Planning with Multi-agent Belief Using Justified Perspectives

Guang Hu, Tim Miller, Nir Lipovetzky

School of Computing and Information Systems The University of Melbourne
Parkville, VIC 3010, AUS
ghu1@student.unimelb.edu.au, tmiller@unimelb.edu.au, nir.lipovetzky@unimelb.edu.au

Abstract

Epistemic planning plays an important role in multi-agent and human-agent interaction domains. Most existing works solve multi-agent epistemic planning problems by either pre-compiling them into classical planning problems; or, using explicit actions and their effects to encode Kripke-based semantics. A recent approach called *Planning with Perspectives* (PWP) delegates epistemic reasoning in planning to external functions using F-STRIPS, keeping the search within the planning algorithm and lazily evaluating epistemic formulae. Although PWP is expressive and efficient, it models S5 epistemic logic and does not support belief, including false belief. In this paper, we extend the PWP model to handle multi-agent belief by following the intuition that agents believe something they have seen until they see otherwise. We call this *justified perspectives*. We formalise this notion of multi-agent belief based on the definition of knowledge in PWP. Using experiments on existing epistemic and doxastic planning benchmarks, we show that our belief planner can solve benchmarks more efficiently than the state-of-the-art baseline, and can model some problems that are infeasible to model using propositional-based approaches.

1 Introduction and Motivation

In epistemic planning problems, agents need to reason about the ontic world and the epistemic world. There is extensive research on epistemic logic reasoning and epistemic planning, each with its own strengths and limitations. Most epistemic planning approaches either explicitly maintain all epistemic relations, such as Kripke frames (Kominis and Geffner 2015; Bolander and Andersen 2011; Bolander 2014), or require an expensive pre-compilation step to convert an epistemic planning problem into a classical planning problem (Muise et al. 2022), which grows exponentially w.r.t. the depth of the epistemic formula used.

Recently, Hu, Miller, and Lipovetzky (2022) proposed a lazy-evaluation approach to epistemic planning called Planning with Perspectives (PWP). They use Functional STRIPS (F-STRIPS) (Geffner 2000) to separate epistemic reasoning from planning by modelling an agent's perspective using external functions, which can be customised for particular domains. The agent perspective function defines which variables each agent 'sees' in each state, and from this, a multi-agent epistemic logic is built using the 'what you get is what you see' paradigm (Gasquet, Goranko, and Schwarzentruber 2014; Bolander 2014; Cooper et al. 2016; Herzig, Lorini, and Maffre 2015). By doing so, epistemic reasoning can be performed without generating and reasoning over all epistemic relations. They show that they can handle more expressive problems than standard PDDL-based epistemic planners, and avoid the costly pre-compilation (Muise et al. 2015, 2022) and maintenance of Kripke models (Bolander 2017; Kominis and Geffner 2015; Le et al. 2018; Fabiano et al. 2020; Bonet and Geffner 2019). In addition, their agent's perspective model only depends on the state variables valuation, so can be applied to model-free planners as long as they expose their current state, such as when planning with simulators (Francès et al. 2017).

The weakness of PWP is that it can plan only with knowledge, but not belief. By following Fagin et al.'s (2003) interpretation of the difference between knowledge and belief, in knowledge logic such as S5, $K_i\varphi \rightarrow \varphi$ is an axiom, while in belief logics such as KD45$_n$, it is not. Thus, approaches such as PWP cannot model problems in which agents can have incorrect beliefs.

In this paper, we extend the PWP approach to model *justified belief*. We call this Planning with Multi-agent Belief using Justified Perspectives. The intuition is that when people reason about something they cannot see, they generate *justified* belief by retrieving information from their 'memory' that supports that belief (Goldman 1979). In our model, this information comes from the states they have observed in the past. So, an agent believes something if they saw it in the past, and has no evidence to suggest it no longer holds. This includes nested beliefs about other agents' beliefs. We illustrate this idea with a classical false-belief example.

Example 1.1. There are two agents, a and b, and there is a coin $c \in \{head,\ tail\}$ inside a box. The coin can only be seen by the agents when they are peeking into the box. The agents can see each other all the time, which means they see whether others are peeking into the box. The actions that agents can take are "*peek*" and "*return*", while the coin can "*flip*" itself at any point in time. The action and outcome of the flip is only visible to the agents who are peeking into the box. Initially, both agent a and b are not peeking, and the coin is $head$. The task is to generate a false belief such that:

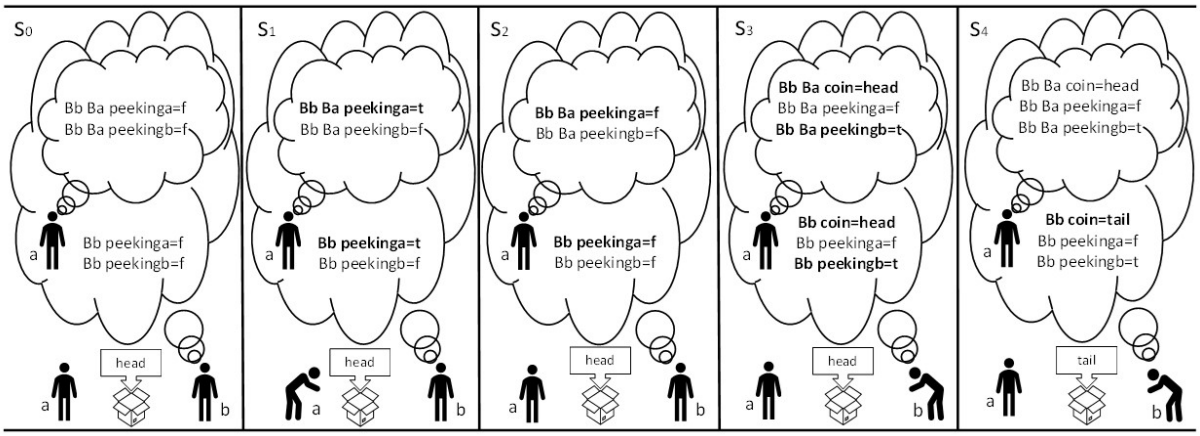

Figure 1: Plan 1.2 with agent b's justified beliefs. Bold text indicates the belief that has changed.

1. the coin is $tail$;
2. agent b believes that the coin is $tail$;
3. agent a believes that the coin is $head$; and
4. agent b believes that agent a believes that the coin is $head$.

A valid plan would be:

Plan 1.1. *peek(a)*, *peek(b)*, *return(a)*, *flip*

It is intuitive that $B_b B_a coin = head$ holds, because agent b saw agent a peeked into the box after the action *peek(b)* and agent a was not peeking into the box while $coin$ flipped.

A more challenging plan to reason about, shown in Figure 1), is:

Plan 1.2. *peek(a)*, *return(a)*, *peek(b)*, *flip*

In this plan, agent a and b no longer peek into the box at the same time, which means, we do not have $K_b K_a coin = head$ in state s_2. All criteria are met as in Plan 1.1, except item 4. For item 4, similarly, agent b recalls that the last time it saw agent a seeing the coin ($S_a coin$) was s_1. However, agent b holds no knowledge or belief on the value of the $coin$ from s_1, which means she cannot generate the justified belief in s_1. Fortunately, agent b gains belief of the coin from s_3. So, agent b still can justify its belief $B_b\ B_a coin = head$:

1. b recalls that the last time agent a peeked inside of the box is s_1; and
2. after b saw a peek, the next time b saw the value of the $coin = head$ is at state s_3.

In a model-free setting, reasoning about this is particularly challenging. Agents do not have access to the action model, so they cannot reason about what other agents have seen. Instead, they can only partially observe states and reconstruct belief by observing states and who else observes each state.

This paper presents a model for KD45$_n$ belief over such plans, with the ability to solve model-free problems. Instead of keeping track of all possible beliefs, we instead use a lazy-evaluation approach that searches through previous states of the plan to re-construct what has been seen, and by who, to evaluate nested belief formulae. Our results show that we can efficiently solve existing benchmarks in epistemic and doxastic planning, even with a simple prototype planner.

2 Background

2.1 Epistemic and Doxastic Logic

The relation between knowledge and belief is not so clear. Some authors state that knowledge is truthful belief (van Ditmarsch et al. 2015; Friedman and Halpern 1994b,a), while others claim that knowledge is truthful *justified* belief (Scherl 2022; Artëmov 2008; Fan and Liau 2017; Grossi and van der Hoek 2014). Bjorndahl and Özgün (2020) define the topology for knowledge and belief based on the different types of justification, while Grossi and van der Hoek (2014) state that belief is generated, endorsed or justified by external arguments. However, there are no definitions of nested belief based on agents' previous observations.

Others use possible worlds to define both knowledge and belief (Stalnaker 2006; Fagin et al. 2003). The idea in these logics is that the agents have a possibility relation $\mathcal{K}_i$ that models whether the agent i can distinguish between two states. Both the knowledge formula $K_i\varphi$ and belief formula $B_i\varphi$ are defined as that φ holds in all the worlds that agent i considers possible. The difference is that possibility relation $\mathcal{K}_i$ in modelling knowledge needs to be *reflective* and *symmetric*[1], while it is *serial* in modelling beliefs. Thus, such approaches have to maintain separate Kripke structures for handling knowledge and belief, and constraints between them must hold to ensure that certain properties hold; e.g. if an agent knows something then it believes it. In this paper, knowledge is based on what an agent currently 'sees', while belief is based on what it currently sees and has seen in the past.

The theoretical foundation for knowledge is the S5 axioms, while for belief are the KD45$_n$ axioms (Fagin et al. 2003). The difference between these two sets of axioms is

[1]Reflective means for all $s \in \mathcal{S}$, (s, s) must be in $\mathcal{K}$, while symmetric means if there is a possibility relation (s, t) in $\mathcal{K}_i$, (t, s) must also be in $\mathcal{K}_i$.

that: S5 includes the axiom $K_i\varphi \rightarrow \varphi$ (Axiom T), which states that an agent's knowledge must be the truth (reflexivity); while KD45$_n$ does not have this axiom, so relations generated between possible worlds are derived from the agent's imperfect information of the world (could be false). Axiom D, replacing T in KD45$_n$, captures $\neg K_i false$, which is preserved by the serial possibility relation.

2.2 Epistemic Planning

Dynamic Epistemic Logic (DEL) approaches use event models to track the agents' knowledge/belief over possible worlds. Most DEL approaches do not support false-belief because they are built on S5 logics. False-belief is challenging to model in DEL because it removes the 'correct' possibility relation between worlds, which results in the agent's belief state becoming isolated (Baral et al. 2022). So, in order to allow agents to recover from false-belief, it requires special sensing or announcing actions (Le et al. 2018; Baral et al. 2022; Li and van Eijck 2022). In addition, it is costly to maintain all agents' possible worlds. Although Kominis and Geffner (2015)'s work captures a fragment of DEL by tracking single agents' belief under the multi-agent setting and updating it by actions in order to achieve better computational performance, it still cannot handle false-belief.

As for the non-DEL-based approach, Muise et al. (2022) define a proper epistemic knowledge base (PEKB) that contains all epistemic formulae as literals. They convert an epistemic planning problem into a classical planning problem using the precondition and conditional effects in actions to update and revise the knowledge and belief literals following some modal axiom, such as those of KD45$_n$. The advantage of their approach is that the model can be solved by any existing classical planner that supports conditional effects. The limitations are: it cannot handle disjunctive belief; the depth of belief is bounded; and the number of literals grows exponentially on the depth of epistemic formulae, so the pre-compilation step has exponential time complexity. (Wan, Fang, and Liu 2021) use a similar method that updates and revises belief knowledge base by implementing their own planner called MEPK, rather than converting to a classical planning problem. By doing so, they lose the advantage of using an existing planner but gain flexibility. However, their approach still cannot handle false-belief.

2.3 Planning with Perspectives Approach

Hu, Miller, and Lipovetzky (2022) propose *Planning with Perspectives* (PWP), which delegates epistemic reasoning to an external solver using F-STRIPS.

Signatures A PWP *signature* is a tuple $\Sigma = (Agt, V, D_{v_1}, \ldots D_{v_n}, R)$, in which Agt is a finite set of agent identifiers (n of them), V is a finite set of variables such that $Agt \subseteq V$ (agent identifiers can be used as variables), D_{v_i} is a possibly infinite domain of constant symbols, one for each variable $v_i \in V$, and R is a finite set of predicate symbols. Domains can be discrete or continuous, and the set of all values is defined as $D = \bigcup_{v \in V} D_v$.

Language The PWP language $\mathcal{L}(\Sigma)$ is defined by the grammar:

$$\varphi ::= r(\vec{t}) \mid \neg\varphi \mid \varphi \wedge \varphi \mid S_i v \mid S_i\varphi \mid K_i\varphi,$$

in which $r \in R$, vectors of terms $\vec{t} \in V \cup D \cup Agt$, $i \in Agt$, and $v \in V$. A relation r is a k-ary propositional relation; $S_i\varphi$ is a visibility formula that means agent i 'sees' the truth value of formula φ, $S_i v$ is a visibility formula that means that agent i 'sees' the value of the variable v, and $K_i\varphi$ is a knowledge formula. 'Seeing' a formula is similar to 'knowing whether' a formula is true or not (Fan, Wang, and van Ditmarsch 2015; Miller et al. 2016). 'Seeing' should not be treated literally: an agent may 'see' the value of a variable by hearing it through a communication channel, for example.

Model A model M is defined $M = (Agt, V, D_{v_1}, \ldots, D_{v_k}, \pi, f_1, \ldots, f_n)$, in which Agt, V, D_{v_i} are as in a signature. A state $s : V \rightarrow D$ is a mapping from variables to values. A *global state* is a total function (a complete assignment for all variables in V), while a *local state* is a partial function (some variables may not be assigned). The expression $s(v)$ denotes the value of variable v in state s. The set of all local and global states is denoted $\mathcal{S}$, while the set of all global states is $\mathcal{S}_G \subsetneq \mathcal{S}$. The set of all models is denoted $\mathcal{M}$. π is an interpretation function $\pi : \mathcal{S} \times R \rightarrow \{true, false\}$ that determines whether the atomic term $r(\vec{t})$ is true in s. r is undefined if any of its arguments t_i is a variable $v \in V$ that is not assigned a value in a local state s, i.e. $v \notin dom(s)$.

Perspective Function The key idea in the PWP model is the *perspective function*. A perspective function for agent i, $f_i : \mathcal{S} \rightarrow \mathcal{S}$ is a function that takes a state and returns a subset of that state, which represents the part of that state that is visible to agent i. The following properties must hold on a perspective function, f_i for all $i \in Agt$ and $s \in \mathcal{S}$:

(1) $f_i(s) \subseteq s$
(2) $f_i(s) = f_i(f_i(s))$
(3) If $s \subseteq s'$, then $f_i(s) \subseteq f_i(s')$

Hu, Miller, and Lipovetzky define general perspective functions, but note that perspective functions can be customised for domains. This provides a level of expressiveness not possible in declarative languages such as PDDL.

Complete Semantics Hu, Miller, and Lipovetzky (2022) propose a sound and complete semantics for PWP[2].

Definition 2.1. Given a model M and state s, the truth of a PWP formula is defined as:

(a) $M, s \vDash r(\vec{t})$ iff $\pi(s, r(\vec{t})) = true$
(b) $M, s \vDash \phi \wedge \psi$ iff $M, s \vDash \phi$ and $M, s \vDash \psi$
(c) $M, s \vDash \neg\varphi$ iff $M, s \nvDash \varphi$
(d) $M, s \vDash S_i v$ iff $v \in \text{dom}(f_i(s))$
(e) $M, s \vDash S_i\varphi$ iff $\forall g \in \mathcal{S}_G$, $M, g[f_i(s)] \vDash \varphi$ or $\forall g \in \mathcal{S}_G$, $M, g[f_i(s)] \vDash \neg\varphi$

[2]They call these semantics the 'non-naïve' semantics, but we use the term 'complete' in this paper as we do not have present a 'naïve' semantics as Hu, Miller, and Lipovetzky (2022) do.

where: $\vec{t}$ is the terms in the relation r, and where $g[s]$ is defined as an override function, where $g[s](v) = s(v)$ if $v \in \text{dom}(s)$, and $g(v)$ otherwise. So, the truth value of $S_i\varphi$ is such that φ is true in every possible state based on the partial view of agent i, or false in every possible state based on the partial view of agent i.

From this, echoing Cooper et al. (2016), the knowledge operation is defined as: $K_i\varphi \leftrightarrow S_i\varphi \wedge \varphi$. That is, agent i knows φ iff φ is true and agent i can see whether φ is true.

Ternary Semantics Hu, Miller, and Lipovetzky (2022) show that the complete PWP semantics has exponential time complexity—in the $S_i\varphi$ definition, we must iterate over all global states $\mathcal{S}_G$. To counter this, they propose a polynomial time *ternary* semantics (Levesque 1998), which contains truth values 0 (false), 1 (true), and $\frac{1}{2}$ (not known). They define a function T, which evaluates the value of the formula, defined as follows (omitting M for readability):

(a) $T[s, r(\vec{t})] = 1$ if $\pi(s_n, r(\vec{t})) = true$; 0 if $\pi(s_n, r(\vec{t})) = false$; $\frac{1}{2}$ otherwise

(b) $T[s, \phi \wedge \psi] = \min(T[s_n, \phi], T[s_n, \psi])$

(c) $T[s, \neg\varphi] = 1 - T[s_n, \varphi]$

(d) $T[s, S_i v] = \frac{1}{2}$ if $i \notin \text{dom}(s_n)$ or $v \notin \text{dom}(s_n)$; 0 if $v \notin \text{dom}(f_i(s_n))$; 1 otherwise

(e) $T[s, S_i\varphi] = \frac{1}{2}$ if $T[s_n, \varphi] = \frac{1}{2}$ or $i \notin \text{dom}(s)$; 0 if $T[f_i(s_n), \varphi] = \frac{1}{2}$; 1 otherwise

They prove that this semantics is sound on all formulae, and is also complete for a fragment of the language known as $\mathcal{NF}$. This fragment $\mathcal{NF}$ captures formulae such as those that do not contain tautologies or contradictions. Given perspective functions for each agent, knowledge formulae in preconditions and effects are evaluated by an F-STRIPS external function that implements the ternary semantics. This approach is considerably more efficient that the PDKB baseline Muise et al. (2022), while offering more expressiveness.

3 Justified Perspective Model

In this section, we add a belief operator, B_i, to the PWP model (Hu, Miller, and Lipovetzky 2022). This belief operator captures the intuition that we believe something if we have seen it before, and we have seen no contradicting evidence since.

3.1 Language and Model

Definition 3.1 (Syntax)**.** The language $\mathcal{L}(\Sigma)$ is defined by the grammar:

$$\alpha ::= r(t_1, \ldots, t_k) \mid \neg\alpha \mid \alpha \wedge \alpha \mid S_i v \mid S_i\alpha \mid K_i\alpha$$
$$\varphi ::= \alpha \mid B_i\varphi$$

$B_i\varphi$ is a belief formula meaning that agent i believes that proposition φ is true. This grammar prohibits formulae such as $S_iB_j\varphi$ or $K_iB_j\varphi$ because seeing or knowing belief are not semantically meaningful in the PWP model—if belief is known then it is knowledge.

Both signature and model are defined as in Section 2.3, except that in this paper we rename their perspective function $f_i(s)$ [3] to be an *observation function* $O_i(s)$, which models what an agent can observe in state s.

3.2 Observation and Justified Perspectives

Now, we define a retrieval function R to retrieve a variable's value from the latest timestamp that the agent had an 'eye' on this variable. From this, we will define the perspective function f_i to construct the agent's *justified perspective*, and reason about the agent's justified belief following the intuition discussed in Section 1.

A sequence of states is denoted as $\vec{s}$, the set of all possible states sequences is denoted as $\vec{S}$, a timestamp is denoted as ts, and the states in a sequence $\vec{s}$ are denoted as $s_0, \ldots, s_n$. Here, the sequence of states in the states that are part of the state-action pairs in a potential plan. A specific state in agent i's perspective $f_i(\vec{s})$ at timestamp n is referred as $f_i(\vec{s})[n]$.

An observation function is defined the same as the perspective function in Section 2.3, except the notation becomes O instead of f.
Then, we can construct a retrieval function, R.

Definition 3.2 (Retrieval function)**.** Given a sequence of states $\vec{s}$, a timestamp ts and a variable v, the retrieval function, $R : \vec{s} \times \mathbf{N} \times V \to D$, is defined as:

$$R(\vec{s}, ts, v) = \begin{cases} s_{ts}(v) & \text{if } v \in s_{ts} \\ s_{max(lts)}(v) & \text{if } lts \neq \{\} \\ s_{min(rts)}(v) & \text{if } rts \neq \{\} \\ None & otherwise \end{cases}$$

where:

$$lts = \{j \mid v \in s_j \wedge j < ts\}$$
$$rts = \{j \mid v \in s_j \wedge 0 \leq ts < j \leq |\vec{s}|\}$$

Here, $\vec{s}$ represents the sequence of states of a plan from a particular perspective, which could be an agent's perspective or the global perspective. The sets lts and rts specify the set of states before and after timestamp ts respectively in which v is seen.

The function R plays a crucial role. If we see an agent i seeing variable v, we know that agent i learns the value of v. However, what value should we believe that i believes? The function R determines this. If the value of variable v exists at time ts, then this is in 'our' perspective, $\vec{s}$, and we see the variable at the same time as i, so R returns the value of v in state s_{ts}. This is the straightforward case.

However, if we do not see variable v at time ts, what value should we assign to agent i's belief? R searches the timestamps before ts to find the most recent reference to v. Intuitively, if we see that agent i sees v at ts, but we do not see the value of v itself at time ts, then we believe that agent i believes the value is the same as the last time we saw v. For

[3] We give a new definition of the perspective function f_i in Definition 3.3

example, if we peek at the coin in the box and see it is a tail, and then we observe agent i peeking at the coin, it implies $B_a coin = tail$ should hold, because $tail$ is the most recent observation of the coin.

If there is no value of v before ts, the function R retrieves the value by searching forward (the timestamps after ts). Intuitively, if we believe that agent i sees v at ts, but we have not seen variable v previously, then we assign i's belief about v the next time we see v after ts.

This is what we see in Plan 1.2 – agent b forms a belief about agent a based on agent b's observation after agent a's observation. If there is no value found about v within $\vec{s}$, then R function returns $None$, as the variable has not been seen from $\vec{s}$.

Other design decisions could be made for R: searching forward first, then backwards; finding the value closest to ts; or 'forgetting' the value of a variable after a certain number of timestamps. Ultimately, there is no 'correct' design here and no design can handle all possible cases, but we believe our choice above is intuitive and justified.

We can now give the definition of a perspective function f_i for agent i. Intuitively, a perspective function models an agent's perspective over the sequence of states in a plan; specifically, an agent's belief about each variable in each state from a given state sequence, which can either be the sequence of global states or another agent's perspectives.

Definition 3.3. A perspective function for agent i, $f_i : \vec{S} \to \vec{S}$, is defined as follows:

$$f_i([s_0, \ldots, s_n]) = [s'_0, \ldots, s'_n]$$

where for all $t \in [0, n]$ and all $v \in \text{dom}(s_t)$:

$$\begin{aligned} s'_t &= \{v \mapsto e \mid lt = max(ats(v)) \wedge e \neq None\}, \\ ats(v) &= \{j \mid v \in \text{dom}(O_i(s_j)) \wedge j \leq t\} \cup \{-1\}, \\ e &= R([s_0, \ldots, s_t], lt, v) \end{aligned}$$

This definition is not so straightforward, so let's give some intuition. First, recall that the sequence $\vec{s} = [s_0, \ldots, s_n]$ can be the perspective of another agent, so it may contain partial states. The set ats contains all timestamps in which agent i sees variable v before state s_t, according to the current perspective. Then, lt is the most recent timestamp up to s_t in which i sees v, which is -1 if agent i has not seen v at all. This tells us the last time that agent i was seen observing variable v in the current perspective. This evidence is used to justify belief (Goldman 1979).

However, if the current perspective represents an agent's perspective, agent j, rather than a global perspective, then agent j may not have seen variable v at time lt—it may have merely observed agent i seeing v, without seeing v itself; e.g. the two agents peek at the coin in the box at different times. We use $R([s_0, \ldots, s_t], lt, v)$ (Definition 3.2) to find what value agent j will believe variable v was in state s_t. That is, the most recent value before s_t or the closest after lt, as defined by R. Therefore, the value $R([s_0, \ldots, s_t], lt, v)$ is the value of v that agent j 'believes' agent i saw; and the perspective function forms a justified perspective of the agent i. We can nest perspective functions arbitrarily to form nested beliefs.

3.3 Semantics

Now, we give two different KD45$_n$ semantics: complete semantics and ternary semantics, which extend their respective S5 semantics from Section 2.3. These have an exponential worst-case time complexity, while the ternary semantics have a polynomial time complexity and have the same properties of incompleteness as their S5 version.

Complete Semantics The complete semantics inherits the definitions of items (a) - (e) in Definition 2.1, but with three minor changes: (1) the frame is a pair $M, \vec{s}$ instead of M, s, i.e. it requires a sequence of states instead of a single state s; (2) the S_i operator uses O_i instead of f_i (our function O_i is equivalent to PWP's f_i perspective function, while our justified perspective function f_i generalises for belief); and (3) the evaluation of atomic propositions is based on the final state of the sequence $\vec{s}$.

The major addition is the semantics [4] for the belief operator:

(g) $M, \vec{s} \vDash B_i\varphi$ iff $\forall \vec{g} \in \vec{S}_G, (M, \vec{g}[f_i(\vec{s})]) \vDash \varphi$

where $\vec{S}_G \in \vec{S}$ is the set of all possible global states sequences and $\vec{g}[\vec{s}] = g_1[s_1], \ldots, g_n[s_n]$.

This definition requires some discussion. At a high level, the definition of $B_i\varphi$ aims to capture is that agent i believes φ if in its past (including present), it saw φ and φ is true: that is, $K_i\varphi$ was true. However, this does not capture situations where φ contains references to variables observed in different states. For example, consider the proposition $B_a(x + y \geq 0)$. If agent a observes $x = 1$ in state s_0, then observes $y = 1$ in state s_1, while not observing x in state s_1 at all, then it is not the case that $M, s_0 \vDash K_a(x + y \geq 0)$ or $M, s_1 \vDash K_a(x + y \geq 0)$ because it does not know the value of y in state s_0 or the value of x in state s_1. However, it seems valid to state that $M, s_1 \vDash B_a(x + y \geq 0)$ because it can remember $x = 1$ from state s_0, and has no evidence to suggest x has changed.

Based on the item (g), $f_a(\vec{s})$ is needed to evaluate the proposition $B_a(x + y \geq 0)$.

In the last timestamp (1), the perspective function identifies the most recent timestamps in which x and y are seen by agent a, which are 0 and 1 respectively. Then, the retrieval function R retrieves the value of x and y, which are $x = 1$ and $y = 1$. So, the last state in agent a's justified perspective $f_a(\vec{s})$ at s_1 is $\{x \to 1, y \to 1\}$. Then, in the previous timestamp, also the first timestamp (0), the lt for x and y identified by the perspective function are 0 and -1. So that, R retrieves x's value is 1, and a's justified perspective at timestamp 0 ($f_a(\vec{s})$ at s_0) is $\{x \to 1\}$.

Then, assuming $D_y = \{-1, 1\}$, by applying the function override $\vec{g}[]$ on a's justified perspective $f_a(\vec{s})$, we have two possible sequences: $\vec{s}_1 = [\{x \to 1, y \to -1\}, \{x \to 1, y \to 1\}]$ and $\vec{s}_2 = [\{x \to 1, y \to 1\}, \{x \to 1, y \to 1\}]$.

Then, we have $M, \vec{s} \vDash B_a(x + y \geq 0) \leftrightarrow (M, \vec{s}_1 \vDash (x + y \geq 0) \wedge M, \vec{s}_2 \vDash (x + y \geq 0))$. Then, based on item

[4] The detailed full semantics can be found in our Appendix: https://github.com/guanghuhappysf128/bpwp/blob/main/ICAPS-23_Supplementarymaterial_final.pdf

(a) in semantics, $M, \vec{s} \vDash B_a(x + y \geq 0)$ holds.

As noted in Section 2.1, there is no underlying definition for our justified belief. So, there is no underlying model to which we can prove soundness or completeness. However, we show our model is sound with respect to KD45$_n$ logic (see the Appendix, Section 3).

Ternary Semantics Now, we show how to implement our model using ternary logic semantics, based on the ternary semantics used by Hu, Miller, and Lipovetzky (2022). This semantics offers a polynomial time complexity logic, compared to the complete semantics, which is exponential in the number of states in the problem. It sacrifices completeness for efficiency. The ternary values for propositions are: 0 denotes false, 1 denotes true, and $\frac{1}{2}$ means the truth value is unknown (unable to be proved).

(g) $\quad T[\vec{s}, B_i\varphi] \quad = \quad T[f_i(\vec{s}), \varphi]$

So, $B_i\varphi = 1$ is 'true' (is equal to 1 in the ternary semantics) if and only if φ is 'true' in agent i's perspective.

Complexity The time complexity for the complete semantics and the ternary semantics are similar to the PWP approach. The only difference is the time complexity for the new justified perspective function.

To evaluate $M, \vec{s} \vDash \varphi$, the worst case scenario is that φ is a belief formula with the depth of d. Then, the justified perspective function complexity is in $\Theta(d \cdot |V|^2 \cdot |\vec{s}|^3)$, which is for each variable, getting the ats, getting $R(\vec{s}, ts, v)$, for s in $\vec{s}$ and for each level of nesting from φ.

For the complete semantics, in $\vec{s}$, each state could have $|V| \times |D|$ possibilities. Thus, the number of possible sequences is $|V \times D|^{|\vec{s}|}$. So, the complexity of the query in the complete semantics is in $\Theta(d \cdot |V|^2 \cdot |\vec{s}|^3 \cdot |V \cdot D|^{|\vec{s}|})$, which is exponential on the input size. While, in the ternary semantics, the complexity of the query is the same as the justified perspective function, assuming item (a) defined in Section 2.3 is in $\Theta(1)$.

Example 3.1 (Example 1.1 with Plan 1.2)**.** In Figure 1, agent b cannot at the same time: 1) see agent a peeking into the box and 2) see what was inside the box at that time, because a and b are no longer peeking into the box at the same time, as they were in s_2 from Plan 1.1. The latest timestamp lt agent a peeked into the box is 1. We can retrieve the value of the $coin$ at timestamp 1 from b's perspective using $R(\vec{s}, 1, coin)$. In agent b's perspective, there is no information about $coin$ until s_3, making the return value of $R(\vec{s}, 1, coin)$ to be $f_b(s)[3](coin) = head$. So, $M, \vec{s} \vDash B_b B_a coin = head$ is equivalent to $M, f_b(\vec{s}) \vDash B_a coin = head$, which is $M, f_a(f_b(\vec{s})) \vDash coin = head$. Therefore, the justified belief of agent b on a false belief about agent a on φ can be generated even if b cannot see both 1) the truth value of φ while 2) seeing agent a seeing φ at the same timestamp (same state).

4 Experiments

We experiment on two benchmark domains (Corridor, and Grapevine), as well as three other domains (Coin, Big Brother Logic and Social-media Network) to evaluate the potential of our model. In this section, we denote d as the depth of the epistemic formulae, $|\mathcal{G}|$ as the number of goal formulae, $|\mathcal{P}|$ as the length of the plan and $|calls|$ as the number of epistemic formulae reasoning.

4.1 Implementation

The source code of the planner, the domain and problem files, as well as experimental results, can be downloaded from https://github.com/guanghuhappysf128/bpwp

Planner Implementation We implemented a simple F-STRIPS planner that supports PDDL and external functions. Since the objective is to demonstrate our model instead of the search algorithm, we use *Breadth First Search* (BFS). The experiments are run on a windows 10 machine with 12 CPUs (Intel i7-8700K CUP 3.70GHz) with 32GB ram.

PDDL encoding As for the encoding, we use the same approach in Hu, Miller, and Lipovetzky (2022). The epistemic formula evaluation only occurs in either the action precondition (Example 4.1) or the goal in PDDL encoding, while the evaluation process itself is done in the external function, which we implemented using python.

4.2 Corridor

Corridor is a benchmark problem in epistemic planning (Muise et al. 2022). Several agents located in different rooms of a corridor try to learn a secret. One of the agents (a in our encoding) has the ability to *move* between rooms, *sense* the secret, *shout* and *shout_lie*. The action *shout* announces the true value of the secret as long as agent a knows the secret (by performing *sense* action before), while the action *shout_lie* announces the false value of the secret. For both actions, agents in the same room or adjacent rooms learn the shouted value of the secret. The objective is to find a plan for agent a that makes some agents believe the secret while some other agents believe the secret is false.

Seeing Rule:

$$\text{sct} \in \text{dom}(O_i(s)) \text{ iff } |s(\text{loc-secret-shout}) - s(\text{loc-}i)| \leq 1$$

Result Results are shown in Table 1. We use the same set of problems as Muise et al. (2022), which contain tasks with false-belief. Since the goals in all of their problems are the same, the number of node generations and node expansions are constant across all problems, while only the number of agents and depth of epistemic formulae affect execution time. We can see that the number of agents and depth of epistemic formula affect the pre-encoding done by PDKB Muise et al. (2022), but not PWP's lazy evaluation.

4.3 Grapevine

Grapevine is another benchmark problem in epistemic planning (Muise et al. 2022). In two adjacent rooms, agents share secrets they have heard previously to all agents in the same room. All agents can *move-left* and *move-right* between two rooms, and *share* the truth about someone's secret or *lie* about it. Both actions require the agent to believe the secret. The objective is to make some agents believe others' secrets without the secret owner's awareness.

Parameters				PWP					PDKB			
$\|Agt\|$	d	$\|\mathcal{G}\|$	$\|\mathcal{P}\|$	$\|Gen\|$	$\|Exp\|$	$\|Calls\|$	TIME(s) Calls	TIME(s) Total	$\|Gen\|$	$\|Exp\|$	TIME(s) Search	TIME(s) Total
3	1	2	5	575	154	185	0.2	0.3	34	16	0.1	0.2
5	1	2	5	575	154	185	0.4	0.5	36	16	0.1	0.2
7	1	2	5	575	154	185	0.6	0.7	37	16	0.1	0.2
3	3	2	5	575	154	185	0.3	0.4	34	16	0.1	0.7
5	3	2	5	575	154	185	0.4	0.5	36	16	0.2	3.3
7	3	2	5	575	154	185	0.7	0.8	37	16	0.6	10.8
3	5	2	5	575	154	185	0.3	0.3	34	16	2.4	39.6
5	5	2	5	575	154	185	0.4	0.5	36	16	217.8	1348.8
7	5	2	5	575	154	185	0.7	0.8	–	–	–	–

Table 1: Experimental results for corridor domain

Parameters				PWP					PDKB			
$\|Agt\|$	d	$\|\mathcal{G}\|$	$\|\mathcal{P}\|$	$\|Gen\|$	$\|Exp\|$	$\|Calls\|$	TIME(s) Calls	TIME(s) Total	$\|Gen\|$	$\|Exp\|$	TIME(s) Search	TIME(s) Total
4	1	2	4	364	68	2234	4.9	5.4	79	15	0.3	0.8
4	2	2	5	442	78	2558	6.0	6.6	61	12	1.6	9.6
4	1	4	6	134492	14781	495432	1942.2	2053.3	88	10	0.2	0.7
4	2	4	7	220022	31709	1056453	5477.5	5707.9	130	17	1.5	9.7
8	1	2	4	892	96	12290	90.8	95.4	152	25	0.7	4.2
8	2	2	5	1114	114	14602	108.9	114.4	186	36	340.4	182.4
8	1	4	11	–	–	–	–	–	5919	194	1.8	5.5
8	2	4	9	–	–	–	–	–	344	29	166.5	329.2

Table 2: Experimental Results for Grapevine Domain

Example 4.1. Agent i shares agent j's secret j-sct:

`action` share(i,j-sct)
`prec` loc-i=x, `epis:` B_i j-sct
`effs` `forall` ?a - agent loc-?a-sct-shared = 0
j-sct=true, loc-j-sct-shared=x

The precondition "`epis:` B_i j-sct" is evaluated by the external function (our model). The effect "loc-j-sct-shared=x" means j-sct has been shared in location x, and "j-sct=true" (if i performs action lie, it would be false).

Seeing Rule:

j-secret $\in$ dom($O_i(s)$) iff s(loc-j-secret-shared) = s(loc-i)

Result Results are shown in Table 2. We use the same set of problems as Muise et al. (2022). Since the epistemic formula is in the precondition of all *share* actions, the number of the external function calls is much larger compared to the corridor domain. In addition, in grapevine, the number of agents increases the branching factor of the problem. Given we implemented just a BFS search algorithm, the planner ran out of time limit (100 minutes) for larger problems.

4.4 Coin

The coin domain is defined in Example 1.1. The objective is to generate some false beliefs.

Seeing Rule:

coin $\in$ dom($O_i(s)$) iff s(peeking-i)

Result Since the coin domain is trivial, results are given in the appendix. A test case with complex goals, $B_aB_bcoin = tail$ and $B_bB_acoin = head$, is worthy of discussion. That is, each agent believes the other believes the coin is a different value.

The plan returned is: *peek(a)*, *peek(b)*, *return(b)*, *flip*, *return(a)*, *peek(b)*. The latest state where b sees that a sees coin is s_4. From s_4, the function R returns $coin = head$ from s_2 (after the first *peek(b)* action), which is the latest state where the coin is in $f_b(\vec{s})$ before s_4.

4.5 Big Brother Logic (BBL)

BBL (Gasquet, Goranko, and Schwarzentruber 2014) contains stationary cameras that can turn and observe a certain angular range in a 2-dimension plane. For example, camera a and camera b are located in positions $(3, 3)$ and $(2, 2)$ respectively, while a target v with value e is located in position $(1, 1)$. The angular range of each camera is $90°$, defined by the observation function O_i. Both cameras have two actions: *clockwise-turn* and *anticlockwise-turn*. For simplicity, we set the angle of turning to be enumerated from the set $\{0°, \pm45°, \pm90°, \pm135°, 180°\}$ and the turning angle to $45°$, but as the external functions are implemented in Python, we can replace this with floating point numbers to model continuous directions. We use the same problems defined by Hu, Miller, and Lipovetzky (2022), but with modified goals to support belief instead of knowledge.

Seeing Rule: From Hu, Miller, and Lipovetzky (2022):

	Parameters				Performance					
	$\|Agt\|$	d	$\|\mathcal{G}\|$	$\|P\|$	$\|Gen\|$	$\|Exp\|$	$\|Calls\|$	TIME(s) $calls$	TIME(s) Total	Goal
BBL01	2	1	1	3	336	85	85	0.0	0.0	$K_b v = e$
BBL02	2	1	1	3	336	85	85	0.1	0.1	$B_b v = e$
BBL03	2	1	2	3	336	85	170	1.1	1.1	$B_b v = e \wedge B_a v = e$
BBL04	2	2	1	5	2728	683	683	90.4	90.6	$B_b B_a v = e$
BBL05	2	2	2	5	2728	683	693	17.3	17.5	$B_a B_b v = e \wedge B_b B_a v = e$
BBL06	2	3	1	5	2728	683	683	178.2	178.5	$B_b B_a B_b v = e$

Table 3: Experimental results for BBL domain

	Parameters				Performance					
	$\|Agt\|$	d	$\|\mathcal{G}\|$	$\|P\|$	$\|Gen\|$	$\|Exp\|$	$\|Calls\|$	TIME(s) $calls$	TIME(s) Total	Goal
SN01	5	1	1	1	64	7	7	0.0	0.0	$B_c p = t$
SN02	5	2	2	1	86	9	11	0.2	0.2	$B_c p = t \wedge B_b B_c p = t$
SN03	5	1	2	2	215	20	25	0.4	0.5	$B_c p = t \neg B_d p = t$
SN04	5	1	1	2	1029	96	193	4.4	4.8	$B_{a,b,c,d,e} p = t$
SN05	5	1	1	2	430	37	56	1.3	1.4	$B_{a,b,d,e} p = t \wedge \neg B_c p = t$
SN06	5	1	1	2	669	62	96	2.1	2.3	$B_{a,c} p = t \wedge \neg B_b p = t \wedge \neg B_d p = t \wedge \neg B_e p = t$

Table 4: Experimental results for SN domain. $B_{a,\ldots,e}\varphi$ is shorthand for each agent in the set $\{a, \ldots, e\}$ believing φ

$$j \in \text{dom}(O_i(s)) \text{ iff}$$
$$\left(|\arctan(\tfrac{|s(y_i)-s(y_j)|}{s(x_i)-s(x_j)}) - s(dir_i)| \leq \tfrac{s(ang_i)}{2}\right) \vee \left(|\arctan(\tfrac{|s(y_i)-s(y_j)|}{s(x_i)-s(x_j)}) - s(dir_i)| \geq \tfrac{360^\circ - s(ang_i)}{2}\right) \quad (1)$$

where (x_i, y_i) is the location of object/agent i.

Result Results are shown in Table 3. Initially, camera b faces 90° in all problems. The optimal solution for camera b to gain any information on variable v is to turn anticlockwise 135°, which is the case of BBL01-03. However, as for camera b to acquire any beliefs about camera a, camera b has to turn clockwise first to see camera a (BBL04-06).

The results show that we can solve multi-agent belief problems in complex domains such as BBL, which would be difficult to encode in a propositional language without external functions.

4.6 Social-media Network

The Social-media Network (SN) domain is an abstract network using bi-directional communication proposed by Hu, Miller, and Lipovetzky (2022). The agents establish two-way communication channels by befriending each other and communicating by posting messages on their homepage or their friends' homepages. Each agent can *Befriend* or *Unfriend* another agent. For example, let a, b, c, d and e be 5 agents and $p = 1$ be the message a wants to share. The secret p can be *post* or *retract* on the homepage of agent a or a's friends' homepage. The objective is to form beliefs about messages. Initially, a is friended with c and d, b is friended c and e, c is friended with a, b and d, d is friended with a, c and e, while e is friended with b and d.

Seeing Rule:

$$p \in \text{dom}(O_i(s)) \text{ iff } \exists j \in Agt, s(\text{friended-}i\text{-}j) \wedge s(\text{post-}p\text{-}j)$$

Results The results are shown in Table 4. For SN01, the plan is simply "post p on a's homepage", while for SN02, with b involved, the plan becomes "post on c's homepage". For SN05, the object is for everyone but c to gain the same belief of p, while SN06 is for everyone who does not believe p except c (a knows p all along, as p is posted by a).

5 Conclusions and Future Work

In this paper, we defined an extension to the PWP S5 logic, to handle belief using perspectives; and embedded this within a model-free planning tool. Although it is not possible to prove the soundness and correctness of our justified belief model, we prove that the new logic satisfies the principles of belief described by the axioms of the logic KD45$_n$ Our experiments demonstrate that we can effectively handle problems of planning with belief, and can do so efficiently even with a simple prototype planner implemented using BFS.

For future work, we will study common belief, which is a challenge to model using perspectives, because common belief is the infinite regress of nested belief, so may not terminate. A potential solution is to find a way to merge each agent's justified perspectives. In addition, besides retrieving information from the agent's memory as justification, other forms of justification, such as argument would be intriguing to be integrated into our model (Goldman 1979).

References

Artëmov, S. N. 2008. The Logic of Justification. *Rev. Symb. Log.*, 1(4): 477–513.

Baral, C.; Gelfond, G.; Pontelli, E.; and Son, T. C. 2022. An action language for multi-agent domains. *Artif. Intell.*, 302: 103601.

Bjorndahl, A.; and Özgün, A. 2020. Logic and Topology for Knowledge, Knowability, and Belief. *Rev. Symb. Log.*, 13(4): 748–775.

Bolander, T. 2014. Seeing is Believing: Formalising False-Belief Tasks in Dynamic Epistemic Logic. In *Proceedings of the ECSI-2014*, 87–107.

Bolander, T. 2017. A Gentle Introduction to Epistemic Planning: The DEL Approach. In *Proceedings of the Ninth Workshop on M4M@ICLA*, 1–22.

Bolander, T.; and Andersen, M. B. 2011. Epistemic planning for single and multi-agent systems. *JANCL*, 21(1): 9–34.

Bonet, B.; and Geffner, H. 2019. Causal Belief Decomposition for Planning with Sensing: Completeness Results and Practical Approximation. *CoRR*, abs/1909.13778.

Cooper, M. C.; Herzig, A.; Maffre, F.; Maris, F.; and Régnier, P. 2016. A Simple Account of Multi-Agent Epistemic Planning. In *ECAI 2016*, 193–201.

Fabiano, F.; Burigana, A.; Dovier, A.; and Pontelli, E. 2020. EFP 2.0: A Multi-Agent Epistemic Solver with Multiple E-State Representations. In *Proceedings of the ICAPS*, 101–109.

Fagin, R.; Halpern, J. Y.; Moses, Y.; and Vardi, M. Y. 2003. *Reasoning About Knowledge*. MIT Press. ISBN 0262562006.

Fan, J.; Wang, Y.; and van Ditmarsch, H. 2015. Contingency and Knowing Whether. *Rew. Symb. Logic*, 8(1): 75–107.

Fan, T.; and Liau, C. 2017. Doxastic Reasoning with Multi-Source Justifications based on Second Order Propositional Modal Logic. In *Proceedings of the 16th AAMAS*, 1529–1531. ACM.

Francès, G.; Ramírez, M.; Lipovetzky, N.; and Geffner, H. 2017. Purely Declarative Action Descriptions are Overrated: Classical Planning with Simulators. In *Proceedings of the 26th IJCAI*, 4294–4301.

Friedman, N.; and Halpern, J. Y. 1994a. A Knowledge-Based Framework for Belief change, Part I: Foundations. In *Proceedings of the 5th TARK*, 44–64.

Friedman, N.; and Halpern, J. Y. 1994b. A Knowledge-Based Framework for Belief Change, Part II: Revision and Update. In *Proceedings of the 4th KR*, 190–201.

Gasquet, O.; Goranko, V.; and Schwarzentruber, F. 2014. Big brother logic: logical modeling and reasoning about agents equipped with surveillance cameras in the plane. In *AAMAS '14*, 325–332.

Geffner, H. 2000. Functional STRIPS: a more flexible language for planning and problem solving. In *Logic-based artificial intelligence*, 187–209.

Goldman, A. I. 1979. What is justified belief? In *Justification and knowledge*, 1–23. Springer.

Grossi, D.; and van der Hoek, W. 2014. Justified Beliefs by Justified Arguments. In *KR 2014*.

Herzig, A.; Lorini, E.; and Maffre, F. 2015. A Poor Man's Epistemic Logic Based on Propositional Assignment and Higher-Order Observation. In *LORI 2015*, volume 9394, 156–168.

Hu, G.; Miller, T.; and Lipovetzky, N. 2022. Planning with Perspectives – Decomposing Epistemic Planning using Functional STRIPS. *jair*, 75: 489–539.

Kominis, F.; and Geffner, H. 2015. Beliefs In Multiagent Planning: From One Agent to Many. In *25th ICAPS*, 147–155.

Le, T.; Fabiano, F.; Son, T. C.; and Pontelli, E. 2018. EFP and PG-EFP: Epistemic Forward Search Planners in Multi-Agent Domains. In *Proceedings of the 28th ICAPS*, 161–170.

Levesque, H. J. 1998. A completeness result for reasoning with incomplete first-order knowledge bases. In *KR*, 14–23.

Li, K.; and van Eijck, J. 2022. Public Announcements, Public Lies and Recoveries. *J. Log. Lang. Inf.*, 31(3): 423–450.

Miller, T.; Felli, P.; Muise, C. J.; Pearce, A. R.; and Sonenberg, L. 2016. 'Knowing Whether' in Proper Epistemic Knowledge Bases. In *Proceedings of the 30th AAAI*, 1044–1050.

Muise, C.; Belle, V.; Felli, P.; McIlraith, S. A.; Miller, T.; Pearce, A. R.; and Sonenberg, L. 2022. Efficient multi-agent epistemic planning: Teaching planners about nested belief. *Artif. Intell.*, 302: 103605.

Muise, C. J.; Belle, V.; Felli, P.; McIlraith, S. A.; Miller, T.; Pearce, A. R.; and Sonenberg, L. 2015. Planning Over Multi-Agent Epistemic States: A Classical Planning Approach. In *Proceedings of the 29th AAAI*, 3327–3334.

Scherl, R. B. 2022. A Situation-Calculus Model of Knowledge and Belief Based on Thinking About Justifications. In *Proceedings of the 20th NMR, Part of the FLoC 2022*, volume 3197, 104–114.

Stalnaker, R. 2006. On logics of knowledge and belief. *Philosophical Studies*, 128(1): 169–199.

van Ditmarsch, H.; Halpern, J. Y.; van der Hoek, W.; and Kooi, B. P. 2015. An Introduction to Logics of Knowledge and Belief. *CoRR*, abs/1503.00806.

Wan, H.; Fang, B.; and Liu, Y. 2021. A general multi-agent epistemic planner based on higher-order belief change. *Artif. Intell.*, 301: 103562.