definition 9. Given the revised epistemic state, the next optimal action can then be decided instantly based on the optimal policy that is already constructed without requiring extra computation.

*Example* 4. Consider the truck agent $\mathbb{A}_t$ from the running example where $\Phi$ is the relevant epistemic state. We have:

```
P1: +!collectMaterial : true ← ProbPlan(Φ, 3);
                               ProbPlan(Φ, 2);
                               ProbPlan(Φ, 1).
P2: +!goOBcollect : true ← moveOtoS1; !waitS1toB; moveS1toB;
                           senseLocation; !load(b).
```

The first plan describes how the truck agent can collect all the materials, i.e. how it can achieve its goal !collectMaterial. Due to some hard constraint (e.g. we only have limited fuel) we rely on the POMDP planning to plan ahead and figure out the best course of action. Since the abstract level considered by the POMDP in the epistemic state $\Phi$ can move from one factory to another in a single step, we consider a decision horizon

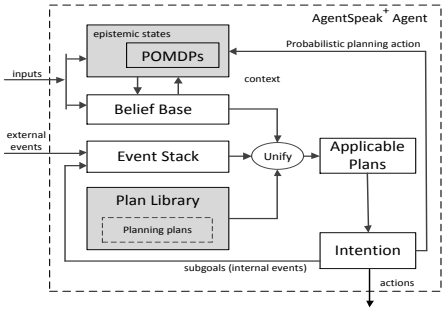


Fig. 3. The revised reasoning cycle for an AgentSpeak$^+$ agent.

of 3. The result of ProbPlan($\Phi$, 3) can for example be the subgoal goOAcollect, i.e. given all the information available to POMDP at the moment, the optimal action is to first visit factory A. During the execution of this subgoal new observations will be collected (e.g. through senseLocation)and will be taken into account when deliberating over ProbPlan($\Phi$, 2) by (implicitly) using the revised epistemic state to find the optimal action/subgoal to collect the remaining two materials.

### C. AgentSpeak$^+$

We are now ready to define our AgentSpeak$^+$ framework:

*Definition* 14 (AgentSpeak$^+$ agent). An AgentSpeak$^+$ agent $\mathbb{A}^+$ is defined as a tuple $\langle \mathsf{BB}^+, \mathsf{EpS}, \mathsf{PLib}^+, \mathsf{E}, \mathsf{A}, \mathsf{I} \rangle$, where the belief base $\mathsf{BB}^+$ now contains belief atoms with an associated probability value, EpS is a set of epistemic states, the plan library $\mathsf{PLib}^+$ contains an additional set of probabilistic planning plans, and E, A and I are as before.

Normally, in AgentSpeak, the context of a plan consists of classical belief atoms. However, in AgentSpeak$^+$ the belief base contains uncertain belief atoms, i.e. we need a way to determine if a given context is *sufficiently plausible*.

*Definition* 15 (Belief entailment). Let $\{h(x_i, m_i, \mathcal{M}) \mid i \in 1, \ldots, k\}$ be a set of exhaustively correlated belief atoms in an agent belief base. The belief atom $h'(x_i)$ is entailed by the agent's belief base BB iff there exists $h(x_i, m_i, \mathcal{M}) \in \mathsf{BB}$ such that $m_i \geq \delta$ with $0 < \delta \geq 1$. The value $\delta$ is context-dependent, and reflects the degree of uncertainty we are willing to tolerate.

*Example* 5 (Example 3 cont'd). The revised belief base contains the belief atoms location(locA, 0.08, $\mathcal{M}$) and location(locB, 0.92, $\mathcal{M}$). Given a threshold $\delta_{\mathsf{Loc}} = 0.9$, only the belief atom location(locB) is entailed.

Notice that we can straightforwardly represent classical belief atoms by associating a probability of 1 with them. Verifying if a context is entailed, *i.e.* a conjunction of belief literals, is done classically based on the entailed belief atoms. As such, we recover classical AgentSpeak entailment of contexts if we enforce that belief entailment is only possible when $\delta = 1$. The revised reasoning cycle of AgentSpeak$^+$ agent is shown in Figure 3. The agent now contains a set of epistemic states, each of which includes a POMDP. A new input can either revise the belief state of a POMDP or be inserted into the agent's belief base BB (i.e. this happens when the input is not related to any of the agent's epistemic states). As needed, during plan execution, the agent can furthermore rely on the POMDP to compute the optimal next step through the use of a probabilistic planning action. Whenever the selected plan (i.e. the one that has been committed as an intention) contains a probabilistic planning action, the corresponding POMDP will be called instead of (directly) relying on the plan library.

## V. Scenario Discussion

We now show how the scenario from the introduction can be expressed using the AgentSpeak$^+$ framework.[4] Even though a prototype implementation has been developed, due to space restrictions we do not evaluate the performance of our implementation against, e.g. a pure POMDP implementation, and instead leave this for future work. Instead, we restrict ourselves to the material collection scenario (which relies heavily on optimal planning) in order to illustrate the benefits offered by AgentSpeak$^+$ over AgentSpeak.

We recall that the goal of our truck agent $\mathbb{A}_t$ is to collect the materials A, B and C from resp. factories A, B and C. We consider a single epistemic state $\Phi$ where the POMDP $\mathcal{M}$ has an action set of 9 subgoals, as discussed earlier in the paper. Each state in the POMDP $\mathcal{M}$ contains information about whether a factory has material available (denoted as FA, FB and FC), whether our agent has already collected the material (denoted as MA, MB and MC) and the location of the agent. For example, the state $s = \{\mathsf{MA}, \mathsf{locA}, \mathsf{FA}, \mathsf{FB}\}$ indicates that the agent has collected material $A$, is currently in factory $A$ and that material is available from factories $A$ and $B$. We define $\mathcal{M}$ in a graphical way to further decompose the state space (e.g. whether one type of material is available at one factory is independent from the other factories). A Dynamic Influence Diagram (DID) [20] $\mathcal{D}$ is obtained for efficient computation (shown in Figure 4). In particular, we decompose the entire state space into a set of chance nodes for FA, FB, FC, MA, MB, MC, and Loc. Correspondingly, belief atoms in our agent $\mathbb{A}_t$ describe the availability (has(X)), the loaded materials (loaded(X)) and the location of the agent ((at(X)) for material/factory $X \in \{a, b, c\}$. As discussed at the end of Section IV-A, a revision to the epistemic state is triggered when a belief atom is added/removed and possible additions/deletions of belief atoms are triggered when the epistemic state is revised.

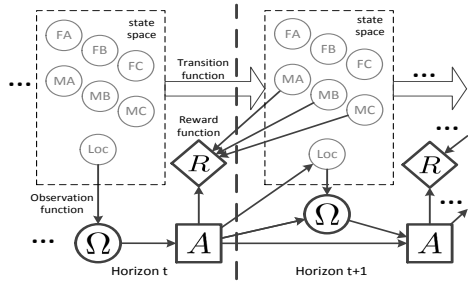


Fig. 4. Graphical representation of POMDP $\mathcal{M}_g$. Here, the entire state space is decomposed into a set of chance nodes. Some causality links for the transition function are omitted for simplicity.

[4]The framework has been implemented by extending Jason [5], an open-source implementation of AgentSpeak. Epistemic states are defined using Java (by extending the Jason interpreter functions) and the corresponding POMDP is constructed using Hugin [20]. During execution, the required POMDP is called through the Hugin API by the Jason interpreter and returns the recommended optimal decision(s).

The plan library for our AgentSpeak$^+$ agent $\mathbb{A}_t$ contains many plans, including:

```
(P3)  +!start : true ← !callFactories, !collectMaterial.
(P4)  +!callFactories : true ← !check(a), !check(b),
                               !check(c).
(P5a) +check(X) : not loaded(X) ← call(X).
(P5b) +check(X) : loaded(X).
(P6a) +!waitS1toB : not s1green ← senseSignal; !waitS1toB.
(P6b) +!waitS1toB : s1green.
(P7)  +!load(X) : at(X) ← pay(X); getMaterial(X),
                          !callFactories.
```

Whenever the agent is activated, the event start is triggered. The plan (P3) deals with this event by calling the factories to check if they have material available (using the domain knowledge encoded in (P5) to only call when material is not yet loaded) and then proceed with collecting those materials. Material collection was previously described in (P1) from Example 4 where it involves a probabilistic planning action, using the POMDP to find the optimal plan to execute. Such a plan can be as described in (P2) from the same example. The plan body of (P2) consists of moving the agent to the signal, applying the domain knowledge in (P6) to determine when it is safe to pass the signal, and then proceed to factory B. Once the agent has completed this action it checks whether it has successfully reached $B$. Once the location is sensed, and the agent is at the required factory, it proceeds to plan (P7) to load the required material. First, e.g. the factory is paid, then the material is loaded and finally the domain knowledge is applied to call those factories from which it still needs to collect material. Importantly, the results of these subgoals (such as moving, loading and calling factories) modify the agent's beliefs and thus also modify the relevant epistemic state. These new beliefs (and uncertainties) are then taken into account in (P1) in order to determine the optimal subgoal for collecting the remaining materials.

This example highlights a number of key benefits offered by the AgentSpeak$^+$ framework. Compared to classical AgentSpeak, we are able to deal with uncertain information. Furthermore, our AgentSpeak$^+$ agent is not fully specified in the design phase; it resorts to probabilistic planning to deal with crucial parts of its execution (e.g. determining the order in which to visit the factories). Compared to a pure POMDP implementation, the AgentSpeak$^+$ framework considerably reduces the overall complexity by relying on domain knowledge encoded on the level of a BDI agent. As such, irrelevant actions such as determining which factories to call and how long to wait at a signal, are omitted from the POMDP dealing with the availability of materials. Furthermore, since planning only happens on-demand, the agent can rely on the simpler plan selection to ensuring maximum reactiveness for most of the subgoals (e.g. when truck agents also have to achieve goals where uncertainty is absent and/or optimality is not required).

## VI. Conclusions

In this paper we proposed the AgentSpeak$^+$ in which we extend the belief base of an agent by allowing it to be represented by one or more epistemic states. An essential part of each epistemic state is a POMDP, which allows us to model the domain knowledge of the partially observable environment and from which we can compute optimal actions when needed. In addition,this allows us to deal in a straightforward way with uncertain information in the environment. To keep the computational complexity low, AgentSpeak$^+$ extends AgentSpeak, which is an agent-programming language based on the BDI paradigm where planning is reduced to the simple task of plan selection. By adding actions to perform on-demand planning, the resulting AgentSpeak$^+$ can both offer good responsiveness while at the same time providing the option for near-optimal planning when needed through the POMDP component. For future work, we plan a full evaluation of our approach, both compared to classical BDI implementations and pure POMDP implementations. We furthermore plan an extension where knowledge from other agents (which are only trusted to some degree) can be employed to improve the domain knowledge currently encoded in the POMDP component.

## References


[1] A. S. Rao and M. P. Georgeff, "An abstract architecture for rational agents," in *Proc. of KR'92*, 1992, pp. 439–449.

[2] N. R. Jennings and S. Bussmann, "Agent-based control systems," *IEEE Control Systems Magazine*, vol. 23, pp. 61–74, 2003.

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

[4] S. Sardina and L. Padgham, "A BDI agent programming language with failure handling, declarative goals, and planning," *AAMAS*, vol. 23, no. 1, pp. 18–70, 2011.

[5] R. H. Bordini, J. F. Hübner, and M. Wooldridge, *Programming Multi-agent Systems in AgentSpeak using Jason*. Wiley-Interscience, 2007.

[6] L. Braubach, W. Lamersdorf, and A. Pokahr, "JADEX: Implementing a BDI-infrastructure for JADE agents," *EXPóin search of innovation*, vol. 3, no. 3, pp. 76–85, 2003.

[7] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *AI*, vol. 101, no. 1, pp. 99–134, 1998.

[8] A. L. Blum and J. C. Langford, "Probabilistic planning in the graphplan framework," in *Recent Advances in AI Planning*. Springer Berlin Heidelberg, 2000, pp. 319–332.

[9] F. Meneguzzi, Y. Tang, K. Sycara, and S. Parsons, "On representing planning domains under uncertainty," in *Proc. of ITA'10*, 2010.

[10] R. Bellman, "A markovian decision process," *Indiana University Mathematics Journal*, vol. 6, pp. 679–684, 1957.

[11] E. A. Hansen, "Solving POMDPs by searching in policy space," in *Proc. of UAI'98*, 1998, pp. 211–219.

[12] J. Ma and W. Liu, "A framework for managing uncertain inputs: An axiomization of rewarding," *IJAR*, vol. 52, no. 7, pp. 917–934, 2011.

[13] S. Parsons and P. Giorgini, "On using degrees of belief in BDI agents," in *Proc. of IPMU'98*, 1998.

[14] A. Casali, L. Godo, and C. Sierra, "A graded BDI agent model to represent and reason about preferences," *AI*, vol. 175, no. 7–8, pp. 1468–1478, 2011.

[15] Y. Chen, J. Hong, W. Liu, L. Godo, CarlesSierra, and M. Loughlin, "Incorporating PGMs into a BDI architecture," in *Proc. of PRIMA'13*, 2013, pp. 54–69.

[16] M. Schut, M. Wooldridge, and S. Parsons, "On partially observable MDPs and BDI models," in *Proc. of UKMAS'02*, 2002, pp. 243–260.

[17] G. I. Simari and S. D. Parsons, "On approximating the best decision for an autonomous agent," in *Proc. of GTDT'04*, 2004.

[18] D. R. Pereira, L. V. Gonçalves, G. P. Dimuro, and A. C. R. Costa, "Constructing BDI plans from optimal POMDP policies, with an application to agentspeak programming," in *Proc. of CLI'08*, 2008, pp. 240–249.

[19] F. Meneguzzi and M. Luck, "Declarative planning in procedural agent architectures," *Expert Systems with Applications*, vol. 40, no. 16, pp. 6508–6520, 2013.

[20] S. K. Andersen, K. G. Olesen, F. V. Jensen, and FrankJensen, "HUGIN - a shell for building bayesian belief universes for expertsystems," in *Proc. of IJCAI'89*, 1989, pp. 1080–1085.