Proceedings, Fifteenth International Conference on
Principles of Knowledge Representation and Reasoning (KR 2016)

Infinite Paths in the Situation Calculus: Axiomatization and Properties

Shakil M. Khan and **Yves Lespérance**
Department of Electrical Engineering and Computer Science
York University, Toronto, Ontario, Canada
Email: {skhan, lesperan}@eecs.yorku.ca

Abstract

The situation calculus has proved to be a very popular formalism for modeling and reasoning about dynamic systems. This otherwise elegant and refined language however lacks a natural way of dealing with "infinite future histories". To this end, in this paper we introduce a new sort ranging over *infinite paths* in the situation calculus and propose an axiomatization for infinite paths. We thus obtain a convenient way of specifying several kinds of notions that involve infinite futures such as temporal properties of non-terminating executions of agents or programs and mental attitudes such as desires and intentions. We prove the correctness of the axiomatization and show that our formalization has some intuitively desirable properties.

Introduction

One of the most prominent formalisms for modeling dynamic domains is the situation calculus (McCarthy and Hayes 1969). Since its introduction, much work has been done to further refine and enrich this language (Reiter 2001). However, this otherwise simple yet elegant language lacks a convenient way of dealing with "infinite future histories". One cannot talk directly about such infinite paths in the situation calculus. While some work has been done to capture the notion of path in the situation calculus, all of these approaches have drawbacks. Many of these deal exclusively with finite paths. For example, while specifying agents' goals and behavior, Shapiro (2005) considers only finite paths. He models a finite path using a pair of situations representing the beginning state and the ending state of the path. Unfortunately, a temporal framework based on such finite paths has limited expressiveness and can't capture arbitrary temporally extended formulae, e.g. the goal to maintain a property ϕ indefinitely far in the future, $\Box\phi$. Also, quantification over these finite paths requires dealing with a pair of situations explicitly which is somewhat clumsy.

Others have looked at infinite paths within the situation calculus, for instance, Lespérance et al. (2000). They introduced the notion of *action selection functions* (also called *ASF* or *strategies*), which are mappings from situations to primitive actions, and showed how ASFs can be used to model infinite paths (see below for details). Their account however does not have paths as a sort and thus does not allow for first-order quantification over paths.

Finally, a third set of approaches utilize other logics along with the situation calculus to express properties of infinite sequences involving situations. For example, (Claßen and Lakemeyer 2008) develops a second-order modal logic inspired by CTL* and dynamic logic that can express properties about executions of (possibly) non-terminating ConGolog programs (De Giacomo, Lespérance, and Levesque 2000). However, the authors derive the semantics of programs and temporal operators from the model theoretic semantics of their logic, rather than using axioms as in the standard situation calculus. Thus, it is not clear if all the desirable features of the situation calculus are inherited in their framework. We discuss other related work in an extended version of this paper (Khan and Lespérance 2015).

To deal with these issues, in this paper we introduce a new sort of infinite *paths* (along with path variables that can be quantified over) in the situation calculus and propose an axiomatization for infinite paths. By adding a new sort ranging over infinite paths in the situation tree, we obtain a convenient way of specifying several kinds of notions that involve infinite futures such as temporal properties of non-terminating executions of agents or programs, and future looking mental attitudes such as desires and intentions, as well as beliefs about the future. We show that our formalization of paths has some intuitively reasonable properties and prove the correctness of the axiomatization.

Background

We adopt the version of the situation calculus as formalized in (Levesque, Pirri, and Reiter 1998; Reiter 2001). We will not go over it here except to note the following components: $\text{Init}(s)$ denotes that s is one of the initial situations, i.e. a situation where no actions have yet occurred; the special constant S_0 is taken to denote the actual initial situation; there is a distinguished binary function symbol do where $do(a, s)$ denotes the successor situation to s resulting from performing the action a; fluents are denoted by predicate and function symbols taking a situation term as their last argument; there is a special predicate $\text{Poss}(a, s)$ used to state that action a is executable in situation s; $s \sqsubset s'$ means that s' can be reached from s by performing a sequence of actions; $s \sqsubseteq s'$ is an abbreviation for $s \sqsubset s' \vee s = s'$. We will use $s \prec s'$

and $s \preceq s'$ to denote that the sequence of actions performed to reach s' from s were all executable. Finally, a situation is called executable if every action in its history was executable, i.e.:

$$\text{Executable}(s) \doteq \forall a, s'.\ do(a, s') \preceq s \supset \text{Poss}(a, s').$$

A dynamic domain can be represented by a *basic action theory* (Reiter 2001) $\mathcal{D}_{bat}$ that includes: (1) action precondition axioms, one per action a characterizing $\text{Poss}(a, s)$, (2) successor state axioms, one per fluent, that succinctly encode both effect and frame axioms and specify exactly how and when the fluent changes (Reiter 2001), (3) initial state axioms describing what is true initially, (4) unique names axioms for actions, and (5) domain-independent foundational axioms Σ describing the structure of situations (Shapiro 2005). All these axioms are first-order except for Σ, which includes a second-order induction axiom for defining the trees of situations.[1]

Infinite Paths in the Situation Calculus

Following (Lespérance et al. 2000), we only consider "realistic" paths; paths involving non-executable actions cannot really occur as they are not realistic. Thus a path in our framework is essentially an infinite sequence of situations, where each situation along the path can be reached by performing some *executable* action in the preceding situation. To allow (first-order) quantification over infinite paths, we in addition introduce a new sort called *paths* in the language with (possibly sub/super-scripted) variables p ranging over paths. We give an axiomatization for infinite paths below.

Thus our formalization of infinite paths is more general than Shapiro's (2005) finite paths. Arbitrary temporally extended formulae such as unbounded maintenance goals can be interpreted using our paths. Moreover, our account is simpler than that of (Lespérance et al. 2000), and unlike them, we allow quantification over paths, which makes our language easier to use.

Before delving into the technical details, let us point out some notational conventions. We will use both state and path formulae denoted by uppercase and lowercase Greek letters, resp., e.g. $\Phi(s)$ and $\phi(p)$. Here s is a free situation variable in which the state formula must be evaluated and p is a free path variable over which the path formula must hold. We sometimes suppress these variables where the intended meaning is clear.

Axiomatization: We now give our axiomatization for infinite paths. We have a predicate $\text{OnPath}(p, s)$, meaning that situation s is on path p. Also, the abbreviation $\text{Starts}(p, s)$ means that s is the starting situation of path p. A path p starts with s iff s is the earliest situation on p:

$$\text{Starts}(p, s) \doteq \text{OnPath}(p, s) \wedge \forall s'.\ \text{OnPath}(p, s') \supset s \preceq s'. \quad \textbf{(D1)}$$

[1] We allow multiple initial situations to support the modeling of mental states and use the foundational axioms Σ given by Shapiro (2005), which are proven to be equivalent to the ones given by (Levesque, Pirri, and Reiter 1998). Nevertheless, since the set of foundational axioms given by (Reiter 2001) is a special case of our Σ (where the only initial situation happens to be S_0), all our results are also entailed by Reiter's Σ.

As shown in (Lespérance et al. 2000), one can use action selection functions (ASFs) to model infinite paths. Recall that ASFs or strategies are mappings from situations to primitive actions. The idea is that given a situation s, an ASF F prescribes an action that the agent must perform in s if she were to follow the path induced by this strategy. An infinite path can then be formalized as a tuple (s, F), where s is the starting situation of the path, and F is a strategy that defines an infinite sequence of situations by specifying an action for every situation starting from s. Thus, one way of axiomatizing paths is by making them correspond to such pairs (s, F):

$$\begin{aligned}(a).\ &\forall p.\ (\exists F, s.\ \text{Executable}(F, s)\\ &\qquad \wedge \forall s'.\ \text{OnPath}(p, s') \equiv \text{OnPathASF}(F, s, s')),\\ (b).\ &\forall F, s.\ \text{Executable}(F, s) \supset (\exists p.\ \text{Starts}(p, s)\\ &\qquad \wedge \forall s'.\ \text{OnPathASF}(F, s, s') \equiv \text{OnPath}(p, s')).\end{aligned} \quad \textbf{(A1)}$$

This second-order axiom says that for every path p, there is an action selection function F and a situation s such that F starting in s is executable, and that F produces exactly the same sequence of situations on p starting from s. Also, for every executable action selection function F and situation s, there is a path p that starts with s and that corresponds exactly to the sequence of situations produced by F starting from s. Here, $\text{OnPathASF}(F, s, s')$ means that the situation sequence defined by (s, F) includes the situation s':

$$\begin{aligned}&\text{OnPathASF}(F, s, s') \doteq\\ &\qquad s \preceq s' \wedge \forall a, s^*.\ s \prec do(a, s^*) \preceq s' \supset F(s^*) = a.\end{aligned} \quad \textbf{(D2)}$$

Also, the situation sequence encoded by a strategy F and a starting situation s is executable iff s is executable, and for all situations s' on this sequence, the action selected by F in s' is executable in s'.

$$\begin{aligned}&\text{Executable}(F, s) \doteq \text{Executable}(s) \wedge\\ &\qquad \forall s'.\ \text{OnPathASF}(F, s, s') \supset \text{Poss}(F(s'), s').\end{aligned} \quad \textbf{(D3)}$$

Another axiom is needed to state that different situation sequences represent different paths.

$$\forall p, p'.\ (\forall s.\ \text{OnPath}(p, s) \equiv \text{OnPath}(p', s)) \equiv p = p'. \quad \textbf{(A2)}$$

Note that, for every situation s on a path, there must be an action that is possible in s, i.e. $\forall p, s.\ \text{OnPath}(p, s) \supset \exists a.\ \text{Poss}(a, s)$. We consider that situations where no action is possible are "artificial". One can always introduce a dummy action $noOp$ that has the precondition that True, and consequently is always executable. Taking paths to be sequences of executable situations means that there may be infinite sequences of successor situations that are not paths; even if the situations on a prefix of a sequence are executable, the presence of a non-executable situation in the sequence means that it is not a path. One could easily modify the above axiomatization to include paths with non-executable situations, and identify the subset of such paths that are executable.

Also, while we focus on infinite paths, finite (executable) paths can be viewed as prefixes of paths since a finite path can always be extended to an infinite one, e.g. by extending the prefix with an infinite sequence of $noOp$ actions.

We now define what it means for a path p' to be a *suffix* of another path p w.r.t. a situation s:

$$\begin{aligned}\text{Suffix}(p', p, s) \doteq \text{OnPath}(p, s) \wedge \text{Starts}(p', s) \\ \wedge \forall s'.\ s \preceq s' \supset (\text{OnPath}(p, s') \equiv \text{OnPath}(p', s')).\end{aligned} \quad \textbf{(D4)}$$

That is, a path p' is a suffix of another path p w.r.t. a situation s iff s is on p, and p' which starts with s, contains exactly the same situations as p starting from s.

Given this, we can talk about properties of infinite paths in a natural way within the language of the situation calculus. For example, one can express that there is a path p starting in situation s such that a certain property Φ holds over all situations on p using the following sentence: $\exists p.\ \text{Starts}(p, s) \wedge (\forall s'.\ \text{OnPath}(p, s') \supset \Phi(s'))$. In an extended version of this paper (Khan and Lespérance 2015), we show how CTL* formulae can be interpreted over the situation calculus with paths and sketch how paths can be utilized in applications involving agents' goals and non-terminating programs. Also, in (Khan and Lespérance 2010), we discuss in detail the use of infinite paths in formalizing prioritized temporally extended goals and their dynamics in the situation calculus.

Properties of Paths

We now show some properties of our axiomatization of paths. Proofs of these properties can be found in (Khan and Lespérance 2015). Let Σ be the set of foundational axioms, and $\mathcal{D}_{path}$ consist of the axiomatization for paths and the associated definitions. Our first property captures the conditions under which a situation can be extended to a path: $\Sigma \cup \mathcal{D}_{path}$ entails that for any executable situation, there is a path that starts with that situation, provided that for any situation there exists an executable action.

$$\begin{aligned}\Sigma \cup \mathcal{D}_{path} \models (\forall s'.\ \exists a.\ \text{Poss}(a, s')) \supset \\ (\forall s.\ \text{Executable}(s) \supset \exists p.\ \text{Starts}(p, s)).\end{aligned} \quad \textbf{(P1)}$$

Again, we maintain that situations with no executable actions are "artificial".

Next, we prove some properties of the starting situation of a path. In particular, we can show that $\Sigma \cup \mathcal{D}_{path}$ entails that (a) any path starts with some situation, (b) the starting situation of any path is unique, and (c) the starting situation of any path is executable.

$$\begin{aligned}&\text{(a). } \Sigma \cup \mathcal{D}_{path} \models \forall p.\ \exists s.\ \text{Starts}(p, s), \\ &\text{(b). } \Sigma \cup \mathcal{D}_{path} \models \forall p, s, s'.\ \text{Starts}(p, s) \wedge \text{Starts}(p, s') \\ &\qquad\qquad \supset s = s', \\ &\text{(c). } \Sigma \cup \mathcal{D}_{path} \models \forall p, s.\ \text{Starts}(p, s) \supset \text{Executable}(s).\end{aligned} \quad \textbf{(P2)}$$

The next two properties deal with the successor situation of a situation on a path that is also on the path. The first states that $\Sigma \cup \mathcal{D}_{path}$ entails that for any situation s on a path p, there is a successor situation $s' = do(a, s)$ on p, and s' can be reached from s by performing an executable action a.

$$\begin{aligned}\Sigma \cup \mathcal{D}_{path} \models \forall p, s.\ \text{OnPath}(p, s) \\ \supset \exists s', a.\ \text{OnPath}(p, s') \wedge s' = do(a, s) \wedge \text{Poss}(a, s).\end{aligned} \quad \textbf{(P3)}$$

Moreover, $\Sigma \cup \mathcal{D}_{path}$ entails that the successor situation of a situation on a path is unique.

$$\begin{aligned}\Sigma \cup \mathcal{D}_{path} \models \forall p, s.\ [\text{OnPath}(p, s) \wedge \\ \text{OnPath}(p, do(a, s)) \wedge \text{OnPath}(p, do(b, s))] \supset a = b.\end{aligned} \quad \textbf{(P4)}$$

The next property deals with the uniqueness of paths: $\Sigma \cup \mathcal{D}_{path}$ entails that if $p \neq p'$, then there is a situation that is on path p but not on path p'.

$$\begin{aligned}\Sigma \cup \mathcal{D}_{path} \models \forall p, p'.\ p \neq p' \supset \\ \exists s.\ (\text{OnPath}(p, s) \wedge \neg\text{OnPath}(p', s)).\end{aligned} \quad \textbf{(P5)}$$

We can also show that $\Sigma \cup \mathcal{D}_{path}$ entails that all situations on a path are executable.

$$\Sigma \cup \mathcal{D}_{path} \models \forall p, s.\ \text{OnPath}(p, s) \supset \text{Executable}(s). \quad \textbf{(P6)}$$

We say that two situations are co-linear if they are the same or if one of them strictly precedes the other. Our next set of properties deal with the structure of situations on paths and shows that paths are essentially linear sequences of situations. First, we have $\Sigma \cup \mathcal{D}_{path}$ entails that any pair of situations on the same path are co-linear:

$$\begin{aligned}\Sigma \cup \mathcal{D}_{path} \models \forall p, s, s'.\ \text{OnPath}(p, s) \wedge \text{OnPath}(p, s') \supset \\ s = s' \vee s \prec s' \vee s' \prec s.\end{aligned} \quad \textbf{(P7)}$$

Secondly, we have $\Sigma \cup \mathcal{D}_{path}$ entails that if situations s and s' are on a given path p, then all situations in the interval defined by these two situations are also on p.

$$\begin{aligned}\Sigma \cup \mathcal{D}_{path} \models \forall p, s, s', s^*.\ (\text{OnPath}(p, s) \wedge \text{OnPath}(p, s') \\ \wedge\ s \preceq s^* \preceq s') \supset \text{OnPath}(p, s^*).\end{aligned} \quad \textbf{(P8)}$$

Finally, we can show that $\Sigma \cup \mathcal{D}_{path}$ entails that two paths can share only one common prefix. Once they branch at some situation, they never merge after that.

$$\begin{aligned}\Sigma \cup \mathcal{D}_{path} \models \forall p_1, p_2, s, a, b, s_1, s_2.\ [\text{OnPath}(p_1, do(a, s)) \\ \wedge\ \text{OnPath}(p_2, do(b, s)) \wedge a \neq b \wedge s \prec s_1 \wedge s \prec s_2 \\ \wedge\ \text{OnPath}(p_1, s_1) \wedge \text{OnPath}(p_2, s_2)] \supset s_1 \neq s_2.\end{aligned} \quad \textbf{(P9)}$$

The next few properties deal with suffixes and prefixes of a given path. The first of these states that $\Sigma \cup \mathcal{D}_{path}$ entails that for any situation s on a path p, there is a suffix of p that starts with s.

$$\Sigma \cup \mathcal{D}_{path} \models \forall p, s.\ \text{OnPath}(p, s) \supset \exists p'.\ \text{Suffix}(p', p, s). \quad \textbf{(P10)}$$

Secondly, we can show that given a path p with starting situation $do(a, s)$, $\Sigma \cup \mathcal{D}_{path}$ entails that there is a path p' s.t. p' starts with s, and p is a suffix of p' starting from $do(a, s)$.

$$\begin{aligned}\Sigma \cup \mathcal{D}_{path} \models \text{Starts}(p, do(a, s)) \supset \\ \exists p'.\ \text{Starts}(p', s) \wedge \text{Suffix}(p, p', do(a, s)).\end{aligned} \quad \textbf{(P11)}$$

Finally, $\Sigma \cup \mathcal{D}_{path}$ entails that any path that starts with a non-initial situation can be extended in the past; formally, for all situations s_1 and s_2, if s_1 strictly precedes s_2 and there is a path p_2 that starts with s_2, then there must also exist a path p_1 such that p_1 starts with s_1 and p_2 is a suffix of p_1 starting from s_2.

$$\begin{aligned}\Sigma \cup \mathcal{D}_{path} \models \forall s_1, s_2, p_2.\ s_1 \prec s_2 \wedge \text{Starts}(p_2, s_2) \supset \\ \exists p_1.\ \text{Starts}(p_1, s_1) \wedge \text{Suffix}(p_2, p_1, s_2).\end{aligned} \quad \textbf{(P12)}$$

We now prove some second-order induction principles for paths and for situations in a path. First, we have $\Sigma \cup \mathcal{D}_{path}$ entails that if some property Q holds for all paths that start with an initial situation, and if whenever Q holds for all paths that start with situation s, then it holds for all paths that start with any successor situation to s, then the property Q holds for all paths.

Theorem 1 (Induction on Paths)**.**

$$\begin{aligned}\Sigma \cup \mathcal{D}_{path} \models \forall Q.\ [\{\forall s, p.\ \text{Init}(s) \wedge \text{Starts}(p, s) \supset Q(p)\} \wedge \\ \{\forall a, s.\ (\forall p.\ \text{Starts}(p, s) \supset Q(p)) \\ \supset (\forall p'.\ \text{Starts}(p', do(a, s)) \supset Q(p'))\}] \\ \supset \forall p.\ Q(p).\end{aligned}$$

Moreover, $\Sigma \cup \mathcal{D}_{path}$ entails that if some property Q holds for the starting situation of a given path p, and if whenever Q holds for a situation s on path p, then it holds for the successor situation to s on p, then the property Q holds for all situations on path p.

Theorem 2 (Induction on Situations in a Path)**.**

$$\begin{aligned}\Sigma \cup \mathcal{D}_{path} \models \forall p, Q.\ [\{\forall s.\ \text{Starts}(p, s) \supset Q(s)\} \wedge \\ \{\forall a, s.\ (\text{OnPath}(p, s) \wedge Q(s) \wedge \text{OnPath}(p, do(a, s))) \\ \supset Q(do(a, s))\}] \\ \supset \forall s.\ \text{OnPath}(p, s) \supset Q(s).\end{aligned}$$

Next, we prove the correctness of our axiomatization. A natural way of capturing the notion of infinite path is by specifying it as a mapping from the set of natural numbers to situations on a path. To this end, we use a function σ of the following sort (here $\mathcal{S}$ denotes the set of all situations): $\sigma : \mathbb{N} \to \mathcal{S}$. We say that such a function σ models a *path sequence* if σ maps the number 0 to an executable situation (representing the starting situation of the path), and for each number n, there is an action a that is executable in the situation s_n produced by $\sigma(n)$ such that σ maps the immediate successor of n (i.e. $n + 1$) to the situation $do(a, s_n)$:

$$\begin{aligned}\text{PathSeq}(\sigma) \doteq \text{Executable}(\sigma(0)) \wedge \\ \forall n.\ \exists a.\ \text{Poss}(a, \sigma(n)) \wedge \sigma(n + 1) = do(a, \sigma(n)).\end{aligned} \quad \textbf{(D5)}$$

We say that a path p matches a path sequence σ if σ is indeed a path sequence, $\sigma(0)$ is the starting situation of p, and for all n, s and a, if $\sigma(n)$ is a situation s on path p, then $\sigma(n + 1)$ is the successor situation $do(a, s)$ of s on p:

$$\begin{aligned}\text{Matches}(p, \sigma) \doteq \text{PathSeq}(\sigma) \wedge (\sigma(0) = s \equiv \text{Starts}(p, s)) \\ \wedge \forall n, s.\ [\sigma(n) = s \wedge \text{OnPath}(p, s) \supset \\ \forall a.\ (\sigma(n + 1) = do(a, s) \equiv \text{OnPath}(p, do(a, s)))].\end{aligned} \quad \textbf{(D6)}$$

Given this formalization, the task of proving correctness of our axiomatization for infinite paths can be reduced to showing that path sequences are isomorphic to paths defined by $\Sigma \cup \mathcal{D}_{path}$, i.e. that there is an one-to-one mapping between these two. To this end, we first show that for any path p, there is a path sequence σ that matches p.[2]

Theorem 3 (Soundness)**.**

$$\Sigma_{\mathbb{N}} \cup \Sigma \cup \mathcal{D}_{path} \models \forall p.\ (\exists \sigma.\ \text{PathSeq}(\sigma) \wedge \text{Matches}(p, \sigma)).$$

[2]Here $\Sigma_{\mathbb{N}}$ is an axiomatization of the natural numbers, i.e., standard second-order Peano arithmetic, for the natural number sort.

Note that this implies that for any path p, there is a countably infinite number of distinct situations on p. Conversely, for any path sequence σ, there is a path p that matches σ.

Theorem 4 (Completeness)**.**

$$\Sigma_{\mathbb{N}} \cup \Sigma \cup \mathcal{D}_{path} \models \forall \sigma.\ \text{PathSeq}(\sigma) \supset \exists p.\ \text{Matches}(p, \sigma).$$

Conclusion

Our main contribution here is twofold: first, we introduced infinite paths in the situation calculus by providing an axiomatization for infinite paths; and second, we proved some desirable properties and showed that infinite paths in the situation calculus are well behaved and indeed correspond to an intuitive notion of paths. To the best of our knowledge, ours is the only work that introduces infinite paths as a sort in the language of the situation calculus.

Our framework thus allows one to specify dynamic domains and processes over them as well as their temporal properties. Also, compared to other formalisms, we inherit all the nice features of the situation calculus, e.g. a reasonable solution to the frame problem (Reiter 2001). Again, our account allows first-order quantification over paths, which contributes to much more readable formulae and makes the modeler's job simpler. Finally, by incorporating infinite paths in the language, we were able to identify useful general properties for reasoning within the situation calculus; e.g. the induction on paths property is a nice general property that follows from our foundational axioms.

In the future, we would like to utilize paths in the situation calculus to develop/further refine various applications involving infinite histories, such as verification of temporal properties of non-terminating programs.

References

Claßen, J., and Lakemeyer, G. 2008. A logic for non-terminating Golog programs. In *Proc. KR-08*, 589–599.

De Giacomo, G.; Lespérance, Y.; and Levesque, H. J. 2000. ConGolog, a concurrent programming language based on the situation calculus. *Artificial Intelligence* 121:109–169.

Khan, S. M., and Lespérance, Y. 2010. A logical framework for prioritized goal change. In *Proc. AAMAS-10*, 283–290.

Khan, S. M., and Lespérance, Y. 2015. *Infinite paths in the situation calculus (extended version)*. Technical Report EECS-2015-05, Dept. of EECS, York University, Canada.
URL: *http://www.eecs.yorku.ca/research/techreports/2015/*

Lespérance, Y.; Levesque, H. J.; Lin, F.; and Scherl, R. 2000. Ability and knowing how in the situation calculus. *Studia Logica* 66(1):165–186.

Levesque, H. J.; Pirri, F.; and Reiter, R. 1998. Foundations for a calculus of situations. *Electronic Transactions of AI (ETAI)* 2(3–4):159–178.

McCarthy, J., and Hayes, P. J. 1969. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence* 4:463–502.

Reiter, R. 2001. *Knowledge in Action. Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press.

Shapiro, S. 2005. *Specifying and Verifying Multiagent Systems using the Cognitive Agents Specification Language (CASL)*. Ph.D. Dissertation, University of Toronto, Canada.