A Model of Rational Agency for Communicating Agents

Shakil M. Khan and Yves Lespérance

Dept. of Computer Science, York University, Toronto, ON, Canada M3J 1P3
{skhan, lesperan}@cs.yorku.ca

Abstract. The Cognitive Agent Specification Language (CASL) is a framework for specifying and verifying complex communicating multiagent systems. In this paper, we extend CASL to incorporate a formal model of means-end reasoning suitable for a multiagent context. In particular, we define a simple model of cooperative ability, give a definition of rational plans, and show how an agent's intentions play a role in determining her next actions. This bridges the gap between intentions to achieve a goal and intentions to act. We also define a notion of subjective plan execution and show that in the absence of interference, an agent that is able to achieve a goal, intends to do so, and is acting rationally and subjectively executing plans, will eventually achieve it.

1 Introduction

Most agent theories [1, 19] suffer from a similar problem: they axiomatize the relation between the different mental attitudes of the agents and the physical states of the world; but they do not account for how the agents will achieve their goals, how they plan and commit to plans. Ideally, an agent's intention to achieve a state of affairs in a situation should drive the agent to intend to execute a plan that she thinks is rational in that situation. In other words, an agent's future directed intentions should lead her to adopt rational plans and eventually achieve her intentions.

Another recent thread in agent theory introduces a procedural component to the framework in an attempt to close the gap between agents' intentions to achieve a state of affairs and their intentional actions, as well as to support the modeling of complex multiagent systems. One example of this is the Cognitive Agent Specification Language (CASL) [26], which is a framework for specifying and verifying complex communicating multiagent systems. However, it is somewhat restricted in the sense that it requires the modeler to specify agent behavior explicitly, and the program that controls the agent's actions need not be consistent with the agent's intentions, or do anything to achieve them.

In this paper, we propose a solution to this problem by extending CASL. In particular, we define rational plans and ability in a multiagent context, and use these notions to link future and present directed intentions. We introduce a special action, the *commit* action, that makes the agent commit to a plan, and define a meta-controller *BehaveRationallyUntil* that has the agent act rationally to achieve a specific goal by choosing and committing to a rational plan, and carrying it out. We also define a notion of subjective execution of plans where the agent must have the required knowledge to execute the plan. Then we show that given that an agent has an intention, she will act to achieve it provided that she is able to do so.

The paper is organized as follows: in the next section, we outline previous work on CASL. In Section 3, we develop a simple formalization of cooperative ability for agents working in a multiagent setting. In Section 4, we define rational plans, relate future and present directed intentions, and discuss what it means for an agent to be behaving rationally and executing plans subjectively. We also state a theorem that links an agent's intentions and abilities to the eventual achievement of her intentions.

2 CASL

In CASL [26], agents are viewed as entities with mental states, i.e., knowledge and goals, and the specifier can define the behavior of the agents in terms of these mental states. CASL combines a declarative action theory defined in the situation calculus with a rich programming/process language, ConGolog [4]. Domain dynamics and agents' mental states are specified declaratively in the theory, while system behavior is specified procedurally in ConGolog.

In CASL, a dynamic domain is represented using an action theory [21] formulated in the situation calculus [16], a second order language for representing dynamically changing worlds in which all changes are the result of named actions. CASL uses a theory that includes the following set of axioms:

- domain-independent foundational axioms describing the structure of situations [9],
- action precondition axioms, one per action,
- successor state axioms (SSA), one per fluent, that encode both effect and frame axioms and specify exactly when the fluent changes [20],
- initial state axioms describing what is true initially including the mental states of the agents,
- axioms identifying the agent of each action, and,
- unique name axioms for actions.

Within CASL, the behavior of agents is specified using the notation of the logic programming language ConGolog [4], the concurrent version of Golog [14]. A typical ConGolog program is composed of a sequence of procedure declarations, followed by a complex action. Complex actions can be composed using constructs that include the ones given in Table 1.[1] These constructs are mostly self-explanatory. Intuitively, $\pi x.\delta$ nondeterministically picks a binding for the

[1] Since we have predicates that take programs as arguments, we need to encode programs as first-order terms as in [4]. For notational simplicity, we suppress this encoding and use formulae as terms directly. Also, here ϕ is used to denote a formula whose fluents may contain a placeholder *now* instead of a situation argument. The

Table 1. Examples of ConGolog Constructs

a,	primitive action
$\phi?$,	wait for a condition
$(\delta_1; \delta_2)$,	sequence
$(\delta_1 \mid \delta_2)$,	nondeterministic choice
$\pi x.\delta$,	nondet. choice of argument
If ϕ Then δ_1 Else δ_2 EndIf,	conditional
While ϕ Do σ EndWhile,	while loop
$\beta(\overrightarrow{p})$,	procedure call.

variable x and performs the program δ for this binding of x. ConGolog also supports nondeterministic iteration, concurrent execution with and without priorities, and interrupts. To deal with multiagent processes, primitive actions in CASL take the agent of the action as argument.

The semantics of the ConGolog process description language is defined in terms of *transitions*, in the style of structural operational semantics [18]. Two special predicates *Final* and *Trans* are introduced, and are characterized by defining axioms for each of the above constructs, where $Final(\delta, s)$ means that program δ may legally terminate in situation s, and where $Trans(\delta, s, \delta', s')$ means that program δ in situation s may legally execute one step, ending in situation s' with program δ' remaining. The overall semantics of a ConGolog program is specified by the *Do* relation:

$$Do(\delta, s, s') \doteq \exists \delta' \cdot (Trans^*(\delta, s, \delta', s') \wedge Final(\delta', s')).$$

$Do(\delta, s, s')$ holds if and only if s' can be reached by performing a sequence of transitions starting with program δ in s, and the remaining program δ' may legally terminate in s'. Here, $Trans^*$ is the reflexive transitive closure of the transition relation $Trans$.

CASL allows the specifier to model the agents in terms of their mental states by including operators to specify agents' information (i.e., their knowledge), and motivation (i.e., their goals or intentions). Following [17, 24], CASL models knowledge using a possible worlds account adapted to the situation calculus. $K(agt, s', s)$ is used to denote that in situation s, *agt* thinks that she could be in situation s'. s' is called a *K-alternative situation* for *agt* in s. Using K, the knowledge or belief of an agent, Know(agt, ϕ, s), is defined as $\forall s'(K(agt, s', s) \supset \phi(s'))$. Two useful abbreviations are also defined: KWhether$(agt, \phi, s) \doteq$ Know(agt, ϕ, s) $\vee$ Know$(agt, \neg\phi, s)$, i.e., *agt* knows whether ϕ holds in s, and KRef$(agt, \theta, s) \doteq$ $\exists t.$Know$(agt, t = \theta, s)$, i.e., *agt* knows who/what θ is. In CASL, K is constrained to be reflexive, transitive and euclidean in the initial situation to capture the

placeholder gets replaced by a situation by an outer construct. $\phi(s)$ is the formula that results from replacing *now* with s. Where the intended meaning is clear, we suppress the placeholder.

fact that agents' knowledge is true, and that agents have positive and negative introspection. As shown in [24], these constraints then continue to hold after any sequence of actions since they are preserved by the successor state axiom for K.

Scherl and Levesque [24] showed how to capture the changes in beliefs of agents that result from actions in the successor state axiom for K. These include knowledge-producing actions that can be either binary sensing actions or non-binary sensing actions. Following [13], the information provided by a binary sensing action is specified using the predicate $SF(a, s)$, which holds if the action a returns the binary sensing result 1 in situation s. Similarly for non-binary sensing actions, the term $sff(a, s)$ is used to denote the sensing value returned by the action.

Lespérance [12] extends the SSA of K in [24] to support two variants of the *inform* communicative action, namely *informWhether* and *informRef*. Here, $inform(inf, agt, \phi)$, $informWhether(inf, agt, \psi)$, and $informRef(inf, agt, \theta)$ mean that *inf* informs *agt* that ϕ currently holds, *inf* informs *agt* about the current truth value of ψ, and *inf* informs *agt* of who/what θ is, respectively. The preconditions of these three actions are as follows:[2]

$$\begin{aligned}
&Poss(inform(inf, agt, \phi), s) \equiv \\
&\qquad \mathrm{Know}(inf, \phi, s) \land \neg\mathrm{Know}(inf, \mathrm{KWhether}(agt, \phi, now), s), \\
&Poss(informWhether(inf, agt, \psi), s) \equiv \\
&\qquad \mathrm{KWhether}(inf, \psi, s) \land \neg\mathrm{Know}(inf, \mathrm{KWhether}(agt, \psi, now), s), \\
&Poss(informRef(inf, agt, \theta), s) \equiv \\
&\qquad \mathrm{KRef}(inf, \theta, s) \land \neg\mathrm{Know}(inf, \mathrm{KRef}(agt, \theta, now), s).
\end{aligned}$$

In other words, the agent *inf* can inform *agt* that ϕ, iff *inf* knows that ϕ currently holds, and does not believe that *agt* currently knows the truth value of ϕ, and similarly for *informWhether* and *informRef*. The SSA for K is defined as follows:

$$\begin{aligned}
&K(agt, s^*, do(a, s)) \equiv \exists s'.\ [K(agt, s', s) \land s^* = do(a, s') \land Poss(a, s') \land \\
&\quad ((BinarySensingAction(a) \land Agent(a) = agt) \supset (SF(a, s') \equiv SF(a, s))) \land \\
&\quad ((NonBinarySensingAction(a) \land Agent(a) = agt) \supset \\
&\qquad\qquad (sff(a, s') = sff(a, s))) \land \\
&\quad \forall inf, \phi.\ (a = inform(inf, agt, \phi) \supset \phi(s')) \land \\
&\quad \forall inf, \psi.\ (a = informWhether(inf, agt, \psi) \supset (\psi(s') \equiv \psi(s))) \land \\
&\quad \forall inf, \theta.\ (a = informRef(inf, agt, \theta) \supset (\theta(s') = \theta(s)))].
\end{aligned}$$

This says that after an action happens, every agent learns that it has happened. Moreover, if the action is a sensing action, the agent performing it acquires knowledge of the associated proposition or term. Furthermore, if the action involves someone informing *agt* that ϕ holds, then *agt* knows this afterwards, and

[2] We modified the preconditions given in CASL by adding the second clause on the right side. Also, the SSA for K presented below is a bit different from that of CASL, and similar to the one given by Lespérance [12].

similarly for *informWhether* and *informRef*. Note that since all agents are aware of all actions, with $inform(inf, agt, \phi)$, every agent learns that ϕ is true. However, with $informWhether(inf, agt, \psi)$ and $informRef(inf, agt, \theta)$, only the addressee learns the truth value/value of ψ/θ. So with the latter, there is some private communication. Also note that this axiom only handles knowledge expansion, not revision.

CASL extends the framework described in [10] to incorporate goal expansion and a limited form of goal contraction. Goals or intentions are modeled using an accessibility relation W over possible worlds (situations, in this case). The W-accessible worlds for an agent are the ones where she thinks that all her goals are satisfied. W-accessible worlds may include worlds that the agent thinks are impossible, unlike Cohen and Levesque's [1] G-accessible worlds. But intentions are defined in terms of the more primitive W and K relations so that the intention accessible situations are W-accessible situations that are also compatible with what the agent knows, in the sense that there is a K-accessible situation in their history. This guarantees that agents' intentions are realistic, that is, agents can only intend things that they believe are possible. Thus we have:

$$\begin{aligned}&\text{Int}(agt, \psi, s) \doteq\\ &\forall now, then.[W(agt, then, s) \land K(agt, now, s) \land now \leq then] \supset \psi(now, then).\end{aligned}$$

This means that the intentions of an agent in s are those formulas that are true for all intervals between situations *now* and *then* where the situations *then* are W-accessible from s and have a K-accessible situation in their history, namely *now*. Intentions are future oriented, and any goal formula will be evaluated with respect to a finite path defined by a pair of situations, a begining situation *now* and an ending situation *then*. This formalization of goals can deal with both achievement goals and maintenance goals. An achievement goal ψ is said to be eventually satisfied if ψ holds in some situation s' over the interval between *now* and *then*.[3] Eventually$(\psi, now, then)$ is defined as $\exists s'.(now \leq s' \leq then \land \psi(s'))$. In [25], Shapiro showed that positive and negative introspection of intentions can be modeled by placing some constraints on K and W. To make sure that agents' wishes and intentions are consistent, W is also constrained to be serial.

The SSA for W which handles intention change in CASL, has the same structure as a SSA for a domain dependant fluent. In the following, $W^+(agt, a, s', s)$ ($W^-(agt, a, s', s)$, respectively) denotes the conditions under which s' is added to (dropped from, respectively) W as a result of the action a:

$$W(agt, s', do(a, s)) \equiv [W^+(agt, a, s', s) \lor (W(agt, s', s) \land \neg W^-(agt, a, s', s))].$$

An agent's intentions are expanded when it is requested something by another agent. After the *request(req,agt,ψ)* action, *agt* adopts the goal that ψ, unless she has a conflicting goal or is not willing to serve *req* for ψ. Therefore, this action

[3] Once again, *now* and *then* are not actual situations, but placeholders for situations that are bound in the definition.

should cause *agt* to drop any paths in W where ψ does not hold. This is handled in W^-:

$$\begin{aligned}
&W^-(agt, a, s', s) \doteq IncompRequest(agt, a, s', s),\\
&IncompRequest(agt, a, s', s) \doteq [\exists req, \psi.\ a = request(req, agt, \psi)\\
&\qquad\qquad \wedge Serves(agt, req, \psi) \wedge \neg \mathrm{Int}(agt, \neg\psi, s)\\
&\qquad\qquad \wedge \exists now.\ K(agt, now, s) \wedge now \leq s' \wedge \neg\psi(do(a, now), s')].
\end{aligned}$$

Here, the *request* action is considered a primitive action. The preconditions of request are:

$$Poss(request(req, agt, \phi), s) \equiv \mathrm{Int}(req, \phi, s).$$

A limited form of intention contraction is also handled in CASL. Suppose that the agent *req* requests *agt* that ψ and later decides it no longer wants this. The requester *req* can perform the action *cancelRequest(req,agt,ψ)*. This action causes *agt* to drop the goal that ψ. *cancelRequest* actions are handled by determining what the W relation would have been if the corresponding *request* action had never happened. This type of goal contraction is handled in W^+, which can be defined as follows:

$$\begin{aligned}
W^+(agt, a, s', s) \doteq\ & \exists s_1.\ W(agt, s', s_1) \wedge \exists a_1.\ do(a_1, s_1) \leq s \wedge Cancels(a, a_1)\\
& \wedge (\forall a^*, s^*.\ do(a_1, s_1) < do(a^*, s^*) \leq s \supset \neg W^-(agt, a^*, s', s^*)).
\end{aligned}$$

Suppose a *cancelRequest* action occurs in situation s. The W relation is first restored to the way it was before the corresponding *request* action occured, i.e., in s_1. Then starting just after the *request*, all the actions a^* that occured in the history of s (say in situation s^*) are considered, and any situation s' in W that satisfies $W^-(agt, a^*, s', s^*)$ is removed from W. $Cancels(a,\ a_1)$ can be defined as follows:

$$\begin{aligned}
&Cancels(a, a') \doteq\\
&\qquad [\exists req, \psi.\ a' = request(req, agt, \psi) \wedge a = cancelRequest(req, agt, \psi)].
\end{aligned}$$

A *cancelRequest* action can only be executed if a corresponding *request* action has occured in the past:

$$Poss(cancelRequest(req, agt, \phi), s) \equiv \exists s'.\ do(request(req, agt, \phi), s') \leq s.$$

3 Simple Cooperative Ability

An agent cannot be expected to eventually achieve an intention just because she has that intention, and she is acting rationally. We also need to make sure that the agent is capable of achieving the goal in the current situation [11]. In a single agent domain, an agent's ability can roughly be defined as her knowledge of a plan that is physically and epistimically executable and whose execution achieves the goal. However, modeling multiagent ability is a more complex problem, since in this case we need to consider the agents' knowledge about each other's knowledge and intentions as well as how they choose actions, behave rationally, etc. In this section, we develop a simple model of cooperative ability of agents suitable for a limited multiagent context in the absence of exogenous actions, i.e., actions whose performance is not intended by the planning agent. In an open multiagent framework, agents' actions may interfere with each other, possibly perturbing their plans. In some cases, there are multiple strategies to achieve a common goal, and the agents may fail unless they coordinate their choice of strategy by reasoning about each other's knowledge, ability, and rational choice. Moreover, agents may have conflicting goals or intentions. To simplify, we restrict our framework by only allowing plans where the actions that the other agents must do are fully specified, i.e., action delegation is possible, but (sub)goal delegation is not. The primary agent, who is doing the planning, is constrained to know the whole plan in advance. Thus, the primary agent is allowed to get help from others, but she can only ask other agents to perform specific actions. As a consequence, we do not need to model the fact that the other agents behave rationally.

When dealing with ability, it is not enough to say that the agent is able to achieve a goal iff she has a physically executable plan, and any execution of this plan starting in the current situation achieves the goal. We should also take into account the epistemic and intentional feasibility of the plan. This is necessary as physical executability does not guarantee that the executor will not get stuck in a situation where it knows that some transition can be performed, but does not know which. For example, consider the plan $(a; \text{If } \phi \text{ Then } b \text{ Else } c \text{ EndIf}) \mid d$, where actions a, b, c and d are always possible, but where the agent does not know whether ϕ holds after a. If the agent follows the branch where the first action is a, she will get stuck due to incomplete knowledge. Hence, the result of deliberation should be a kind of plan where the executor will know what to do next at every step, a plan that does not itself require deliberation to interpret. To deal with this, De Giacomo *et al.* [3] defined the notion of *Epistemically Feasible Deterministic Programs* (EFDPs) for single agent plans and characterized deliberation in terms of it. Note that EFDPs are deterministic, since they are the result of deliberation and their execution should not require making further choices or deliberation.

Since we are dealing with cooperative multiagent ability, we also need to make sure that the cooperating agents intend to perform the requested actions when it is their turn to act. In our framework, we extend the notion of EFDP to handle simple multiagent plans. A program is called an *Epistemically and Intentionally Feasible Deterministic Program* (EIFDP) in situation s for agent agt, if at each step of the program starting at s, agt always has enough infomation to execute the next action in the program, or knows that the executor of the next action is another agent, and that this agent has enough information to execute this action and intends to do it. Put formally:

$$EIFDP(agt, \delta, s) \doteq \forall \delta', s'.\, Trans^*(\delta, s, \delta', s') \supset LEIFDP(agt, \delta', s'),$$
$$LEIFDP(agt, \delta, s) \doteq \text{Know}(agt, Final(\delta, now), s) \vee$$

$$\begin{aligned}&\exists\delta'.\ \mathrm{Know}(agt, UTrans(\delta, now, \delta', now), s) \vee\\&\exists\delta', a.\ \mathrm{Know}(agt, Agent(a) = agt \wedge UTrans(\delta, now, \delta', do(a, now)), s) \vee\\&\exists\delta', agt'.\ \mathrm{Know}(agt, \exists a.\ UTrans(\delta, now, \delta', do(a, now))\\&\qquad\qquad\qquad\wedge Agent(a) = agt' \neq agt\\&\qquad\qquad\qquad\wedge \mathrm{Int}(agt', \exists s'.\ s' \leq then \wedge Do(a, now, s'), now), s).\end{aligned}$$

Thus to be an EIFDP, a program must be such that all configurations reachable from the initial program and situation, involve a *Locally Epistimically and Intentionally Feasible Deterministic Program* (LEIFDP). A program is called LEIFDP in a situation with respect to an agent, if the agent believes that the program is currently in its *Final* configuration, or believes that she is the agent of the next action and knows what unique transition (with or without an action) it can perform next, or believes that someone else agt' is the agent of the next action, that agt' knows what the action is and intends to do it next, and knows what unique transition the program can perform next with this action. Here, $UTrans(\delta, s, \delta', s)$ means that the program δ in s can perform a unique transition, which takes the agent to s' with the remaining program δ'.

Using EIFDP, the ability of an agent can be defined as follows:[4]

$$\begin{aligned}Can(agt, \psi(now, then), s) \doteq \exists\delta.\ &\mathrm{Know}(agt, EIFDP(agt, \delta, now)\\&\wedge \exists s'.\ Do(\delta, now, s') \wedge \forall s'.\ (Do(\delta, now, s') \supset \psi(now, s')), s).\end{aligned}$$

Thus, an agent can achieve a goal in situation s, iff she knows of a plan δ that is an EIFDP, is executable starting at s, and any possible execution of the plan starting in the current situation brings about the goal.

We use the following as our running example (adapted from [17]) throughout the paper. Consider a world in which there is a safe with a combination lock. If the safe is locked and the correct combination is dialed, then the safe becomes unlocked. However, dialing the incorrect combination will cause the safe to explode. The agent can only dial a combination if the safe is intact, and it is not possible to change the combination of the safe. Initially, the agent Agt_1 has the intention to open the safe, but does not know the combination. However, she knows that Agt_2 knows it. She also knows that Agt_2 is willing to serve her, and that Agt_2 does not have the intention of not informing her of the combination of the safe. Here are some of the axioms that we use to model this domain:

$$\begin{aligned}&sf_1)\ Poss(a, s) \supset [Exploded(do(a, s)) \equiv\\&\qquad\exists c, agt.\ (a = dial(agt, c) \wedge Comb(s) \neq c) \vee Exploded(s)].\\&sf_2)\ Poss(dial(agt, c), s) \equiv \neg Exploded(s).\\&sf_3)\ Agent(dial(agt, c)) = agt.\\&sf_4)\ \neg Exploded(S_0).\\&sf_5)\ W(Agt_1, s, S_0) \equiv \neg Locked(s).\end{aligned}$$

[4] Note that this definition of *Can* handles non-achievement goals, as there are two situation placeholdes in ψ, i.e., *now* and *then*. However, an achievement goal $\psi(now)$ can be placed inside an Eventually block to provide both *now* and *then*.

The first axiom, a successor state axiom, states that the safe has exploded after doing action a iff a denotes the action of dialing the wrong combination, or if the safe has already exploded. The second axiom, a precondition axiom, states that it is possible to dial a combination for the safe in situation s iff the safe is intact in s. The third axiom is an agent axiom and defines the agent of the *dial* action. The last two axioms are initial situation axioms, and state that the safe is initially intact, and that Agt_1 initially only intends to open the safe, respectively. From now on, we will use D_{safe} to denote the set of axioms that we use to model this safe domain (see [8] for the complete axiomatization).

Now, consider the follwing plan:[5]

$$\begin{aligned}\sigma_{safe} = &\, requestAct(Agt_1, Agt_2, informRef(Agt_2, Agt_1, Comb(s)));\\ &\, informRef(Agt_2, Agt_1, Comb(s)); dial(Agt_1, Comb(s)).\end{aligned}$$

So, the plan is that Agt_1 will request Agt_2 to inform her of the combination of the safe, Agt_2 will inform Agt_1 of the combination of the safe, and finally, Agt_1 will dial the combination to open the safe. We claim that σ_{safe} is an EIFDP in the initial situation for Agt_1, and that Agt_1 is able to achieve her intention of opening the safe in the initial situation:

Theorem 1.

$$\begin{aligned}&a.\ D_{safe} \models EIFDP(Agt_1, \sigma_{safe}, S_0).\\ &b.\ D_{safe} \models Can(Agt_1, \neg Locked(now), S_0).\end{aligned}$$

(a) holds as all configurations reached by σ_{safe} starting in S_0 are LEIFDP. (b) holds as Agt_1 knows of a plan (i.e., σ_{safe}), which she knows is an EIFDP and is executable, and knows that any execution of this plan ends up in a situation where the safe is unlocked.

4 From Intentions That to Intentions to Act

In this section, we define rational plans and extend CASL to model the role of intention and rationality in determining an agent's actions. This bridges the gap between future directed intentions and present directed ones. We also discuss a notion of subjective plan execution and present a theorem that relates intention and ability to the eventual achievement of intended goals.

Before going further, let us discuss the communication actions that we will use in our framework. Like in CASL, we use three primitive informative communication actions, namely, *inform*, *informWhether*, and *informRef*. However, unlike in CASL, we provide two intention transfer communication actions, *request* and *requestAct*, and these are defined in terms of *inform*.[6] The *request*

[5] *requestAct* is an abbreviation introduced in the next section; it denotes a special kind of request, namely, the request to perform an action.

[6] A similar account of request was presented by Herzig and Longin [5], where it is defined as inform about intentions, and the requested goals are adopted via cooperation principles.

action can be used by an agent to request another agent to achieve some state of affairs, whereas *requestAct* involves an agent's request to another agent to perform some particular complex action starting in the next situation. Formally,

$$\begin{aligned}&\mathit{request}(\mathit{req}, \mathit{agt}, \phi) \doteq \mathit{inform}(\mathit{req}, \mathit{agt}, \mathrm{Int}(\mathit{req}, \phi, \mathit{now})).\\ &\mathit{requestAct}(\mathit{req}, \mathit{agt}, \delta) \doteq \mathit{request}(\mathit{req}, \mathit{agt}, \exists s', a.\ \mathit{Do}(\delta, \mathit{do}(a, \mathit{now}), s')\\ &\qquad\qquad\qquad\qquad\qquad \wedge \mathit{now} < s' \leq \mathit{then} \wedge \mathit{Agent}(\delta) = \mathit{agt}).\end{aligned}$$

Here *Agent(δ)=agt* means that the agent of all actions in δ is *agt*. In our specification, we only allow sincere requests. That is, an agent can perform a request if the request is not contradictory to her current intentions. So defining requests as informing of intentions is reasonable. However, since requests are modeled in terms of *inform*, and since we are using true belief, the usual preconditions of the *inform* action are relaxed:

$$\begin{aligned}&\mathit{Poss}(\mathit{inform}(\mathit{inf}, \mathit{agt}, \phi), s) \equiv\\ &\qquad \mathrm{Know}(\mathit{inf}, \phi', s) \wedge \neg\mathrm{Know}(\mathit{inf}, \mathrm{KWhether}(\mathit{agt}, \phi', \mathit{now}), s),\end{aligned}$$

where, ϕ' is ϕ with all $\mathrm{Int}(\mathit{inf}, \phi'', \mathit{now})$ are replaced by $\neg\mathrm{Int}(\mathit{inf}, \neg\phi'', \mathit{now})$. This axiom says that the agent *inf* can inform *agt* that ϕ, iff *inf* knows that ϕ' currently holds, and does not believe that *agt* currently knows the truth value of ϕ', where ϕ' is defined as above. Note that, if we use ϕ instead of ϕ' in the above axiom, the account would be overly strict. For instance, in the safe domain, σ_{safe} is a rational plan for Agt_1 in the initial situation. However, initially, Agt_1 does not have the intention that Agt_2 informs her the combination of the safe. So if we use ϕ instead of ϕ' in the axiom, we can not show that σ_{safe} is rational, since it requires Agt_1 to know that she has the intention before she can inform about it. So we relax the requirements so that the agent only needs to know that she does not have the opposite intention.

To facilitate the cancellation of requests, we also provide two actions, namely, *cancelRequest*, and *cancelReqAct*. Unlike CASL where *cancelRequest* is primitive, we define it using *inform*. These two actions are defined as follows:

$$\begin{aligned}&\mathit{cancelRequest}(\mathit{req}, \mathit{agt}, \psi) \doteq \mathit{inform}(\mathit{req}, \mathit{agt}, \neg\mathrm{Int}(\mathit{req}, \psi, \mathit{now})),\\ &\mathit{cancelReqAct}(\mathit{req}, \mathit{agt}, \delta) \doteq\\ &\quad \mathit{cancelRequest}(\mathit{req}, \mathit{agt}, \exists s^*, s^+, \mathit{prev}.\ \mathit{prev} = \mathit{do}(\mathit{requestAct}(\mathit{req}, \mathit{agt}, \delta), s^+)\\ &\qquad\qquad \wedge s^+ < \mathit{now} \leq s^* \leq \mathit{then} \wedge \mathit{Do}(\delta, \mathit{prev}, s^*) \wedge \mathit{Agent}(\delta) = \mathit{agt}).\end{aligned}$$

Now let us look at what plans are rational for an agent. To keep the theory simple, we only consider conditional plans. An agent that is acting rationally, should prefer some plans to others. To this end, we define an ordering on plans.

$$\begin{aligned}&\succeq (\mathit{agt}, \delta_1, \delta_2, s) \doteq \forall s'.\ K(\mathit{agt}, s', s) \wedge \exists s''.\ \mathit{Do}(\delta_2, s', s'') \wedge W(\mathit{agt}, s'', s)\\ &\qquad\qquad \supset [\exists s''.\ \mathit{Do}(\delta_1, s', s'') \wedge W(\mathit{agt}, s'', s)].\end{aligned}$$

That is, a plan δ_1 is as good as another plan δ_2 in situation s for an agent *agt* iff for all W-accessible situations that can be reached by following δ_2 from a situation that is K-accessible from s (say s'), there exists a W-accessible situation that can be reached from s' by following δ_1. In other words, δ_1 is at least as good as δ_2 if it achieves the agent's goals in all the possible situations where δ_2 does.

Using EIFDP and the $\succeq$ relation, we next define rational plans. A plan δ is said to be *rational* in situation s for an agent *agt* if the following holds:

$$\begin{aligned} \mathit{Rational}(agt, \delta, s) \doteq \forall \delta'.\ &\succeq (agt, \delta', \delta, s) \supset \succeq (agt, \delta, \delta', s) \\ &\wedge \mathit{EIFDP}(agt, \delta, s) \\ &\wedge \forall \delta''.\ (\mathit{SubPlan}(\delta'', \delta) \wedge \delta'' \neq \delta \supset \neg \mathit{Rational}(agt, \delta'', s)). \end{aligned}$$

Thus, a rational plan in a situation s, is a plan that is as good as any other plan in s, is an EIFDP in s, and is *minimal*, i.e., no sub-plan of the plan is rational. The latter guarantees that no unnecessary actions are included in the plan (see [8] for the definition of $\mathit{SubPlan}$).

For example, consider the plan σ_{safe}. We claim that σ_{safe} is as good as any other plan available to Agt_1 in the initial situation, and that σ_{safe} is rational in the initial situation.

Theorem 2.

$$\begin{aligned} &a.\ D_{safe} \models \forall \sigma.\ \succeq (Agt_1, \sigma_{safe}, \sigma, S_0). \\ &b.\ D_{safe} \models \mathit{Rational}(Agt_1, \sigma_{safe}, S_0). \end{aligned}$$

Since this plan achieves Agt_1's intention of opening the safe starting in any situation that is K-accessible to S_0, (a) holds. (b) follows from the fact that σ_{safe} is as good as any other plan in S_0, is an EIFDP in S_0, and no subplan of σ_{safe} is rational in S_0.

In most cases, there are many rational plans (i.e., ways of achieving as many goals as possible). The decision of which plan the agent commits to is made based on pragmatic/non-logical grounds. We do not model this here. Instead, we introduce a $commit(agt, \delta)$ action that will model the agent's commiting to a particular plan δ, more specifically, commiting to executing δ next. The action precondition axiom for the *commit* action is as follows:

$$\mathit{Poss}(commit(agt, \delta), s) \equiv \neg \mathrm{Int}(agt, \neg \exists s^*.\ s \leq s^* \leq then \wedge Do(\delta, now, s^*), s).$$

That is, the agent *agt* can commit to a plan δ is situation s, iff the agent currently does not have the intention that the actions in the plan do not happen next.

Next, we extend the SSA for W to handle intention revision as a result of the agent's commitment to a rational plan, and also as a result of other agents' *requestAct* and *cancelReqAct* actions. This axiom has a similar structure to that of CASL; however, we modify W^- as follows:

$$W^-(agt, a, s', s) \doteq \mathit{IncompRequest}(agt, a, s', s) \vee\ \mathit{IncompCommit}(agt, a, s', s).$$

Here, *IncompCommit* handles the expansion of the agent's intentions that occur when a *commit* action occurs. We define *IncompCommit* as follows:

$$\begin{aligned}
IncompCommit(agt, a, s', s) \doteq {} & [\exists \delta .\ a = commit(agt, \delta) \wedge Rational(agt, \delta, s) \\
& \wedge \exists s^*.\ s^* \leq s' \wedge K(agt, s^*, s) \\
& \wedge \neg \exists s^{**}.\ (s^* < s^{**} \leq s' \wedge Do(\delta, do(a, s^*), s^{**}))].
\end{aligned}$$

So, after the performance of a *commit* action in s, a W-accessible situation s' in s will be dropped from *agt*'s new set of W-accessible situations if the plan to which *agt* is commiting is rational, and the committed action does not happen between the interval defined by the pair of situations, a K-accessible situation accessible from the current situation, and the W-accessible situation s'.

The definition of W^+ remains unchanged. Note that if exogenous actions are allowed, agents need to revise their commitments when an exogenous action occurs by uncommiting from the currently committed plan, and committing to a new rational plan. We return to this issue in Section 5.

We now show that our formalization of intentions has some desirable properties:

Theorem 3.

a. $\models \neg \mathrm{Int}(agt, \neg\phi, s) \wedge Serves(agt, req, \phi) \supset$
$\mathrm{Int}(agt, \phi, do(request(req, agt, \phi), s))$.

b. $\models \neg \mathrm{Int}(agt, \neg \exists s'.\ Do(\delta, now, s') \wedge now \leq s' \leq then, s) \supset$
$\mathrm{Int}(agt, \exists s'.\ Do(\delta, now, s') \wedge now \leq s' \leq then, do(commit(agt, \delta), s))$.

(a) says that if an agent *agt* does not have the intention that not ϕ in s, then she will have the intention that ϕ in the situation resulting from another agent *req*'s request to *agt* that ϕ in s, provided that she is willing to serve *req* on ϕ . (b) states that if an agent *agt* does not have the intention of not performing a complex action δ in s, then she will have the intention of performing it after she commits to it.

Commit provides a way to link future directed intentions and present directed ones. We next specify a generic meta-controller for an agent that arbitrarily chooses a rational plan, commits to it, and executes it. Then we can prove a theorem about the relationship between intention, ability, and the eventual achievement of an intended goal. This theorem serves as a proof of soundness of our agent theory.

The following meta-controller allows us to refer to the future histories of actions that may occur for an agent who is behaving rationally until ψ holds. Rational behavior until ψ can be defined as follows (we assume that there are no exogenous actions):

```
$BehaveRationallyUntil(agt, \psi(now), s) \doteq$
  $\pi \delta .\ Rational(agt, \delta, now)?; commit(agt, \delta);$
  While $\neg \psi(now)$ Do
```

```
        If $\exists a.\ \mathrm{Int}(agt, do(a, now) \leq then, now) \land Agent(a) = agt)$ Then
            $[\pi a.\ (\mathrm{Int}(agt, do(a, now) \leq then, now) \land Agent(a) = agt)?; a]$
        Else
            $[\exists a.\ [\mathrm{Int}(agt, do(a, now) \leq then, now) \land Agent(a) \neq agt = agt']?;$
            $(\pi a'.\ \mathrm{Int}(agt', do(a', now) \leq then, now)?; a')]$
        EndIf
    EndWhile.
```

That is, rational behavior until ψ can be defined as arbritarily choosing a rational plan, committing to it, and then executing it as long as ψ does not hold. A rational plan can have actions by the planning agent as well as actions by other agents. When it is the planning agent's turn to act, she should perform the action she intends to perform next; otherwise, she should wait for the other agent to act. Note that, when it is the other agent's turn, it will always perform the action that it is supposed to perform because rational plans are EIFDP. Also note that we only deal with achievement goals here.

One problem with CASL is that the execution of plans is viewed from the system's perspective rather than from the agent's perspective. So, although CASL includes operators that model agents' knowledge and goals, the system behavior is simply specified as a set of concurrent processes. These processes may refere to the agents' mental states, but they don't have to. To deal with this problem, Lespérance [12] proposed an account of subjective plan execution in CASL that ensures that the plan can be executed by the agent based on its knowledge state. Here we extend this to deal with multiagent plans and to consider other agents' intentions. We define the subjective execution construct $\mathrm{Subj}(agt, \delta)$ as follows:

$$\begin{aligned}
&Trans(\mathrm{Subj}(agt,\delta), s, \gamma, s') \equiv \exists \delta'.\ (\gamma = \mathrm{Subj}(agt, \delta') \land \\
&\quad [\mathrm{Know}(agt, Trans(\delta, now, \delta', now), s) \land s = s' \lor \\
&\quad \exists a.\ (\mathrm{Know}(agt, Trans(\delta, now, \delta', do(a, now)) \land Agent(a) = agt, s) \\
&\qquad \land s' = do(a, s)) \lor \\
&\quad \exists agt'.\ (\mathrm{Know}(agt, \exists a.\ Trans(\delta, now, \delta', do(a, now)) \land Agent(a) = agt' \\
&\qquad\qquad\qquad \land \mathrm{Int}(agt', \exists s^*.\ s^* \leq then \land Do(a, now, s^*), now), s) \\
&\qquad \land s' = do(a, s))]), \\
&Final(\mathrm{Subj}(agt, \delta), s) \equiv \mathrm{Know}(agt, Final(\delta, now), s).
\end{aligned}$$

This means that when a program is executed subjectively by an agent *agt*, the system can make a transition only if *agt* knows that it can make this transition, and if the transition involves a primitive action by another agent, then the transition is possible provided that *agt* also knows that the other agent will intend to perform the action. A subjective execution may legally terminate only if the agent knows that it may. Next, we present our "success theorem":[7]

[7] The construct *AllDo* is a strict version of *Do* that requires that all possible executions of a program terminate successfully.

Theorem 4 (From Commitment and Ability to Eventuality).

$$\begin{aligned}\models\ & [\mathrm{OInt}(agt, \mathrm{Eventually}(\gamma, now, then), s) \\ & \wedge Can(agt, \mathrm{Eventually}(\gamma, now, then), s) \\ & \wedge \mathrm{Int}(agt, \mathrm{Eventually}(\psi, now, then), s)] \supset \\ & \qquad AllDo(\mathrm{Subj}(agt, BehaveRationallyUntil(agt, \psi, s)), s).\end{aligned}$$

Intuitively, if in some situation, an agent intends to achieve some goal and is able to achieve it, then the agent will eventually achieve the goal in all rational future histories subjectively executed from that situation. $\mathrm{OInt}(agt, \psi, s)$ means that ψ is all the intentions that *agt* has in s. This construct is useful as we have to assume that the agent is able to achieve all her intentions. If this is not the case, the *BehaveRationallyUntil* operator does not guarantee that a specific goal (i.e., ψ) will be achieved. If there are exogenous actions, then a more generic meta-controller can be defined. We discuss this in the next section.

We also have the following corrolary for the safe domain:

Theorem 5.

$$D_{safe} \models AllDo(\mathrm{Subj}(Agt_1, BehaveRationallyUntil(Agt_1, \neg Locked, S_0)), S_0).$$

We have shown in Theorem 1(b) that Agt_1 can achieve her intention of opening the safe in the initial situation. Moreover, by sf_5, the only intention of Agt_1 is to open the safe. It follows from Theorem 4 that Agt_1 will eventually open the safe if she behaves rationally starting in S_0.

5 Discussion and Future Work

In this paper, we presented a formal theory of agency that deals with simple multiagent cooperation and shows how future directed intentions and present directed ones can be related. An agent's current rational plans depend on her current intentions. The *commit* action models how the agent's intentions can be updated to include a commitment to a rational plan. Using this, we have formulated a planning framework for multiple cooperating and communicating agents in CASL. We specified how an agent's future directed intentions will lead the agent to adopt a rational plan and then carry it out using the meta-controller *BehaveRationallyUntil*.

To relate agents' intentions with their actions, Cohen and Levesque [1, 2] required that the agents not procrastinate with respect to their intentions (AKA the *no infinite deferral* assumption). However, this assumption is unintuitive as it should follow from other axioms of the theory, rather than be imposed separatly. A similar account was presented by Rao and Georgeff [19]. A more intuitive account was presented in [28], where Singh showed that rather than having it as an assumption, the no infinite deferral principle can be derived from the theory. However, he does not explicitly address the interaction between knowledge and actions and its relationship with ability. Another account was presented by Sadek [22], where he incorporated a backward chaining planning mechanism in his framework. However, his account is limited in the sense that it uses hardcoded perlocutionary or rational effects of actions rather than actual effects. Although independently motivated, our account closely resembles the one in [15], where a similar notion of commitment to actions was introduced to relate intentions and actions. However, that framework does not model rationality and provide a success theorem. There has also been related works that extend agent programming languages to support declarative goals [7, 23].

The theory presented here is a part of our ongoing research on the semantics of speech acts and communication in the situation calculus. In [8], we present an extended version of our framework where we allow exogenous actions. To deal with these unintended actions, an agent needs to revise the plan it is committed to whenever an exogenous action occurs. In other words, she needs to un-commit from the previously committed plan, consider the new set of rational plans, and commit to one of them. We handle the un-commiting part in the SSA for W. The agents' commitment to a new rational plan is handled using a more general meta-controller. This controller iterates the *BehaveRationallyUntil* program as long as the goal remains un-achieved and there is a plan that is rational in the current situation. In [8], we also define a notion of conditional commitment, and model some simple communication protocols using it.

Our current agent theory is overly simplistic in many ways. One strict constraint that we have is that we do not allow cooperating agents to choose how they will achieve the goals delegated to them by assuming that the planning agent knows the whole plan in advance. Only one agent is assumed to do planning. In future work, we will try to relax this restriction and to model some interaction protocols that involve multiple planning agents.

References

1. Cohen, P., Levesque, H.: Intention is Choice with Commitment. In: Artificial Intelligence, 42:(2-3), (1990) 213–361
2. Cohen, P., Levesque, H.: Rational Interaction as the Basis for Communication. In: Cohen P., Morgan, J., Pollack, M. (eds.): Intentions in Communication. Cambridge, MA, MIT Press (1990) 221–255
3. De Giacomo, G., Lespérance, Y., Levesque, H., Sardina, S.: On the Semantics of Deliberation in IndiGolog – from Theory to Implementation. In: Fensel, D., Giunchiglia, F., McGuiness, D., Williams, M. (Eds.): Principles of Knowledge Representation and Reasoning KR 02 (2002) 603–614
4. De Giacomo, G., Lespérance, Y., Levesque, H.: ConGolog, a Concurrent Programming Language Based on the Situation Calculus. Artificial Intelligence, 121 (2000)
5. Herzig, A., Longin, D.: A Logic of Intention with Cooperation Principles and with Assertive Speech Acts as Communication Primitives. In: Proc. of AAMAS 02 (2002)
6. Hoare, C.: Communicating Sequential Processes. Prentice Hall Int. (1985)
7. van der Hoek, W., Hindriks, K., de Boer, F., Meyer, J.-J.Ch.: Agent Programming with Declarative Goals. In: Castelfranchi, C., Lespérance, Y. (eds.): Intelligent Agents VII, Proc. of ATAL 00, LNAI 1986 (2000)
8. Khan, S.: M.Sc Thesis. In preparation (2004)

9. Lakemeyer, G., Levesque, H.: AOL: A Logic of Acting, Sensing, Knowing, and Only-Knowing. In: Proc. of KR 98 (1998) 316–327
10. Lespérance, Y., Levesque, H. J., Lin, F., Marcu, D., Reiter, R., Scherl, R.: Foundations of a Logical Approach to Agent Programming. In: Wooldridge, M., Muller, J., Tambe, M. (eds.): Intelligent Agents Vol. II - Proc. of ATAL 95 (1996) 331–346
11. Lespérance, Y., Levesque, H., Lin, F., Scherl, R.: Ability and Knowing How in the Situation Calculus. Studia Logica 66(1) (2000) 165–186
12. Lespérance, Y.: On the Epistemic Feasibility of Plans in Multiagent Systems Specifications. In: Meyer, J.-J.Ch., Tambe M. (Eds.): Intelligent Agents VIII, Proc. of ATAL 01, Seattle, WA, USA (2001)
13. Levesque, H.: What is planning in the presence of sensing? In: Proc. of the Thirteenth National Conference on Artificial Intelligence, Portland, OR (1996) 1139–1146
14. Levesque, H., Reiter, R., Lespérance, Y., Lin, F., Scherl, R.: GOLOG: A Logic Programming Language for Dynamic Domains. J. of Logic Programming, 31 (1997)
15. van Linder, B., van der Hoek, W., Meyer, J.-J. Ch.: Formalising Motivational Attitudes of Agents : On Preferences, Goals, and Commitments. In: Wooldridge, M., Muller, J., Tambe, M. (eds.): Intelligent agents II LNAI 1037 (1996) 17–32
16. McCarthy, J., Hayes, P.: Some Philosophical Problems from the Standpoint of Artificial Intelligence. Machine Intelligence, 4 (1969) 463–502
17. Moore, R.: A Formal Theory of Knowledge and Action. In: Hobbs J., Moore, R. (eds.): Formal Theories of the Commonsense World. Ablex (1985) 319–358
18. Plotkin, G.: A Structural Approach to Operational Semantics. Technical Report DAIMI-FN-19, Computer Science Dept., Aarhus University, Denmark (1981)
19. Rao, A., Georgeff, M.: Modeling Rational Agents within a BDI-architecture. In: Fikes, R., Sandewall, E. (eds.): Proc. of KR&R 91 (1991) 473–484
20. Reiter, R.: 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: Papers in the Honor of John McCarthy. San Diego, CA, Academic Press (1991)
21. Reiter, R.: Knowledge in Action. Logical Foundations for Specifying and Implementing Dynamical Systems. MIT Press (2001)
22. Sadek, M.: Communication Theory = Rationality Principles + Communicative Act Models. In: Proc. of AAAI 94 Workshop on Planning for Interagent Comm. (1994)
23. Sardiña, S., Shapiro, S.: Rational Action in Agent Programs with Prioritized Goals. In: Proc. of AAMAS 03 (2003) 417–424
24. Scherl, R., Levesque, H.: The Frame Problem and Knowledge-Producing Actions. In: Proc. of the Eleventh National Conference on Artificial Intelligence. Washington, DC, AAAI Press/The MIT Press (1993) 689–695
25. Shapiro, S.: PhD Thesis. In preparation (2004)
26. Shapiro, S., Lespérance, Y.: Modeling Multiagent Systems with the Cognitive Agents Specification Language - A Feature Interaction Resolution Application. In: Castelfranchi, C., Lespérance, Y. (eds.): Intelligent Agents Vol. VII - Proc. of ATAL 00, LNAI 1986 (2001) 244–259
27. Shapiro, S., Lespérance, Y., Levesque, H.: Specifying Communicative Multi-Agent Systems with ConGolog. In: AAAI Fall 1997 Symp. on Comm. Act. in Humans and Machines (1997) 75–82
28. Singh, M.: Multiagent Systems: A Theoretical Framework for Intentions, Know-How, and Communications. LNAI 799 (1994)