An Action Language for Reasoning about Beliefs in Multi-Agent Domains

Chitta Baral and **Gregory Gelfond**
Ira A. Fulton School of Engineering
Arizona State University
Tempe, AZ 85281-8809
chitta@asu.edu,logicprogrammer@gmail.com

Enrico Pontelli and **Tran Cao Son**
Department of Computer Science
New Mexico State University
Las Cruces, NM 88011
epontell|tson@cs.nmsu.edu

Abstract

The paper presents a novel action language for multi-agent domains, $m\mathcal{A}$+ , which generalizes action languages to multi-agent domains. The language allows representation and reasoning about different types of actions such as ontic (world-altering) actions, sensing actions, and announcements. It also allows the specification of agents' dynamic awareness w.r.t. action occurrences. $m\mathcal{A}$+ considers three different kinds of awareness role: fully aware, partially aware, and completely ignorant (oblivious) of the action's occurrence and its effects. The semantics of $m\mathcal{A}$+ relies on states (pointed Kripke models), used to encode the agent's knowledge and the real state of the world, and is defined by a transition function. The paper identifies a class of definite action theories whose set of initial states is finite and thus can be implemented. Finally, $m\mathcal{A}$+ is related to other formalisms for reasoning about actions in multi-agent domains.

Introduction and Motivation

Reasoning about Actions and Change (RAC) has been an important goal in AI since McCarthy's 1959 work (McCarthy 1959), where he illustrates "programs with common-sense" through reasoning about actions to fulfill "wants". In the early days of AI, RAC played a driving role in the development of logics (especially, non-monotonic logics) (McCarthy 1980; McDermott and Doyle 1980; Reiter 1980) and languages to represent knowledge. It gave rise to the important "frame problem"[1] (McCarthy and Hayes 1969) that was the focus of AI research for several decades (Brown 1987; Hanks and McDermott 1987; Kartha 1994; Lifschitz 1987; Lin and Reiter 1994; Reiter 1991; 2001; Sandewall 1994; Shanahan 1997; Scherl and Levesque 1993). During the 1990's, significant progress was made in reasoning about actions and in solving the frame problem under various conditions (Baral and Gelfond 1997; Baral 1995; Gelfond and Lifschitz 1993; Giunchiglia, Kartha, and Lifschitz 1997; Giunchiglia and Lifschitz 95; Lifschitz 1997; Kartha and Lifschitz 1994; McCain and Turner 1995; Turner 1997). However, the focus was mostly on a single agent operating in an environment. The relatively fewer proposals on multiple agents (Baral and Gelfond 1997; Lin and Shoham 1995) do not consider the interplay between the agents' knowledge, nor they consider actions that impact the agents' knowledge about other agents' knowledge. One significant contribution of the works in this decade was the development of action and domain specification languages with independent semantics (Baral and Gelfond 1997; Gelfond and Lifschitz 1993; 1998; Giunchiglia and Lifschitz 98), which allow the precise statement and proof of results about frame problems and about the encoding of planning domains in various logics. In summary, while earlier work on RAC gives us some ideas on how to design action specification languages, address frame problems, and develop planners, they barely touch these issues in a multi-agent setting.

To illustrate our points, let us introduce an example that will be used throughout the paper.

Example 1 Consider a simple scenario with three agents A, B, and C, where agent A, a friend of C (a double agent in the organization of B), is in the custody of a hostile agent B. A needs C's help to escape. A also needs to take the key without B knowing about it and makes C aware that he has the key. He cannot do that while B is watching. C can only be aware of A's action if he is present and watching A. C can make A aware of his presence by making noise (e.g., shouting). A simple plan that helps A to escape is as follows.

Agent A waits for C to be present, distracts B, signals C (so that C is watching A), and then takes the key.

The plan requires actions such as *"distract"* and *"signal"* which, at first glance, may seem like world changing actions, such as picking up a block; but upon further analysis, these actions alter the world in such a way that some subsequent actions (e.g., take the key) have impact not only on the world, but also on agents' knowledge and beliefs about the world and about other agents' knowledge and beliefs. For example, after A distracts B and takes the key, B will not know that A has the key, and will believe that A does not have it; A knows that B does not know that A has the key. Similarly, when A signals C and takes the key, C will know that A has the key, and A will know that C knows that. □

[1]The frame problem is to represent the principle that properties of the world (*fluents*) do not change unless they are forced to change due to an action. In AI, this was first mentioned in (McCarthy and Hayes 1969), but one can trace the notion to Leibniz (Leibniz c 1679) and Newton's laws of motion (Newton 1687).

Multi-agent domains and reasoning about agents' knowledge about other agents' knowledge has been explored in the Theoretical Aspect of Rationality and Knowledge (TARK) and Dynamic Epistemic Logic (DEL) communities (Baltag and Moss 2004; Boella and van der Torre 2005; Fagin et al. 1995; Gerbrandy 2006; Herzig and Troquard 2006; Meyer 2000; Sauro et al. 2006; Spaan, Gordon, and Vlassis 2006; van Benthem, van Eijck, and Kooi 2006; van der Hoek, Jamroga, and Wooldridge 2005; van Ditmarsch, van der Hoek, and Kooi 2007). More recent proposals (Baltag and Moss 2004; van Benthem, van Eijck, and Kooi 2006; van Ditmarsch, van der Hoek, and Kooi 2007) address the issue of describing how to *change* knowledge and beliefs. Among the various proposals, the formalisms of action models (Baltag and Moss 2004) and update models (van Benthem, van Eijck, and Kooi 2006) have been well accepted by the community. While action and update models are good tools to compute change in knowledge, they hard-code the roles of agents; thus, they do not allow the dynamic manipulation of the roles of agents (through actions such as "distract" and "signal"), that is needed to reason and plan for goals of the kind we mentioned in the example (A wants to be free). It is well-accepted that the knowledge and belief of agents about the world and about others' beliefs can be represented by a pointed Kripke structure (defined later). Changes caused by the execution of actions are reflected by updates on the pointed Kripke structure. Let us continue with Example 1.

Example 2 The states of the world in Example 1 can be described using the following fluents:[2] $has\_key(x)$—x has the key; $custody(x,y)$—x is in the custody of y; $present(x)$—x is present; and $watching(x,y)$—x is watching y; x and y denote different agents in $\{A,B,C\}$.

Let us consider the situation in which A is in custody of B, B is watching A, C is not watching A, nobody has the key, and everybody is present. Everyone knows that A is in custody of B, B is watching A, C is not watching A, nobody has the key, and A and B are present. A and C are aware that C is present but B is unaware of the presence of C.

The state of the world as well as the state of knowledge of A, B, and C can be represented using the pointed Kripke structure shown in Figure 1, with two state symbols s_1 and s_2, whose interpretations[3] are given by the sets $\{watching(B,A), custody(A,B), present(C)\} \cup P_{AB}$ and $\{watching(B,A), custody(A,B)\} \cup P_{AB}$, where $P_{AB} = \{present(B), present(A)\}$, respectively. The links labeled A,B,C represent the accessibility relations of agents, indicating their uncertainty about the world. A double-circled node indicates the real state of the world.

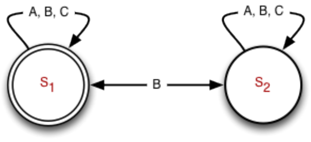

Figure 1: A pointed Kripke Structure

[2]We also need fluents for describing the location of the agents. For simplicity, we only describe whether an agent is present or not.

[3]Fluents listed in the interpretation are true.

Assume that the agent A distracts agent B. Intuitively, B will not be watching A after the execution of this action, i.e., the real state of the world can be encoded by the interpretation $s_3 = \{custody(A,B), present(C)\} \cup P_{AB}$. Accordingly, we have $s_4 = s_2 \setminus \{watching(B,A)\}$. The resulting structure, however, depends on awareness of the agents about the execution of the action and its effects. The structure in Fig. 2 describes the situation when *all agents* are aware of the action's execution and effects. In this case, everyone has the same knowledge that B is no longer watching A which is also true in the real state of the world.

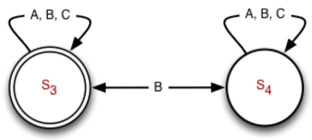

Figure 2: Everyone is aware that A distracts B

The situation is different when only A and B are aware of the action's execution and effects. As such, only A and B have the knowledge that B is no longer watching A, while C still believes that B is watching A and that everybody believes that B is watching A; B is still not aware of the presence of C. This is depicted in Fig. 3.

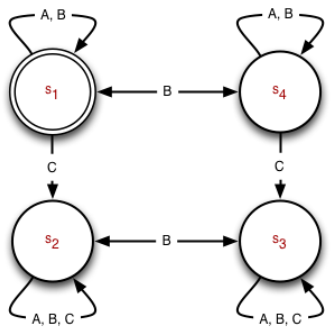

Figure 3: Everyone but C is aware that A distracts B

To account for different contexts, in which an action is executed, approaches to modeling changes using *program models* or *update models* (e.g., (Baltag and Moss 2004; van Benthem, van Eijck, and Kooi 2006)) would need to develop different update models. For the two situations discussed above, the update models are given in Fig. 4, the left and right model will update the structure in Fig. 1 to the structures in Fig. 2 and Fig. 3, respectively. □

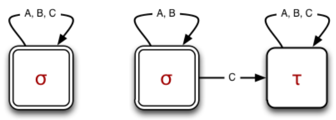

Figure 4: Update models

The discussion in Example 2 shows that approaches to modeling actions and change based on update models are not elaboration tolerant. Furthermore, the approach leaves one critical question unanswered, namely "how should an update model be constructed given an English description of the effects of an action?" For instance, given the statement about the effect of the action *distract*: "A distracts B causes B not watching A," why do we use the two update models in Fig. 4 and how do we construct them.

In this paper we propose a novel action language, $m\mathcal{A}$+ , for multi-agent domains with the following salient features:

- In $m\mathcal{A}$+ , we allow dynamic specification of roles of agents and separate the specification of action effects from the agents' roles. This is important as this allows specification of actions like signal and distract. The impact of this is that, using $m\mathcal{A}$+ , plans can be made to manipulate other agents' observability. All this could not be specified in $m\mathcal{A}$ (Baral et al. 2010) or in any of the existing DEL-based formalisms.
- The semantics of $m\mathcal{A}$+ is defined using the notion of update models from dynamic epistemic logic. This not only make the semantic characterization simpler, but serves as one of the bridges between research in DEL and reasoning about actions using high level languages.

Preliminaries

Belief Formulae and Kripke Structures

We assume an environment with a set $\mathcal{AG}$ of n agents. The state of the world can be characterized by a set $\mathcal{F}$ of propositional variables, called *fluents*. We will be concerned with the belief of agents about the environment and about the belief of other agents. For this purpose, we adapt the logic of knowledge and the notations used in (Fagin et al. 1995). We associate to each agent i a modal operator $\mathbf{B}_i$ and represent the belief of an agent as belief formulae in a logic extended with these operators. Formally:

- *Objective formulae:* an *objective formula* is a propositional formula built over $\mathcal{F}$ and propositional logical connectives $\vee$, $\wedge$, $\rightarrow$, $\neg$, etc. $f \in \mathcal{F}$ is called an *fluent atom.* A *fluent literal* is either $f \in \mathcal{F}$ or its negation $\neg f$.
- *Belief formulae:* a *belief formula* (or, simply, *formula*) is:
  - an objective formula;
  - a formula of the form $\mathbf{B}_i\varphi$ where φ is a belief formula;
  - a propositional combination of belief formulae;
  - a formula of the form $\mathbf{E}_\alpha\varphi$ or $\mathbf{C}_\alpha\varphi$ where φ is a formula and non-empty set $\alpha \subseteq \mathcal{AG}$.

Formulae of the form $\mathbf{E}_\alpha\varphi$ and $\mathbf{C}_\alpha\varphi$ are referred to as *group formulae*. Whenever $\alpha = \mathcal{AG}$, we simply write $\mathbf{E}\varphi$ and $\mathbf{C}\varphi$. $\mathcal{L}_{\mathcal{AG}}$ is the language of all belief formulae over $\mathcal{F}$ and $\mathcal{AG}$.

Intuitively, belief formulae are used to describe the beliefs of agents concerning the state of the world as well as about the beliefs of other agents. For example, the formula $\mathbf{B}_1 f$ states that "*1 believes that f is true*," while $\mathbf{B}_1\mathbf{B}_2 p$ states that "*1 believes that 2 believes that p is true*."

Definition 1 (Kripke Structure) *A Kripke structure is a tuple $\langle S, \pi, \mathcal{B}_1, \ldots, \mathcal{B}_n\rangle$, where S is a set of state symbols, π is a function that associates an interpretation of $\mathcal{F}$ to each element of S, and $\mathcal{B}_i \subseteq S \times S$ for $1 \leq i \leq n$.*

A pointed Kripke structure *(or* state*) is defined as a pair (M, s) where M is a Kripke structure and $s \in S$.*

Satisfaction between formulae and states is defined next.

Definition 2 *Given a formula φ, a Kripke structure $M = \langle S, \pi, \mathcal{B}_1, \ldots, \mathcal{B}_n\rangle$, and a state symbol $s \in S$:*

1. $(M, s) \models \varphi$ if φ is a fluent formula and $\pi(s) \models \varphi$;
2. $(M, s) \models \mathbf{B}_i\varphi$ if for each t such that $(s, t) \in \mathcal{B}_i$, $(M, t) \models \varphi$;
3. $(M, s) \models \neg\varphi$ if $(M, s) \not\models \varphi$;
4. $(M, s) \models \varphi_1 \vee \varphi_2$ if $(M, s) \models \varphi_1$ or $(M, s) \models \varphi_2$;
5. $(M, s) \models \varphi_1 \wedge \varphi_2$ if $(M, s) \models \varphi_1$ and $(M, s) \models \varphi_2$.
6. $(M, s) \models \mathbf{E}_\alpha\varphi$ if $(M, s) \models \mathbf{B}_i\varphi$ for every $i \in \alpha$.
7. $(M, s) \models \mathbf{C}_\alpha\varphi$ if $(M, s) \models \mathbf{E}^k_\alpha\varphi$ for every $k \geq 0$ where $\mathbf{E}^1_\alpha\varphi = \mathbf{E}_\alpha\varphi$ and $\mathbf{E}^{k+1}_\alpha = \mathbf{E}_\alpha(\mathbf{E}^k_\alpha\varphi)$.

For readability, we use $M[S]$, $M[\pi]$, and $M[i]$, to denote the components S, π, and $\mathcal{B}_i$ of M, respectively.

Example 3 The pointed Kripke structure in Fig. 1 is described by (M_1, s_1) with $M_1 = \langle\{s_1, s_2\}, \pi, \mathcal{B}_A, \mathcal{B}_B, \mathcal{B}_C\rangle$ where $\mathcal{B}_A = \mathcal{B}_C = \{(s_1, s_1), (s_2, s_2)\}$, $\mathcal{B}_B = \{(s_1, s_1), (s_2, s_2), (s_1, s_2), (s_2, s_1)\}$ and $\pi(s_1)(f) = true$ iff $f \in \{custody(A, B), watching(B, A), present(C)\} \cup P_{AB}$ and $\pi(s_2)(f) = true$ iff $f \in \{watching(B, A), custody(A, B)\} \cup P_{AB}$.[4] It is common to visualize a Kripke structure as a graph, with $M[S]$ as nodes and with labeled edges derived from $M[i]$. A double circle represents the distinguished state symbol of a pointed Kripke structure. $\square$

Intuitively, a Kripke structure denotes the possible worlds considered by the agents—and the presence of multiple worlds denotes uncertainty and presence of different beliefs. The relation $(u, v) \in \mathcal{B}_i$ denotes that the belief of agent i about the real state of the world is insufficient for her to distinguish between the state of the world described by u and the one described by v. If $M[\pi](u) \models \varphi$ and $M[\pi](v) \models \neg\varphi$, then i is uncertain about the truth of φ. For example, we can easily check that $(M_1, s_1) \models present(C)$, $(M_1, s_2) \models \neg present(C)$, and $(s_1, s_2) \in \mathcal{B}_B$, and thus, B is not aware of the presence of C.

The following axioms are often used in classifying Kripke structures:

(**K**)	$\forall i \in \mathcal{AG}, \forall\varphi, \psi \in \mathcal{L}_{\mathcal{AG}}$	$[(\mathbf{B}_i\varphi \wedge \mathbf{B}_i(\varphi \Rightarrow \psi)) \Rightarrow \mathbf{B}_i\psi]$
(**T**)	$\forall i \in \mathcal{AG}, \forall\psi \in \mathcal{L}_{\mathcal{AG}}$	$[\mathbf{B}_i\psi \Rightarrow \psi]$
(**4**)	$\forall i \in \mathcal{AG}, \forall\psi \in \mathcal{L}_{\mathcal{AG}}$	$[\mathbf{B}_i\psi \Rightarrow \mathbf{B}_i\mathbf{B}_i\psi]$
(**5**)	$\forall i \in \mathcal{AG}, \forall\psi \in \mathcal{L}_{\mathcal{AG}}$	$[\neg\mathbf{B}_i\psi \Rightarrow \mathbf{B}_i\neg\mathbf{B}_i\psi]$
(**D**)	$\forall i \in \mathcal{AG}$	$[\neg\mathbf{B}_i false]$

A Kripke structure is **S5** if it satisfies **K**, **T**, **4**, and **5**.

Definition 3 *A set of belief formulae X is said to be* **S5**-*(resp.* **K**-*,* **T**-*,* **4**-*, and* **5**-*)* satisfiable *(or* consistent*) if there exists a* **S5**-*(resp.* **K**-*,* **T**-*,* **4**-*, and* **5**-*) Kripke structure M and a state $s \in M[S]$ such that $(M, s) \models \psi$ for every $\psi \in X$. (M, s) is then referred to as a* model *of X.*

Update Models

Program models are used to represent action occurrences using structures similar to pointed Kripke structures and they describe the effects of an action on states using an update operator. The original paper (Baltag and Moss 2004) deals with sensing actions and announcement actions. The approach is extended to world-altering (a.k.a. ontic) actions in (van Benthem, van Eijck, and Kooi 2006) and is called update model.

[4]We will often give the function π indirectly by listing the interpretation associated to the state symbols of the Kripke structure. For instance, we will write $s_1 = \{custody(A, B), watching(B, A), present(C)\} \cup P_{AB}$.

An $\mathcal{L}_{\mathcal{AG}}$-substitution is a set

$$\{p_1 \rightarrow \varphi_1, \ldots, p_n \rightarrow \varphi_n\}$$

where each p_i is a distinct fluent and each φ_i is formula in $\mathcal{L}_{\mathcal{AG}}$. We will implicitly assume that for each $p \in \mathcal{F} \setminus \{p_1, \ldots, p_n\}$, the substitution contains $p \rightarrow p$. The set of all $\mathcal{L}_{\mathcal{AG}}$-substitutions is denoted with $SUB_{\mathcal{L}_{\mathcal{AG}}}$.

Definition 4 (Update Model) *An* update model $\mathbf{\Sigma}$ *is a tuple* $(\Sigma, \{R_i \mid i \in \mathcal{AG}\}, pre, sub)$ *where*

- Σ *is a set, whose elements are called* events*;*
- *each* R_i *is a binary relation on* Σ*;*
- $pre : \Sigma \rightarrow \mathcal{L}_{\mathcal{AG}}$ *is a function mapping each event* $a \in \Sigma$ *to a formula in* $\mathcal{L}_{\mathcal{AG}}$*; and*
- $sub : \Sigma \rightarrow SUB_{\mathcal{L}_{\mathcal{AG}}}$.

A update instance ω *is a pair* $(\mathbf{\Sigma}, e)$ *where* $\mathbf{\Sigma}$ *is an update model* $(\Sigma, \{R_i \mid i \in \mathcal{AG}\}, pre, sub)$ *and e, referred to as a* designated event*, is a member in* Σ.

Intuitively, an update model represents different views of an action occurrence which are associated with the observability of agents. Each view is represented by an event in Σ. The designated event is the one that agents who are aware of the action occurrence will observe. The relation R_i describes agent i's uncertainty on action execution—i.e., if $(\sigma, \tau) \in R_i$ and event σ is performed, then agent i may believe that event τ is executed instead. pre defines the action precondition and sub specifies the changes of fluent values after the execution of an action. Update models are graphically represented similarly to Kripke structures, while update instances are similar to pointed Kripke structures. The update instance on the left of Fig. 4 is $(\mathbf{\Sigma_1}, \sigma)$ where

$$\mathbf{\Sigma_1} = (\{\sigma\}, \{R_x \mid x \in \{A, B, C\}\}, pre_1, sub_1)$$

and $R_A = R_B = R_C = \{(\sigma, \sigma)\}$, $pre_1(\sigma) = true$, and $sub_1(\sigma) = \{watching(B, A) \rightarrow false\}$.

Definition 5 (Updates by an Update Model) *Let* M *be a Kripke structure and* $\mathbf{\Sigma} = (\Sigma, \{R_i \mid i \in \mathcal{AG}\}, pre, sub)$ *be an update model. The* update operator*, induced by* $\mathbf{\Sigma}$*, defines a Kripke structures* $M' = M \otimes \mathbf{\Sigma}$*, where*

- $M'[S] = \{(s, \tau) \mid s \in M[S], \tau \in \Sigma, (M, s) \models pre(\tau)\}$,
- $((s, \tau), (s', \tau')) \in M'[i]$ *iff* $(s, s') \in M[i]$ *and* $(\tau, \tau') \in R_i$*, and*
- *For* $f \in \mathcal{F}$, $M'[\pi]((s, \tau)) \models f$ *iff* $f \rightarrow \varphi \in sub$ *and* $(M, s) \models \varphi$.

Intuitively, the Kripke structure M' is obtained from the component-wise cross-product of the old structure M and the update model $\mathbf{\Sigma}$, by (*i*) removing pairs (s, τ) s.t. (M, s) does not satisfy the action precondition; and (*ii*) removing links of the form $((s, \tau), (s', \tau'))$ from the cross product of $M'[i]$ and R_i if $(s, s') \notin M[i]$ or $(\tau, \tau') \notin R_i$. The former ensures that the executability condition of the action is satisfied. The latter guarantees that each agent's accessibility relation is updated according to the update model.

Example 4 Continuing with the previous examples, let us compute $M' = M_1 \otimes \mathbf{\Sigma_1}$:

- $M'[S]=\{(s_1, \sigma), (s_2, \sigma)\}$. Let $u=(s_1, \sigma)$ and $v=(s_2, \sigma)$.
- $M'[A] = M'[C] = \{(u, u), (v, v)\}$ and $M'[B] = \{(u, u), (v, v), (u, v), (v, u)\}$.
- $M'[\pi](s_3)(f) = true$ iff $f \in s_3$ and $M'[\pi](s_4)(f) = true$ iff $f \in s_4$.

Observe that the Kripke structure in Fig. 2 is obtained from M' by renaming u and v to s_3 and s_4 respectively. $\square$

An update template is a pair $(\mathbf{\Sigma}, \Gamma)$ where $\mathbf{\Sigma}$ is an update model with the set of events Σ and $\Gamma \subseteq \Sigma$. The update of a state (M, s) given a update template $(\mathbf{\Sigma}, \Gamma)$ is a set of states, denoted by $(M, s) \otimes (\mathbf{\Sigma}, \Gamma)$, where for each $(M', s') \in (M, s) \otimes (\mathbf{\Sigma}, \Gamma)$, it holds that $M'=M \otimes \mathbf{\Sigma}$ and $s' = (s, \tau)$ where $\tau \in \Gamma$ and $s' \in M'[S]$. We can easily check that $(M_1, s_1) \otimes (\mathbf{\Sigma_1}, \{\sigma\})$ results in a pointed Kripke structure which is bisimilar to the pointed Kripke structure on the left of Fig. 2.

The Language $m\mathcal{A}$+

The language $m\mathcal{A}$+ is built over a signature $\langle \mathcal{AG}, \mathcal{F}, \mathcal{A} \rangle$, where $\mathcal{AG}$ is a finite set of agent identifiers, $\mathcal{F}$ is a set of fluents, and $\mathcal{A}$ is a set of actions. Each action in $\mathcal{A}$ is an action the agents in the domain are capable of performing. For the sake of simplicity, we restrict initially our attention to domains where all agents have the same capabilities in terms of actions. We organize the description of $m\mathcal{A}$+ in two parts: *(1)* the description of the initial beliefs, and *(2)* the description of the actions and their effects.

Describing the Initial Beliefs

Syntax: In designing a specification of a domain in which multiple agents are operating, we need to provide a description of the initial configuration of the agents' beliefs about the world and about others' beliefs. In $m\mathcal{A}$+ , this is accomplished through *initial statements*:

$$\textbf{initially} \ \ \varphi \tag{1}$$

where φ is a belief formula. Intuitively, this statement says that φ is true in the initial situation.

Example 5 The initial situation of the domain in Example 1 can be partially expressed by the following statements:

$$\textbf{initially} \ \ \mathbf{C}(custody(A, B) \wedge watching(B, A))$$
$$\textbf{initially} \ \ \mathbf{C}(\neg has\_key(x)) \ \ (x \in \{A, B, C\})$$
$$\textbf{initially} \ \ \mathbf{C}(present(x)) \ \ (x \in \{A, B\})$$

These statements say that A is in the custody of B, B is watching A, nobody has the key, A and B are present, and everyone knows all these facts.

If C is present and both A and B are unaware of the presence of C, we can add the following statements:

$$\textbf{initially} \ \ present(C)$$
$$\textbf{initially} \ \ \mathbf{C}(\neg \mathbf{B}_B(present(C)) \wedge \neg \mathbf{B}_B(\neg present(C)))$$
$$\textbf{initially} \ \ \mathbf{C}(\neg \mathbf{B}_A(present(C)) \wedge \neg \mathbf{B}_A(\neg present(C)))$$

Observe that in single-agent domains the state space described by the initial states declarations is always finite—bounded by $2^{|\mathcal{F}|}$. This property allows us to employ available computational technologies (e.g., answer set solvers, forward search planners) that rely on the existence of a finite number of initial states. The same property does not hold in the case of general initial state descriptions in $m\mathcal{A}$+ . For example, it is easy to see that if (M, s) is a pointed Kripke structure such that $(M, s) \models \varphi$ for some formula φ, then $(M', s) \models \varphi$ for any M' obtained from M by adding any number of new state symbols.

One way to address such problem is to restrict the type of Kripke structures considered for the initial situations. To this end, we observe that several examples found in the literature have the following properties: (*a*) the initial situation can be described by a set of statements involving the *common knowledge* among all agents; (*b*) the common knowledge relates to whether a particular agent is aware or not of a property of the world; and (*c*) the beliefs of agents coincide with their knowledge about the world. We introduce a class of formulae that will help us towards this objective.

Definition 6 (Restricted Formulae)

(i) *Each objective formula φ is a restricted formula;*

(ii) *For $i \in \mathcal{AG}$ and φ an objective formula each belief formula of the form $\mathbf{B}_i\varphi$, $\mathbf{B}_i\varphi \vee \mathbf{B}_i\neg\varphi$, $\neg\mathbf{B}_i\varphi \wedge \neg\mathbf{B}_i\neg\varphi$ are restricted formulae;*

(iii) *For $i \in \mathcal{AG}$ and ψ is an objective formula or a restricted formula defined in Item (ii), $\mathbf{C}(\psi)$ is a restricted formula; and*

(iv) *No other formula is a restricted formula.*

Intuitively, a restricted formula is such that: (*i*) describes the "concrete" state of the world; (*ii*) expresses the belief of agent i about the state of the world; (*iii*) states that agent i has a belief about a property (but the specific belief is unknown); (*iv*) encodes the lack of a belief about a property.

Semantics: The considerations made earlier about restricting the type of Kripke structures used in the initial states are realized by allowing only restricted formulae in the initial statements (requirements *(a)* and *(b)*) and by considering only **S5** structures (to meet the requirement *(c)*). Let a **S5-state** be a state (M, s) where M satisfies **S5**.

It can be shown that if (M, s) is an **S5**-state whose states are reachable from s then it is equivalent to an **S5**-state (M', s') such that for every pair of state symbols $u, v \in M'[S]$, the interpretations associated to u and v are different, i.e., $M'[\pi](u) \not\equiv M'[\pi](v)$, which means that the set of state symbols in M' is finite. The detail can be found in (Baral et al. 2011). Thus, we can use this results to define a restricted set of initial statements, i.e., those using only restricted formulae, that allows us to focus on **S5**-states of bounded size, as formalized in the following statement. Given a set of initial statements $\mathcal{I}$, let us denote $T_{\mathcal{I}} = \{\psi \mid (\textbf{initially } \mathbf{C}\psi) \in \mathcal{I}\} \cup \{\psi \mid (\textbf{initially } \psi) \in \mathcal{I},\ \psi \text{ is an objective formula}\}$.

Definition 7 (Definite Initial Description) *A set of initial statements $\mathcal{I}$ is* definite *if*

1. *$T_{\mathcal{I}}$ is a set of restricted formulae;*
2. *For every agent i,*
   (a) *$T_{\mathcal{I}} \not\models \mathbf{C}(\mathbf{B}_i\psi)$ holds for every fluent formula ψ s.t. there exists no φ satisfying $\mathbf{C}(\mathbf{B}_i\varphi) \in T_{\mathcal{I}}$ and $\models \varphi \rightarrow \psi$;*
   (b) *$T_{\mathcal{I}} \not\models \mathbf{C}(\mathbf{B}_i\psi \vee \mathbf{B}_i\neg\psi)$ holds for every fluent formula ψ s.t. there exists no φ satisfying $\mathbf{C}(\mathbf{B}_i\varphi \vee \mathbf{B}_i\neg\varphi) \in T_{\mathcal{I}}$ and $\models \varphi \rightarrow \psi$.*

$\mathcal{I}$ is said to be complete *if for each fluent $f \in \mathcal{F}$, either* [**initially** f] *or* [**initially** $\neg f$] *belongs to $\mathcal{I}$. $\mathcal{I}$ is said to be* consistent *if $T_{\mathcal{I}}$ is consistent.*

Intuitively, the conditions state that $\mathcal{I}$ completely characterizes the common knowledge of the agents about the beliefs of agent i about the world. We can prove that all **S5**-initial states of a complete and consistent definite action theory are equivalent to each other.

Theorem 1 *For each consistent and definite set of initial statements $\mathcal{I}$, there exists a finite number of initial* **S5**-*states $(M_1, s_1), \ldots, (M_k, s_k)$ such that for every* **S5**-*state (M, s) of $\mathcal{I}$ there exists $1 \leq i \leq k$ such that (M, s) is equivalent to (M_i, s_i). Furthermore, if $\mathcal{I}$ is complete then $k = 1$.*

Actions

In $m\mathcal{A}$+ , an agent is allowed to use three types of actions: *ontic actions* (or *world-changing actions*), *sensing actions*, and *announcement actions*. Intuitively,

- An *ontic* action is used to modify certain properties of the world—e.g., the agent A takes the key in Example 1; an agent turns a switch, causing the light to turn on;
- A *sensing* action is used by an agent to refine its belief about the world, by making observations (e.g., agent C watches the agent A); the effect of the sensing action is to reduce the amount of uncertainty of the agent;
- A truthful *announcement* action is used by an agent to affect the beliefs of the agents receiving the communication—we operate under the assumption that agents receiving an announcement always believe what is being announced.

Each action $a \in \mathcal{A}$ falls in exactly one of the three categories. We consider the following statements for $a \in \mathcal{A}$.

- *Executability condition*: an executability condition is a statement of the form:
$$\textbf{executable } a \textbf{ if } \psi \tag{2}$$
where ψ is a belief formula; a is executable if ψ is true.
- *Dynamic law*: let a be an ontic action; a dynamic law is a statement of the form:
$$a \textbf{ causes } \ell \textbf{ if } \psi \tag{3}$$
where ℓ is a fluent literal and ψ is a belief formula. The part " **if** ψ" will be omitted when ψ is a tautology. Intuitively, if the state of the world and of the beliefs matches the condition ψ, then the real state of the world is affected by the change described by ℓ after the execution of a.
- *Observation law*: let a be a sensing action; an observation law has the form:
$$a \textbf{ determines } f \tag{4}$$
where f is a fluent. Statements of type (4) encode a sensing action a which enable the agent(s) to learn about the value of the fluent f.
- *Announcement law*: let a be an announcement action; an announcement law has the form:
$$a \textbf{ announces } \varphi \tag{5}$$
where φ is a formula.

Example 6 The actions for the domain in Example 1:

- $distract(x, y)$: x distracts y so that y does not watch x. The action is executable if y is watching x. This can be expressed by the law:
$$distract(x, y) \textbf{ executable } watching(y, x)$$

The action causes y not to watch x. This is represented by

$distract(x, y)$ **causes** $\neg watching(y, x)$.

- $get\_key(x)$: x takes the key. A cannot take the key while B is watching A. This is expressed as follows.

  $get\_key(A)$ **executable** $\neg watching(B, A)$.
  $get\_key(A)$ **causes** $has\_key(A)$.
  $get\_key(x)$ **causes** $has\_key(x)$ for $x \in \{B, C\}$.

- $signal(x, y)$: x signals y to watch x;

  $signal(x, y)$ **executable** $\neg watching(y, x)$.
  $signal(x, y)$ **causes** $watching(y, x)$.

- $lookAround(x, y)$: x is looking around for y and will be able to determine whether y is present or not.

  $lookAround(x, y)$ **determines** $present(y)$.

- $shout(x)$: x announces his presence;

  $shout(x)$ **announces** $present(x)$. □

Observability of Actions

In a single-agent setting, it is reasonable to assume that if an agent observes an action occurrence and knows the action specification then he/she will be aware of the effects of the action. In a multi-agent setting, it is possible for an agent to be completely unaware of an action occurrence—e.g., an agent may be too far to hear an announcement. It is also possible for an agent to be aware of the action occurrence (e.g., performed by another agent) but unaware of the effects of the action. For example, an agent might be told that someone fires a shot but unaware of whether the bullet hit the target, i.e., he is aware of the action occurrence but unaware of the result of the action. Even when an agent executes an action, he might not observe the its effects.

Given an action occurrence, we will divide agents into three groups: *(a)* those aware of the action occurrence and of the effects of the actions, *(b)* those aware of the action occurrence but unaware of the effects of the actions, and *(c)* those completely oblivious of both the action occurrence and its effects. These categories are called *frames of reference*, and are dynamic in nature. Frames of reference are specified by a class of statements, called observation axioms:

$$x \textbf{ observes } a \textbf{ if } \varphi \qquad (6)$$

$$x \textbf{ aware of } a \textbf{ if } \varphi \qquad (7)$$

where x is a set of agents, a is an action, and φ is a fluent formula. Intuitively, a statement of the form (6) indicates that if the action a occurs and the condition φ is true in the real state of the world, then x is the set of agents who are completely knowledgeable about the action occurrence and its consequences. A statement of the form (7) indicates that, if the action a occurs and the condition φ is true in the real state of the world, then x is the set of agents who are aware of the action occurrence but cannot observe the action's effects. If an agent does not occur in an observation axiom with respect to an action occurrence, or if the condition φ is not satisfied, then the agent will be oblivious of the action occurrence and its consequences. For simplicity, we assume that, in the case of ontic actions, each agent is either fully aware or completely oblivious. For all other types of actions, all three frames of reference may be present.

Example 7 The following set of observation axioms describes the observability of agents in the domain in Exp. 6:

$\{x, y\}$ **observes** $distract(x, y)$
$\{x\}$ **observes** $get\_key(x)$
$\{y\}$ **observes** $get\_key(x)$ **if** $watching(y, x)$
$\{x, y\}$ **observes** $signal(x, y)$
$\{x\}$ **observes** $lookAround(x, y)$
$\{A, B, C\}$ **observes** $shout(x)$

where x and y denote distinct agents. □

A *domain specification* (or *domain*) D in $m\mathcal{A}$+ is a collection of statements of the forms (2)-(7). D is *consistent* if for every pair of dynamic laws (a **causes** f **if** φ) and (a **causes** $\neg f$ **if** ψ) in D, $\varphi \wedge \psi$ is inconsistent.

Definition 8 (Action Theory) *An* action theory *in $m\mathcal{A}$+ is a pair (D, I) where D is a consistent domain specification and I is a set of* **initially** *statements.*

For simplicity, we will not consider the action executors in the formalization in this paper. We will also assume that agents do not reason about action occurrences retrospectively.[5] Taking all of these aspects into consideration will be an important task for our future work.

Semantics of $m\mathcal{A}$+

Let D be a consistent domain specification. Given a state (M, s) and an action $a \in \mathcal{A}$ which occurs in the executability law[6] a **executable** ψ in D, we say that a is *executable* in a state (M, s) if $(M, s) \models \psi$. Let us define

$$\alpha(a, M, s) = \left\{ i \in \mathcal{AG} \,\middle|\, \begin{array}{l} [i \textbf{ observes } a \textbf{ if } \varphi] \in D, \\ (M, s) \models \varphi \end{array} \right\}$$

$$\beta(a, M, s) = \left\{ i \in \mathcal{AG} \,\middle|\, \begin{array}{l} [i \textbf{ aware of } a \textbf{ if } \varphi] \in D, \\ (M, s) \models \varphi \end{array} \right\}$$

and $\gamma(a, M, s) = \mathcal{AG} \setminus (\alpha(a, M, s) \cup \beta(a, M, s))$. Intuitively, $\alpha(a, M, s)$, $\beta(a, M, s)$, and $\gamma(a, M, s)$ are the agents that are fully aware, partially aware, and oblivious about the effects of the execution of action a in the state (M, s). In the following, we will often write α, β, and γ instead of $\alpha(a, M, s)$, $\beta(a, M, s)$, $\gamma(a, M, s)$ respectively, whenever it is clear from the context what a and (M, s) are.

We will now define the semantics of $m\mathcal{A}$+ by constructing, for each action a executable in a state (M, s), an update template $\omega^a_{M,s} = (\mathbf{\Sigma}_a, \Gamma)$ with $\mathbf{\Sigma}_a = (\Sigma_a, \{R_i\}, pre, sub)$. We will assume that the executability condition a **executable** ψ belongs to D.

Sensing Actions

For an observation law a **determines** f in D, $\omega^a_{M,s}$ is defined as follows:

- $\Sigma_a = \{\sigma, \tau, \epsilon\}$ and $\Gamma = \{\sigma, \tau\}$;
- $R_i = \{(\sigma, \sigma), (\tau, \tau), (\epsilon, \epsilon)\}$, for $i \in \alpha$;
- $R_i = \{(\sigma, \sigma), (\tau, \tau), (\epsilon, \epsilon), (\sigma, \tau), (\tau, \sigma)\}$, for $i \in \beta$;

[5] An agent observing that a dead turkey could infer that a shooting action has occurred and the number of bullets has changed, etc.

[6] Recall that we assume that each action appears only in one executability law.

- $R_i = \{(\sigma, \epsilon), (\tau, \epsilon), (\epsilon, \epsilon)\}$, for $i \in \gamma$;
- $pre(\sigma) = f \wedge \psi$, $pre(\tau) = \neg f \wedge \psi$, and $pre(\epsilon) = \top$.
- $sub : \Sigma_a \rightarrow SUB_{\mathcal{L}_{\mathcal{AG}}}$ where $sub(x) = \emptyset$ for $x \in \Sigma_a$, i.e., sub maps every event to the identity substitution.

Intuitively, σ, τ, and ϵ, called *events*, represent the different views of an occurrence of a. σ (resp. τ) indicates an event determining that f is true (resp. false) happens while ϵ encodes that nothing happens. σ and τ are the designated events, representing what occurs, and thus their precondition requires $f \wedge \psi$ and $\neg f \wedge \psi$, respectively. ϵ encodes that the action does not occur, which is what oblivious agents will be seeing and thus is not a designated event. Because sensing actions do not change the real state of the world, sub maps every event to the identity substitution. The relations R_i indicate that agents in α would be able to distinguish world states in which f is true from those in which f is false; those in β are aware of the fact that α learn the value of f but they themselves are oblivious of such value; agents in γ are completely unaware of the action execution.

Example 8 Let us consider the domain D given in the Examples 6-7 and the state (U_1, s_1) in Fig. 5 (left). Assume that $lookAround(A, C)$ is executed. We have that $\alpha = \{A\}$, $\beta = \emptyset$, and $\gamma = \{B, C\}$. This leads to the update template on the right of Fig. 5 with $pre(\sigma) = present(C)$, $pre(\epsilon) = \top$, $present(\tau) = \neg present(C)$, and $sub(x) = \emptyset$ for every $x \in \{\sigma, \tau, \epsilon\}$. The result of executing $lookAround(A, C)$ in

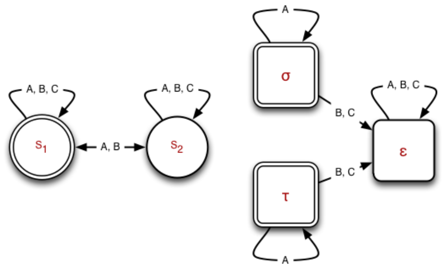

Figure 5: State (U_1, s_1) and $\omega^{lookAround(A,C)}_{U_1,s_1}$

(U_1, s_1) is the state in Fig. 6.

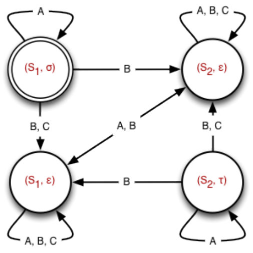

Figure 6: Execution of $lookAround(A, C)$ in (U_1, s_1)

Proposition 2 *Let D be a consistent domain, (M, s) be a state, and a be a sensing action that is executable in (M, s). Assume that $(M', s') \in (M, s) \otimes \omega^a_{M,s}$. Then, it holds that*

- $\forall f \in Sens_D(a)$, $[(M', s') \models \mathbf{C}_\alpha \ell$ iff $(M, s) \models \ell]$ *where* $\ell \in \{f, \neg f\}$*;*
- $\forall f \in Sens_D(a)[(M', s') \models \mathbf{C}_\beta(\mathbf{C}_\alpha f \vee \mathbf{C}_\alpha \neg f)]$*;*
- $\forall i \in \gamma(a, M, s), \forall \ell[(M', s') \models \mathbf{B}_i \ell$ iff $(M, s) \models \mathbf{B}_i \ell]$

*where $Sens_D(a) = \{f \mid a$ **determines** f in $D\}$.*

The proposition shows that agents in $\alpha(a, M, s)$ will know the truth values of the sensed fluents; the agents in $\beta(a, M, s)$ know that the agents in $\alpha(a, M, s)$ know the truth values of the sensed fluents but do not know the truth value of these fluents; and nothing changes for oblivious agents.

Announcement Actions

For an announcement law a **announces** φ in D, $\omega^a_{M,s}$ is defined as follows:

- $\Sigma_a = \{\sigma, \tau, \epsilon\}$ and $\Gamma = \{\sigma\}$;
- $R_i = \{(\sigma, \sigma), (\tau, \tau), (\epsilon, \epsilon)\}$, for $i \in \alpha$;
- $R_i = \{(\sigma, \sigma), (\tau, \tau), (\epsilon, \epsilon), (\sigma, \tau), (\tau, \sigma)\}$, for $i \in \beta$;
- $R_i = \{(\sigma, \epsilon), (\tau, \epsilon), (\epsilon, \epsilon)\}$, for $i \in \gamma$;
- $pre(\sigma) = \varphi \wedge \psi$, $pre(\tau) = \neg\varphi \wedge \psi$, and $pre(\epsilon) = \top$;
- $sub : \Sigma_a \rightarrow SUB_{\mathcal{L}_{\mathcal{AG}}}$ where $sub(x) = \emptyset$ for $x \in \Sigma_a$.

Update template for an announcement is slightly different from that of sensing actions in that there is only one designated event, which says that φ is true and there is no link from τ to σ for agent in β. R_i indicates that after the execution, $i \in \alpha$ will know φ while $i \in \beta \cup \gamma$ does not. However, $i \in \beta$ will know that $j \in \alpha$ knows about φ.

Example 9 Assume that $shout(C)$ is executed in the state (U_1, s_1) in Fig. 5 (left). We have that $\alpha = \{A, B, C\}$, $\beta = \gamma = \emptyset$. The update template is on the left of Fig. 7 with $pre(\sigma) = present(C)$, $pre(\tau) = \neg present(C)$, $pre(\epsilon) = \top$, and $sub(x) = \emptyset$ for every $x \in \{\sigma, \tau, \epsilon\}$. The result of executing $shout(C)$ in (U_1, s_1) is on the right of Fig. 7, which is equivalent to the state with a single state symbol (σ, s_1) in the shaded box, due to result of the previous section.

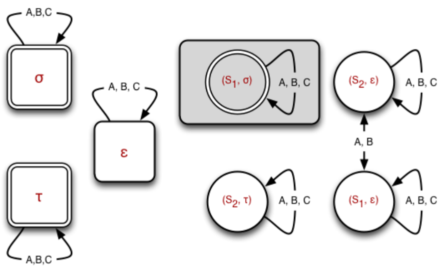

Figure 7: $\omega^{shout(C)}_{U_1,s_1}$ and Result

The next proposition is similar to Proposition 2.

Proposition 3 *Let D be a consistent domain, (M, s) be a state, and a* **announces** *φ belongs to D. Assume that a is executable in (M, s) and $(M', s') \in (M, s) \otimes \omega^a_{M,s}$. Then,*

- $(M', s') \models \mathbf{C}_\alpha \varphi$*;*
- $(M', s') \models \mathbf{C}_\beta(\mathbf{C}_\alpha \varphi \vee \mathbf{C}_\alpha \neg\varphi)$ *and;*
- $\forall i \in \gamma(a, M, s), \forall \ell'.[(M', s') \models \mathbf{B}_i \ell'$ iff $(M, s) \models \mathbf{B}_i \ell']$.

Ontic Actions

For an ontic action a in D, $\omega^a_{M,s}$ is defined as follows:

- $\Sigma_a = \{\sigma, \epsilon\}$ and $\Gamma = \{\sigma\}$;
- $R_i = \{(\sigma, \sigma), (\epsilon, \epsilon)\}$, for $i \in \alpha$;
- $R_i = \{(\sigma, \epsilon), (\epsilon, \epsilon)\}$, for $i \in \gamma$;
- $pre(\sigma) = \psi$ and $pre(\epsilon) = \top$.

- $sub : \Sigma_a \rightarrow SUB_{\mathcal{L}_{\mathcal{AG}}}$ where $sub(\epsilon) = \emptyset$ and
  $$sub(\sigma) = \{f \rightarrow \varphi \vee f \mid a \textbf{ causes } f \textbf{ if } \varphi \text{ in } D\} \ \cup \{f \rightarrow \neg\varphi \wedge f \mid a \textbf{ causes } \neg f \textbf{ if } \varphi \text{ in } D\}$$
  where $e_a^+ = \{f \mid a \textbf{ causes } f \textbf{ if } \varphi \text{ in } D\}$ and $e_a^- = \{f \mid a \textbf{ causes } \neg f \textbf{ if } \varphi \text{ in } D\}$.

Update template for an ontic action is simpler than those for sensing/announcement actions, since there are only two groups of agents: those who aware of the effects of the action and those who are oblivious. Similar to announcement actions, there is only one designated event. The effects of an action occurrence is that only agents who are aware of its occurrence will know the effects.

Example 10 Let us continue with the domain D given in the Exp. 6-7 and the state (U_2, s_1) which is the result of the execution of $shout(C)$ in (U_1, s_1) given in Fig. 5 (left). For the simplification of the presentation, we use the state with the single symbol in the shaded box of the state on the right of Fig. 7 and $U_2[S] = \{s_1\}$, $U_2[A]{=}U_2[B]{=}U_2[C]{=}\{(s_1, s_1)\}$, and s_1 is given in Exp. 2. We observe that $distract(A, B)$ is executable in (U_2, s_1) and $\alpha{=}\{A, B\}$, $\beta{=}\emptyset$, and $\gamma{=}\{C\}$. This leads to the update template on the left of Fig. 8 with $pre(\sigma){=}watching(B, A)$ and $pre(\epsilon){=}\top$ and $sub(\sigma){=}\{watching(B, A) \ \rightarrow \ false\}$ and $sub(\epsilon){=}\emptyset$. The state (U_2, s_1), the update template, and the result of the execution of $distract(A, B)$ is given in Fig. 8 (top-left, top-right, bottom and shaded). Observe that the interpretation of (σ, s_1) is $\{custody(A, B), present(C), present(A), present(B)\}$. However, the interpretation of (ϵ, s_1) is $\{watching(B, A),$ $custody(A, B), present(C), present(A), present(B)\}$.

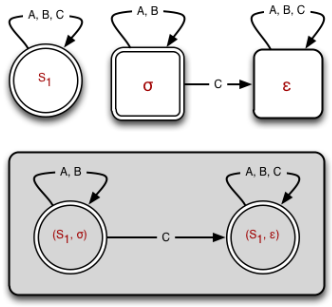

Figure 8: (U_2, s_1), $\omega_{U_2,s_1}^{distract(A,B)}$, and result of the execution of $distract(A, B)$ in (U_2, s_1)

Proposition 4 *Let D be a consistent domain, (M, s) be a state, and a be a world-altering action that is executable in (M, s). Assume that $(M', s') \in (M, s) \otimes \omega_{M,s}^a$. Then, the following holds*

- *for every $i \in \alpha(a, M, s)$, a* **causes** *ℓ* **if** *φ in D, and $u, v \in M[S]$,*
  - *$(\sigma, u) \models \ell$ if $(M, u) \models \varphi$; and*
  - *$(u, v) \in M[i]$ iff $((\sigma, u), (\sigma, v)) \in M'[i]$*
- *for every $i \in \gamma(a, M, s)$ and literal ℓ, $(M', s') \models \mathbf{B}_i\ell$ iff $(M, s) \models \mathbf{B}_i\ell$.*

The proposition states that agents in $\alpha(a, M, s)$ know the effects of the action execution while all other agents do not see any changes.

Entailment

The entailment relation in $m\mathcal{A}+$ action theories is defined between action theories and queries of the form:

$$\varphi \textbf{ after } Plan \tag{8}$$

where φ is a belief formula and *Plan* is defined as follows:

Definition 9 (Plan) *For a domain description D,*
- *The empty plan $[\,]$ is a plan;*
- *If $a \in \mathcal{A}$ and P is is a plan then $(a; P)$ is also a plan;*
- *If φ is a formula and P_1, P_2 are plans, then* **if** *φ* **then** *P_1* **else** *P_2 is also a plan.*

We will next define the notion of entailment between $m\mathcal{A}+$ action theories and queries of the form (8).

A *belief state* (or *b-state*) is a set of states. For a b-state B and an action a, we say that a is executable in B if a is executable in every state (M, s) in B. With a slight abuse of notation, we define

$$\Phi_D(a, B) = \begin{cases} \{\bot\} & \text{if } a \text{ is not executable in some states in } B \\ \bigcup_{(M,s)\in B}(M, s) \otimes \omega_{M,s}^a & \text{otherwise} \end{cases}$$

where $\bot$ denotes that the execution of a in B fails.

Let P be a plan and B be a b-state. The set of b-states resulting from the execution of P in B, denoted by $\Phi_D^*(P, B)$, is defined as follows.

- If P is the empty plan then $\Phi_D^*(P, B) = B$;
- If P is a plan of the form $(a; P')$ then $\Phi_D^*(P, B) = \Phi_D^*(P', \Phi_D(a, B))$
- If P is of the form **if** φ **then** P_1 **else** P_2, then
  $$\Phi_D^*(P, B) = \bigcup_{(M,s)\in B} \Xi_D^*(P, M, s)$$
  where
  $$\Xi_D^*(P, M, s) = \begin{cases} \Phi_D^*(P_1, \{(M, s)\}) & \text{if } (M, s) \models \varphi \\ \Phi_D^*(P_2, \{(M, s)\}) & \text{otherwise} \end{cases}$$

where, we define $\{\bot, \dots\} = \{\bot\}$ and $\Phi_D(P, \{\bot\}) = \{\bot\}$ for every plan P.

Intuitively, the execution of P in B can go through several paths, each path might finish in a set of states. It is easy to see that if one of the states reached on a path during the execution of P is $\bot$ (the failed state) then the final result is the execution of P in B is $\{\bot\}$. $\Phi_D^*(P, B) = \{\bot\}$ indicates that the execution of P in B fails.

Definition 10 (Initial State/b-State) *Let (D, I) be an action theory. An initial state of (D, I) is a state (M, s) such that for every statement* [**initially** *φ] in I, $(M, s) \models \varphi$.*

The initial b-state *of (D, I) is the collection of all initial states of (D, I).*

We are now ready to define the notion of entailment.

Definition 11 (Entailment) *An action theory (D, I) entails the query φ* **after** *Plan, denoted $(D, I) {\models} \varphi$* **after** *Plan, if*
- *$\Phi_D^*(Plan, I_0) \neq \{\bot\}$ and*
- *$(M, s) \models \varphi$ for each $(M, s) \in \Phi_D^*(Plan, I_0)$*

where I_0 is the initial b-state of (D, I).

It is easy to see that for the action theory (D, I), where D is the domain given in Exp. 6-7 and I is given in Exp. 5 (represented by (U_1, s_1) in Fig. 5, left), and $\alpha = shout(C); signal(A, C); distract(A, B); get\_key(A)$

it holds that $(D, I) \models \mathbf{C}_{A,C} has\_key(A)$ **after** α and $(D, I) \models \mathbf{B}_B \neg has\_key(A)$ **after** α.

Related Work and Discussion

The proposed language, $m\mathcal{A}$+ , is built on the advances made in RAC and DEL. It is strongly related to our previous attempts in developing an action language for multi-agent systems but differs in several ways. In (Baral et al. 2010), we suggested that ASP can be used for multi-agent reasoning; we showed how ASP can be used to find Kripke models of a modal theory, how ASP can be used to formulate the muddy children and the sum-and-product problems. In the process, we formulated the global announcement and ask-and-truthfully answer actions in ASP. In (Pontelli et al. 2010), we presented the multi-agent action language $m\mathcal{A}$ and showed how reasoning and planning can be done in $m\mathcal{A}$ using Prolog. In (Baral and Gelfond 2010), we extrapolated the work in (Baltag and Moss 2004) on multi-agent reasoning in the context of dynamic epistemic logic to the case where agents are classified into three types and suggest directions to combine DEL with RAC.

As we have mentioned, $m\mathcal{A}$+ differs significantly from $m\mathcal{A}$ in that it allows for the dynamic specification of roles of agent and separate the specification of action effects from the agents' role. This is also a key difference between $m\mathcal{A}$+ and DEL's formalisms proposed in (Baltag, Moss, and Solecki 1998; Baltag and Moss 2004; van Benthem, van Eijck, and Kooi 2006). However, the semantics of $m\mathcal{A}$+ relies on the update models developed in this line of works.

In (Herzig, Lang, and Marquis 2005), an approach to reasoning about actions and beliefs in multi-agent domains, relying on update models, has been proposed. As with other frameworks, this approach does not separate the specification of agents' roles and action effects and thus differs from $m\mathcal{A}$+ in this regards. Furthermore, the approach does not consider announcement actions and does not discuss how such an update model should be constructed.

The idea of using progression-based planning based on epistemic states and update models has recently been expanded in (Bolander and Andersen 2011), with particular focus on decidability issues, and working with restricted forms of accessibility in the Kripke structures; (Löwe, Pacuit, and Witzel 2010) builds on DEL to achieve an analogous objective. Neither of these proposals designs an action language for domain descriptions or separately addresses the declarative specification of levels of observability. Indeed, (Bolander and Andersen 2011) indicates the development of languages suitable for describing actions in a form closer to traditional planning, with parameters for describing observers as one of the main problems left open.

Let us conclude the section with a discussion on nondeterministic actions. Nondeterminism occurs for various reasons. We will consider two different types: sensing actions that may fail and nondeterministic ontic actions. They can be added to $m\mathcal{A}$+ using the following syntax.

$$a \textbf{ may\_determine } f \tag{9}$$

$$a \textbf{ causes } \ell_1 \vee \ldots \vee \ell_k \textbf{ if } \varphi \tag{10}$$

Statements of the type (9) encode a sensing action a which enable the agent(s) to learn about the value of the fluent f. However, a might fail leaving the agent unaware of the value of the fluent being sensed. Statements of the type (10) indicate that the action a might cause any of $\ell_1, \ldots, \ell_k$ to become true, still preserving the consistency of the state of the world. Update templates of these types of actions are given next. We will still use ψ to denote the executability of an action a. For an observation law a **may_determine** f executable in D, $\omega^a_{M,s}$ is defined as follows:

- $\Sigma_a = \{\sigma, \tau, \epsilon, \rho\}$ and $\Gamma = \{\sigma, \tau\}$;
- $R_i = \{(\sigma, \sigma), (\tau, \tau), (\epsilon, \epsilon), (\rho, \rho)\}$, for $i \in \alpha$;
- $R_i = \{(\sigma, \sigma), (\tau, \tau), (\epsilon, \epsilon), (\rho, \rho), (\sigma, \tau), (\tau, \sigma), (\rho, \sigma), (\sigma, \rho), (\rho, \tau), (\tau, \rho)\}$, for $i \in \beta$;
- $R_i = \{(\sigma, \epsilon), (\tau, \epsilon), (\rho, \epsilon), (\epsilon, \epsilon)\}$, for $i \in \gamma$;
- $pre(\sigma) = f \wedge \psi$, $pre(\tau) = \neg f \wedge \psi$, $pre(\rho) = \psi$, and $pre(\epsilon) = \top$.
- $sub : \Sigma_a \rightarrow SUB_{\mathcal{L}_{\mathcal{AG}}}$ where $sub(x) = \emptyset$ for $x \in \Sigma_a$, i.e., sub maps every event to the identity substitution.

The update template for an ontic nondeterministic action can be built using a similar construction given in the previous section for ontic actions. We need some extra notations. Let

$$D_a = \left\{ \begin{array}{c} a \textbf{ causes } \ell^1_1 \vee \ldots \vee \ell^1_{k_1} \textbf{ if } \varphi_1 \\ \ldots \\ a \textbf{ causes } \ell^n_1 \vee \ldots \vee \ell^n_{k_n} \textbf{ if } \varphi_n \end{array} \right\}$$

be the set of laws of the form 10 in D. We denote with $S_1, \ldots, S_n$ a sequence of sets of literals such that $\emptyset \neq S_i \subseteq \{\ell^i_1, \ldots, \ell^i_{k_i}\}$ and S_i is consistent. Let $\mathcal{S}$ denote the set of all these sequences. Let

$$D(S_1, \ldots, S_n) = D \setminus D_a \cup \bigcup_{i=1}^{n} \{a \textbf{ causes } l \textbf{ if } \varphi_i \mid l \in S_i\}.$$

Let $\sigma_{S_1,\ldots,S_n}$ be the substitution for the event σ defined for the action a with respect to $D(S_1, \ldots, S_n)$ as defined in the previous section. We now define $\omega^a_{M,s}$ for an ontic-nondeterministic action a that is executable in (M, s):

- $\Sigma_a = \{\sigma(\hat{S}) \mid \hat{S} \in \mathcal{S}\} \cup \{\epsilon\}$ and $\Gamma = \{\sigma(\hat{S}) \mid \hat{S} \in \mathcal{S}\}$;
- $R_i = \{(u, u) \mid u \in \Sigma_a\}$, for $i \in \alpha$;
- $R_i = \{(u, \epsilon) \mid u \in \Sigma_a\}$, for $i \in \gamma$;
- $pre(\sigma(\hat{S})) = \psi$ for each $\hat{S} \in \mathcal{S}$ and $pre(\epsilon) = \top$.
- sub:$\Sigma_a \rightarrow SUB_{\mathcal{L}_{\mathcal{AG}}}$ where $sub(\sigma(\hat{S}))=\sigma_{\hat{S}}$ and $sub(\epsilon)=\emptyset$.

Conclusion and Future Work

We presented a high-level action description language, $m\mathcal{A}$+ , for multi-agent domains. The language semantics is defined via action and update models. It separates the specification of action effects and the agents' roles. We also identify a reasonably large classes of action theories for which hypothetical queries can be answered and planning can be computed using state-of-the-art reasoners and planners, respectively. This will be our focus in the immediate future.

Acknowledgement Chitta Baral acknowledges the support by ONR-MURI and IARPA. Enrico Pontelli and Tran Cao Son acknowledge the support by NSF-IIS 0812267 grant.

References

Baltag, A., and Moss, L. 2004. Logics for epistemic programs. *Synthese*.

Baltag, A.; Moss, L.; and Solecki, S. 1998. The logic of public announcements, common knowledge, and private suspicions. In *7th TARK*, 43–56.

Baral, C., and Gelfond, M. 1997. Reasoning about effects of concurrent actions. *J. Log. Program.* 31(1-3):85–117.

Baral, C., and Gelfond, G. 2010. On representing actions in multi-agent domains. In *Proceedings of the Symposium on Constructive Mathematics*.

Baral, C.; Gelfond, G.; Pontelli, E.; and Son, T. C. 2010. Modeling multi-agent scenarios involving agents knowledge about other's knowledge using asp. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, 259–266. International Foundation for Autonomous Agents and Multiagent Systems.

Baral, C.; Gelfond, G.; Pontelli, E.; and Son, T. C. 2011. An action language for mutli-agent domains. Technical report.

Baral, C. 1995. Reasoning about Actions : Nondeterministic effects, Constraints and Qualification. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 2017–2023. Morgan Kaufmann Publishers, San Francisco, CA.

Boella, G., and van der Torre, L. W. N. 2005. Enforceable social laws. In Dignum, F.; Dignum, V.; Koenig, S.; Kraus, S.; Singh, M. P.; and Wooldridge, M., eds., *4rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), July 25-29, 2005, Utrecht, The Netherlands*, 682–689. ACM.

Bolander, T., and Andersen, M. 2011. Epistemic Planning for Single and Multi-Agent Systems. *Journal of Applied Non-Classical Logics* 21(1).

Brown, F., ed. 1987. *Proceedings of the 1987 worskshop on The Frame Problem in AI.* Morgan Kaufmann, CA, USA.

Fagin, R.; Halpern, J.; Moses, Y.; and Vardi, M. 1995. *Reasoning about Knowledge*. MIT press.

Gelfond, M., and Lifschitz, V. 1993. Representing actions and change by logic programs. *Journal of Logic Programming* 17(2,3,4):301–323.

Gelfond, M., and Lifschitz, V. 1998. Action languages. *ETAI* 3(6).

Gerbrandy, J. 2006. Logics of propositional control. In Nakashima, H.; Wellman, M. P.; Weiss, G.; and Stone, P., eds., *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, May 8-12, 2006*, 193–200. ACM.

Giunchiglia, E., and Lifschitz, V. 95. Dependent fluents. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 1964–1969. Morgan Kaufmann Publishers, San Mateo, CA.

Giunchiglia, E., and Lifschitz, V. 98. An action language based on causal explanation: preliminary report. In *Proceedings of AAAI 98*, 623–630.

Giunchiglia, E.; Kartha, G.; and Lifschitz, V. 1997. Representing action: indeterminacy and ramifications. *Artificial Intelligence* 95:409–443.

Hanks, S., and McDermott, D. 1987. Nonmonotonic logic and temporal projection. *Artificial Intelligence* 33(3):379–412.

Herzig, A., and Troquard, N. 2006. Knowing how to play: uniform choices in logics of agency. In Nakashima, H.; Wellman, M. P.; Weiss, G.; and Stone, P., eds., *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, May 8-12, 2006*, 209–216.

Herzig, A.; Lang, J.; and Marquis, P. 2005. Action Progression and Revision in Multiagent Belief Structures. In *Sixth Workshop on Nonmonotonic Reasoning, Action, and Change (NRAC)*.

Kartha, G., and Lifschitz, V. 1994. Actions with indirect effects: Preliminary report. In *KR 94*, 341–350.

Kartha, G. 1994. Two counterexamples related to Baker's approach to the frame problem. *Artificial Intelligence* 69:379–391.

Leibniz, G. c. 1679. *An Introduction to a Secret Encyclopdia.*

Lifschitz, V. 1987. Formal theories of action. In Brown, F. M., ed., *The Frame Problem in Artificial Intelligence, Proceedings of the 1987 Workshop*, 35–58.

Lifschitz, V. 1997. On the Logic of Causal Explanation (Research Note). *Artif. Intell.* 96(2):451–465.

Lin, F., and Reiter, R. 1994. State constraints revisited. *Journal of Logic and Computation* 4(5):655–67. Special Issue on Action and Processes.

Lin, F., and Shoham, Y. 1995. Provably correct theories of action. *Journal of the ACM* 42(2):293–320.

Löwe, B.; Pacuit, E.; and Witzel, A. 2010. Planning Based on Dynamic Epistemic Logic. Technical report, ILLC, University of Amsterdam.

McCain, N., and Turner, H. 1995. A causal theory of ramifications and qualifications. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 1978–1984. Morgan Kaufmann Publishers, San Mateo, CA.

McCarthy, J., and Hayes, P. 1969. Some philosophical problems from the standpoint of artificial intelligence. In Meltzer, B., and Michie, D., eds., *Machine Intelligence*, volume 4. Edinburgh: Edinburgh University Press. 463–502.

McCarthy, J. 1959. Programs with common sense. In *Proceedings of the Teddington Conference on the Mechanization of Thought Processes*, 75–91. London: Her Majesty's Stationery Office.

McCarthy, J. 1980. Circumscription—a form of nonmonotonic reasoning. *Artificial Intelligence* 13(1, 2):27–39,171–172.

McDermott, D., and Doyle, J. 1980. Nonmonotonic logic I. *Artificial Intelligence* 13(1,2):41–72.

Meyer, J.-J. 2000. Dynamic logic for reasoning about actions and agents. In Minker, J., ed., *Logic-Based Artificial*

Intelligence. Boston/Dordrecht: Kluwer. 281–311. Chapter 13.

Newton, I. 1687. *Philosophiae Naturalis Principia Mathematica*.

Pontelli, E.; Son, T. C.; Baral, C.; and Gelfond, G. 2010. Logic programming for finding models in the logics of knowledge and its applications: A case study. *Theory and Practice of Logic Programming* 10(4-6):675–690.

Reiter, R. 1980. A logic for default reasoning. *Artificial Intelligence* 13(1,2):81–132.

Reiter, R. 1991. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Lifschitz, V., ed., *Artificial Intelligence and Mathematical Theory of Computation*. Academic Press. 359–380.

Reiter, R. 2001. *KNOWLEDGE IN ACTION: Logical Foundations for Describing and Implementing Dynamical Systems*. MIT Press.

Sandewall, E. 1994. *Features and Fluents: The representation of knowledge about dynamical systems*. Oxford University Press.

Sauro, L.; Gerbrandy, J.; van der Hoek, W.; and Wooldridge, M. 2006. Reasoning about action and cooperation. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, 185–192. New York, NY, USA: ACM.

Scherl, R., and Levesque, H. 1993. The frame problem and knowledge producing actions. In *Proceedings of the 12th National Conference on Artificial Intelligence*, 689–695. AAAI Press.

Shanahan, M. 1997. *Solving the frame problem: A mathematical investigation of the commonsense law of inertia*. MIT press.

Spaan, M. T. J.; Gordon, G. J.; and Vlassis, N. A. 2006. Decentralized planning under uncertainty for teams of communicating agents. In Nakashima, H.; Wellman, M. P.; Weiss, G.; and Stone, P., eds., *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, May 8-12, 2006*, 249–256.

Turner, H. 1997. Representing actions in logic programs and default theories. *Journal of Logic Programming* 31(1-3):245–298.

van Benthem, J.; van Eijck, J.; and Kooi, B. P. 2006. Logics of communication and change. *Inf. Comput.* 204(11):1620–1662.

van der Hoek, W.; Jamroga, W.; and Wooldridge, M. 2005. A logic for strategic reasoning. In Dignum, F.; Dignum, V.; Koenig, S.; Kraus, S.; Singh, M. P.; and Wooldridge, M., eds., *4rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), July 25-29, 2005, Utrecht, The Netherlands*, 157–164. ACM.

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