# Better Eager Than Lazy? How Agent Types Impact the Successfulness of Implicit Coordination

**Thomas Bolander**
DTU Compute
Technical University of Denmark
tobo@dtu.dk

**Thorsten Engesser** and
**Robert Mattmüller** and **Bernhard Nebel**
Faculty of Engineering, University of Freiburg, Germany
{engesser,mattmuel,nebel}@cs.uni-freiburg.de

## Abstract

Epistemic planning can be used for decision making in multi-agent situations with distributed knowledge and capabilities. Recent work proposed a new notion of strong policies with implicit coordination. With this it is possible to solve planning tasks with joint goals from a single-agent perspective without the agents having to negotiate about and commit to a joint policy at plan time. We study how and under which circumstances the decentralized application of those policies leads to the desired outcome.

## Introduction

One important task in multi-agent systems is to collaboratively reach a joint goal with multiple autonomous agents (e.g. robots and humans). For instance, if there is a group of robots that are supposed to reach target locations, they have to develop a plan that enables each robot to accomplish its goal. Taking, for instance the situation in Figure 1, where the circular robot $C$ wants to go to the cell marked by the solid circle and the square robot $S$ wants to reach the place with the solid square (the empty circle and square will only become important later). One could come up with the following plan: (i) $C$ moves to 2 and then to 4, (ii) $S$ moves to 2 and then to target location 3, and (iii) $C$ finally moves to target location 2.

This plan could be generated *centrally* by an external observer and then communicated to the two agents, which will execute it. We will assume, however, that all plans are developed by the agents in a *distributed* fashion. Assuming that the two agents can observe everything in the world, have full knowledge of their goals, and execution is deterministic, they both can come up with the same plan as above and execute this plan in a distributed way. If they came up with different plans but have anticipated that the other agents might deviate, then the joint execution might still be successful. We have to make strong assumptions about the *planning agent types*, though, as we will demonstrate.

The problem of planning and executing in a distributed fashion becomes significantly more difficult if we drop the assumption about full observability. In order to illustrate this point, let us again consider the situation in Figure 1, but unlike before, let us assume that each robot knows about their own target positions with certainty (the solid circle and square), but there is uncertainty about the target position of the other robot (the empty circle and square are considered as possible target positions for $C$ and $S$, respectively). This means that we still assume a common goal, namely each robot wants that in the end all the robots have reached their target positions. However, these target positions are not common knowledge. In such a situation, we will consider *policies* instead of plans, which can branch on observations and sensing actions. As it turns out, an agent can still come up with a successful policy which is *implicitly coordinated*, i.e., contains only steps such that the acting agent knows that her step contributes to reaching the goal. The key for generating such policies is to take *perspective shifts*, i.e., picturing oneself in the shoes of the other agent. Giving general success guarantees for the joint execution of *policy profiles* in a partially observable setting appears to be much more difficult than in the former case of full observability, though.

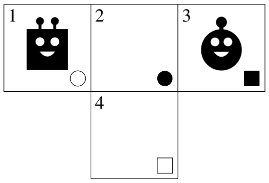


Figure 1: Multi-robot coordination example

Although taking into account the plans of other agents to achieve cooperation has been identified as an interesting topic of artificial intelligence research for a considerable amount of time (Konolige and Nilsson 1980), the application of implicit coordination has been limited almost exclusively to the fields of probabilistic robotics (Stulp, Isik, and Beetz 2006; Anderson and Papanikolopoulos 2008; Hollinger et al. 2009) and Dec-POMDPs (Spaan, Gordon, and Vlassis 2006). While existing classical planning approaches rely on continual (re-)planning (Brenner and Nebel 2009), the work we present in this paper is situated in the context of (multi-agent) epistemic planning, which can be approached algorithmically either by compilation to classical planning (Albore, Palacios, and Geffner 2009; Kominis and Geffner 2015; Muise et al. 2015) or by search in the space of “nested” (Bolander and Andersen 2011;

Engesser et al. 2017) or "shallow" knowledge states (Petrick and Bacchus 2002; 2004; Petrick and Foster 2013). Our concepts can be considered related to recent work in temporal epistemic logics (Bulling and Jamroga 2014; Jamroga and Ågotnes 2007; Jamroga and van der Hoek 2004), which addresses a question similar to ours, namely what groups of agents can jointly achieve under imperfect information. These approaches are based on concurrent epistemic game structures. Our approach is different in a number of ways, including: 1) As in classical planning, our actions and their effects are explicitly and compactly represented in an action description language (using the event models of Dynamic Epistemic Logic (van Ditmarsch, van der Hoek, and Kooi 2007)); 2) Instead of joint actions we have sequential action execution, where the order in which the agents act is not predefined; 3) None of the existing solution concepts considered in temporal epistemic logics capture the stepwise shifting of perspective underlying our notion of implicitly coordinated policies.

We expect implicit coordination to be helpful in settings of human robot collaboration, particularly when explicit coordination (i.e., through detailed instructions by the human) is inconvenient. In this regard, we consider our paper a first important step towards a general theory of implicit coordination in terms of identifying basic concepts and proving first results.

The rest of the paper is structured as follows. In Section 2, we describe the formal framework for representing states as the one in Figure 1, and how actions can change these states. Section 3 formalizes the notions of policies, policy profiles and their executions. In Section 4, we analyze the conditions under which the execution of policy profiles can be successful.

# Theoretical Background: DEL

## Epistemic States and Perspective Shifts

To represent planning problems as the one described above we need a formal framework where: (1) agents can reason about the first- and higher-order knowledge and ignorance of other agents; (2) both fully and partially observable actions can be described in a compact way. Dynamic Epistemic Logic (DEL) satisfies these conditions. We first very briefly recapitulate the foundations of DEL, following the conventions of Bolander and Andersen (Bolander and Andersen 2011).

In the following we will define epistemic languages, epistemic states and epistemic actions. All of these are defined relative to a given finite set of *agent names* (or simply *agents*) $\mathcal{A}$ and a given finite set of *atomic propositions* $P$. To keep the exposition simple, we will not mention the dependency on $\mathcal{A}$ and $P$ in the following. The *epistemic language* $\mathcal{L}_{\text{KC}}$ is

$$\varphi ::= \top \mid \bot \mid p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_i\varphi \mid C\varphi$$

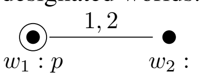
As usual, we read $K_i\varphi$ as "agent $i$ knows $\varphi$" and $C\varphi$ as "it is common knowledge that $\varphi$". Formulas are evaluated in *epistemic models* $\mathcal{M} = \langle W, (\sim_i)_{i\in\mathcal{A}}, V\rangle$ where the *domain* $W$ is a non-empty finite set of *worlds*; $\sim_i \subseteq W^2$ is an equivalence relation called the *indistinguishability relation* for agent $i$; and $V : P \to \mathcal{P}(W)$ assigns a *valuation* to each atomic proposition. For $W_d \subseteq W$, the pair $(\mathcal{M}, W_d)$ is called an *epistemic state* (or simply a *state*), and the worlds of $W_d$ are called the *designated worlds*. A state is called *global* if $W_d = \{w\}$ for some world $w$ (called the *actual world*), and we then often write $(\mathcal{M}, w)$ instead of $(\mathcal{M}, \{w\})$. We use $S^{\text{gl}}$ to denote the set of global states. For any state $s = (\mathcal{M}, W_d)$ we let *Globals*$(s) = \{(\mathcal{M}, w) \mid w \in W_d\}$. We define truth in states as follows, where the propositional cases are standard and hence left out:

$$\begin{array}{lll} (\mathcal{M}, W_d) \models \varphi & \text{iff} & (\mathcal{M}, w) \models \varphi \text{ for all } w \in W_d \\ (\mathcal{M}, w) \models K_i\varphi & \text{iff} & (\mathcal{M}, w') \models \varphi \text{ for all } w' \sim_i w \\ (\mathcal{M}, w) \models C\varphi & \text{iff} & (\mathcal{M}, w') \models \varphi \text{ for all } w' \sim w \end{array}$$

where $\sim$ is the transitive closure of $\bigcup_{i\in\mathcal{A}} \sim_i$.

A state $(\mathcal{M}, W_d)$ is called a *local state* for agent $i$ if $W_d$ is closed under $\sim_i$. Given a state $s = (\mathcal{M}, W_d)$, the *associated local state* of agent $i$, denoted $s^i$, is $(\mathcal{M}, \{v \mid v \sim_i w \text{ and } w \in W_d\})$. Going from $s$ to $s^i$ amounts to a *perspective shift* to the local perspective of agent $i$.

**Example 1.** Consider the global state $s = (\mathcal{M}, w_1)$ given as follows, where the nodes represent worlds, the edges represent the indistinguishability relations (reflexive edges left out), and ◉ is used for designated worlds:

$s =$ 

Each node is labeled with the name of the world, and the list of atomic propositions true at the world. In the state $s$, the proposition $p$ is true but agent 1 does not know this: $s \models p \wedge \neg K_1 p$. Hence from the local perspective of agent 1, $p$ cannot be verified, and we correspondingly have $s^1 \not\models p$ and $s^1 \not\models \neg p$.

## Epistemic Actions and Product Update

To model actions, we use the *event models* of DEL. An *event model* is $\mathcal{E} = \langle E, (\sim_i)_{i\in\mathcal{A}}, \text{pre}, \text{post}\rangle$ where the *domain* $E$ is a non-empty finite set of *events*; $\sim_i \subseteq E^2$ is an equivalence relation called the *indistinguishability relation* for agent $i$; $\text{pre} : E \to \mathcal{L}_{\text{KC}}$ assigns a *precondition* to each event; and $\text{post} : E \to \mathcal{L}_{\text{KC}}$ assigns a *postcondition* to each event. For all $e \in E$, *post*$(e)$ is a conjunction of literals (atomic propositions and their negations, including $\top$ and $\bot$). For $E_d \subseteq E$, the pair $(\mathcal{E}, E_d)$ is called an *epistemic action* (or simply *action*), and the events in $E_d$ are called the *designated events*. Similar to states, $(\mathcal{E}, E_d)$ is called a *local action* for agent $i$ when $E_d$ is closed under $\sim_i$.

Each event of an action represents a different possible outcome. By using multiple events $e, e' \in E$ that are indistinguishable (i.e. $e \sim_i e'$), it is possible to obfuscate the outcomes for some agent $i \in \mathcal{A}$, i.e. modeling partially observable actions. Using event models with $|E_d| > 1$, it is also possible to model sensing actions and nondeterministic actions (Bolander and Andersen 2011).

The *product update* is used to specify the successor state resulting from the application of an action in a state. Let a state $s = (\mathcal{M}, W_d)$ and an action $a = (\mathcal{E}, E_d)$ be given with $\mathcal{M} = \langle W, (\sim_i)_{i\in\mathcal{A}}, V\rangle$ and $\mathcal{E} = \langle E, (\sim_i)_{i\in\mathcal{A}}, \text{pre}, \text{post}\rangle$.

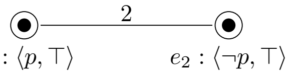
Then the *product update* of $s$ with $a$ is defined as $s \otimes a = (\langle W', (\sim'_i)_{i\in\mathcal{A}}, V'\rangle, W'_d)$ where

- $W' = \{(w,e) \in W \times E \mid \mathcal{M}, w \models \mathrm{pre}(e)\}$;
- $\sim'_i = \big\{((w,e),(w',e')) \in (W')^2 \mid w \sim_i w' \;\&\; e \sim_i e'\big\}$;
- $V'(p) = \{(w,e) \in W' \mid \mathrm{post}(e) \models p$ or
  $(\mathcal{M}, w \models p \text{ and } \mathrm{post}(e) \not\models \neg p)\}$;
- $W'_d = \{(w,e) \in W' \mid w \in W_d$ and $e \in E_d\}$.

We say that $a = (\mathcal{E}, E_d)$ is *applicable* in $s = (\mathcal{M}, W_d)$ if for all $w \in W_d$ there is an event $e \in E_d$ s.t. $(\mathcal{M}, w) \models \mathrm{pre}(e)$.

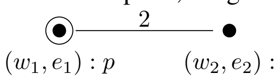
**Example 2.** Consider the following epistemic action $a = (\mathcal{E}, \{e_1, e_2\})$, using the same conventions as for epistemic states, except each event is labeled with $\langle \mathrm{pre}(e), \mathrm{post}(e)\rangle$:

$a =$


2
$e_1 : \langle p, \top\rangle$
$e_2 : \langle \neg p, \top\rangle$

It is a private sensing action for agent 1, where agent 1 privately gets to know the truth value of $p$, since $e_1$ and $e_2$ are distinguishable to agent 1 (and *in*distinguishable to agent 2). Letting $s$ be the state from Example 1, we get:

$s \otimes a =$



After the private sensing of $p$ by agent 1, agent 1 will know that $p$ is true, but agent 2 will still not: $s \otimes a \models K_1 p \wedge \neg K_2 p$.

In the following, we will systematically identify isomorphic states.

## Tasks, Policies and Executions

In this paper we consider *cooperative* planning tasks, that is, planning tasks in which the agents plan towards a joint goal (Engesser et al. 2017). Each action in a planning task is assumed to be executable by a unique agent, called the *owner* of the action. More precisely, given a set of actions $A$, an *owner function* is a mapping $\omega : A \to \mathcal{A}$ from actions to their owners. Mapping each action to a unique agent can be done without loss of generality, since semantically equivalent duplicates can always be added to the action set.

**Definition 1.** A *planning task* $\Pi = \langle s_0, A, \omega, \gamma\rangle$ consists of a global state $s_0$ called the *initial state*; a finite set of actions $A$; an owner function $\omega : A \to \mathcal{A}$; and a *goal formula* $\gamma \in \mathcal{L}_{\mathrm{KC}}$. We require that each $a \in A$ is local for $\omega(a)$.

**Example 3.** Consider the planning task $\langle s_0, \{a_1, a_2\}, \omega, p\rangle$ with initial state $s_0 = ⊙$ and two semantically equivalent actions $a_1 = ⊙\, e_1 : \langle \top, p\rangle$ and $a_2 = ⊙\, e'_1 : \langle \top, p\rangle$ for the owners $\omega(a_1) = 1$ and $\omega(a_2) = 2$ (both actions making the goal $p$ true unconditionally). Both the initial state $s_0$ and the effects of the actions $a_1$ and $a_2$ are fully observable for both agents. Intuitively, a solution should prescribe the action $a_1$ for agent 1 or the action $a_2$ for agent 2.

### Policies and Executions

Instead of working with sequential plans, our plans are going to be policies, representing instructions that can be individually followed by each of the agents. We impose some requirements on these policies to be reasonable.

First, we require actions to be applicable in all states they are assigned to (APP). Since the policies are required to be applicable from the agent perspective, we also require *uniformity* (UNIF). I.e., if the policy $\pi$ prescribes some action $a$ to agent $i$ in state $s$ and agent $i$ cannot distinguish $s$ from some other state $t$, then $\pi$ has to prescribe the same action $a$ for $i$ in $t$ as well. Finally, we require $\pi$ to be *unambiguous* for all agents in the sense that in each state $s$ where an agent $i$ is supposed to act according to $\pi$, $\pi$ is *deterministic* for agent $i$ (DET). More formally:

**Definition 2.** Let $\Pi = \langle s_0, A, \omega, \gamma\rangle$ be a planning task. Then a *policy* $\pi$ for $\Pi$ is a partial mapping $\pi : S^{\mathrm{gl}} \hookrightarrow \mathcal{P}(A)$, s. t.

(APP) f. a. $s \in S^{\mathrm{gl}}$, $a \in \pi(s)$: $a$ is applicable in $s$,

(UNIF) f. a. $s, t \in S^{\mathrm{gl}}$ s. t. $s^{\omega(a)} = t^{\omega(a)}$, $a \in \pi(s)$: $a \in \pi(t)$,

(DET) f. a. $s \in S^{\mathrm{gl}}, a, a' \in \pi(s)$ s. t. $\omega(a) = \omega(a')$: $a = a'$

The properties (UNIF) and (APP) together imply *knowledge of preconditions*, i.e., the property that in each state, an agent who is supposed to perform a particular action must also *know* that the action is applicable in that state. Note also that we must allow policies to sometimes prescribe multiple actions of different owners to the same state. This is because the set of indistinguishable states can differ between the agents. In some indistinguishable state for agent 1, action $a$ might be a necessary step towards the goal, while in some indistinguishable state for agent 2, action $b$ might be a necessary step towards the goal. Thus, by uniformity, both $a$ and $b$ have to be prescribed to the original state.

To characterize the different outcomes of agents acting according to a common policy, we define the notion of policy executions. As in more classical non-epistemic settings, relevant questions are whether the execution process terminates or not, and if it does, whether a goal state is reached.

**Definition 3.** An *execution* of a policy $\pi$ from a global state $s_0$ is a maximal (finite or infinite) sequence of alternating global states and actions $(s_0, a_1, s_1, a_2, s_2, \ldots)$, such that for all $m \geq 0$,

(1) $a_{m+1} \in \pi(s_m)$, and

(2) $s_{m+1} \in \mathit{Globals}(s_m \otimes a_{m+1})$.

An execution is called *successful* for a planning task $\Pi = \langle s_0, A, \omega, \gamma\rangle$, if it is a finite execution $(s_0, a_1, s_1, \ldots, a_n, s_n)$ such that $s_n \models \gamma$.

In the following, we will restrict our attention to policies that are guaranteed to achieve the goal after a finite number of steps. More formally, this means that all of their executions must be successful. As in nondeterministic planning, we call such policies *strong* (Cimatti et al. 2003).

**Definition 4.** For a planning task $\Pi = \langle s_0, A, \omega, \gamma\rangle$, a policy $\pi$ is called *strong* if $s_0 \in \mathrm{Dom}(\pi) \cup \{s \in S^{\mathrm{gl}} \mid s \models \gamma\}$ and for each $s \in \mathrm{Dom}(\pi)$, any execution of $\pi$ from $s$ is successful for $\Pi$. A planning task $\Pi$ is called *solvable* if a strong policy for $\Pi$ exists. For $i \in \mathcal{A}$, we call a policy $\pi$ *$i$-strong* if it is strong and $\mathit{Globals}(s_0^i) \subseteq \mathrm{Dom}(\pi) \cup \{s \in S^{\mathrm{gl}} \mid s \models \gamma\}$.

When a policy is $i$-strong it means that the policy is strong and defined on all the global states that agent $i$ cannot initially distinguish between. It follows directly from the definition that any execution of an $i$-strong policy from any of those initially indistinguishable states will be successful. So if agent $i$ comes up with an $i$-strong policy, it means that agent $i$ knows the policy to be successful.

We introduce the notion of reachability to talk about states that can occur during executions.

**Definition 5.** Given global states $s_0$ and $s$, we call $s$ *reachable* from $s_0$ if there are sequences of actions $a_1, \dots, a_n$ and states $s_1, \dots, s_n = s$ such that $a_{m+1}$ is applicable in $s_m$ and $s_{m+1} \in \mathit{Globals}(s_m \otimes a_{m+1})$ for all $m = 0, \dots, n-1$. We call $s$ reachable from $s_0$ *by following* a policy $\pi$ if it is part of an execution $(s_0, a_1, \dots, s, \dots)$ of $\pi$.

A strong policy $\pi$ is *implicitly coordinated* in the sense that at any point during its execution, at least one agent knows that it can execute a particular action as part of the strong policy $\pi$. This is formalized by the following proposition, that follows straightforwardly from Def. 4 and the uniformity condition in Def. 2.

**Proposition 1.** *Let $\pi$ be a strong policy for $\langle s_0, A, \omega, \gamma\rangle$ and let $s$ be a non-goal state reachable from $s_0$ by following $\pi$. Then for some $i \in \mathcal{A}$: $\pi(s) \cap \{a \mid \omega(a) = i\} \neq \emptyset$ and $\pi$ is an $i$-strong policy for $\langle s, A, \omega, \gamma\rangle$.*

Note that there are planning tasks where $i$-strong policies exist only for some of the agents. If these policies require other agents to act in some future states, Proposition 1 implies that these agents are able to find their own $i$-strong policies when re-planning from these states. However, to simplify our analysis, we will mostly consider tasks where there is an $i$-strong policy for all of the agents right away. Furthermore, we assume that the agents try to plan for all contingencies (as seen from their local perspective), so that it never becomes necessary to change policy due to unexpected actions by other agents. The relevant notion of “planning for all contingencies” in this setting is captured by what we call *maximality* of strong policies.

**Definition 6.** We call an $i$-strong policy $\pi$ for planning task $\Pi$ a *maximal $i$-strong policy* for $\Pi$ if $\pi$ is a strong policy for $\langle s, A, \omega, \gamma\rangle$ for all states $s$ such that: (1) $s$ is reachable from some $s_0' \in \mathit{Globals}(s_0^i)$, and (2) $\langle s, A, \omega, \varphi\rangle$ is solvable.

### Policy Profiles

Besides the centralized scenario in which one agent plans centrally for all agents, or equivalently, in which the involved agents can already coordinate on a common plan at plan time, we especially want to study the scenario in which the agents *cannot* coordinate their plans, but rather have to come up with plans individually. Those plans can easily differ, not only because of different reasoning capabilities of the different agents, but also because of their non-uniform knowledge of the initial state and of action outcomes. For our formal analysis, we define a *policy profile* for a planning task $\Pi$ to be a family $(\pi_i)_{i\in\mathcal{A}}$, where each $\pi_i$ is a policy for $\Pi$. We assume actions to be instantaneous and executed asynchronously. This leads us to the following generalization of executions to policy profiles.

**Definition 7.** An *execution* of a policy profile $(\pi_i)_{i\in\mathcal{A}}$ is a maximal (finite or infinite) sequence of alternating global states and actions $(s_0, a_1, s_1, \dots)$, such that for all $m \geq 0$,

(1) $a_{m+1} \in \pi_i(s_m)$ where $i = \omega(a_{m+1})$, and

(2) $s_{m+1} \in \mathit{Globals}(s_m \otimes a_{m+1})$.

We call such an execution successful if it is a finite execution $(s_0, a_1, s_1, \dots, a_n, s_n)$ such that $s_n \models \gamma$.

Note that there are two sources of nondeterminism for executions. One as a result from the possibility of multiple policies prescribing actions for their respective agents (item 1 in Def. 7). The other one results from the possibility of nondeterministic action outcomes (item 2 in Def. 7). Definition 4 implies that strong policies are *closed* in the usual sense that following a strong policy cannot lead to an “off-policy” non-goal state where the policy is undefined (Cimatti et al. 2003). As a first positive result, we can now show that *policy profiles* consisting of maximal strong policies are also *closed* in the sense that they do not produce dead-end executions, i. e. executions ending in a non-goal state where some of the policies in the profile are undefined. By inductive application of Proposition 1, we can show that in each execution step from $s$ via $a = \pi_i(s)$ to a non-goal state $s'$, the policy $\pi_i$ must be defined for $s'$, and by maximality of the other policies $\pi_j$, $j \neq i$, the policies of all other agents have to be defined in $s'$ as well. Hence:

**Proposition 2.** *Let $(\pi_i)_{i\in\mathcal{A}}$ be a policy profile where each $\pi_i$ is a maximal $i$-strong policy for task $\Pi$. Then $s \in \mathrm{Dom}(\pi_i)$ for all agents $i \in \mathcal{A}$ and non-goal states $s \in S^{gl}$ occurring in arbitrary executions $(s_0, a_1, \dots, s, \dots)$ of $(\pi_i)_{i\in\mathcal{A}}$.*

If all agents have *one strong policy in common* which all of them follow, then at execution time, the goal is guaranteed to be eventually reached. If, however, each agent acts on *its individual strong policy*, then the incompatibility of the individual policies may prevent the agents from reaching the goal, even though each individual policy is strong. The following example illustrates what may go wrong.

**Example 4.** Let $\Pi = \langle s_0, \{a_1, a_2\}, \omega, p\rangle$ be the planning task described in Example 3, and let $(\pi_1, \pi_2)$ be the policy profile consisting of the maximal 1-strong policy $\pi_1$ assigning only $a_2$ to $s_0$, and the maximal 2-strong policy $\pi_2$ assigning only $a_1$ to $s_0$. Here each agent expects the other agent to do the work, since the policy $\pi_1$ for agent 1 specifies the action $a_2$ belonging to agent 2 and vice versa. This profile has only one execution, the empty one, which is unsuccessful.

This shows that agents following maximal $i$-strong policies may still not reach the goal. The issue we see here is that the agents run into a “deadlock”, and the underlying reason is that both agents are “lazy”, expecting the other agent to act. In the following, we will discuss for which types of agents (lazy, eager, ...) and for which combinations of them success can or cannot be guaranteed.

## Agent Types

We distinguish between different agent types by distinguishing between the types of policies they produce. To that end, we identify agents with mappings from planning tasks to policies. Additionally, the agent mapping must be associated with the part the agent plays in the planning task, since a policy that is lazy from one agent's perspective may be eager from another agent's perspective; e.g. the policy $\pi_1$ in Example 4 is lazy for agent 1, but eager for agent 2.

**Definition 8.** A *planning agent* (or simply *agent*) is a pair $(i, T)$, where $i$ is an agent name and $T$ is a mapping from planning tasks to policies, such that $T(\Pi)$ is an $i$-strong policy for $\Pi$, whenever such a policy exists.

The requirement of $T(\Pi)$ being $i$-strong, whenever such a policy exists, comes from the fact that each agent should produce a policy that it knows will be successful, whenever it is possible to form such a policy. We can now extend the definition of executions to *groups* of agents $(i, T_i)_{i\in\mathcal{A}}$.

**Definition 9.** Let $(i, T_i)_{i\in\mathcal{A}}$ be a group of agents and let $\Pi$ be a planning task. Then the *executions* by $(i, T_i)_{i\in\mathcal{A}}$ of $\Pi$ are the executions of the policy profile $(T_i(\Pi))_{i\in\mathcal{A}}$.

## Lazy and Naively Eager Agents

We already saw that agents being lazy can be problematic. To formally capture laziness (and its dual, eagerness), we note that laziness essentially means having a preference against using one's own actions, and planning with someone else's actions instead. Similarly, eagerness means preferring one's own actions over someone else's. Intuitively, an agent has a preference for (or against) a set of actions if whenever a policy produced by that agent is defined, it prescribes at least one preferred action (or no unpreferred action), unless violating the preference is unavoidable in that state.

**Definition 10.** For a state $s$, a policy $\pi$, and a set of actions $A'$, we say that $\pi$ *uses* $A'$ in $s$ if $\pi(s) \cap A' \neq \emptyset$. Then we say that agent $(i, T)$ *has preference*

(1) *for* (the actions in) $A'$ if for all $\Pi$ and all $s \in \mathrm{Dom}(T(\Pi))$, policy $T(\Pi)$ uses $A'$ in $s$ unless no $i$-strong policy for $\Pi$ uses $A'$ in $s$, and

(2) *against* (the actions in) $A'$ if for all $\Pi$ and all $s \in \mathrm{Dom}(T(\Pi))$, policy $T(\Pi)$ does not use $A'$ in $s$ unless every $i$-strong policy for $\Pi$ uses $A'$ in $s$.

Unfortunately, *preference against* a set of actions is not the same as *preference for* its complement, which is why we need both notions. We can now define laziness as preference against one's own actions, that is, we call an agent $(i, T)$ *lazy* if it has preference against the actions in $\{a \in A \mid \omega(a) = i\}$.

To formalize in which sense Example 4 is problematic, we still have to define deadlocks, which intuitively are states where (1) something still *needs to be done*, where (2) it is known that something *can be done*, but where (3) nothing *will be done* because of incompatible individual policies.

**Definition 11.** A *deadlock* for a policy profile $(\pi_i)_{i\in\mathcal{A}}$ is a global state $s$ such that (1) $s$ is not a goal state, (2) $s \in \mathrm{Dom}(\pi_i)$ for some $i \in \mathcal{A}$, and (3) $\omega(a) \neq i$ for all $i \in \mathcal{A}$ and $a \in \pi_i(s)$.

Requirement (2) is included to distinguish deadlocks from *dead ends*, where none of the agents' policies prescribe an action, not even for another agent. From the above definitions and Example 4 we immediately get the following result.

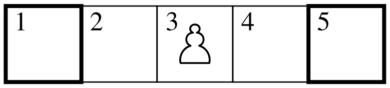


Figure 2: Planning task—move chess piece left or right.

**Proposition 3.** *There are solvable planning tasks for which all executions by lazy agents result in a deadlock.*

To avoid deadlocks, we define (naively) eager agents as agents who have a preference *for* their own actions. That is, we call an agent $(i, T)$ *naively eager* if it has a preference for the actions in $\{a \in A \mid \omega(a) = i\}$. They are called *naively* eager since it will turn out that in their eagerness they can *interfere* with other agents' plans and executions in a harmful way. But still, we first get the positive result that eagerness prevents the deadlocks we observed for lazy agents.

**Proposition 4.** *Let* $\Pi$ *be a planning task and* $(i, T_i)_{i\in\mathcal{A}}$ *be a group of naively eager agents. If each* $\pi_i = T_i(\Pi)$ *is a maximal i-strong policy, then all executions of* $(\pi_i)_{i\in\mathcal{A}}$ *are deadlock-free.*

*Proof sketch.* Assume for contradiction that $s$ is a deadlock for $(\pi_i)_{i\in\mathcal{A}}$. Then there has to exist an agent name $i \in \mathcal{A}$ and an action $a$ such that $a \in \pi_i(s)$ with $\omega(a) = j$ and $j \neq i$. Because $\pi_j$ is a maximal $j$-strong policy, we have $s \in \mathrm{Dom}(\pi_j)$. Then there also has to exist an $a' \in \pi_j(s)$ with $\omega(a') = j$, since $(j, T_j)$ is naively eager and has preference for its own actions. This contradicts item (3) of Definition 11. □

**Example 5.** Consider the scenario in Fig. 2. The chess piece can be moved left by agent 1 and right by agent 2 (one cell at a time). Everything is fully observable. The goal is to move the piece to one of the highlighted target cells. Every possible naively eager policy $\pi_1$ of agent 1 must assign action *left* to every non-goal state, and similarly, every naively eager policy $\pi_2$ of agent 2 must assign *right* to every non-goal state. Using $s_i$ to denote that the piece is in cell $i$, one possible execution of any such policy profile $(\pi_1, \pi_2)$ is the infinite sequence $(s_3, \textit{left}, s_2, \textit{right}, s_3, \textit{left}, \ldots)$, which is clearly not successful.

This shows that naively eager agents may also not reach the goal since they can potentially produce infinite executions.

**Proposition 5.** *There are solvable planning tasks for which some executions by naively eager agents are infinite.*

## Optimally Eager Agents

In order to address the stated problem, we will now consider agents who always try to simplify the problem by reaching states closer to the goal. This means that the agents should come up with *optimal policies*, policies that reach the goal in the fewest number of steps. In order to formally define optimal policies, we need the notion of *cost*. The cost of a policy can be defined as its worst-case execution length, that is, the number of actions in its longest possible execution. An optimal policy is then one of minimal cost. However, due to partial observability, different agents might assign different costs to the same policy, and hence disagree on which policies have minimal cost.

For instance, in a variant of Example 5, agent 1 might not know whether the chess piece is initially in cell 3 or 4, and agent 2 might not know whether it is initially in cell 2 or 3. Then agent 1 would assign cost 2 to the "go right" strategy, but cost 3 to the "go left" strategy (according to the knowledge of agent 1, the chess piece might initially be in cell 4, and hence 3 cells away from cell 1). Conversely, agent 2 would assign cost 2 to the "go left" strategy and cost 3 to the "go right" strategy. If both agents choose strategies of minimal cost, they would choose opposing strategies: agent 1 would want agent 2 to go right, and agent 2 would want agent 1 to go left. Clearly this will result in a deadlock.

To remedy this, we need the agents to measure cost in a "perspective-sensitive" way: the assigned cost takes the different perspectives of the involved agents into account.

**Definition 12.** Let $\pi$ be a strong policy for a planning task $\Pi$. The *perspective-sensitive cost* (or simply *cost*) of $\pi$ from a state $s \in \mathrm{Dom}(\pi)$, denoted $\kappa_\pi(s)$, is defined as:

$$\kappa_\pi(s) = \begin{cases} 0 & \text{if there exists no } a \in \pi(s) \\ 1 + \max_{a \in \pi(s), s' \in \mathit{Globals}(s^{\omega(a)} \otimes a)} \kappa_\pi(s') & \text{else.} \end{cases}$$

We extend this to local states $s$ with $\mathit{Globals}(s) \subseteq \mathrm{Dom}(\pi)$ by letting $\kappa_\pi(s) := \max_{s' \in \mathit{Globals}(s)} \kappa_\pi(s')$.

The following proposition captures the intuition that perspective-sensitive costs can only increase with additional uncertainty (by shifting perspective), and that in each global state $s$ with $\pi(s) \neq \emptyset$, one or more actions can be identified as the ones maximizing the perspective-sensitive cost for the successor state and thus defining the value of $\kappa_\pi(s)$. We will need this to prove deadlock-freedom in Proposition 7.

**Proposition 6.** *For any policy $\pi$, epistemic state $s$ and agent $i \in \mathcal{A}$, it holds that $\kappa_\pi(s) \leq \kappa_\pi(s^i)$. Moreover, if $\kappa_\pi(s) > 0$, then there is an action $a \in \pi(s)$ such that $\kappa_\pi(s) = \kappa_\pi(s^{\omega(a)})$.*

It can be verified that in the variant of Example 5 with partial observability about the initial state of the chess piece, both the "go left" and the "go right" strategy will have the same (perspective-sensitive) cost 3. The point is that the cost assigned to the "go left" strategy will be measured from the local state of the owner of the "go left" action, and similarly for "right", as seen from the Def. 12.

**Definition 13.** A policy $\pi$ for a planning task $\Pi = \langle s_0, A, \omega, \gamma \rangle$ is called *subjectively optimal* if for all $s \in \mathrm{Dom}(\pi)$, all $a \in \pi(s)$ and all $\omega(a)$-strong policies $\pi'$ for $\langle s, A, \omega, \gamma \rangle$ we have $\kappa_{\pi'}(s^{\omega(a)}) \geq \kappa_\pi(s^{\omega(a)})$.

**Definition 14.** Given a set of actions $A'$, we say that agent $(i, T)$ is *subjectively optimal with preference for the actions in $A'$*, if for all $\Pi$: (1) $T(\Pi)$ is an $i$-strong subjectively optimal policy if such a policy exists, and (2) $T(\Pi)$ uses $A'$ in each $s \in \mathrm{Dom}(\pi)$ unless no $i$-strong subjectively optimal policy for $\Pi$ uses $A'$ in $s$.

We call an agent that is subjectively optimal with preference for its own actions *optimally eager*. That is, a planning agent $(i, T)$ is called *optimally eager* if it is subjectively optimal with preference for the actions in $\{a \in A \mid \omega(a) = i\}$.

In the variant of Example 5 with partial observability about the initial state, optimally eager agents will always be successful. They assign the same cost to both the "go left" and "go right" strategies, but are eager, and will hence prefer the policy where they act themselves. So initially they specify conflicting actions. Assume agent 1 gets to act first and moves one cell left. In the resulting state, agent 2 assigns cost 3 to the "move right" strategy and only cost 2 to the "move left" strategy. Hence agent 2 will not try to prevent agent 1 from moving the chess piece to the far left.

On the other hand, an *optimally lazy* agent (which we could define analogously, by first defining *subjective optimality with preference against* own actions) would exhibit the same deadlock potential as naively lazy agents. We can also see this in Example 4, where both policies are in fact subjectively optimal ones. Our focus will thus be on optimally eager agents. We can indeed show that optimally eager agents do not produce deadlocks.

**Proposition 7.** *Let $\Pi$ be a planning task and $(i, T_i)_{i \in \mathcal{A}}$ be a group of optimally eager agents. If each $\pi_i = T_i(\Pi)$ is a maximal $i$-strong policy, then all executions of $(\pi_i)_{i \in \mathcal{A}}$ are deadlock-free.*

*Proof sketch.* Let $s$ be a reachable non-goal state. We analyze *waiting chains*, i. e., sequences of agents $i^1, \ldots, i^{n+1}$, such that (abbreviating $\pi_{i^j}$ as $\pi^j$, and $\kappa_{\pi^j}$ as $\kappa^j$), for all $j = 1, \ldots, n$, (1) there is no $a \in \pi^j(s)$ with $\omega(a) = i^j$, and (2) there is an $a \in \pi^j(s)$ with $\kappa^j(s^{\omega(a)}) = \kappa^j(s)$ and $\omega(a) = i^{j+1}$. By Proposition 6 and the definition of subjective optimality, we have $\kappa^{j+1}(s) \leq \kappa^{j+1}(s^{\omega(a)}) \leq \kappa^j(s^{\omega(a)}) = \kappa^j(s)$ for all $j = 1, \ldots, n$. This implies that no agent can occur more than once in a waiting chain, since that would directly contradict its eagerness. Thus, if $s \in \mathrm{Dom}(\pi^1)$ and $\pi^1(s) \neq \emptyset$ for some agent $i^1 \in \mathcal{A}$, then there has to exist a maximal waiting chain $i^1, \ldots, i^n$, where the last agent $i^n$ has an action $a \in \pi^n(s)$ such that $\omega(a) = i^n$. □

We can also show that all agents being optimally eager prevents infinite executions in the simple setting with uniform observability. We call a planning task $\langle s_0, A, \omega, \gamma \rangle$ *uniformly observable* if all agents share the same indistinguishability relations, both in the initial state $s_0$ and in all actions $a \in A$, which is tantamount to assuming that there is a *single* agent planning in the *belief space* of a partially observable nondeterministic (POND) problem (Bonet and Geffner 2000).

**Proposition 8.** *Let $\Pi$ be a uniformly observable and solvable planning task and let $(i, T_i)_{i \in \mathcal{A}}$ be a group of optimally eager agents. Then all executions by $(i, T_i)_{i \in \mathcal{A}}$ of $\Pi$ are finite.*

*Proof sketch.* Let $\pi_i = T_i(\Pi)$ for each agent $i \in \mathcal{A}$. Then for any transition $(\ldots, s, a, s', \ldots)$ occurring in an execution, we have $\kappa_{\pi_i}(s') \leq \kappa_{\pi_i}(s) - 1$ for the acting agent $i = \omega(a)$. Due to uniform observability and optimality, $\kappa_{\pi_i}(s') = \kappa_{\pi_j}(s')$ for any $j \in \mathcal{A}$ with $s' \in \mathrm{Dom}(\pi_j)$. Thus, by monotonicity, the execution ends after at most $\kappa_{\pi_i}(s')$ more actions. □

This means that in the uniformly observable setting, we can guarantee each execution to be successful, given all agents are optimally eager and act with respect to a maximal strong policy. Our result follows directly from Propositions 7 and 8.

**Proposition 9.** *Let* $\Pi$ *be a uniformly observable planning task and* $(i, T_i)_{i\in\mathcal{A}}$ *be a group of optimally eager agents. If each* $\pi_i = T_i(\Pi)$ *is a maximal* $i$*-strong policy, then all executions of* $(\pi_i)_{i\in\mathcal{A}}$ *are successful.*

Unfortunately, if there is non-uniform observability, optimally eager agents cannot always prevent infinite executions, as we see in the following example.

**Example 6.** Consider another variant of Example 5, where the initial position of the chess piece is again fully observable, but where the information about possible target cells is non-uniformly distributed. The initial state is given as $s_3 = \begin{tikzcd} \underset{at_3, t_1}{\bullet} \arrow[r, dash, "1"] & \underset{at_3, t_1, t_5}{\odot} \arrow[r, dash, "2"] & \underset{at_3, t_5}{\bullet} \end{tikzcd}$ with $at_i$ meaning that the piece is in cell $i$, and $t_i$ meaning that cell $i$ is a target position. The joint goal is $\gamma = (t_1 \to at_1) \land (t_5 \to at_5)$. Since agent 1 only knows that cell 1 is a target while agent 2 only knows that cell 5 is one, optimally eager agents would produce policies where they move the piece always in their own direction. Similar to Example 5, an infinite execution would then be $(s_3, \textit{left}, s_2, \textit{right}, s_3, \textit{left}, \ldots)$.

We can see from Example 6 that it is generally not possible to solve the problem of infinite executions just by imposing restrictions on the types of agents. Since, in this example, for each state $s$ and agent $i$ there is only one possible choice of action as part of an $i$-strong policy (*left* for agent 1, *right* for agent 2), every conceivable combination of planning agents produces infinite executions. Hence we get the following:

**Proposition 10.** *For every group of at least two agents* $(i, T_i)_{i\in A}$ *there exists a partially observable and solvable planning task* $\Pi$ *that has unsuccessful executions by* $(i, T_i)_{i\in A}$ *of* $\Pi$.

It is important to note that planning tasks with non-uniform knowledge do exist in which implicit coordination by optimally eager agents is guaranteed to be successful, i.e., without the potential occurrence of infinite executions. In particular, by allowing communication between the agents to be modeled directly as part of the planning task (using announcement actions), it is possible to solve more problems. One example in this class of planning tasks is the robots example from the introduction. To guarantee the existence of strong policies, we enable a robot that has reached its target position to publicly announce that fact as its final action (e.g., by visibly powering down).

A subjectively optimal policy for the square robot (that can be easily extended to a maximal, optimally eager one) is depicted in Figure 3. Solid edges denote actions and dashed edges denote indistinguishability. For clarity, only such indistinguishability edges are shown that refer to the agent designated to act and that, via uniformity, enforce inclusion of some action in the policy. Here, the square robot starts by moving out of the way of the circular robot, in order to allow the circular robot to move to the leftmost cell. This is because only from this position, the circular robot can make sure that the square robot will be able to reach its goal cell. Independently of the actual goal cell of the square robot, the square robot will then be able to move there and power down, after which the circular robot can finish the task. Note that this strategy will succeed for the given global initial state no matter which strong policy the circular robot chooses, just provided it is subjectively optimal. If the actual goal cell for the circular robot was the leftmost one, an optimally eager circular robot would already try to announce and power down earlier when having reached its destination. This contingency is covered by the maximal version of the policy (or with re-planning).

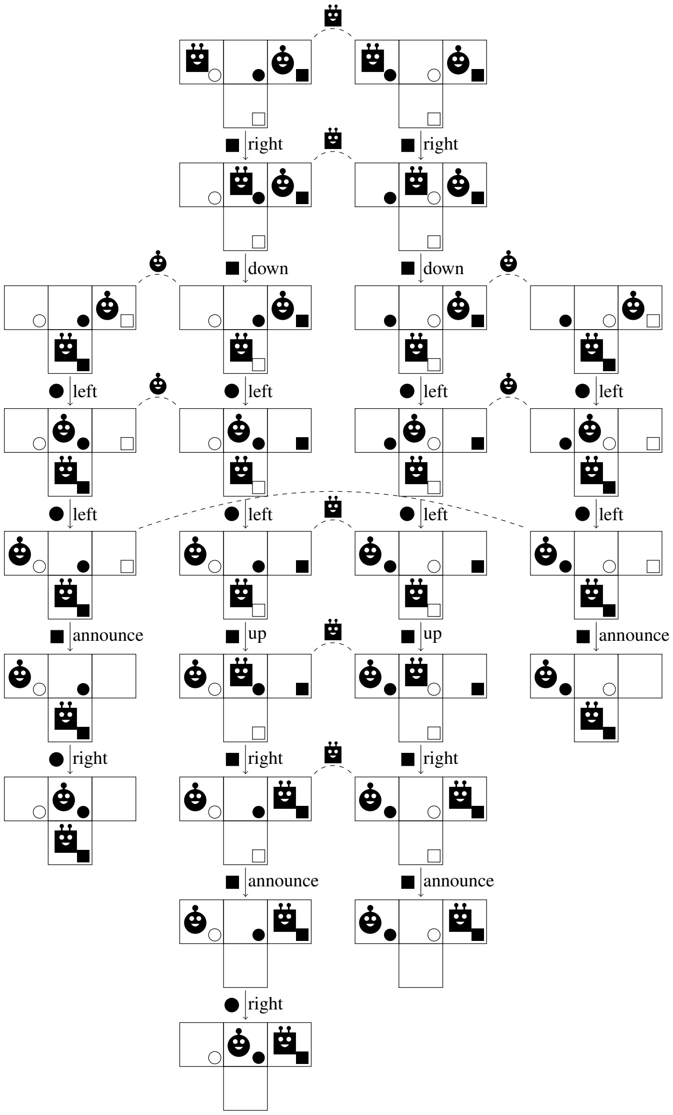


Figure 3: Depiction of a strong policy for the robots example

## Conclusion and Discussion

We investigated how agent types impact the successfulness of implicit coordination in cooperative multi-agent planning with distributed knowledge and capabilities. We distinguished between *lazy* and *eager* agents and saw that lazy agents may produce *deadlocks* (waiting for one another to move), a problem that does not show up with eager agents. However, it turned out that over-eager agents can produce *infinite executions* instead (unintentionally working against each other), which can only be avoided under rather strong assumptions, namely if the agents optimize what we termed perspective-sensitive costs and if they have uniform observability. Under non-uniform observability, even optimally eager agents may unintentionally sabotage each other.

This means that there is no general positive result for non-uniformly observable settings such as the motivating multi-robot coordination example with uncertain target positions (Fig. 1). Still, in that particular example, implicit coordination does work and we can guarantee a successful execution taking both robots to their targets, if we allow the first robot that reaches its target to publicly announce that fact. However, to ensure successful implicit coordination, the square robot has to move first, and the total number of moves (excluding the announcement) will be 7 instead of 5 as in the full observability case.

For future work, we plan to investigate whether unintentional sabotage (like the infinite executions we observed in Example 6) can be avoided if we change our assumptions about the type of reasoning performed by the agents. E.g., one assumption we made in this paper is that policies are *history-independent* functions of the current state. In contrast, we could allow agents to infer additional knowledge based on previous states and the assumption of optimality of the other agents' past actions. In our example, the move of the chess piece in one direction should act implicitly as announcement that there is a target cell on that side (since otherwise, the action would complicate the problem). Similarly, in the robots example, it should be possible for the square robot to signal to the circle agent that the bottom cell is the target cell just by performing a *noop* action on that cell. Moving upwards would signal that the target cell is the one on the right. Thus, having explicit noop actions would render the additional announcement actions unnecessary. We believe that by extending our solution concept to include this kind of reasoning, we will be able to solve a wider range of cooperative tasks using implicit coordination.

## References

Albore, A.; Palacios, H.; and Geffner, H. 2009. A translation-based approach to contingent planning. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*, 1623–1628.

Anderson, M., and Papanikolopoulos, N. 2008. Implicit cooperation strategies for multi-robot search of unknown areas. *Journal of Intelligent and Robotic Systems* 53(4):381–397.

Bolander, T., and Andersen, M. B. 2011. Epistemic planning for single and multi-agent systems. *Journal of Applied Non-Classical Logics* 21(1):9–34.

Bonet, B., and Geffner, H. 2000. Planning with incomplete information as heuristic search in belief space. In *Proceedings of the 5th International Conference on Artificial Intelligence Planning Systems (AIPS 2000)*, 52–61.

Brenner, M., and Nebel, B. 2009. Continual planning and acting in dynamic multiagent environments. *Autonomous Agents and Multi-Agent Systems* 19(3):297–331.

Bulling, N., and Jamroga, W. 2014. Comparing variants of strategic ability: how uncertainty and memory influence general properties of games. *Autonomous Agents and Multi-Agent Systems* 28(3):474–518.

Cimatti, A.; Pistore, M.; Roveri, M.; and Traverso, P. 2003. Weak, strong, and strong cyclic planning via symbolic model checking. *Artificial Intelligence* 147(1–2):35–84.

Engesser, T.; Bolander, T.; Mattmüller, R.; and Nebel, B. 2017. Cooperative epistemic multi-agent planning for implicit coordination. In *Proceedings of the 9th Workshop on Methods for Modalities (M4M 2017)*, 75–90.

Hollinger, G.; Singh, S.; Djugash, J.; and Kehagias, A. 2009. Efficient multi-robot search for a moving target. *The International Journal of Robotics Research* 28(2):201–219.

Jamroga, W., and Ågotnes, T. 2007. Constructive knowledge: what agents can achieve under imperfect information. *Journal of Applied Non-Classical Logics* 17(4):423–475.

Jamroga, W., and van der Hoek, W. 2004. Agents that know how to play. *Fundam. Inform.* 63(2–3):185–219.

Kominis, F., and Geffner, H. 2015. Beliefs in multiagent planning: From one agent to many. In *Proceedings of the 25th International Conference on Automated Planning and Scheduling (ICAPS 2015)*, 147–155.

Konolige, K., and Nilsson, N. J. 1980. Multiple-agent planning systems. In *Proceedings of the 1st Annual National Conference on Artificial Intelligence (AAAI 1980)*, 138–142.

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 *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI 2015)*, 3327–3334.

Petrick, R. P. A., and Bacchus, F. 2002. A knowledge-based approach to planning with incomplete information and sensing. In *Proceedings of the 6th International Conference on Artificial Intelligence Planning Systems (AIPS 2002)*, 212–222.

Petrick, R. P. A., and Bacchus, F. 2004. Extending the knowledge-based approach to planning with incomplete information and sensing. In *Proceedings of the 14th International Conference on Automated Planning and Scheduling (ICAPS 2004)*, 2–11.

Petrick, R. P. A., and Foster, M. E. 2013. Planning for social interaction in a robot bartender domain. In *Proceedings of the 23rd International Conference on Automated Planning and Scheduling (ICAPS 2013)*, 389–397.

Spaan, M. T.; Gordon, G. J.; and Vlassis, N. 2006. Decentralized planning under uncertainty for teams of communicating agents. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006)*, 249–256.

Stulp, F.; Isik, M.; and Beetz, M. 2006. Implicit coordination in robotic teams using learned prediction models. In *Proceedings 2006 IEEE International Conference on Robotics and Automation (ICRA 2006)*, 1330–1335.

van Ditmarsch, H. P.; van der Hoek, W.; and Kooi, B. 2007. *Dynamic Epistemic Logic*. Springer Heidelberg.