Implementing Epistemic Planning on a Robot

Abstract

Research has provided evidence that Dynamic Epistemic Logic (DEL) is capable of representing significant aspects of Theory of Mind (ToM) in autonomous robots. This has allowed robots to perform perspective shifts on their representation of the world, and pass higher-order false-belief tasks. In this paper, we present a robotic implementation of DEL-based epistemic planning, and benchmark it on a number of human-robot collaboration scenarios. Furthermore, we introduce a novel partition refinement algorithm for computing unique bisimulation contractions, used in the planning algorithm.

1 Introduction

One important application of epistemic planning is Human-Robot Collaboration (HRC), that is, multiple robots and humans collaboratively working towards a joint goal. The inclusion of human collaborators greatly increases the complexity, since explicit human-robot coordination can be slow and unreliable (Thomaz, Hoffman, and Cakmak 2016). Instead, it is desirable to build systems capable of *implicit coordination*, that is, coordination based upon shared information of the world and the ability to take the perspective of the other agents.

Consider a scenario with a human and a service robot, where the human says: "I am looking for my glasses. Can you help me find them?". Assuming the service robot adopts the goal to help, it becomes a *joint goal* between the human and the robot to locate the glasses. It is an epistemic goal: the goal is for the human to *know* where the glasses are. The robot might come up with the following sequential plan: *findLocationOfGlasses*, *announceLocationOfGlasses*. While this plan works correctly in the nominal case, it could lead to socially awkward behaviour if the human finds the glasses herself first. The robot, still trying to execute its two tasks, would then announce: "I found your glasses! They are in your hands!". To solve this issue, the robot needs two essential abilities: the ability to replan and the ability to take the perspective of the human. Only when being able to take the perspective of the human, will the robot realise that when the human has picked up the glasses, she will know where they are, and the goal has been reached. And only when being able to replan, will the robot be able to adapt its plan to the observed actions of others.

One possible approach to building such a system is to build it on the theoretical framework of *epistemic planning with implicit coordination* proposed by Engesser et. al. (2017) and further detailed by Bolander et. al. (2018). Using Dynamic Epistemic Logic (DEL), they encode the communicative and other epistemic effects of actions in the same action representation language as regular ontic actions. Furthermore, they suggest finding *policies* (mapping states to actions) instead of sequential plans in order to handle non-deterministic execution and uncertainty about the plans chosen by other agents. Their paper gives a formal definition of the planning framework and analyses its theoretical properties. Reifsteck et. al. (2019) describes one possible algorithm and implementation capable of finding such policies using a Monte-Carlo Tree Search over epistemic states, branching on applicable actions and on the global states of perspective-shifted states.

In this paper, we describe another method for finding such policies using an AND-OR graph search algorithm. Our primary contributions are:

- Extensions to the theoretical framework introduced by Bolander et. al. (2018) allowing for the handling of planner uncertainty and an improved partition refinement algorithm for computing unique bisimulation contracted states. (Section 2)
- A detailed description of an AND-OR graph search algorithm that implements the specification of this epistemic planning framework. (Section 3)
- An implementation of the aforementioned algorithm capable of solving several tasks involving implicit coordination and belief synthesis, in both simulated and robotic settings (Section 4)

2 Preliminaries

Epistemic States and Actions The version of DEL used in this paper is based on the version used by Bolander (2018), which is essentially classic DEL (Baltag, Moss, and Solecki 1998) with the extension of *postconditions*, *edge-conditioned event models*, *observability propositions* and *multi-pointed models*. Let Ψ be a set of predicates of first-order logic, $\mathcal{O}$ a finite set of *objects*, and $\mathcal{A}$ a finite set of *agents*. The epistemic language $\mathcal{L}(\Psi, \mathcal{O}, \mathcal{A})$, abbreviated

$\mathcal{L}$, is then given by

$$\phi ::= \top \mid \bot \mid P(c) \mid i{\lhd}j \mid \neg\phi \mid \phi \wedge \phi \mid B_i\phi$$

where $i, j \in \mathcal{A}$, $P \in \Psi$ is a predicate of arity $ar(P) \in \mathbb{N}$ and $c \in (\mathcal{O} \cup \mathcal{A})^{ar(P)}$. The set of atoms of the form $P(c)$ or $i{\lhd}j$ is denoted Atm. The special atoms $i{\lhd}j$ are *observability atoms* with the intended interpretation that agent *i* is currently observing agent *j* and all actions performed by agent j. Formulas $B_i\phi$ are read as: "agent i believes ϕ".

Definition 1. *An* epistemic model *is $\mathcal{M} = (W, R, L)$, where W is a non-empty finite set of* worlds*, $R\colon \mathcal{A} \to \mathscr{P}(W \times W)$ assigns to each agent $i \in \mathcal{A}$ an* accessibility relation *R_i; and $L\colon W \to \mathscr{P}(Atm)$ assigns to each world w a* labelling *(the set of atoms true in w). An* (epistemic) state *is a pair $s = (\mathcal{M}, W_d)$ where $W_d \subseteq W$ denotes the* designated worlds*, and each world $w \in W$ is* reachable *from a designated world $w_d \in W_d$, i.e., (w_d, w) is in the transitive closure of $\cup_{i\in\mathcal{A}} R_i$. The definition of reachability is extended to sets of worlds in the obvious way.*

We will always throughout this work identify states with their restriction to the worlds reachable from W_d, i.e. we implicitly delete unreachable worlds. The logical entailment of a formula in a given state is defined as follows, with standard semantics for the propositional cases:

- $(\mathcal{M}, \{w\}) \vDash p$ iff $p \in L(w)$ where $p \in Atm$
- $(\mathcal{M}, \{w\}) \vDash B_i\phi$ iff $\forall v \in W, (w, v) \in R_i \Rightarrow (\mathcal{M}, v) \vDash \phi$
- $(\mathcal{M}, W_d) \vDash \phi$ iff $(\mathcal{M}, \{w\}) \vDash \phi$ for all $w \in W_d$

We will implicitly assume $i{\lhd}i$ to be true in all models, all worlds and for all agents, i.e., all agents are always observing their own actions. When $W_d = \{w_0\}$ for some w_0, we call w_0 the *actual world*, and the state is called *global*. We will often write $s = ((W, R, L), w_0)$ for $s = ((W, R, L), \{w_0\})$. The set of all global states will be denoted S^{gl}. For any state $s = (\mathcal{M}, W_d)$, let $Globals(s) = \{(\mathcal{M}, w) \mid w \in W_d\}$. Given a global state $s = ((W, R, L), w_0)$, the accessibility relations R_i models the *run-time beliefs* of agent i, i.e., what agent i will believe if being in state s. In the actual world w_0, agent i considers all worlds v s.t. $(w_0, v) \in R_i$ to be possible, i.e., i will believe exactly what is true in all those worlds v. Note that an agent i may have false beliefs and not consider the actual world to be possible, i.e., we might have $(w_0, w_0) \notin R_i$. Higher-order beliefs are represented via sequences of accessibility relations, e.g. "agent i believes that agent j believes ϕ" is true in s iff $s \models B_iB_j\phi$, which holds when ϕ is true in all worlds accessible from the actual world w_0 by the composite relation $R_i; R_j$.

A state $s = ((W, R, L), W_d)$ that is not global can be considered shorthand for the set of global states it contains, $Globals(s)$. We use the set of worlds in W_d to model *plan-time indistinguishability* from the perspective of a particular agent. An agent i might consider to toss a coin and represent the two different possible outcomes by two worlds w and v. The agent then knows that when the coin has been tossed, the actual world will either be w or v, but a priori it doesn't know which. When agent i considers the coin toss and its consequences (at plan time), it will then represent the situation

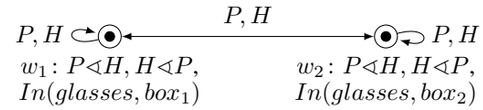


Figure 1: The initial state $s_0 = (\mathcal{M}, \{w_1, w_2\})$ of Example 1. The worlds are represented as nodes, the accessibility relations are represented as labelled edges and an interior dot ⦿ marks designated worlds.

resulting from the coin toss by a state with $W_d = \{w, v\}$. There will be no R_i-link between w and v, since at run time, when the action has actually been executed, the agent will not be in doubt about whether the actual world is w or v. If the agent instead considered to make the coin toss inside a dice cup and not observe the outcome, the resulting state would have $W_d = \{w, v\}$ and $R_i = W_d \times W_d$.

A state $(\mathcal{M}, W_d)$ is a *local state* for agent i if W_d is closed under R_i. The *associated local state* of an agent i, given a state $s = (\mathcal{M}, W_d)$, is defined as $s^i = (\mathcal{M}, \{v \mid (w, v) \in R_i \text{ and } w \in W_d)\}$. The operation of going from s to s^i corresponds to a *perspective shift* into the view of agent i.

Example 1. *Consider the "missing glasses" scenario from the introduction. We have a robot P and a human H. Initially, both of them know that the glasses are in box_1 or box_2, but they do not know which, modelled by the initial epistemic state s_0 of Figure 1. World w_1 represents the possibility that the glasses are in box_1 and w_2 that they are in box_2. The state is both local for P and H. The presence of the observability propositions $P{\lhd}H$ and $H{\lhd}P$ means that in both w_1 and w_2, the agents are co-present (observing each other). The accessibility relations encode that both agents consider both worlds possible, i.e., P believes the glasses to be in one of the boxes, $s_0 \vDash B_P(In(glasses, box_1) \vee In(glasses, box_2))$, but doesn't know which, $s_0 \vDash \neg B_P In(glasses, box_1) \wedge \neg B_P In(glasses, box_2)$ and likewise for H.*

Definition 2. *An* event model *is $\mathcal{E} = (E, Q, pre, post)$, where E is a non-empty finite set of* events*; $Q\colon E \times E \to (\mathcal{A} \to \mathcal{L})$ is an* edge-condition mapping*; $pre\colon E \to \mathcal{L}$ assigns a* precondition *to each event; and $post\colon E \to \mathcal{L}$ assigns a* postcondition *to each event with the restriction that for all $e \in E, post(e)$ is a conjunction of atoms and their negations. An* (epistemic) action *for agent i is an expression of the form $i{:}a$ with $a = (\mathcal{E}, E_d)$, where $\mathcal{E} = (E, Q, pre, post)$ is an event model and $E_d \subseteq E$ denotes the* designated events.

An epistemic action $i{:}a$ is an action executed by agent i (we can read $i{:}a$ as "i does a"). When $a = (\mathcal{E}, E_d)$, then E_d represents the events that from the perspective of agent i might occur when a is executed. An action with $|E_d| > 1$ thus represents an action that is non-deterministic from the perspective of agent i, e.g. the previously considered action of tossing a coin; or the action of peeking into the dice cup after having performed the coin toss (where only the first is objectively non-deterministic). For each pair of events e, f

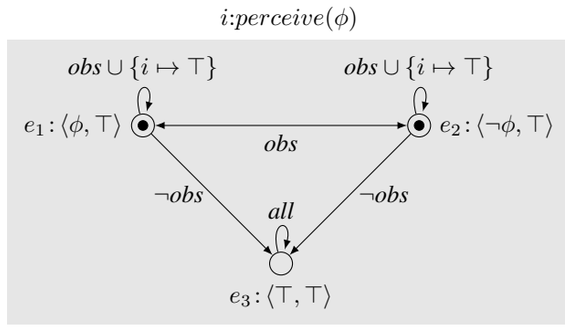


Figure 2: The generic perception action $i{:}perceive(\phi)$. Each event $e \in E$ is a node labelled by $\langle pre(e), post(e) \rangle$ and each edge (e, f) is labelled by the mapping $Q(e, f)$. When $Q(e, f)(i) = \bot$ for some $e, f \in E$ and $i \in \mathcal{A}$ we will leave it out. We here use the partial mappings $obs = \{j \mapsto j{\lhd}i \mid j \neq i\}$ and $\neg obs = \{j \mapsto \neg j{\lhd}i \mid j \neq i\}$ representing those agents who are currently observing, respectively not observing, agent i. Furthermore, we let $all = \{j \mapsto \top \mid j \in \mathcal{A}\}$.

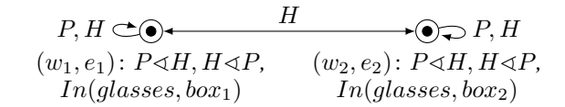
and each agent i, the formula $Q(e, f)(i)$ expresses the condition under which there is an accessibility edge from e to f for agent i. Applying an action to a specific state is achieved through the *product update*, which computes the resulting successor state.

Definition 3. *Let $s = ((W, R, L), W_d)$ and $i{:}a = i{:}(\mathcal{E}, E_d)$ where $\mathcal{M} = (W, R, L)$ and $\mathcal{E} = (E, Q, pre, post)$. The* product update *of s with $i{:}a$ is defined as $s \otimes i{:}a = ((W', R', L'), W'_d)$ where $W' = \{(w, e) \in W \times E \mid (\mathcal{M}, w) \models pre(e)\}$; $R'_j = \{((w, e), (v, f)) \in W' \times W' \mid (w, v) \in R_j$ and $(\mathcal{M}, w) \models Q(e, f)(j)\}$; $L'((w, e)) = \{p \in Atm \mid post(e) \models p$ or $((\mathcal{M}, w) \models p$ and $post(e) \not\models \neg p)$; and $W'_d = \{(w, e) \in W' \mid w \in W_d$ and $e \in E_d\}$. Furthermore, $i{:}a$ is said to be* applicable *in s iff for all $w \in W_d$ there exists an event $e \in E_d$ s.t. $(\mathcal{M}, w) \models pre(e)$.*

You now also need to restrict the product update to the reachable worlds in order to make sure the updated model is an epistemic state. The easiest solution is to add a sentence at the end of the definition saying that we will identify the product update with its restriction to the worlds reachable from W'_d. Nicolai: In the definition of an epistemic state we have stated that epistemic states are implicitly assumed to have all unreachable worlds removed

Example 2. *Extending Example 1, for each agent i and formula ϕ we can define an action $i{:}perceive(\phi)$, provided in Figure 2, modelling that agent i senses whether ϕ holds (e.g. $P{:}perceive(In(glasses, box_1))$ models that P checks whether the glasses are in box_1). The action is defined with the assumption that agent i privately observes whether ϕ holds or not (e.g. by peeking into box_1 in the case of $\phi = In(glasses, box_1)$). However, the agents currently observing i will see the sensing action taking place and will hence know that agent i comes to know whether ϕ.*

Letting s_0 be the state from Figure 1 and $a = perceive(In(glasses, box_1))$, we can compute the result $s_1 = s_0 \otimes P{:}a$ of performing $P{:}a$ in s_0:



P no longer has any (run-time) uncertainty regarding the location of the glasses, i.e. $s_1 \models B_P In(glasses, box_1) \vee B_P In(glasses, box_2)$. H still doesn't know their location, but since H saw P look into box_1, H knows that P knows their location, $s_1 \models B_H(B_P In(glasses, box_1) \vee B_P In(glasses, box_2))$. Had s_0 contained a third agent H' with the same accessibility relation as H, but not observing any other agents, the updated model s_1 would instead contain 4 worlds. Two of them would be a copy of s_0, representing that H' didn't learn anything from the action $P{:}a$ (didn't see anything happen).

Bisimulation and bisimulation contraction In order to deal with indistinguishability between states, we introduce a notion of bisimulation. Our bisimulations are essentially defined as usual in epistemic logic, except we need to account for our models potentially having multiple designated worlds (multi-pointed models). Bisimulations between multi-pointed models was defined by Bolander et al. (2011), however not in a way that guaranteed correspondence between modal equivalence and bisimilarity. We here improve the definition and prove that it satisfies the required correspondence. We also extend the classic partition refinement algorithm to guarantee constructing a unique bisimulation contraction of any model (unique up to identity).

Definition 4. *A* bisimulation *between states $s = ((W, R, L), W_d)$ and $t = ((W', R', L'), W'_d)$ is a binary relation $Z \subseteq W \times W'$ satisfying:*

[atom] *If $(w, w') \in Z$, then $L(w) = L'(w')$.*

[forth] *If $(w, w') \in Z$ and $(w, v) \in R_i$, then there exists v' such that $(v, v') \in Z$ and $(w', v') \in R'_i$.*

[back] *If $(w, w') \in Z$ and $(w', v') \in R'_i$, then there exists v such that $(v, v') \in Z$ and $(w, v) \in R_i$.*

[designated] *If $w \in W_d$, then there exists a $w' \in W'_d$ with $(w, w') \in Z$, and vice versa.*

Two states s and t are called bisimilar *if there exists a bisimulation between them, and we then write $s \leftrightarrows t$.*

Proposition 1. *Two states are bisimilar iff they are modally equivalent (satisfy the same set of formulas).*

If the proof goes out, then the proof *has* to be in an extended version somewhere, and there has to be some text explaining this.

Proof. $\Rightarrow$: Suppose $s = ((W, R, L), W_d)$, $t = ((W', R', L'), W'_d)$, that Z is a bisimulation between s and t, and $s \models \phi$. Then prove $t \models \phi$. By our truth conditions, this is equivalent to proving that $((W', R', L'), w') \models \phi$ for every $w' \in W'_d$. Choose $w' \in W'_d$ arbitrarily. By the [designated] condition, there exists $w \in W_d$ with $(w, w') \in Z$. Now note that Z is a standard bisimulation between $((W, R, L), w)$ and $((W', R', L'), w')$ (it satisfies [atom], [back] and [forth] and $(w, w') \in Z$), and it hence follows that the two states are modally equivalent (Blackburn, de Rijke, and Venema 2001). From $s \models \phi$ and $w \in W_d$, we now immediately get $((W, R, L), w) \models \phi$, and hence $((W', R', L'), w') \models \phi$ by modal equivalence.

$\Leftarrow$: Suppose two states $s = ((W, R, L), W_d)$ and $t = ((W', R', L'), W'_d)$ are modally equivalent. We need to find a bisimulation Z' between them. From s we build a new model $s' = ((W'', R'', L''), w_0)$ with $W'' = W \cup \{w_0\}$,

It can be confusing that $t = ((W', R', L'), W'_d)$, but $s' = ((W'', R'', L''), w_0$, and not t'

$L(w_0) = \emptyset$, $R_i'' = R_i$ for all i, and we add a fresh agent j with $R_j = \{(w_0, w) \mid w \in W_d\}$. We construct t' from t in the same way, using w_0' as the designated world and using the same fresh agent j. By induction on the structure of the formula, we can now prove s' and t' to be modally equivalent: All cases except $B_j\phi$ are trivial, and the case of $B_j\phi$ holds due to the modal equivalence of s and t ($s' \models B_j\phi$ iff $s \models \phi$ iff $t \models \phi$ iff $t' \models B_j\phi$). Since s' and t' are modally equivalent, and they are standard single-pointed epistemic models, there must exist a bisimulation Z between them (Blackburn, de Rijke, and Venema 2001) (note that our models are finite and hence trivially image-finite). Let $Z' = Z \cap (W \times W')$. Since Z is a bisimulation relation on $(W \cup \{w_0\}) \times (W' \cup \{w_0'\})$, Z' must satisfy the conditions [atom], [forth] and [back] for the states s and t. It now suffices to prove that Z' also satisfies [designated]. So let $w \in W_d$. Then $(w_0, w) \in R_j$, and since $(w_0, w_0') \in Z$ and Z is a bisimulation, from [forth] we get the existence of a world w' with $(w, w') \in Z$ and $(w_0', w') \in R_j$. From $(w_0', w') \in R_j$ we can infer $w' \in W_d'$, as required. The other direction is shown symmetrically. □

With this is place, this allows us to precisely define the notion of indistinguishability between states: Two local states are indistinguishable for agent i if they are modally equivalent, i.e., satisfy the same formulas. Given Proposition 1, this notion of indistinguishability between local states correspond to bisimilarity between the states.

Since the union of two bisimulations is itself a bisimulation, there exists a unique largest bisimulation between any two states (Blackburn and van Benthem 2006). An *autobisimulation* is a bisimulation between a model and itself. The quotient of a state wrt. its largest autobisimulation is called the *bisimulation contraction* of the state (Blackburn and van Benthem 2006). For multi-pointed epistemic models, the designated worlds of the bisimulation contraction will be all equivalence classes containing at least one of the original designated worlds. It is well-known that the largest autobisimulation can be computed using the partition refinement algorithm (Aceto, Ingolfsdottir, and Srba 2012). The partition refinement algorithm initially creates a partition of the worlds into a set of blocks B, where worlds having the same label are put into the same block $b \in B$. It then refines the partition by splitting blocks in such a way that all worlds of a block in the new partition will have the same accessibility edges to other blocks of the old partition. This refinement process is repeated iteratively until saturation. In the final partition, two worlds are in the same block iff related by the largest autobisimulation. See Aceto et al. (2012) for details. The only thing non-standard in our case is that we also have to deal with the designated worlds, that is, we need to prove that the autobisimulation defined by partition refinement satisfies [designated]. However, this trivially holds.

Since two worlds are in the same block after partition refinement iff they are related by the largest autobisimulation, we can use partition refinement to compute the bisimulation contraction of a state s. We simply run the partition refinement algorithm on s, and after termination we create a new state with the computed blocks as worlds, the relation between blocks as world relations, and the designated worlds as those blocks that contain designated worlds of the original state s. We can also use partition refinement to compute whether two states s and t are bisimilar.

Proposition 2. *Let $s = ((W, R, L), W_d)$ and $t = ((W', R', L'), W_d')$ be states, and let $s \sqcup t$ denote the disjoint union of s and t. Let B be the set of blocks resulting from running partition refinement on $s \sqcup t$. Then s and t are bisimilar iff the following holds: 1) For each $w \in W_d$, the block $b \in B$ containing w also contains some world $w' \in W_d'$; 2) For each $w' \in W_d'$, the block $b \in B$ containing w' also contains some world $w \in W_d$.*

Proof. For any state s' and any world w, let $s'(w)$ denote the submodel of s' generated by w and having w as its designated world (the submodel of s' generated by w is the restriction of s' to the worlds reachable from w by some sequence of accessibility edges (Blackburn, de Rijke, and Venema 2001)). Partition refinement on $s \sqcup t$ produces the largest autobisimulation on $s \sqcup t$, so any two worlds $w \in W$ and $w' \in W'$ will be in the same block after partition refinement iff $s(w)$ is bisimilar to $t(w')$.

$\Leftarrow$: Suppose 1) and 2) hold after partition refinement. To prove that s and t are bisimilar, by Proposition 1 it suffices to prove that they are modally equivalent. Hence suppose $s \models \phi$. We then want to prove $t \models \phi$, i.e., we want to prove $((W', R', L'), w') \models \phi$ for all $w' \in W_d'$ (the other direction is symmetric). Let $w' \in W_d'$ be chosen arbitrarily. Now by condition 2 we get the existence of a block b and a world $w \in W_d$ with $w, w' \in b$. Hence $s(w)$ and $t(w')$ are bisimilar and thus modally equivalent, by Proposition 1. From $s \models \phi$ and $w \in W_d$, we now immediately get $((W, R, L), w) \models \phi$, implying $s(w) \models \phi$, and hence $t(w') \models \phi$, which finally gives $((W', R', L'), w') \models \phi$, as required.

$\Rightarrow$: Suppose s and t are bisimilar, and let B be the set of blocks resulting from partition refinement on $s \sqcup t$. We need to prove properties 1) and 2). We only prove 1), the other property being proved symmetrically. So choose $w \in W_d$ arbitrarily. We then need to show that the block containing w also contains a world from W_d'. Let Z be a bisimulation relation between s and t (which exists since s and t are assumed bisimilar). Then by [designated], there exists a world $w' \in W_d'$ with $(w, w') \in Z$. Hence w must also be related to w' by the largest bisimulation, which is the union of all bisimulations. The largest bisimulation is computed by partition refinement, so we get that w and w' must belong to the same block. □

One limitation of the standard partition refinement algorithm is that given two distinct bisimilar states, their respective bisimulation contractions are only guaranteed to be isomorphic, not identical. This prevents constant-time lookup of bisimilar states through hashing of contracted models. For this reason, we here introduce a more sophisticated *ordered partition refinement* procedure which given two bisimilar states always produces a single unique bisimulation contraction suitable for hashing. The intuition behind ordered partition refinement is to introduce a total ordering on the blocks which does not depend on the initial naming

of the worlds, hence guaranteeing uniqueness of the resulting contracted state. First we need to define a few concepts with regards to orderings. A total order on world labels can trivially be found (e.g. lexicographical). An *indexing* for a given epistemic model is a mapping $f\colon W \to \mathbb{N}$ satisfying $f(w_1) = f(w_2) \Rightarrow L(w_1) = L(w_2)$. A *block* is a number in the image of f. A block n is said to *contain* the worlds w with $f(w) = n$. In this way, an indexing f induces a partition of the worlds into blocks, and each block has a unique label by construction (the label shared by all the worlds in the block). A *signature* σ_w is a set of $\mathcal{A} \times \mathbb{N}$ pairs representing the outgoing relations of a specific world w, i.e., $(i, n) \in \sigma_w$ means that w has an i-edge to block n. We denote the set of all signatures, with some arbitrary total ordering, as Σ.

Definition 5. *Given a state $s = ((W, R, L), W_d)$ the* ordered partition refinement *procedure proceeds as follows:*

1. *Create the initial indexing f by letting each world w map to the position of its label in the total ordering of labels.*
2. *Refinement step: For every block n considered in increasing order, do the following. Let $\sigma_0 < \ldots < \sigma_m$ denote the ordered sequence of unique signatures in block n. If $f(w) = n$ and σ_k,$k \in [1, m]$, is the signature of w, then $f(w) \leftarrow max_{x \in W}\{n \mid f(x) = n\} + 1$*

The refinement step is repeated until a fixpoint is reached, and the contracted model is then induced from the blocks of the partition as earlier described.

I'd like to look at this again when the earlier mentioned issue regarding T has been settled. I think it needs to be revised a bit, but let's see.

Lemma 1. *Let $s = ((W, R, L), W_d)$ and $s' = ((W', R', L'), W'_d)$ be two epistemic states, and Z a bisimulation between s and s'. Suppose that we concurrently perform partition refinement on both s and s', with the block indexing f and f' respectively. Then the following invariant holds at the beginning of each refinement step:*

(a) f and f' induce partitions of equal size, i.e., they have the same number of blocks.
(b) For any pair $(w, w') \in Z$, there is a unique n with $f(w) = n$ and $f'(w') = n$.

Proof. By induction on the steps of the algorithm. The base case is at the beginning of the first refinement step, i.e., after step 1 of ordered partition refinement. Since the states are bisimilar and both have each world reachable from a designated world (Definition 1), it follows from [designated], [forth] and [back] that for all $w \in W$ there must exist $w' \in W'$ such that $(w, w') \in Z$ and vice versa. It then follows from [atom] that $\{L(w) \mid w \in W\} = \{L'(w') \mid w' \in W'\}$, and therefore step 1 creates partitions of the same size for both s and s', proving (a). Now consider any $(w, w') \in Z$. By [atom] it follows that $L(w) = L(w')$ and from the initial partition step it directly follows that $f(w) = f(w')$, thereby proving (b). This concludes the base case.

For the induction step, suppose (a) and (b) hold at the beginning of refinement step N. We now want to prove that the refinement step maintains both (a) and (b), i.e. that both also hold before step $N + 1$. We will assume that only a single block n will be split during the refinement step. Since this split maintains (a) and (b), the case with multiple blocks immediately follows. Due to (a) and the fact that we consider the blocks according to some total ordering, the n'th block will be considered at the same time in both s and s'. Let $\{w, v\} \in W$ s.t. $(w, v) \in R_i$ and w has signature σ_k. From the bisimilarity of s and s', and the [forth] condition, there must exist $w', v' \in W'$ such that $(w, w'), (v, v') \in Z$ and $(w', v') \in R'_i$, where w' has signature $\sigma'_{k'}$. Due to (b) it holds that $f(w) = f(w')$ and $f(v) = f(v')$, and therefore $\sigma_k = \sigma'_{k'}$. Due to (b) and the fact the w was chosen arbitrarily, block n in s and s' must contain the same set of unique signatures, and therefore $k = k'$. Since the unique signatures of s and s' are considered according to some total ordering, w and w' will be put into the same index block thus maintaining (a) and (b). □

Theorem 1. *Let s and s' be bisimilar epistemic states. Computing the bisimulation contraction of both s and s' using ordered partition refinement results in identical states.*

Proof. Let Z be a bisimulation between states $s = ((W, R, L), W_d)$ and $s' = ((W', R', L'), W'_d)$. We prove that the contracted models induced by running ordered partition refinement on states s and s' will result in identical states. It is sufficient to prove that the last partition of s and s' have ordered blocks with 1) pairwise identical labellings, 2) pairwise identical accessibility edges to other blocks, and 3) the same blocks of s and s' contain designated worlds. Let f and f' be the block indexing from running ordered partition refinement on s and s' respectively. By Lemma 1, f and f have the same number of blocks, i.e., the same image. *Proof of 1)*: Let n be in the image of f and f'. We want to show that block n of s has the same label as block n of s'. Choose $w \in W$ with $f(w) = n$. There exists $w' \in W'$ with $(w, w') \in Z$. By [atom], $L(w) = L(w')$. This proves that the two blocks has the same label, as the label of a block by construction is the unique label shared by all worlds of the block. *Proof of 2)*: Let $(w, v) \in R_i$ with $f(w) = n$ and $f(v) = m$. Due to $s \leftrightarrows s'$ there exists $w', v' \in W'$ s.t. $(w, w') \in Z$, $(v, v') \in Z$. From Lemma 1 it follows that $f'(w') = n$ and $f'(v') = m$. From [forth] it follows that $(w', v') \in R'_i$, as required. The other direction is symmetric. *Proof of 3)*: Let $w \in W_d$ with $f(w) = n$. From [designated] we get the existence of a $w' \in W'_d$ s.t. $(w, w') \in Z$. From Lemma 1 it then follows that $f'(w') = n$, as required. The other direction is symmetric. □

Epistemic Planning Tasks and Policies With epistemic states and actions defined we can now, following Bolander et al. (2018), describe how these concepts can be used for planning. We use S_i^{min} to denote the set of bisimulation contracted (by ordered partition refinement) local states for agent i in which W_d is a minimal set closed under R_i.

Definition 6. *A* planning task *for an agent i is a tuple $\Pi_i = (s_0, A, \gamma)$, where $s_0 \in S_i^{min}$, called the* initial state*, A is a finite set of epistemic actions called the* action library *and γ is a formula called the* goal formula*.*

Example 3. *We can now finalize the formalization of the scenario described in the introduction, as the planning task*

$\Pi_P = (s_0, A, \gamma)$ *where* s_0 *is the state described in Example 1,* A *is the action library consisting of* $i{:}perceive(\phi)$, $i{:}announce(\phi)$ *and* $i{:}ontic(\psi, \phi)$ *actions for* $i = P, H$ *from Figure 2 and 3, and* $\gamma = B_H In(glasses, box_1) \vee B_H In(glasses, box_2)$.

Following the definition of joint policies in (Engesser et al. 2017) we here define policies with our notion of S_j^{min}.

Definition 7. *Let* $\Pi_i = (s_0, A, \gamma)$ *be a planning task. A policy* $\pi = (\pi_j)_{j \in \mathcal{A}}$ *consists of partial mappings* $\pi_j{:}S_j^{min} \to A$ *such that for all* $s \in S_j^{min}$, *if* $\pi_j(s) = k{:}a$ *then* $k = j$ *and* $j{:}a$ *is applicable in* s. *We use* $Dom(\pi)$ *to denote the set of global states* s *with some* j *such that* $\pi_j(s^j)$ *is defined.*

Policies only contain bisimulation contracted states to ensure that all bisimilar states are represented by the same entry in the policy thereby ensuring that an agent will execute the same action in all indistinguishable states, while keeping the policy of finite size.

Our notion of a policy being a solution to a planning task is provided below. We use $\lfloor s \rfloor$ to denote the bisimulation contraction of s using ordered partition refinement.

Definition 8. *Let* $\Pi_i = (s_0, A, \gamma)$ *be a planning task and* π *a policy for* Π_i. *An* execution *of* π *is a maximal alternating sequence of states and actions* $(s_0, j_1{:}a_1, s_1, j_2{:}a_2, \ldots, j_n{:}a_n, s_n)$ *such that for* $m \geq 0$, $j_{m+1}{:}a_{m+1} \in \{\pi_{j_{m+1}}(g^{j_{m+1}}) \mid g \in Globals(s_m)\}$ *and* $s_{m+1} = \lfloor s_m \otimes j_{m+1}{:}a_{m+1} \rfloor$. *An execution of* π *is* successful *for* Π_i *if it is finite and its last element* s_n *satisfies* $s_n \vDash \gamma$. *The policy* π *is an* implicitly coordinated policy *for planning task* Π_i *if* $Globals(s_0) \subseteq Dom(\pi) \cup \{s \in S^{gl} \mid s \vDash \gamma\}$ *and for each* $s \in Dom(\pi)$, *any execution of* π *from* s *is successful for* Π.

Is that subset condition really still relevant? Now you start out with a local state from the perspective of the planning agent. That agent has to be able to verify that the goal will be reached. For the empty policy, the requirement would be $s_0 \models \gamma$. This is already from the perspective of i, so that's fine. I don't see the need of any additional requirements.

As explained in Bolander et al. (2018), implicitly coordinated policies satisfy that at any point during execution, at least one agent knows that it can execute a particular action as part of the policy, and can individually verify the remaining policy to be a solution to the planning task. This is what makes the policies implicitly coordinated.

3 Algorithm

We will now present our planning algorithm for computing implicitly coordinated policies with optimal worst-case execution length. The algorithm is a BFS variant of the classical AND-OR search (Nilsson 1971) on a rooted graph. Distinct nodes in the graph will represent distinct states. Therefore state and node will be used interchangeably in the following, and we will identify nodes with the states they represent. The algorithm generally follows the following two node expansion rules: 1) Or-node: Create a child and-node for every action applicable in the state of the node; 2) And-node: Create a child or-node for every global state contained in the state of the node. When branching on an action $j{:}a$ from an or-node s, the algorithm will first calculate the product update $s' = s \otimes j{:}a$, and perform a bisimulation contraction using ordered partition refinement. This will produce a unique minimal state for each set of bisimilar states. In order to increase efficiency all identical states are represented by the same node in the graph.

Algorithm 1: Planning algorithm

Input: Planning task $\Pi_i = (s_0, A, \gamma)$
Output: Policy π for Π_i, or *failure*

```
1  initialize(i)
2  while not frontier.empty() do
3      s ← frontier.pop_front()
4      for each j:a ∈ A do
5          s' ← s^j
6          if not s'.applicable(j:a) then continue
7          s' ← s' ⊗ j:a
8          s' ← bisimulation_contraction(s')
9          if bisimilar_exists(s') then continue
10         S_and ← S_and ∪ {s'}
11         for each s'' ∈ Globals(s') do
12             s'' ← bisimulation_contraction(s'')
13             if bisimilar_exists(s'') then continue
14             S_or ← S_or ∪ {s''}
15             if s'' ⊨ γ then
16                 update_solved_dead(s'')
17             else
18                 frontier.push_back(s'')
19     update_solved_dead(s)
20     if root.is_solved() then return extract_policy()
21     if root.is_dead() then return failure
22 return failure
```

Pseudocode The pseudocode of our planning algorithm is provided by Algorithm 1. We now detail the individual lines of the code. Line 1 has been contracted due to space limitations. It performs some bookkeeping by creating a root node from s_0, and adding it to the graph as an and-node. It then essentially runs lines 11-18 where s' is replaced by the root. Thereby the root becomes solved if all of its globals become solved.

Lines 2-6 iterate the frontier starting at the root, s_0, and check for all actions $j{:}a \in A$ if they are applicable from the perspective of the action owner j. Note that the search strategy used is BFS, and as such, the frontier is a queue.

Lines 7-10 perform the product update, and the bisimulation contraction. If a bisimilar state exists anywhere in the graph, it will be set as a child of the current node s. Otherwise, a new node is created and added to the graph as a child of the current node s. Note that the bisimilarity check is done only on nodes of the same type, to prevent a global state t to be caught by its child t', e.g. when $Globals(t) = \{t'\}$.

Lines 11-18 iterate the global states of s' and repeat the previously described bisimilarity check for s''. Since s'' is an or-node it is checked whether it fulfills the goal formula γ in which case it is set to solved and propagated, otherwise it is added to the frontier for further expansion.

Lines 16 and 19 check (after each found goal and each node expansion, respectively) if a given node, t, is *solved*, *dead* or *undefined* (status not yet known), updates a flag on the node accordingly, and propagates this up the graph in case the node is *solved* or *dead*. The propagation recursively updates the status of all parents if the status of the updated node changes (i.e., if a parent t' of t changes from unde-

fined to solved or dead, then the parents of t' are updated as well). The intuition behind a solved node is that it either (for or-nodes) is a goal state or has an action which necessarily leads to a goal state, or (for and-nodes) all its global states necessarily lead to a goal state. We define a node as being expanded if all its children have been generated. Formally, we define *solved*(t) recursively as follows:

- If t is an and-node, then *solved*(t) is true iff t has been expanded and all its children are solved.
- If t is an or-node, then *solved*(t) is true iff $t \vDash \gamma$ or t has a solved child.

Similarly, we define *dead*(t) as:

- If t is an and-node, then *dead*(t) is true iff t has a dead child.
- If t is an or-node, then *dead*(t) is true iff t has been expanded and all its children are dead.

We define an *undefined* node as one which is neither *solved* nor *dead*.

Lines 20 and 21 check for termination. If *dead*$(root)$ then no policy exists, else if *solved*$(root)$ the policy is extracted and returned. The policy extraction performs two graph traversals, only visiting solved nodes. The first traversal is bottom-up, starting at the solved leaf nodes, marking each node with a cost, defined as the min-max (or-and nodes respectively) distance to leaf nodes, i.e. minimizing the worst case length to a solved lead node. The second traversal is top-down, starting at the root, visiting (for or-nodes) the child with the lowest cost, and (for and-nodes) all children. At an or-node, given the pair of the state t attached to the node, and the action $j{:}a$ to the child with the lowest cost, the policy π_j is extended with the assignments $\lfloor (t')^j \rfloor \mapsto j{:}a$ for all $t' \in Globals(t)$. Note that even though the graph contains cycles, the extracted policy does not. When traversing from the root to a leaf node, by always choosing the lowest cost child at an or-node, the cost will always decrease by at least one per step. If two of the same nodes existed on a traversed path from root to leaf, the cost must have had a non-decreasing change at some step, which is a contradiction.

Theorem 2. *Algorithm 1 is sound and complete. That is, given a planning task Π_i as input to Algorithm 1, if there exists an implicitly coordinated policy for Π_i, then Algorithm 1 will return such a policy (completeness); and if Algorithm 1 returns a policy π, it will be an implicitly coordinated policy for Π_i (soundness).*

A formal proof of Theorem 2 remains future work.

4 Implementation & Experiments

In order to validate the algorithm in practice and test its capabilities, we have implemented an epistemic planner based upon the algorithm and integrated it into a robotics system[1].

[1]The source code can be found on https://github.com/Zeltex/Pepper_Planner

Implementation A planner has been implemented in the C++ programming language. It supports loading planning domains from a declarative PDDL-like (McDermott 2000) language. The planner has furthermore been integrated as an extension to the robotic platform described by Dissing and Bolander (2020). In brief, a Softbanks Pepper robot equipped with additional Intel RealSense D435i RGB+D cameras is used. A perception pipeline is used to detect actions and thereby continuously maintain the epistemic state. When a goal is received, the planner is invoked with the current internal state as the initial planning state. The found policy is passed to an execution system which on all state changes look up the new state in the policy and executes any action with the robot itself marked as action owner.

Experimental Evaluation We have tested our planner on an extended version of the the *cubes and boxes* domain described by Dissing and Bolander (2020). We denote the robot P and the human H. In case of multiple human agents, they will be denoted as $H_i, i \in \mathbb{N}$. The set of objects $\mathcal{O}$ consists of two distinct types of objects: uniquely coloured cubes named $cube_{color}$ and numbered boxes named box_i where $i \in \mathbb{N}$. The cubes can then be put into boxes, represented by the binary predicate $In(cube, box)$, or picked up, $In(cube, H)$. The agents can then perform the four different types of generic actions (Figure 2–3). This domain has been used to build two scenarios: The first involves nondeterministic perception actions thereby demonstrating the value of non-sequential (conditional) plans, while the second involves false beliefs and demonstrates the value of having an explicit representation of observability.

Scenario I: Block Search Task

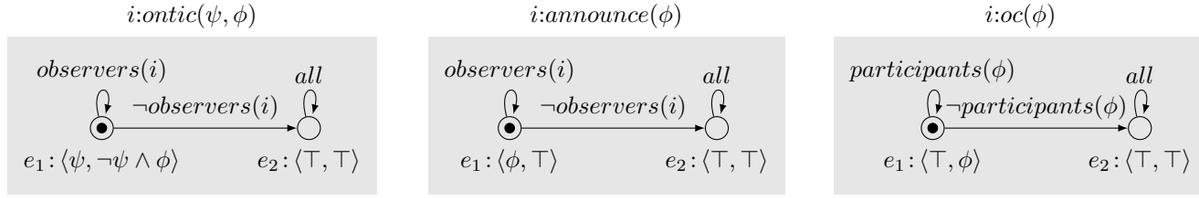
In this first scenario, two agents P and H, each want to know the location of some particular coloured cube. The setup consists of four boxes $box_1, \ldots, box_4$ and four uniquely colored cubes. The cubes selected by the agents are respectively denoted $cube_P$ and $cube_H$. The boxes are placed such that box_1, box_2 can be reached by the human and the other two by the robot. Each box contains one randomly selected cube and has a lid such that the agents can only determine the content of the box by lifting the lid and looking into the box. The goal formula is then of the form

$$\gamma = \bigvee_{i=1}^{4} B_P In(cube_P, box_i) \wedge \bigvee_{i=1}^{4} B_H In(cube_H, box_i)$$

Here the planner finds a policy where both agents look into each of their boxes and announce if they have found the other agent's cube.

Scenario II: False Belief Synthesis Task

In this second scenario, the robot and an agent H_1 have to collaborate to deceive another agent H_2, i.e., convince H_2 of some fact which is not true. The scenario involves two boxes box_1, box_2 and a cube $cube$. The shared goal between



$$observers(i) = \{j \mapsto j \lhd i \mid j \in \mathcal{A}\} \qquad all = \{i \mapsto \top \mid i \in \mathcal{A}\} \qquad participants(\phi) = \{i \mapsto \top \mid \exists j \in \mathcal{A} \text{ s.t. } i \lhd j \text{ in } \phi\}$$
$$\neg observers(i) = \{j \mapsto \neg j \lhd i \mid j \in \mathcal{A}\} \qquad \neg participants(\phi) = \{i \mapsto \top \mid \neg\exists j \in \mathcal{A} \text{ s.t. } i \lhd j \text{ in } \phi\}$$

Figure 3: The three remaining generic actions which are used in this paper in addition to $i{:}perceive(\phi)$ (See Figure 2). $i{:}ontic(, \psi, \phi)$ models a physical change in the world, e.g. $H{:}ontic(\top, In(cube_{red}, box_1))$ represents that agent H puts the red cube into box 1 ; $i{:}announce(phi)$ models that agent i announces whether i believes that ϕ holds or not; $i{:}oc(\phi)$ models a change in observability, e.g. $P{:}oc(P \lhd H \wedge H \lhd P)$ represents that P causes P and H to start observing each other.

H_1 and P is to deceive H_2 into believing that the cube is in box 1, that is $\gamma = \neg In(cube, box_1) \wedge B_{H_2} In(cube, box_1)$. [2]

For this scenario the robot has a special *distract* action modelled as $P{:}oc(\neg H_2 \lhd P \wedge \neg H_2 \lhd H_1)$ while only H_1 is able to perform the needed ontic (move cube) actions. The initial state $(\mathcal{M}, w_0)$ consists of a single world where all agents have full observability of each other and the cube is in box_1, $L(w_0) = \bigcup_{\forall i,j \in \mathcal{A}} \{i \lhd j\} \cup In(cube, box_2)$. The planner then finds a policy where H_1 first moves the cube into box_2, the robot then distracts H_2 allowing H_1 to move the cube back into box_1 without H_2 noticing.

Benchmarking

All experiments were performed on a machine with an Intel Core i9-7900X @ 4.0GHz and 128 GB of RAM. The algorithm implements four distinct optimisations; RU (unreachable worlds removed from states), GS (all identical nodes merged to one, i.e. graph search), BC (bisimulation contraction on states), and H (hashing is used to compare states). Three scenario instances were tested; Search(2) which is scenario I as described above, Search(1) which is a simplified scenario I where only one cube exists ($cube_H$), and FB Synth which is scenario II as described above. The scenario instances were run with all optimisations enabled and all but a single optimisation enabled.

The results can be seen in Table 1. Of note, disabling BC results in unchanged memory usage, while taking around 20% shorter to run. This is likely because the states were fairly small and therefore did not benefit from the contraction.

		Full	-RU	-GS	-BC	-H
Search(1)	[s]	0.10	1.45	-	**0.08**	1.07
	[MB]	**20**	38		**20**	**20**
Search(2)	[s]	17.1	-	-	**13.4**	-
	[MB]	**185**			**185**	
FB Synth	[s]	0.09	0.25	0.15	**0.07**	0.20
	[MB]	24	35	46	24	**19**

Table 1: Benchmarks. "-" is for 1 hour timeout

5 Related work

A significant amount of work on implementing epistemic planning has appeared in recent years, using several different techniques. Several authors (Kominis and Geffner 2015; Muise et al. 2015) propose a reduction of epistemic planning into a classical planning problem, through a compilation step which produces special belief propositions which can then be manipulated by STRIPS actions, thus allowing the usage of existing planners, but with the restriction to sequential plans and a fixed maximal depth of epistemic reasoning. Hu, Miller, and Lipovetzky (2019) extend pre-existing solvers supporting the use of external black-box functions with epistemic reasoning procedures. This allows them to support reasoning to an arbitrary depth including common knowledge, but has so far only been applied to **S5** logic, i.e. only reasoning about the knowledge of agents, not their (potentially incorrect) beliefs. Huang et. al. (2017) presents an AND-OR search-based contingent planning algorithm and a matching epistemic planner MEPK. They develop a framework for epistemic planning based upon a multi-agent KD45 logic and rewrite epistemic formulas into a special normal form suitable for planning. This allows them to do belief updates and belief revision, but prevents them from representing common belief.

Another line of work builds on the epistemic action language $m\mathcal{A}^*$ (Baral et al. 2015) allowing for formal specification of ontic, sensing and announcement actions. Le et. al. (2018) describes an epistemic planner EFP which performs a forward search over $m\mathcal{A}^*$ actions. It supports reasoning to an arbitrary depth, common knowledge and both beliefs and knowledge modalities, but its breadth-first search only produces sequential plans. Fabiano et. al. (2020) describes an extended language $m\mathcal{A}^\rho$ and a matching epistemic planner EFP 2.0, which replaces the underlying Kripke state representation with *possibilities* (Gerbrandy

[2] The fact that H_2 does not participate in the planning process can here be modelled by just not assigning any actions to H_2. Further machinery would be needed in order to handle adversarial settings, i.e. H_2 having his own goal.

and Groeneveld 1997). This representation inherently collapses bisimilar states, thereby pruning the search space.

There also exists prior work integrating epistemic planning with robotics systems in order to facilitate shared human-robot plan execution, e.g. featuring a Theory-of-Mind manager maintaining an explicit knowledge base for each agent, allowing the robot to provide the minimal information necessary to complete a plan (Devin and Alami 2016; Devin, Clodic, and Alami 2017).

6 Discussion and Future Work

A significant limitation of our current approach is the lack of support for belief updates, in that our framework does not currently support that an agent perceives some piece of information which they do not consider to be possible. This issues arises from the fact that seriality of epistemic models is not preserved under product update with actions having arbitrary preconditions, e.g. suppose for some state state $s = (\mathcal{M}, w_0)$ that some agent i falsely believes that some proposition p is true, that is, for all v s.t. $(w_0, v) \in R_i$ it holds that $(\mathcal{M}, v) \models p$. Now some other agent arrives and announces $\neg p$. Updating s with this announcement has the effect of deleting all worlds w' s.t. $(\mathcal{M}, w') \not\models \neg p$ including all of the aforementioned worlds, thus resulting in agent i now believing in *everything*. One possible solution, which remains future work, is to extend our framework with plausibility models (Baltag and Smets 2008). A second limitation is that the current algorithm is not yet capable of finding cyclic policies. Suppose that in the coin tossing domain, the goal is to have the coin showing heads. Due to the non-deterministic coin toss, there exists no acyclic policy which is guaranteed to succeed, and it can therefore not be solved by our planner. Investigating how to generalise to strong cyclic policies and support this in the algorithm remains future work. A third limitation is in the explicit representation of uncertainty as the set of all possible worlds, the size of which can potentially be exponential in the number of propositions. Investigating compacted representations remains future work.

References

[Aceto, Ingolfsdottir, and Srba 2012] Aceto, L.; Ingolfsdottir, A.; and Srba, J. 2012. The algorithmics of bisimilarity. *Advanced Topics in Bisimulation and Coinduction* 52:100–172.

[Baltag and Smets 2008] Baltag, A., and Smets, S. 2008. A qualitative theory of dynamic interactive belief revision. *Logic and the foundations of game and decision theory (LOFT 7)* 3:9–58.

[Baltag, Moss, and Solecki 1998] Baltag, A.; Moss, L. S.; and Solecki, S. 1998. The logic of public announcements and common knowledge and private suspicions. In Gilboa, I., ed., *Proceedings of the 7th Conference on Theoretical Aspects of Rationality and Knowledge (TARK-98)*, 43–56. Morgan Kaufmann.

[Baral et al. 2015] Baral, C.; Gelfond, G.; Pontelli, E.; and Son, T. C. 2015. An action language for multi-agent domains: Foundations. *arXiv preprint arXiv:1511.01960*.

[Blackburn and van Benthem 2006] Blackburn, P., and van Benthem, J. 2006. Modal logic: A semantic perspective. In *Handbook of Modal Logic*. Elsevier.

[Blackburn, de Rijke, and Venema 2001] Blackburn, P.; de Rijke, M.; and Venema, Y. 2001. *Modal Logic*, volume 53 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge, UK: Cambridge University Press.

[Bolander and Andersen 2011] Bolander, T., and Andersen, M. B. 2011. Epistemic planning for single- and multi-agent systems. *Journal of Applied Non-Classical Logics* 21:9–34.

[Bolander et al. 2018] Bolander, T.; Engesser, T.; Mattmüller, R.; and Nebel, B. 2018. Better eager than lazy? how agent types impact the successfulness of implicit coordination. In *Sixteenth International Conference on Principles of Knowledge Representation and Reasoning*.

[Bolander 2018] Bolander, T. 2018. Seeing is believing: Formalising false-belief tasks in Dynamic Epistemic Logic. In *Jaakko Hintikka on Knowledge and Game-Theoretical Semantics*. Springer. 207–236.

[Devin and Alami 2016] Devin, S., and Alami, R. 2016. An implemented theory of mind to improve human-robot shared plans execution. In *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 319–326. IEEE.

[Devin, Clodic, and Alami 2017] Devin, S.; Clodic, A.; and Alami, R. 2017. About decisions during human-robot shared plan achievement: Who should act and how? In *International Conference on Social Robotics*, 453–463. Springer.

[Dissing and Bolander 2020] Dissing, L., and Bolander, T. 2020. Implementing theory of mind on a robot using Dynamic Epistemic Logic. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*.

[Engesser et al. 2017] Engesser, T.; Bolander, T.; Mattmüller, R.; and Nebel, B. 2017. Cooperative epistemic multi-agent planning for implicit coordination. In *Proceedings of Methods for Modalities*, volume 243 of *Electronic Proceedings in Theoretical Computer Science*, 75–90.

[Fabiano et al. 2020] 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 International Conference on Automated Planning and Scheduling*, volume 30, 101–109.

[Gerbrandy and Groeneveld 1997] Gerbrandy, J., and Groeneveld, W. 1997. Reasoning about information change. *Journal of logic, language and information* 6(2):147–169.

[Hu, Miller, and Lipovetzky 2019] Hu, G.; Miller, T.; and Lipovetzky, N. 2019. What you get is what you see: Decomposing epistemic planning using functional STRIPS. *arXiv preprint arXiv:1903.11777*.

[Huang et al. 2017] Huang, X.; Fang, B.; Wan, H.; and Liu, Y. 2017. A general multi-agent epistemic planner based on higher-order belief change. In *IJCAI*, 1093–1101.

[Kominis and Geffner 2015] Kominis, F., and Geffner, H. 2015. Beliefs in multiagent planning: From one agent to

many. In *Twenty-Fifth International Conference on Automated Planning and Scheduling*.

[Le et al. 2018] Le, T.; Fabiano, F.; Son, T. C.; and Pontelli, E. 2018. EFP and PG-EFP: Epistemic forward search planners in multi-agent domains. In *Twenty-Eighth International Conference on Automated Planning and Scheduling*.

[McDermott 2000] McDermott, D. M. 2000. The 1998 AI planning systems competition. *AI magazine* 21(2):35–35.

[Muise et al. 2015] Muise, C.; Belle, V.; Felli, P.; McIlraith, S.; Miller, T.; Pearce, A. R.; and Sonenberg, L. 2015. Planning over multi-agent epistemic states: A classical planning approach. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

[Nilsson 1971] Nilsson, N. J. 1971. *Problem-solving methods in artificial intelligence*. McGraw-Hill computer science series. McGraw-Hill.

[Reifsteck et al. 2019] Reifsteck, D.; Engesser, T.; Mattmüller, R.; and Nebel, B. 2019. Epistemic multi-agent planning using Monte-Carlo Tree Search. In Benzmüller, C., and Stuckenschmidt, H., eds., *KI 2019: Advances in Artificial Intelligence*, 277–289. Cham: Springer International Publishing.

[Thomaz, Hoffman, and Cakmak 2016] Thomaz, A.; Hoffman, G.; and Cakmak, M. 2016. Computational human-robot interaction. *Foundations and Trends in Robotics* 4(2-3):105–223.