Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems 1

Raymond Reiter

The MIT Press
Cambridge, Massachusetts
London, England

1 Introduction

Heraclitus reminds us that we cannot step into the same river twice. As goes the river, so goes virtually every modeling task in artificial intelligence, computer animation, robotics, software agents, decision and control theory, simulation, databases, programming languages, etc. The world simply won't sit still, and all attempts to model any but its simplest features must take change seriously. It is not trivial to address this problem in its full generality, as suggested by the following partial list of phenomena that a comprehensive theory of dynamical systems and autonomous agents must accommodate:

- The causal laws relating actions to their effects.
- The conditions under which an action can be performed.
- Exogenous and natural events.
- Probabilistic action occurrences and action effects.
- Decision theory: Determining what to do and when to do it.
- Complex actions and procedures.
- Discrete and continuous time.
- Concurrency.
- Continuous processes.
- Hypothetical and counterfactual reasoning about action occurrences and time.
- Perceptual actions and their effects on an agent's mental state.
- Deciding when to look and what to look for.
- Unreliable sensors and effectors.
- Agent beliefs, desires and intentions and how these influence behaviour.
- Real time (resource bounded) behaviour.
- Non deliberative (reactive) behaviour.
- Revising an agent's beliefs in the presence of conflicting observations.
- Planning a course of actions.
- Execution monitoring of a course of actions; recognizing and recovering from failures.

Despite the many existing disciplines that focus on modeling dynamical systems of one kind or another, a story as general as this has yet to be told. This is not to say that your average system modeler lacks for tools of the trade; there are plenty of formalisms to choose from, including Petri nets, process algebras, dynamic and temporal logics, finite automata, Markov decision processes, differential equations, STRIPS operators, influence diagrams, etc. But as this list suggests, what's available is more like a Tower of Babel than a unifying representational and computational formalism. To be fair, this state of affairs is the natural outcome of disciplines organized by their applications; discrete event control theory is concerned with different problems than, say, programming language design, and neither appear to have anything in common with semantics for tense in natural language. We all solve problems that arise in our own, sometimes narrowly circumscribed fields of specialization, and in this sense, we are like the proverbial blind men, each acting in isolation, and each trying to figure out the elephant. Nevertheless, one can't help thinking that there really is an elephant out there, that at a suitable level of abstraction, there must be a unifying "theory of dynamics", one that subsumes the many special purpose mechanisms that have been developed in these different disciplines, and that moreover accommodates all the features of autonomous dynamical systems listed above.

During the past 15 years or so, a number of researchers in artificial intelligence have been developing mathematical and computational foundations for dynamical systems that promise to deliver the elephant.[1] The methodological foundations of all these approaches —indeed, of much of the theory and practice of artificial intelligence—are based on what Brian Smith [**?**] has called the *Knowledge Representation Hypothesis*:

> *Any mechanically embodied intelligent process will be comprised of structural ingredients that a) we as external observers naturally take to represent a propositional account of the knowledge that the overall process exhibits, and b) independent of such external semantical attribution, play a formal but causal and essential role in engendering the behaviour that manifests that knowledge.*

Adopting this hypothesis has a number of important consequences:

1. We are naturally led to employ mathematical logic as a foundation for the "propositional account of the knowledge that the overall process exhibits" called for in part a) of the hypothesis.
2. This "propositional account" differs substantially from the state-based approaches central, for example, to decision and control theory. Instead of explicitly enumerating states and their transition function, the Knowledge Representation Hypothesis favours sentences—descriptions of what is true of the system and its environment, and of the causal laws in effect for that domain.
3. Part b) of the Knowledge Representation Hypothesis calls for a causal connection between these sentences and the system's *behaviours*. How can sentences lead to behaviour? In logic, sentences beget sentences through logical entailment, so it is natural to view system behaviours as appropriate *logical consequences* of the propositional account of the domain. On this perspective, the *computational* component of a logical representation for a dynamical system consists of deduction. Determining how a sys-

1 Needless to say, this is still just a promise.

tem behaves amounts to *deducing how it must behave*, given the system's description.

4. Providing a propositional account for some domain amounts to giving an abstract *specification* for that problem. Even by itself, having a non-procedural specification for a problem domain is a good thing; at least it's clear what modeling assumptions are being made. But in addition to this, because these are logical specifications, one can hope to prove, entirely within the logic, various properties of the specification. In other words, there is a direct mechanism, namely logical deduction, for establishing correctness properties for the system.
5. In those cases where deduction can be performed efficiently, these system specifications are also *executable*. This means that, as a side effect of providing a logical specification, we often obtain a simulator for the system.

This book deals with a logical approach to modeling dynamical systems based on a dialect of first order logic called the *situation calculus*, a language first proposed by John McCarthy in 1963 [**?**]. The material presented here has evolved over a number of years in response to the needs of the University of Toronto Cognitive Robotics Project, and therefore, has been heavily influenced by the problems that arise there. Broadly speaking, the newly emerging field of cognitive robotics has, as its long term objectives, the provision of a uniform theoretical and implementation framework for autonomous robotic or software agents that reason, act and perceive in changing, incompletely known, unpredictable environments. It differs from "traditional" robotics research in emphasizing "higher level" cognition as a determiner for agent behaviours. Therefore, one focus of cognitive robotics is on modeling an agent's beliefs and their consequences. These include beliefs about what is true of the world it inhabits, about the actions that it and other agents (nature included) can perform and the effects of those actions on the agent and its environment, about the conditions under which such actions can be performed, about the mental states and physical abilities of its fellow agents in the world, and about the outcomes of its perceptions. In keeping with the Knowledge Representation Hypothesis, these beliefs are represented as logical sentences, in our case, using the situation calculus. Such beliefs condition behaviour in a variety of ways, and one goal of cognitive robotics is to provide a theoretical and computational account of exactly how it is that deliberation can lead to action. None of this is meant to suggest that these objectives are peculiar to cognitive robotics; many control theorists and roboticists are concerned with modeling similar phenomena. What distinguishes cognitive robotics from these other disciplines is its emphasis on beliefs and how they condition behaviour, and by its commitment to the Knowledge Representation Hypothesis, and therefore, to logical sentences as the fundamental mathematical representation for dynamical systems and agent belief states. Nevertheless, despite these differences in emphasis and methodology, the representational and computational problems that must be solved are generic to all dynamical system modeling. Basically, we are all trying to do the same thing.

This book is about modeling and implementing autonomous agents and dynamical systems in the situation calculus. It addresses many, but certainly not all, of the issues enumerated at the beginning of this chapter; to get an idea of its coverage, glance through the Table of Contents. In addition to its focus on theoretical foundations, the book also emphasizes implementations. Because these are in the logic programming language Prolog, the programs are quite compact and therefore, are given in their entirety. All these programs are freely available and—Need I say it?—without guarantees of any kind; Appendix **??** tells you how to download them.