Munindar P. Singh

Multiagent Systems

A Theoretical Framework for Intentions, Know-How, and Communications

Foreword by Michael N. Huhns

Springer-Verlag
Berlin Heidelberg New York
London Paris Tokyo
Hong Kong Barcelona
Budapest

Lecture Notes in Artificial Intelligence 799

Subseries of Lecture Notes in Computer Science
Edited by J. G. Carbonell and J. Siekmann

Lecture Notes in Computer Science
Edited by G. Goos and J. Hartmanis

Series Editors

Jaime G. Carbonell
School of Computer Science, Carnegie Mellon University
Schenley Park, Pittsburgh, PA 15213-3890, USA

Jörg Siekmann
University of Saarland
German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany

Author

Munindar P. Singh
Information Systems Division, MCC
3500 W. Balcones Center Drive, Austin, TX 78759-5398, USA

CR Subject Classification (1991): I.2.11, C.2.4, D.4.7, F.3.2, I.2

ISBN 3-540-58026-3 Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-58026-3 Springer-Verlag New York Berlin Heidelberg

CIP data applied for

Typesetting: Camera ready by author
SPIN: 10131065 45/3140-543210 - Printed on acid-free paper

Foreword

Distributed artificial intelligence (DAI) is a melding of artificial intelligence with distributed computing. From artificial intelligence comes the theory and technology for constructing or analyzing an intelligent system. But where artificial intelligence uses psychology as a source of ideas, inspiration, and metaphor, DAI uses sociology, economics, and management science for inspiration. Where the focus of artificial intelligence is on the individual, the focus of DAI is on the group. Distributed computing provides the computational substrate on which this group focus can occur.

However, DAI is more than just the design of intelligent systems. It also provides insights and understanding about interactions among humans, as they organize themselves into various groups, committees, societies, and economies in order to improve their lives. For example, economists have been studying multiple agents for more than two hundred years, ever since Adam Smith in the eighteenth century, with the goal of being able to understand and predict economies. Economics provides ways to characterize masses of agents, and these are useful for DAI. But in return, DAI provides a means to construct artificial economies that can test economists' theories before, rather than after, they are applied.

Distributed artificial intelligence has become a growing and maturing subfield of computer science. Since the first organized gathering of researchers in DAI at an MIT workshop in 1979, there have been twelve DAI Workshops in the U.S.A., five MAAMAW Workshops in Europe, two CKBS Workshops in England, two MACC Workshops in Japan, and numerous meetings associated with other conferences. A substantial body of results, in the form of theories and working systems, has already been produced. As I write this foreword in late 1993, there are plans underway for five DAI-related colloquia in the next six months and an International Conference the following year. This level of interest around the globe is significant. It is indicative of the importance that DAI has attained in computer science, and of the quality and quantity of research that is being produced by its international research community.

Moreover, DAI is growing, even at a time when AI itself is not. I think there are three major reasons for this: (1) DAI deals with open systems, i.e., systems that are too large or unpredictable to be completely characterized—most real systems are of this type; (2) DAI is the best way to characterize or design distributed computing systems; and (3) DAI provides a natural way to view intelligent systems. I will elaborate on each of these reasons in turn.

First, real systems cannot be meaningfully closed and bounded for analysis purposes. No matter how they are defined, they will always be subject to new information from outside themselves, causing unanticipated outcomes. For example, to analyze fully the operation of a banking system and produce answers to such questions as "How many of the customers will try to access the banking system at the same time, and will the system be able to handle the resulting load?" one must attempt to include all of the people that use the system. This is infeasible. By taking an open systems approach and a social perspective, DAI provides notions of systems of commitment and joint courses of action that permit such questions to be considered naturally.

Second, DAI is the best way to characterize or design distributed computing systems. Information processing is ubiquitous. There are computer processors seemingly everywhere, embedded in all aspects of our environment. My office has five, in such places as my telephone and my clock, and this number does not consider the electrical power system, which probably uses hundreds in getting electricity to my office. The large number of processors and the myriad ways in which they interact makes distributed computing systems the dominant computational paradigm today.

But there is a concomitant complexity in all this processing and interaction that is difficult to manage. One effective way is by considering such distributed computing systems in anthropomorphic terms. For example, it is convenient to think that "my toaster *knows* when the toast is done," and "my coffee pot *knows* when the coffee is ready." When these systems are interconnected so they can interact, then they should also *know* that the coffee and toast should be ready at approximately the same time. In these terms, my kitchen becomes more than just a collection of processors—a distributed computing system—it becomes a *multiagent system.*

Third, DAI also provides a natural way to view intelligent systems. Much of traditional AI has been concerned with how an agent can be constructed to function intelligently, with a single locus of internal reasoning and control implemented in a Von Neumann architecture. But intelligent systems do not function in isolation—they are at the very least a part of the environment in which they operate, and the environment typically contains other such intelligent systems. Thus, it makes sense to view such systems in societal terms.

In support of this view, there is a fundamental principle that I find appealing and applicable here: *cognitive economy.* Cognitive economy is the idea that given several, equally good explanations for a phenomenon, a rational mind will choose the most economical, i.e., the simplest. The simplest explanations are the ones with the most compact representation, or the lowest computational cost to discover and use, or the minimum energy, or the fewest variables or degrees of freedom. Cognitive economy is manifested by an agent choosing the simplest representation that is consistent with its perceptions and knowledge. It is the basis for McCarthy's circumscription and accurately characterizes many aspects of human visual perception.[1]

There are several important ramifications for an agent that adheres to this idea. When applied to an agent's beliefs about its environment, cognitive economy leads an agent to believe in the existence of other agents: characterizing the environment as changing due to the actions of other agents is simpler than trying to cope with a random and unpredictable environment. (This is possibly why, when confronted with a complex and often incomprehensible world, ancient cultures concocted the existence of gods to explain such events as eclipses and the weather. Believing that a god is making it rain is simpler than understanding the physics of cloud formation.) When applied to the unknown internals (whether beliefs, desires, and intentions or states and next-state functions) of other agents, cognitive economy causes an agent to presume that other agents are just like itself, because that is the simplest way to represent them. (This is possibly why hypothesized gods are typically human-like.)

Hence, an agent must construct representations, albeit economical ones, that accurately cover its perceptions of the environment. Representations are simplifications that make certain problems easier to solve, but they must be sufficient for the agent to make realistic predictions about how its actions will change the environment. If an agent had *no* representations, it could still act, but it would be inefficient. For example, it would wander aimlessly if it did not know something about a graph it was traversing to reach a goal. The agent could treat the environment as deterministic and completely under its control—a STRIPS-like approach—but this would be inaccurate and not robust. The agent could model the unpredictability of the environment using statistics, but this would inform the agent only what it should do on the average, not specifically what it should do now. Of the many things that an agent

[1] "Rube Goldberg" devices are fascinating for people simply because they violate this principle of cognitive economy.

could choose to represent, agents are among the most important because they purposefully change the environment. It is thus rational for an agent to create and maintain internal representations of other agents; otherwise, it would simply react to the unpredictable (to it) changes in its environment that are caused by the other agents, and its own behavior would be inefficient.

What might be the nature of these representations? Agent architectures based on beliefs, desires, and intentions are common in AI. The beliefs are a representation of the environment, and form the basis upon which the agent chooses its actions. When an agent represents other agents, it must represent what *they* believe, desire, and intend. The other agents have beliefs about (i.e., representations of) this agent, and its beliefs and representations of them, *ad infinitum*. But this must converge, because representations are (by my definition) *simplifications* of the things being represented, and representations of representations of representations ..., soon have no content. A first-order representation for another agent is that it is the same as one's representation for oneself. The representation can then be refined via perception and interaction.

Believing that there are other agents in its environment will cause an agent to act differently, e.g., *benevolently* and *predictably*. First, a benevolent agent might clean up its environment, if it believes that other agents will act similarly, because it knows that it can operate more efficiently in a more orderly environment. For example, it might remove an obstacle that is blocking the path of other agents, under the belief that other agents will also help remove obstacles from its own path.

Second, as stated by Randall Davis, "An agent should act predictably." This implies

- the agent should have a self model with which it is consistent, i.e., its beliefs should be consistent with each other and its actions should be consistent with its beliefs, and

- the agent should have a representation of what other agents believe about it (a representation of their representation of it), and should act in accord with that.

By acting predictably, an agent will reduce conflicts with other agents, thereby increasing not only its own effectiveness, but also the effectiveness of all of the agents. Enabling this behavior is just one of the important capabilities for agents that needs to be researched.

Research Directions

There are two basic ways by which research progress has been made in DAI: (1) by extending single-agent concepts to multiple agents, and (2) by developing uniquely multiagent concepts for which there are no single-agent analogs. Examples of the first are extensions of belief revision and nonmonotonic reasoning to groups of agents, while examples of the second are negotiation, cooperation, content-based communication, and the design of environments in which autonomous and independently-developed agents are guaranteed to interact fairly.

A field is defined by the researchers and practitioners that consider themselves to be working in it, and by the collection of papers, results, and working systems that they produce. A substantial body of results has been produced in DAI, consisting of the proceedings of at least 24 Workshops, held in Europe, Asia, and North America; seven edited collections of DAI papers; numerous research monographs; several special journal issues; and working systems in such areas as manufacturing and process control, scheduling, and decision support. What is missing from all of this is a principled, comprehensive methodology for characterizing and constructing the essential component of a DAI system—an agent's cognitive structure, which determines its behavior and interaction with other agents. This book supplies such a methodology.

This book is also a return to the roots of DAI—the Contract Net—in the following sense. An agent has both knowledge and capabilities. The contract net provided a way for an agent to advertise its capabilities and employ them in assisting other agents. However, much of the work in DAI since the contract net was first described in 1978 has focused on the knowledge of the agents, rather than on their capabilities. This book provides a principled way to represent and discuss the capabilities of agents—not what they think and know, but what they can and will do. From a utilitarian viewpoint, this latter is of *far* greater importance.

For the future, there needs to be experimentation in DAI to validate the theoretical advances exemplified by this monograph. The experiments need to be conducted in both physical and computational environments: there is both difficulty and power in each. But experimentation in small, controlled worlds is not an effective way to establish meaningful relationships between agents and their environment. Such agents would not need or establish the type of relationships required for them to function effectively in real environments, i.e., they would not scale up. Also, the agents themselves need to be long-lived and adaptable. Agents that are restarted each time they are given a problem to solve are not confronting important aspects of autonomy.

There is still no uniquely multiagent aspect to learning. That is, group learning is so far nothing more than replicated individual learning. One promising possibility is based on the observation that an agent does not need to learn or remember something if it can rely on another agent to know or learn it. This affects what an agent chooses to learn, not how it chooses to learn it.

Most importantly, agents must have the ability to acquire and use representations of each other. This is what is needed for negotiation, cooperation, coordination, and multiagent learning. What should be the contents of these representations? This book provides the answer.

Michael N. Huhns

Preface

It is well-known that the future of computing lies in distributed computing. Distributed computing systems are of great significance in a number of current and future applications of computer science. For example, they are central to systems for electronic data interchange, air traffic control, manufacturing automation, computer supported cooperative work, and electronic banking, as well as in robotics and heterogeneous information systems. As the nature of computing comes to be increasingly characterized by networking and resource-integration, distributed computing systems will occur in all key applications.

The expansion and increasing importance of distributed computing presents us with a number of outstanding problems. I introduce these problems in terms of three main desiderata for distributed systems. One, practicable distributed systems must be *heterogeneous*. Reasons for this include the needs to (a) preserve past investment in diverse systems, (b) facilitate introduction of new technology piecemeal, and (c) optimize platform usage by using the most appropriate platform for each task. Two, the components of feasible distributed systems must in general be *locally autonomous*. Reasons for this include the needs to (a) manage security, (b) enable incremental change, and (c) obey legal requirements. Three, deployable distributed systems must behave in a manner that is not just *predictable*, but also *controllable* by their end users. Reasons for this include the needs to (a) behave correctly in critical applications, and (b) empower the ultimate users of technology, which is a key prerequisite to introducing it into novel application domains. A number of other requirements on distributed systems can also be stated, but the above are the most relevant for our purposes.

In light of the above, we must ensure that different components of a distributed system interact with one other in a manner independent of their internal implementations. The question then arises as to how we may specify how these interactions are to take place, so that components may independently be upgraded without affecting the correctness of the entire system. Most extant research on this problem concerns itself with the low-level aspects of interaction:

typically, it worries about the formatting of data rather than its content.

Another set of problems pertains to requirements acquisition. Capturing requirements for complex systems, which is a difficult problem in centralized computing, becomes harder and more urgent for distributed computing. We need to capture both the end users' requirements and the intermediate design requirements by which the desired interactions among different components can be used to guide their implementation. Consequently, the need for high-level specification techniques is more pressing than ever. For the same reason, there is need for formalization of any proposed specification approaches.

The program of research whose initial steps are reported in the present monograph addresses the issues described above. It does not entirely solve them, for they are too complex and have many facets, but it addresses their deepest aspects directly. I propose that we think of distributed systems as composed of intelligent entities with intentions, beliefs, and know-how. These entities interact through high-level communications in order to affect each others' intentions and beliefs and, thereby, actions. I submit that, when properly formalized and understood for their computational content, the intentions and know-how of agents, and the communications that take place among them, are important scientific abstractions for complex distributed systems.

These abstractions help us meet our three desiderata quite naturally. One, the heterogeneity of system components is hidden behind concepts that are independent of implementation; interactions among the components similarly occur in a content-based and implementation-independent manner. Two, the components can easily be designed to be autonomous, being influenced by other components only to the extent desired. Such components may also be used as mediators to shield existing applications. Three, specifications phrased in terms of the proposed concepts are natural for end users and more easily correspond to their wishes and expectations.

I develop a semantics of intentions and know-how in a general model of actions and time. Using this semantics, I also provide a semantics for the different modes of communication, including, e.g., promises and prohibitions. The proposed framework involves actions, possible and actual, abstract and concrete, that agents perform. This enables us to use intentions, know-how, and communications as more than just conceptual descriptions. Their formal semantics is useful for comparing implementations and for creating design tools. It aids us in stating constraints on system behavior that more naturally capture users' requirements. The proposed framework can thus serve as a foundation on which to develop specific approaches and methodologies for specifying, designing, and implementing complex systems.

I use the term *agent* to refer to the intelligent components of distributed systems when viewed in this manner; I use the term *multiagent systems* to describe the composite systems themselves. Of course, multiagent systems are really distributed systems and have no privileged approach to computation distinct from other distributed systems. But we should not allow ourselves to be distracted by this. The power of the proposed approach resides in its making high-level concepts available for specification and design and in supplying a formal notion of correctness in terms of those concepts. Of course, almost every abstraction can be reduced to lower-level abstractions, but that does not by itself make it useless. For example, just because programming languages can be compiled all the way to microcode or hardware does not mean that we should not use them. Similarly, multiagent systems can be reduced to ordinary distributed systems, but it may not be productive to do so.

It turns out that the approach I follow is closely related, in spirit, to work in distributed artificial intelligence (DAI). Concepts such as knowledge, intentions, know-how, and communications are of key importance in DAI. The work described here can thus be seen as contributing to DAI. However, I view DAI as a means to solve the problems in distributed computing, not as an end in itself.

One always takes a risk when attempting to formalize some previously informal concepts. This risk is the acutest for mental concepts, such as intentions. One can never precisely capture every informal aspect of such concepts, which may in fact be mutually contradictory. The yardstick I use in deciding upon a particular formalization is its utility to the main program of research, namely, the high-level specification of multiagent systems. If a formalization makes impossible to naturally derive an important result, that would be reason to discard it. If a formalization requires going into the innards of an intelligent system, rather than giving a high-level specification, that too would be reason to discard it.

This is not to say that other potential approaches are necessarily useless: just that, for purposes of computing, I see them as less fit than the chosen approach. Indeed, the history of science indicates that informal concepts can be formalized as several coexisting concepts. For example, the sixteenth century concept of *impetus* has been formalized as the present-day concepts of *momentum* and *kinetic energy*. Both can be valid: the first would be preferable for computing the force required to stop a moving object in a certain time and the second for computing the energy that must be dissipated in order to stop it.

I have sought to make the present work accessible to a wide audience, including graduate students and researchers in computer science (including

distributed computing and artificial intelligence), cognitive science, and philosophy. The main focus of the proposed theory is in computer science, however. No special background is required beyond a familiarity with logic and some mathematical maturity. A knowledge of temporal and modal logic would help, but is not essential.

Outline of this Monograph

The rest of this monograph is organized as follows. Chapter 1 discusses multiagent systems in some detail and shows what it means to take the intentional stance towards them. It also discusses the state of the art in computer science as it pertains to multiagent systems and points out how the present work fits in with it.

Chapter 2 motivates and develops the basic formal model, which considers the actions of agents in a framework of branching time. It admits simultaneous actions by several agents and allows the actions of different agents to be performed out of synchronization. These features help make the model correspond more closely to the multiagent systems that may occur in practice than would have been possible otherwise. The way in which all these features are brought together is novel.

In this chapter, I also motivate and describe *strategies*, which are programs denoting abstract actions. Strategies are used to define intentions and know-how in a way that makes them easy to interrelate. A notion of knowledge is also needed to properly define know-how. I describe the standard modal one, which I use here. Finally, I compare the proposed framework to the theories of action in linguistics, philosophy, and artificial intelligence.

Chapter 3 motivates a definition of intentions that is suitable for multiagent systems. It includes a discussion of the numerous dimensions of variation of intentions as studied in the pertinent artificial intelligence and philosophy literature. My goal is not to produce a philosophical treatise. However, it is crucial to obtain an understanding of the key issues, though from a strictly computer science perspective. After this discussion, I proceed to motivate and present a formalization of intentions and consider its strengths and limitations.

Chapter 4 motivates, defines, and formalizes the next core primitive in this monograph: know-how. Ability is an important special case of know-how, so it is treated first to clarify the subsequent formalization of know-how. Two formalizations are provided, one that considers the basic actions of agents directly, and another that considers abstractions over these actions. Though

these formalizations have different architectural consequences, they have the same logical properties. This is as one would expect, because merely considering abstractions should not change the intrinsic meaning of a concept.

Next, Chapter 5 relates the concepts of intentions and know-how to each other. This is where the theory of this monograph really begins to fall into place. I prove the key theorem showing how an agent who (a) intends something, (b) has the necessary skills, (c) persists long enough with his intentions, and (d) is rational enough to actually use his know-how, can in fact succeed with his intention. It is "obvious" theorems like this one that are the hardest to prove formally, because, unless all the necessary ingredients are correctly formalized, spurious and counterintuitive consequences can easily result. Indeed, previous attempts to prove theorems akin to this suffer from shortcomings, which I detail.

Lastly, Chapter 6 is an exercise in using the definitions of intentions and know-how to give a semantics of communications. Like many other approaches, the proposed approach to communications is based on speech act theory. However, it differs from classical formalizations of speech acts in that it cleanly separates the semantic aspects from the syntactic and pragmatic aspects. I show that most previous attempts at the semantics of speech acts are not really semantics at all, in that they do not give the conditions of satisfaction of sentences in a formal model as required in a Tarskian model-theoretic semantics. I give a semantics of speech acts that states their conditions of satisfaction in the proposed technical framework. This facilitates the statement of various requirements and correctness constraints on communications in multi-agent systems with respect to the intentions and know-how of the participants. This semantics is used in a formal analysis of the well-known contract net protocol.

Acknowledgments

This monograph strongly reflects my personal philosophy about computational agents and their role in computer science. Thus, it is especially relevant that I acknowledge those who have influenced me and helped me formulate these views.

This monograph is based on my Ph.D. dissertation completed at the University of Texas at Austin. I am deeply indebted to my advisers, Nicholas Asher and Allen Emerson, for their constant encouragement and advice on all matters technical and presentational. They guided my work without ever attempting to impose their own will upon it. Nicholas listened patiently to my

numerous harangues and pointed out the weaknesses of my various proposals. Allen helped relate my work to temporal logic and distributed computing and each instance of that exercise helped identify some superfluity or the other.

I am also indebted to the other members of my doctoral committee, Hans Kamp, Woody Bledsoe, and Robert Boyer, for their patience and trust. Bob Boyer, in particular, gave valuable comments on a previous draft that have helped me clarify and correct some claims.

I should take this occasion to thank my undergraduate professors. Shachin Maheshwari inspired me to undertake research in computer science and stressed the interplay between its practical and theoretical aspects. Niraj Sharma introduced me to distributed computing in a way that has held my interest, even in periods when I haven't actually pursued it.

Among the people I was fortunate enough to interact with in the course of my graduate career, Michael Huhns is the one I am most indebted to. He kept me thinking about multiagent systems, first in artificial intelligence and later in classical distributed computing. I learned a lot about computer science research from him. He has been a pillar of support and has given me great advice on professional and other matters.

I also benefited a lot from conversations with Benjamin Kuipers and Robert Koons, who kept me in touch with two opposite subjects that my research touches upon: reasoning about physical systems and the philosophy of mind, respectively. Rob gave me useful comments on previous versions of parts of this book. Several other people have helped me with their advice and suggestions. Kurt Konolige initially got me interested in theories of actions and intentions. Norman Martin encouraged me to pursue research in logic as applied to artificial intelligence. Manfred Krifka and Daniel Bonevac taught me much of what I know about the applications of logic in natural language.

Over the years, I have had useful discussions with a number of people, including Larry Stephens, Jürgen Müller, Wlodek Zadrozny, Raimo Tuomela, Nigel Seel, Anand Rao, Michael Georgeff, Jeff Rosenschein, Martha Pollack, Candy Sidner, and Graham Oddie. I have also gained much from interactions with fellow graduate students at the University of Texas and elsewhere, especially, Pankaj Mehra and Sakthi Subramanian. Several colleagues have helped shape my view of computer science, even though I did not always talk to them about the research reported here. These include Christine Tomlinson, Phil Cannata, Greg Lavender, Paul Attie, Greg Meredith, Wei-Min Shen, Jim Barnett, and Paul Portner.

Many of the changes from my dissertation to the present version were caused by comments from Jürgen Müller, Raimo Tuomela, Mike Huhns, Allen

Emerson, and Mona Singh. Jürgen, in particular, read a previous version extremely carefully and gave me lots of useful technical comments.

I am indebted most of all to my family for their continuous support. My son, Amitoj, was born after I completed my dissertation. He has tolerated my distractions with my research and always given me something to smile about. My wife encouraged me to pursue my research and was always there for me. She read several parts of this book, as well as papers in which some other parts were published. Her suggestions led to several improvements in the presentation. My parents gently nudged me towards completion. My father read draft versions of Chapters 1 and 3, and suggested numerous stylistic improvements, which I have tried to incorporate elsewhere also. My brother's frequent phone calls helped graduate school pass by easily. He and my sister-in-law and nephews were always exceedingly hospitable on my numerous visits to their home, mostly to work summers or attend conferences in the area.

This research was supported by various agencies. The University of Texas awarded me a Microelectronics and Computer Development (MCD) Fellowship. Kurt Konolige invited me to the Artificial Intelligence Center of SRI International for the summers of 1987 and 1988. Nicholas Asher and Carlota Smith supported me from a National Science Foundation grant (# IRI-8945845) to the Center for Cognitive Science, University of Texas. Gerhard Barth made it possible for me to visit DFKI (the German Research Center for Artificial Intelligence), Kaiserslautern, Germany during the summer of 1991. I have been supported most of all by the Microelectronics and Computer Technology Corporation. The director of the Artificial Intelligence Laboratory, Elaine Rich, supported me from June 1989 to August 1990. As director of the Carnot Heterogeneous Database Project, Philip Cannata supported me from September 1991 to May 1993. Phil encouraged me to pursue my research on multiagent systems, while giving me a great opportunity to acquire research expertise in heterogeneous databases. After Phil left, Darrell Woelk, who took over as director, has successfully managed to maintain a productive research climate in extremely tough times.

Munindar Paul Singh

Contents