Rational Teams: Logical Aspects of Multi-Agent Systems

Huib Aldewereld[1] Wiebe van der Hoek[2] John-Jules Meyer[1]

[1]Intelligent Systems Group
Dept of Computer Science, Utrecht University
P.O. Box 80089, 3508 TB Utrecht,
The Netherlands
e-mail: huib@cs.uu.nl, jj@cs.uu.nl

[2]Department of Computer Science
University of Liverpool
Liverpool L69 7ZF
United Kingdom
wiebe@cs.uu.nl

Abstract. This paper describes a possible extension to the logical framework KARO to make it able to handle multiple agents. The KARO framework, which was initially developed by van der Hoek, van Linder and Meyer, makes use of epistemic and dynamic logics to express rational agents. These agents were capable of reasoning about their own actions and, somewhat limited, about the actions of other agents. This paper intend to extend the KARO system such that one is also able to reason about cooperation (and perhaps even competition) between agents.

1 Introduction

The last few years much research is done on intelligent agents. Research is especially done on the logical side of single-agents (cf. [1] and [2]). The research of intelligent agents has not limited itself towards single agents alone; many researchers are also interested in the cooperation of multiple agents. To specify systems of multiple agents working together several logical extentions of the classical proposition logic have been proposed, all with their own way of handling the problems that present themselves. There are frameworks that use mere epistemic logic for knowledge handeling ([1], [3]) and frameworks that use apart from that more specific logics to cope with aspects such as time ([4], [5]).

There is one single agent framework that stands out because of its use of dynamic logic for facilitating reasoning about actions and their results; this system is known as the KARO system [6]. KARO, however, is a specification language to describe merely single agents.

Creating a multi-agent system is not as easy as extending the existing operators with subscripts to denote different agents ([7]; it requires a lot of tweaking of the lifted operators and in some cases redefining concepts. Besides that there exist several different kinds of multi-agent systems; agents can work together in a team (by voluntary cooperation or due to obligations and requests of other team members), but it is also possible that groups of agents are appointed to work together and might therefore be directed by a leader, which can, of course, be an agent itself. It would be nice if a framework could describe all these kinds of multi-agent systems.

In this paper we propose a possible extension of the KARO framework, in such that it will become able to describe multi-agent systems as well. Moreover, we will try to create a system that should be easily extendable to describe any kind of multi-agent system, not only the team-based cooperations that we use in this paper. We will start in the next section with defining the semantics of the system. This is followed by the definitions and explanations of the new operators in section 3. We describe these operators in groups that are intuitively related.

2 Preliminaries

In this section we discuss the basic syntax and semantics of the KARO multi-agent system. In subsection 2.1 we introduce the syntax and semantics of single-agent KARO, which we will extend in subsection 2.2 such that it is able to describe multi-agent systems.

2.1 Basic syntax and semantics

The basis of the multi-agent system that we describe in this paper is the KARO framework developed by van der Hoek, van Linder and Meyer [6]; therefore we tried to develop our operators to match the single-agent operators as close as possible, while, of course, expanding their functionality to be able to cope with multiple agents. We will start this section by defining the base syntax and semantics of single-agent KARO, which we will expand to suffice for multi-agent KARO.

First of all we define the language $\mathscr{L}$ that consists of propositional atoms and the connectives $\wedge, \vee, \rightarrow, \leftrightarrow, \neg$ to build formulas $\varphi, \psi, \ldots$ with their usual meaning; the canonical tautology $\top$ is defined to be $p \vee \neg p$ for p some arbitrary propositional atom, and the canonical contradiction $\bot$ is defined to be $\neg\top$. The pure propositional language will be denoted with $\mathscr{L}_0$. We will extend this to the core language $\mathscr{L}_c$ (thus $\mathscr{L}_c \subseteq \mathscr{L}_0$) to deal with actions, epistemic and motivational attitudes.

Let us define a set $\mathscr{A} = \{1, \ldots, n\}$ which denotes the agents in the world, and let i be a variable over this denoting some arbitrary agent. Apart from that let us define a set Ac of actions that are either atomic actions (At = $\{a, b, \ldots\}$ wich are all of the same, atomic, length) or composed actions($\alpha, \beta, \ldots$). Composed actions are confirm actions ($\texttt{confirm}\,\varphi$ or, shorthand, $\varphi?$), sequential actions ($\alpha;\beta$) or conditional actions, where conditional actions might be if-then-else ($\texttt{if}\ \varphi\ \texttt{then}\ \alpha\ \texttt{else}\ \beta$) or while actions ($\texttt{while}\ \varphi\ \texttt{do}\ \alpha$).

The actions in Ac, say α, can be used to build new formulas to express the possible *result* of the execution of α by an agent i. We define the formula $\langle \texttt{do}_i(\alpha)\rangle\varphi$ to express that φ is a result of i's execution of α. The *opportunity* for i to do α will be denoted by $\langle \texttt{do}_i(\alpha)\rangle\top$ and i's *capability* of performing the action α by $\mathbf{A}_i\alpha$. The formula $[\texttt{do}_i(\alpha)]\varphi$ is shorthand for $\neg\langle \texttt{do}_i(\alpha)\rangle\neg\varphi$, thus expressing that all possible worlds resulting from the performance of α by i imply φ.

Next let us define the semantics of our system in terms of Kripke models. The models we use are of the kind $\mathcal{M} = \langle W, \pi, \texttt{D}, \texttt{I}, \texttt{M}\rangle$, where W is a non-empty set of worlds. To determine whether a formula $\varphi \in \mathscr{L}$ holds in a state, we write $\mathcal{M}, w \vDash \varphi$, where

$(\mathcal{M}, w)$ is called a state; $\pi, \mathtt{D}, \mathtt{I}$ and $\mathtt{M}$ tell us how to interpret the propositional atoms (i.e. truth assignments) and the Dynamical, Informational and Motivational operators, respectively. Concerning the propositional part we say that $\mathcal{M}, w \vDash p$ if and only if $\pi(w)(p) = true$; the logical connectives are interpreted in $\mathcal{M}, w$ as usual (see [9] for details).

We define for agent i a modal operator X_i to be a necessity operator for a relation R_i (or a function f_i) if the truth of $X_i\varphi$ at state $\mathcal{M}, w$ is defined as the truth of φ in all states s for which $R_i ws$ (or $s \in f_i(w)$, respectively).

The Dynamical part of the models ($\mathtt{D}$) consists or two functions, **r** and **c** which denote the results of actions and the capabilities of agents, respectively. The Informational part consists of the modal operator concerning epistemics, e.g. beliefs (**B**). The Motivational part can be specified as $\mathtt{M} = \langle$W,C,Agenda$\rangle$, where W determines the desirability relation of an agent in a state, C denotes the choices made by an agent in a state, and Agenda records the commitments of agents per state.

2.2 Multi-agent extensions

In order to make a valid system it is important to define the kind of multi-agent systems that are described by the extended system. Since there are many kinds of multi-agent systems and these systems can be very different the definition that we provide might not work for all sorts of multi-agent systems. We try to define the aspects of the framework as universal as possible, i.e. we want the definitions to work with as many different kinds of multi-agent systems as possible; we did not succeed (yet) in doing so for all definitions. The definitions that we provide can all be used in team-based systems 1[1] (from [8]).

Definition 1. Teams:

- *A team is a group of agents that are jointly trying to achieve their common* desires.

Next let us discuss the extensions to the base system we provided in the previous section. For simplicity and easy denoting we give a team of agents a name, T, and its members new designations, $\{\tau_1, \ldots, \tau_k\}$. Moreover, from now on we will always assume that a team has k members,which are denoted by τ_i (for $1 \leq i \leq k$), unless stated otherwise.

Now let us define a new set of atomic actions At', which contains the atomic parts of multi-actions. Multi-actions are those actions that require more than one agent to perform, and the parts of these multi-actions are the parts that the participating agents need to perform. Thus, a multi-action that has to be performed by two different agents, e.g `lift a heavy table`, contains two actions parts, which are `lift left side` and `lift right side` respectively. Since we want to reason about actions in team context we define a set of atomic action tuples At*, which denote those actions that a team as a whole can perform. Since a team is a group of agents we can break these team-actions into single-agent actions or multi-action parts, thus, when considering a team of

[1] The definitions provided throughout this paper might be sufficient to describe other kinds of teams or multi-agent systems as well.

k agents a team-action m can be seen as a tuple $(m_1, \ldots, m_k)$, where m_i denotes the action performed by the i^{th} member of the team; moreover since the members of the team can perform both atomic actions (At) and atomic multi-action parts (At'), it holds that either $m_i \in$ At or $m_i \in$ At'. Next we extend the set of atomic action tuples (At*) with the team variants of the confirm, sequential composition and conditional actions to obtain the set of composed team-actions (Ac*).

Note that we allow individual agents to merely perform atomic actions, i.e. actions of atomic length, to overcome possible synchronization problems. And since we demand that members of a team always perform such atomic actions, we introduce a special atomic action which changes nothing to the world, i.e. the null-action. Also note that when a team (T) performs a (composed) team-action (μ) it means that the members of T perform their designated parts at the same time, i.e. $do_T(\mu) \equiv do_{\tau_1}(\mu_1) \parallel \ldots \parallel do_{\tau_k}(\mu_k)$.

3 Multi-agent Operators

In this section we discuss the operators we added to the existing KARO logical framework in order to describe multi-agent systems. First we describe the operators concerning the informational level in subsection 3.1. Next we discuss in subsection 3.2 the dynamic and action operators. After that the operators related to the motivational level are discussed in subsection 3.3. And finally in subsection 3.4 we discuss the social operators.

3.1 Informational Level

One of the main aspects of agents is their capability to reason. Agents know things and can use this knowledge to make changes in the world and adjust their desires and goals to match their knowledge.

The KARO system used standard S5 models to denote the knowledge of the agents. Knowledge is, in our opinion, however, much too strong for agent systems since axiom T (i.e. $\mathbf{K}_i\varphi \rightarrow \varphi$, also called reflexivity) requires all the things the agents know to be true. So instead of using knowledge to express reasoning of the agents we have chosen to use beliefs, as they at least allow us to express feasibility in reasoning. The notion of belief we use is based on the KD45 system. The most significant difference between beliefs and knowledge is that agents can believe things that are actually not true at that moment.

Since we are developing a multi-agent system we define some aspects about the beliefs in teams. We introduce the notion of *everybody believes* (**EB**), analogous to the notion of 'everybody knows' [9]; where $\mathbf{EB}_T\varphi$ denotes that all members in the team T believe that φ holds. This is, however, not enough to denote the beliefs of the team as a whole. In fact, we want the team as a whole to have the same reasoning capabilities as individual agents, e.g. the team should believe its beliefs and its disbeliefs. A notion that satisfies these conditions is known for epistemic frameworks as 'common knowledge'. We introduce a notion analogous to 'common knowledge', adapted to be usable in the KD45 system, which we will call *mutual belief* (**MB**).

Definition 2. Beliefs in Multi-Agent Environments[2]**:**

$A1$ $All\ (instances\ of)\ propositional\ tautologies.$

$A2$ $(\mathbf{B}_i\varphi \wedge \mathbf{B}_i(\varphi \to \psi)) \to \mathbf{B}_i\psi$ $\quad K$

$A3$ $\mathbf{B}_i\bot \to \bot$ $\quad D$

$A4$ $\mathbf{B}_i\varphi \to \mathbf{B}_i\mathbf{B}_i\varphi$ $\quad 4$

$A5$ $\neg\mathbf{B}_i\varphi \to \mathbf{B}_i\neg\mathbf{B}_i\varphi$ $\quad 5$

$A6$ $\mathbf{EB}_T\varphi \leftrightarrow \bigwedge_{i=1}^{k} \mathbf{B}_{\tau_i}\varphi$ $\quad EB$

$A7$ $\mathbf{MB}_T\varphi \to \mathbf{EB}_T\varphi$

$A8$ $\mathbf{MB}_T\varphi \to \mathbf{EB}_T\mathbf{MB}_T\varphi$

$A9$ $(\mathbf{MB}_T\varphi \wedge \mathbf{MB}_T(\varphi \to \psi)) \to \mathbf{MB}_T\psi$

$A10$ $\mathbf{MB}_T(\varphi \to \mathbf{EB}_T\varphi) \to (\mathbf{EB}_T\varphi \to \mathbf{MB}_T\varphi)$

$R1$ $\dfrac{\varphi \quad \varphi \to \psi}{\psi}$ $\quad Modus\ Ponens$

$R2$ $\dfrac{\varphi}{\mathbf{B}_i\varphi}$ $\quad Necessitation$

$R3$ $\dfrac{\varphi}{\mathbf{EB}_T\varphi}$ $\quad EB\ Necessitation$

$R4$ $\dfrac{\varphi}{\mathbf{MB}_T\varphi}$ $\quad MB\ Necessitation$

Note that obtaining something like mutual belief is extremely difficult in real world settings[3] where communication between agents is not reliable. To cope with this problem we demand that communication between agents is perfect, i.e. if an agent sends a message he can be absolutely sure that the other agent received it.

3.2 Dynamic/Action Level

Another important aspect of agents is their ability to act, and reason about their actions and the results of these actions. In order to reason about their actions and action results we introduce several operators. Most of these multi-agent operators make use of the idea that the actions of a team are done by its members; therefore most of the following operators depend on the single-agent variants of these operators[4].

One of the aspects of actions that we want agents to reason about is the fact whether a certain state or proposition can be reached; we want them to check and examine the implementability of formulas. A formula is implementable if a plan (a series of sequential actions) exists such that it is practically possible to achieve the formula by performing the plan.

A team is practically possible to achieve a formula (say φ) by performing an action (or action sequence, say α) when it holds that the team has the opportunity and ability to perform α and in the state resulting from the performance of α φ always holds.

[2] Note that rules $R2$ and $R3$ are derivable from $R4$, we merely included them for completeness.

[3] See [9] or [10] for the conditions to obtain common knowledge.

[4] See [6] for details on the single-agent operators.

Definition 3. Ability, Practical Possibility and Implementabilty:

$$\mathbf{A}_T\beta \equiv \bigwedge_{i=1}^{k}(\mathbf{A}_{\tau_i}\beta_i)$$
$$\mathbf{PracPoss}_T(\beta,\varphi) \equiv \langle \mathrm{do}_T(\beta)\rangle\top \wedge [\mathrm{do}_T(\beta)]\varphi \wedge \mathbf{A}_T\beta$$
$$\mathcal{M}, s \models \mathbf{Impl}_T\varphi \Leftrightarrow \exists n \in \mathbb{N}\ \exists b_1, \dots b_n \in \mathrm{At}^*\ (\mathcal{M}, s \models \mathbf{PracPoss}_T(b_1; \dots; b_n, \varphi))$$

Since **Implementability** denotes whether a formula or state can be reached by a certain plan, we use it to define a new operator called **Never**, previously unknown in the KARO framework, which denotes that there is no possible way to achieve a certain state or formula.

We also introduce the notions of **Joint-Can** and **Joint-Cannot** which denote that a team believes that they are (un)able (i.e. practical possible) to achieve a certain state.

Definition 4. Never, Joint-Can and Joint-Cannot ($\varkappa \in \{i, T\}$)**:**

$$\mathbf{Never}_\varkappa\varphi \equiv \neg\mathbf{Impl}_\varkappa\varphi$$
$$\mathbf{Joint{-}Can}_T(\beta.\varphi) \equiv \mathbf{MB}_T\mathbf{PracPoss}_T(\beta,\varphi)$$
$$\mathbf{Joint{-}Cannot}_T(\beta.\varphi) \equiv \mathbf{MB}_T\neg\mathbf{PracPoss}_T(\beta,\varphi)$$

An important aspect of reasoning about actions is the ability to determine the results of actions. Since teams are performing actions that are combined of parts that are performed by the team members we would like to "construct" the results of a team action from the results of these action parts. One problem with this approach is that the action parts might not seem to be very intuitive, e.g. what is the result of one hand performing a handshake? On the other hand this approach allows us to reason about possible synchronization problems that might arise in multi-agent environments[5].

One other problem with this approach is that when you try to determine the results of your action you probably do not (yet) know what the other members of your team are doing. Besides that it is desirable that, in some cases, action parts have a different result when the other, perhaps matching, action part is not performed. For instance, when agent τ_i is performing the action `lift left side` while agent τ_j refrains from performing his part, i.e. `lift right side`, the result of the action should be that all items on the table fall off the right side of the table. However, when τ_j does synchronize and performs his part, we would like the result to be that the table with all items on it is lifted.

To overcome these problems we redefine the function **r**[6] which was defined in single-agent KARO for the purpose of result determination. Moreover our new result

[5] In this paper, though, these problems are eliminated by the assumptions that all agents synchronize all actions, that all actions are of the same (atomic) length and that communication is perfect.

[6] The function **r** is part of the dynamic logic and provides the resulting worlds of an action performed by an agent in a world, i.e. $\mathbf{r}(i, a)(\mathcal{M}, s)$ will return all worlds resulting from the performance of a by i in the state $\mathcal{M}, s$.

function will use an "open-ended" approach (similar to the notion of Alternating Temporal Logic, see [11]). This means that the result function will return a set containing all worlds that might result from doing the action. From these worlds the actual resulting worlds are determined by combining the sets found by all team members.

Definition 5. Result function:

- $\boldsymbol{r}(\tau_i, \beta_i)(\mathcal{M}, s)$ *yields the set of all states* $(\mathcal{M}', s')$ *resulting from* τ_i *doing* β_i.
- *For* $\beta = (\beta_1, \ldots, \beta_k) \in \mathrm{At}^*$ *and* $T = \{\tau_1, \ldots, \tau_k\} \subseteq \mathscr{A}$, $\boldsymbol{r}(T, \beta)(\mathcal{M}, s) = \bigcap_{i=1}^{k} \boldsymbol{r}(\tau_i, \beta_i)(\mathcal{M}, s)$

Now we have the means to determine the results of atomic actions in teams we want to extend this definition such that it can handle composed team actions as well. Therefore we overwrite the **r*** function, which handled composed actions in single-agent KARO.

Definition 6. Extensions of r ($\varkappa \in \{i, T\}$)**:**

$$
\begin{array}{ll}
\mathbf{r}^*(\varkappa, a)(\mathcal{M}, s) & = \boldsymbol{r}(\varkappa, a)(\mathcal{M}, s) \\
\mathbf{r}^*(i, any)(\mathcal{M}, s) & = \forall a \in \mathrm{At} \bigcup \mathbf{r}^*(i, a)(\mathcal{M}, s) \\
\mathbf{r}^*(T, any)(\mathcal{M}, s) & = \forall a \in \mathrm{At}^* \bigcup \mathbf{r}^*(T, a)(\mathcal{M}, s) \\
\mathbf{r}^*(\varkappa, \alpha)\{(\mathcal{M}_1, s_1), \ldots, & = \bigcup_{j=1}^{n} \mathbf{r}^*(\varkappa, \alpha)(\mathcal{M}_j, s_j) \\
\quad (\mathcal{M}_n, s_n)\} & \\
\mathbf{r}^*(i, \mathtt{confirm}\,\varphi)(\mathcal{M}, s) & = (\mathcal{M}, s) \text{ if } \mathcal{M}, s \models \varphi \text{ and } \emptyset \text{ otherwise} \\
\mathbf{r}^*(T, \mathtt{joint-confirm}\,\varphi)(\mathcal{M}, s) & = (\mathcal{M}, s) \text{ if } \mathcal{M}, s \models \varphi \text{ and } \emptyset \text{ otherwise} \\
\mathbf{r}^*(\varkappa, \alpha_1; \alpha_2)(\mathcal{M}, s) & = \mathbf{r}^*(\varkappa, \alpha_2)(\mathbf{r}^*(\varkappa, \alpha_1)(\mathcal{M}, s)) \\
\mathbf{r}^*(\varkappa, \mathtt{if}\,\varphi\,\mathtt{then}\,\alpha_1 & = \mathbf{r}^*(\varkappa, \alpha_1)(\mathcal{M}, s) \text{ if } \mathcal{M}, s \models \varphi \text{ and} \\
\quad \mathtt{else}\,\alpha_2\,\mathtt{fi})(\mathcal{M}, s) & \quad \mathbf{r}^*(\varkappa, \alpha_2)(\mathcal{M}, s) \text{ otherwise} \\
\mathbf{r}^*(T, \mathtt{while}\,\varphi & = (\mathcal{M}', s') \text{ iff } \exists k \in \mathbb{N}\, \exists \mathcal{M}_0, s_0 \ldots \exists \mathcal{M}_k, s_k \\
\quad \mathtt{do}\,\beta\,\mathtt{od})(\mathcal{M}, s) & \quad [\mathcal{M}_0, s_0 = \mathcal{M}, s \& \mathcal{M}_k, s_k = \mathcal{M}', s' \& \forall j < k \\
 & \quad [\mathcal{M}_{j+1}, s_{j+1} \in \mathbf{r}^*(T, \mathtt{confirm}\,\varphi; \beta)(\mathcal{M}_j, s_j)] \\
 & \qquad\qquad \mathcal{M}', s' \models \neg\varphi]
\end{array}
$$

Finally we introduce a new operator, inspired by temporal logics, which we will call Until, to check whether a formula remain true until some condition is met. The Until will derive its temporal aspect from actions. Since the performance of an action takes time, we can use the performance of an atomic action as the standard of time[7]. We define the Until to coincide with the intuition we have of the term; i.e. $\varphi\ \mathrm{Until}_T\ \psi$ holds if T can reach a state by doing some number of actions where ψ necessarily holds, and in the meantime, in each state we pass φ holds. Since it does not need to be specified what actions we need to do to achieve that state, we introduce the *any* set, which is the union of all atomic actions all agents can do in a world.

Definition 7. Until:

$$\varphi\ \mathrm{Until}_T\ \psi \equiv \left[\mathrm{do}_T\left(((\varphi \wedge \neg\psi)?; any)^*; (\neg\varphi \vee \psi)?\right)\right]\psi$$

[7] So the time it takes to perform an atomic action equals one "time unit".

3.3 Motivational Level

The basic motivational level of agents is related to the first operators we discuss, namely Wishes and Choice. Since our teams cooperate on the basis of their desires we can introduce the desire of the team as the intersection of the desires of the team members. Notice that for other kinds of multi-agent systems the notion of desire is not as evident as presented here. What can we say about the desires of a group leaded by a "super"-agent? Is the desire of that team equivalent to the desire of the leader? This is an aspect of multi-agent systems that is very vague, and needs a lot of refinement.

Similar to the single-agent KARO framework we introduce the idea that a team should make choices and select the wishes they are willing to pursue. The choice operator in single-agent KARO is dependent on the semantically defined function C, that is part of the motivational aspect of the Kripke models. We extend this function to include the choices made by teams in the worlds.

Definition 8. Wishes & Choice:

$$\mathbf{W}_T\varphi \equiv (\bigwedge_{i=1}^{k} \mathbf{W}_{T_i}\varphi)$$
$$\mathcal{M}, s \models \mathbf{C}_T\varphi \Leftrightarrow \varphi \in \mathrm{C}(T, s)$$

Desires are a nice feature to have in a framework, but if the agents do not do anything with them, they are just hopeless dreams that will never be reached. In order to make the desires an useful attribute of our agents we need to introduce goals. However, the fact that the agents wish something to be true is only one aspect of a goal, the state they want to be true should be reachable, i.e. implementable. Besides that we want the agents to continu to try and achieve that state, and when done to inform the rest of the team that they achieved their objective. In order to achieve this last notion of persistency we make use of weak-goals (derived from Cohen, Levesque e.a., see [12] and [13]).

Proposition 1. *An agent has a* **Weak-Goal** *with respect to a team to bring about φ if either of these conditions holds:*

- *The agent has a normal goal (i.e. single-agent goal) to bring about φ, that is, the agent does not yet believe that φ is true, desires φ to be true and sees a way to implement this.*
- *The agent believes that φ is true or will never be true,* but *has as a goal that the status of φ be mutually believed by all the team members.*

Proposition 2. *A team T has a* **Joint-Goal** *to bring about φ when all of these conditions are satisfied:*

- *φ should not be true when the team adopts it as its goal;*
- *φ should be the chosen wish of the team, i.e. the team should be willing to pursue φ;*
- *φ should be implementable by the team, i.e. the team should not have an impossible desire as their goal;*

– *All members of T should try to achieve their weak-goal φ with respect to T as long as there is no mutual belief established that φ is either achieved or unachievable.*

Definition 9. Joint-Goal & Weak-Goal:

$$\begin{aligned}
\mathbf{Weak{-}Goal}^{T}_{\tau_i}\varphi \equiv\ & (\neg\mathbf{B}_{\tau_i}\varphi \wedge \neg\mathbf{B}_{\tau_i}(\mathbf{Never}_T\varphi) \wedge \mathbf{Goal}_{\tau_i}\varphi) \vee \\
& (\mathbf{B}_{\tau_i}\varphi \wedge \mathbf{Goal}_{\tau_i}(\mathbf{MB}_T\varphi)) \vee \\
& (\mathbf{B}_{\tau_i}(\mathbf{Never}_T\varphi) \wedge \mathbf{Goal}_{\tau_i}(\mathbf{MB}_T(\mathbf{Never}_T\varphi))) \\
\mathbf{Joint{-}Goal}_T\varphi \equiv\ & \mathbf{MB}_T(\neg\varphi \wedge \mathbf{W}_T\varphi \wedge \mathbf{C}_T\varphi \wedge \mathbf{Impl}_T\varphi) \wedge \\
& [\bigwedge_{i=1}^{k} \mathbf{MB}_T(\mathbf{Weak-Goal}^{T}_{\tau_i}\varphi \\
& \qquad\qquad \mathrm{Until}_T \\
& (\mathbf{MB}_T(\varphi) \vee \mathbf{MB}_T(\mathbf{Never}_T\varphi))]
\end{aligned}$$

The last operator we introduce in this section is the Possible-Intend operator, which is required to achieve all aspects of intentions (described in [14] and [15]). As Bratman describes in [14] intentions consist of two aspects, namely achievement intentions and maintenance intentions[8]. One might notice that achievement intentions are already fully implemented in the framework through the use the choice operator in the joint-goal definition. The choice operator ensures that conflicting formulas are not adopted as a goal.

Maintenance intentions are partly implemented by the use of the joint- and weak-goals. These operators make sure that a team pursues a chosen formula until they believe it is either achieved or unachievable. Maintenance intentions, however, should also allow the agents to drop a goal when they are not interested anymore. To achieve this in our framework we make use of possible intentions and the `commit_to` action, the latter is described in section 3.4.

Definition 10. Possible Intentions:

$$\begin{aligned}
\mathbf{Joint{-}PossInt}_T(\beta,\varphi) \equiv\ & \mathbf{Joint{-}Can}_T(\alpha,\varphi) \wedge \\
& \mathbf{MB}_T(\mathbf{Joint{-}Goal}_T\varphi)
\end{aligned}$$

In our framework intentions mean that the team mutually believes that the goal they have chosen to pursue is obtainable. This combined with the altered definition of the `commit_to` action is enough to ensure all aspects of intentions in the framework.

3.4 Social Level

In order to be able to achieve the goals, wishes and intentions specified in the previous subsection the agents need to commit themselves to the plan they devised and the goals they adopted. Since the plan or goal of the team is a tuple of actions that are to be performed by the members of the team, the commit action specified in single-agent KARO

[8] Maintenance intentions make sure an agent will keep on trying to achieve its intention as long as he believes it is not avhieved already or unachievable.

does not have to be changed much. The only feature missing is that agents may also commit to actions that *the team* is intending[9]. Therefore we redefine the `commit_to` action as follows.

Definition 11. Commitments:

- *When doing a* `commit_to` *on an action α of which the agent does not intend the consequences φ (i.e. $\neg(\mathbf{PossIntend}_i(\alpha, \varphi)$ for some φ) or when that action is part of a multi-action and the team the agent is a member of does not intend the consequences φ of that multi-action (i.e. $\neg\mathbf{Joint{-}PossIntend}_T(\beta, \varphi))$ for some φ and β when $\alpha \in \beta$) the result is the "empty world" $\mathcal{E}$[10], i.e. an agent can only commit to actions of which either it or its team intends the consequences;*
- *If successful, the agent's agenda is updated with α in the world it is in;*
- *Moreover, also in all other possible worlds that are related to this world, either by performing actions or by considering epistemic alternatives, the agenda is updated appropriately.*

Likewise we redefine the `uncommit` action to include this property, i.e. agents should only be allowed to uncommit when neither themselves nor the team they are part of intends the result of the action anymore. Moreover, we have included that agents are able to uncommit when they believe that other members of their team are not committed to their part of the team plan.

Definition 12. Uncommitment:

- *An agent can only perform an* `uncommit` *with respect to α when it is committed to α;*
- *When uncommitting, the actions related to α (subactions) that are still on its agenda are removed from the agenda (in this world and all other related worlds, either by performing actions or by considering epistemic alternatives);*
- *More importantly, an agent is only capable of performing an* `uncommit` *with respect to α if it does not intend, individually (i.e. $\neg\mathbf{PossIntend}_i(\alpha, \varphi)$, for some φ) or as part of a team (i.e. $\neg\mathbf{Joint{-}PossIntend}_T(\beta, \varphi)$ for some φ and multi-action β, where $\alpha \in \beta$), the consequences, φ, of α anymore. And when α is part of a multi-action, and the agent believes that other members of the team did not commit towards their part of the plan, the agent is also able to perform an uncommit toward α.*

Let us introduce a new operator to determine whether a team is actually committed to an action in order to simplify reasoning about team commitments. This operator will be dependent on the single-agent variant, which returns true when the agent is committed to the specified action.

[9] The original `commit_to` action does include the ability to commit to actions that the agent intends. Since a member of a team might not intend the action it has to perform in favour of the team, it has to be able to commit to that action if the team intends the action.

[10] The special state $\mathcal{E} = (\mathcal{M}, \epsilon)$, where ϵ is a world where all impossible actions lead to; from there, no other actions can be performed anymore.

Definition 13. Joint-Commitment:

$$\mathbf{Joint\!-\!Commit}_T\beta \equiv \bigwedge_{i=1}^{k}(\mathbf{MB}_T(\mathbf{Com}_{\tau_i}\beta_i) \wedge \mathbf{Com}_{\tau_i}\beta_i$$

The last operator we discuss in this section is the agenda. The team agenda (or joint-agenda) is the "object" that stores the actions that the team has committed itself to. This joint-agenda is not centralized, but distributed among the members of the team[11]. In order to cope with agents being a member of multiple teams, and with the fact that a team might commit itself to more than one action, i.e. the team is not necessarily single-minded, we demand that actions written to the agenda are of a specific format. So instead of just writing the action (say β_i) to the agenda, we write $(\beta_i; \beta, T)$ to the agenda, and thus including the multi-action it is part of and the team on whose behalf the action is to be performed.

Definition 14. Joint-Agenda:

$$\begin{aligned}\mathbf{Joint\!-\!Agenda}_T = \{\beta|\\ &(\beta_1 : \beta, T) \in \mathbf{Agenda}_{\tau_1}\\ &\vdots\\ &(\beta_k : \beta, T) \in \mathbf{Agenda}_{\tau_k}\}\end{aligned}$$

The agenda we use in this paper is a non-"physical" agenda in that it is maintained by all members of the team together. Each member remembers the actions it is committed to himself, and by doing so according the specific format they allow other members (or even agents or persons outside the team) to construct an agenda for the team as a whole.

4 Conclusions & Future Work

In this paper we have proposed an extension of the single-agent KARO framework to describe multi-agent systems. We focussed on the ability to describe and handle teams of rational agents. Although the extension is succesful in describing these kinds of multi-agent systems, it is far from complete. We would like the multi-agent KARO framework to develop into a complete framework able of describing most, if not all, kinds of multi-agent systems possible. Therefore we tried to define the new operators as universal as possible, and we have succeeded in making various of the new operators capable of usage in multi-agent systems that are beyond focus for this paper.

One of the main problems in trying to make the framework capable of handling all sorts of multi-agent systems is the wishes operator. It is not clear what the wishes of multi-agent cooperations other than teams might be, not even to mention the wishes of

[11] It is, however, quite imaginable to have a special member in the team that manages the agenda and sees to it that the other members of the team fulfil their commitments.

competitive systems. Much research has to be performed on this before an attempt can be made to make an "universal" multi-agent framework.

One aspect that needs further refinement in the current multi-agent KARO is the fact that we used mutual beliefs. It is know that obtaining mutual belief (and common knowledge, which has the same problems) is very difficult in normal settings. We solved this in this paper by assuming perfect communication, which is usualy not available in real life settings. Perhaps alternatives to this problem become available, or methods might be found that facilitate the obtaining of mutual belief.

References

1. A.S. Rao and M.P. Georgeff. Modeling rational agents within a BDI-architecture. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, pages 473–484. Morgan Kaufmann, 1991.
2. M. Wooldridge. Intelligent agents. In *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999.
3. M.P. Singh, A.S. Rao, and M.P. Georgeff. Formal methods in DAI: Logic-based representations and reasoning. In *Introduction to Distributed Artificial Intelligence*, chapter 8. MIT Press, 1998.
4. E.A. Emerson. Temporal and modal logis. In *Handbook of Theoretical Computer Science*, chapter 16, pages 995–1072. Elsevier Science Publishers b.b., 1990.
5. E. Sandewal and Y. Shoham. Non-monotonic temporal reasoning. In *The Handbook of Logic in AI and Logic Programming*, volume 4, pages 439–484. Oxford University Press, 1995.
6. W. van der Hoek, B. van Linder, and J.-J. Ch. Meyer. An integrated modal approach to Rational Agents. In *Foundations of Rational Agency*, M. Woolridge and A. Rao (eds.), Kluwer, Dordrecht, 1999, pp. 37–75.
7. M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.
8. H. Aldewereld. Rational Teams: Logical Aspects of multi-Agent Systems. *Preprint Series, Cognitive Artificial Intelligence Master's Theses.* http://preprints.phil.uu.nl, Department of Philosophy, Utrecht University, 2002.
9. J.-J. Ch. Meyer and W. van der Hoek. *Epistemic Logic for AI and Computer Science*. Cambridge University Press, 1995.
10. R. Fagin, J.Y. Halpern, Y. Moses, and M.Y. Vardi. *Reasoning about Knowledge*. The MIT Press, 1995.
11. R. Alur, T.A. Henzigger, and O. Kupferman. Alternating temporal logic. In *Proc. 38th IEEE Symposium on Foundations of Computer Science*, pages 100–109. 1997.
12. P. Cohen and H. Levesque. Teamwork. *Nous*, 24(4):487–512, 1991.
13. P. Cohen, H. Levesque, and I. Smith. On team formation. In J. Hintikka and R. Tuomela, editors, *Contemporary Action Theory*. Synthese, 1997.
14. M.E. Bratman. *Intentions, Plans and Practical Reasoning*. Harvard University Press, Cambridge, MA, 1987.
15. P. Cohen and H. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42(3):213–261, 1990.