Computationally Grounded Theories of Agency[*]

Michael Wooldridge
Department of Computer Science, University of Liverpool
Liverpool L69 7ZF, United Kingdom

M.J.Wooldridge@csc.liv.ac.uk

Abstract

In this paper, I motivate, define, and illustrate the notion of computationally grounded *theories of agency. A theory of agency is said to be computationally grounded if we can give the theory an interpretation in terms of some concrete computational model. This requirement is essential if we are to claim that the theories we develop can be understood as expressing properties of real multiagent systems. After introducing and formally defining the concept of a computationally grounded theory of agency, I illustrate the idea with reference to $\mathcal{VSK}$ logic, a formalism for reasoning about agent systems that has a semantics defined with respect to an automata-like model of agents. $\mathcal{VSK}$ logic is an extension of modal epistemic logic, which allows us to represent what information is visible to an agent, what it sees, and what it knows. We are able to prove that formulae of $\mathcal{VSK}$ logic correspond directly to properties of agents.*

1 Introduction

Artificial Intelligence (AI) is a broad church, which encompasses many different sub-fields. Researchers from such wildly differing disciplines as philosophy, cognitive science, mathematical logic, and economics all count themselves as carrying out AI research. AI draws strength from this diversity, which has led to an enormously robust and rich subject area. One aspect of the interdisciplinary nature of AI is that it is both an *engineering* and a *scientific* discipline. Engineers in AI aim to gain an understanding of how to build computer systems that are capable of intelligent behaviour. Scientists in AI aim to develop *theories* that explain intelligent behaviour.

Even if you do not view multiagent systems as a sub-field of AI, I hope you will accept that it shares many of the characteristics of AI as a research discipline. Like AI itself, multiagent systems is a broad church, taking input from a number of diverse disciplines, including economics, sociology, ethology, and of course sub-fields of computer science such as distributed systems. The goals of the multiagent systems project are also similar in nature to those of the AI project. The eventual success — or otherwise — of the multiagent systems project will have implications that are every bit as profound as the success of the AI project. As in AI, researchers in multiagent systems can also be broadly divided into those that do engineering (and so aim to understand how we can build societies of agents that exhibit various kinds of social behaviour, such as cooperative problem solving and negotiation), and those that do science (and so aim to develop theories that explain and predict the behaviour of such social systems).

In this paper, I am primarily concerned with multiagent systems as a scientific endeavour. My goal is to motivate and explain a particular requirement — *computational grounding* — for a certain class of agent theories. I take my cue for this requirement from the following observation: What makes multiagent systems unique as a research discipline, distinct from such disciplines as sociology, economics, and game theory, is that it is concerned with *computational* models. This observation will be self-evident to practitioners of multiagent systems, who naturally make use of computational models to implement and evaluate their systems. When we consider the *theory* of multiagent systems, however, the situation is less clear-cut. Many theories have been put forward to explain aspects of rational agency and multiagent systems. However, I claim that these theories have, for the most part, not been *grounded* in any computational model. In the remainder of this paper, I will explain in detail what I mean by computational grounding, and attempt to justify the claim that, for many kind of theories of agency and multiagent systems (though by no means all), computational grounding is an important requirement. To illustrate these ideas, I present $\mathcal{VSK}$ logic, in which formulae can be interpreted directly with respect to a computational model. I conclude with a discussion on open issues.

[*] Much of the work on $\mathcal{VSK}$ logic summarised in this paper was carried out jointly with Alessio Lomuscio.

2 Theories of Multiagent Systems

Computer science is, as much as it is about anything, about developing formalisms and theories that enable us to model, understand, and reason about computational systems. Many formalisms have been proposed for reasoning about intelligent agents and multi-agent systems, and using these formalisms, many theories have been developed that attempt to explain aspects of multiagent behaviour.

Most formalisms for multiagent systems research have taken as their starting point either game theory [2, 16, 19] or mathematical logic [29]. Researchers using game theory focus primarily on interactions between self-interested agents. A typical research programme starts by identifying a particular type of interaction scenario (such as the prisoner's dilemma [1]), and then asks what the best strategy is for any agent placed in such a scenario. Another common research programme involves asking how an interaction protocol can be engineered so that if participating agents behave rationally, then certain desirable outcomes are guaranteed (such as the maximisation of social welfare) — this is mechanism design.

In this paper, I will focus on logical theories of rational agents and multiagent systems. A typical research programme involves attempting to develop a logical axiomatization of some phenomenon of interest, and then investigating the extent to which the logical consequences of this axiomatization correspond to our understanding of the phenomenon. Some of the best-known papers in the multiagent systems field are examples of such work — these include Cohen and Levesque's well-known theory of intention [4], Rao and Georgeff's work on formalising the belief-desire-intention paradigm [14, 26], and the many attempts to formalise social phenomena, such as the semantics of speech acts [5, 22], teamwork [11], cooperative problem solving [30], cooperation protocols [7], argumentation [10], and the dynamics of mental states [12].

Most formalisms for expressing axiomatic theories of multiagent systems have been based upon *modal logic* with Kripke, or possible worlds semantics [3]. Following Hintikka's pioneering work on the use of modal logic for formalising knowledge and belief [8], the idea is to use modal operators to represent an agent's attitudes — its beliefs, desires, and the like. (The motivations for this approach, and the technicalities of Kripke semantics are beyond the scope of this paper; see [21, 29, 27] for introductions.) As an approach to characterising an agent's attitudes, Kripke semantics have much in their favour. In particular:

- the associated mathematics of *correspondence theory* makes it comparatively easy to prove properties of modalities, and in particular, to prove soundness and completeness results;
- Kripke semantics allow us to remain silent with respect to the internal structure of an agent.

Despite these advantages, most such formalisms have one main disadvantage: they are not *computationally grounded*, in the following sense.

Suppose we have some set of programs, $\Pi = \{\pi_1, \pi_2, \ldots\}$. Think of these as JAVA or PASCAL programs, for example. We want to show that one of these programs $\pi \in \Pi$ corresponds to, or implements some theory of agency. The theory of agency is represented as a formula φ of some logical language $\mathcal{L}$. This logic might be Cohen and Levesque's intention logic [4], for example, or Rao and Georgeff's BDI logic [14, 26]. How might we go about showing that program π implements theory φ?

The *semantics* of $\mathcal{L}$ will be given with respect to a set $mod(\mathcal{L})$ of logical models for $\mathcal{L}$. Formally, the semantics of $\mathcal{L}$ will be given by a function

$$[\![\ldots]\!]_{\mathcal{L}} : wff(\mathcal{L}) \to \wp(mod(\mathcal{L}))$$

which assigns to every formula φ of $\mathcal{L}$ a set of models — those in which φ is satisfied.

The semantics of a program π are given in terms of a function

$$[\![\ldots]\!]_{\Pi} : \Pi \to \wp(\mathcal{C})$$

which maps a program to a set of *computations*. Think of a computation as a "run", or possible history of a program. Thus if $c \in [\![\pi]\!]_{\Pi}$, then the computation $c \in \mathcal{C}$ represents one possible run of the program π. The set of all such runs represents the meaning of the program π — its semantics.

Now, in general, there is *no relationship* between models $mod(\mathcal{L})$ for $\mathcal{L}$ and computations $\mathcal{C}$. For example, in Cohen and Levesque's logic, an agent's beliefs and goals are characterised model-theoretically in terms of belief and goal accessibility relations; but we have no way of systematically associating such relations with arbitrary computer programs. In simple terms, this means that we will in general be unable to characterise the behaviour of π in terms of the logic $\mathcal{L}$, and to return to our motivating example, we have no way of showing that π implements the theory φ.

I believe that computational grounding is not simply a desirable property that we can choose to ignore. If a logic of agency is not computationally grounded, then this must throw doubt on the claim that this logic can be useful for reasoning about computational agent systems. If we really intend our theories to be theories of *computational* systems, then computational grounding is an issue that must be addressed. In [25], I point out that computational grounding is essential if we are to treat agent theories as specifications for systems.

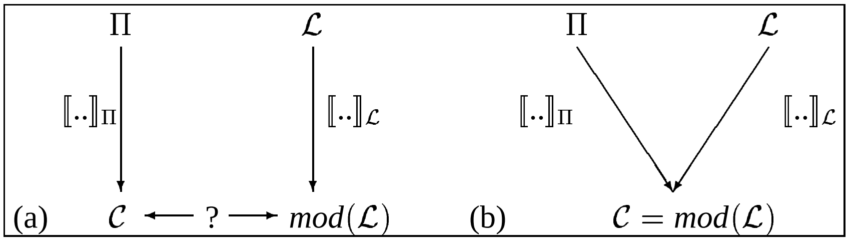

Figure 1. Computationally ungrounded (a) and computationally grounded (b) logics.

To be computationally grounded, a theory does not necessarily have to have a direct interpretation in terms of program computations. An alternative is to define a function

$$f : \wp(\mathcal{C}) \to \mathit{wff}(\mathcal{L})$$

that would take as input a set of computations, representing the semantics of a program, and from them derive a formula of $\mathcal{L}$ representing the properties of this program.

Of course, not all formalisms for reasoning about multiagent systems are ungrounded. The best known example of a grounded formalism for reasoning about multiagent systems is also one of the most successful: epistemic logic [6]. Epistemic logic is the modal logic of knowledge. The semantics of epistemic logic are given in terms possible worlds, but crucially, these possible worlds are given a *concrete interpretation* in terms of the states of computer processes:

> [An agent] ... x is said to carry the information that p in world state s, written $s \models K(x, p)$, if for all world states in which x has the same value as it does in s, the proposition p is true. [9, p36]

Because the semantics of the knowledge modality in epistemic logic are given in terms of the states of programs, when we write a formula of epistemic logic there is some justification for the claim that this formula is expressing a property of *programs*. Program computations can be understood as models for epistemic logic, and in this sense, epistemic logic is computationally grounded. The notion of computationally grounded logics is illustrated in Figure 1.

In the remainder of this paper, I describe $\mathcal{VSK}$ logic, an extension of epistemic logic that maintains the computational grounding of the original formalism. The presentation of this logic is in two parts. First, the computational model that underpins $\mathcal{VSK}$ logic is presented. In the sections that follow, the logic itself is developed, and I show how possible axioms of $\mathcal{VSK}$ logic can be understood as expressing properties of the computational model.

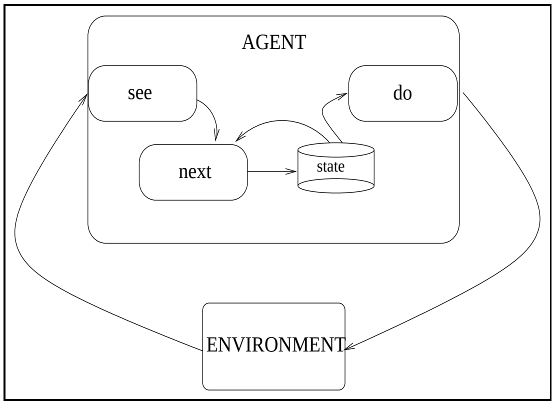

Figure 2. A computational model.

3 A Computational Model

In this section, we present a computational model of agents and the environments they occupy — see Figure 2. First, it is assumed that the environment may be in any of a set $E = \{e, e', \ldots\}$ of instantaneous states, and that the (single) agent occupying this environment may be in any of a set $L = \{l, l', \ldots\}$ of *local* states. Agents are assumed to have a repertoire of possible actions available to them, which transform the state of the environment — we let $Ac = \{\alpha, \alpha', \ldots\}$ be the set of actions. We assume a distinguished member *null* of *Ac*, representing the "noop" action, which has no effect on the environment.

In order to represent the effect that an agent's actions have on an environment, we introduce a *state transformer* function, $\tau : E \times Ac \to E$ (cf. [6, p154]). Thus $\tau(e, \alpha)$ denotes the environment state that would result from performing action α in environment state e. Note that our environments are *deterministic*: there is no uncertainty about the result of performing an action in some state. Dropping this assumption is not problematic, but it does make the formalism somewhat more convoluted.

In order to represent what is knowable about the environment, we use a *visibility function*, $\nu : E \to (\wp(E) \setminus \emptyset)$. The idea is that if the environment is *actually* in state e, then it is impossible for any agent in the environment to distinguish between e and any member of $\nu(e)$. We require that ν partitions E into mutually disjoint sets of states, and that $e \in \nu(e)$, for all $e \in E$. For example, suppose $\nu(e_2) = \{e_2, e_3, e_4\}$. Then the intuition is that the agent would be unable to distinguish between e_2 and e_3, or between e_2 and e_4. Note that visibility functions are *not* intended to capture the everyday notion of visibility, as in "object x is visible to the agent".

We will say ν is *transparent* if $\nu(e) = \{e\}$. Intuitively, if ν is transparent, then it will be possible for an agent observing the environment to distinguish every different environment state.

Formally, an environment *Env* is a 4-tuple $\langle E, \tau, \nu, e_o \rangle$, where E is a set of environment states as above, τ is a state transformer function, ν is a visibility function, and $e_0 \in E$ is the initial state of *Env*.

From Figure 2, we can see that an agent has three functional components, representing its sensors (the function *see*), its next state function (*next*), and its action selection, or decision making function (*do*). Formally, the perception function $see : \wp(E) \to P$ maps sets of environment states to *percepts* — we denote members of P by $\rho, \rho', \ldots$. The agent's next state function $next : L \times P \to L$ maps an internal state and percept to an internal state; and the action-selection function $do : L \to Ac$ simply maps internal states to actions.

The behaviour of an agent can be summarised as follows. The agent starts in some state l_0. It then observes its environment state e_0 through the visibility function $\nu(e_0)$, and generates a percept $see(\nu(e_0))$. The internal state of the agent is then updated to $next(l_0, see(\nu(e_0)))$. The action selected by the agent is then $do(next(l_0, see(\nu(e_0))))$. This action is performed, and the agent enters another cycle.

Together, an environment/agent pair comprise a *system*. The *global* state of a system at any time is a pair containing the state of the agent and the state of the environment. Let $G = E \times L$ be the set of all such global states. We use g (with annotations: $g, g', \ldots$) to stand for members of G. A *run* of a system can be thought of as an infinite sequence:

$$g_0 \xrightarrow{\alpha_0} g_1 \xrightarrow{\alpha_1} g_2 \xrightarrow{\alpha_2} g_3 \xrightarrow{\alpha_3} \cdots \xrightarrow{\alpha_{u-1}} g_u \xrightarrow{\alpha_u} \cdots$$

A sequence $(g_0, g_1, g_2, \ldots)$ over G represents a run of an agent $\langle see, next, do, l_0 \rangle$ in an environment $\langle E, \tau, \nu, e_0 \rangle$ iff:

1. $g_0 = \langle e_0, next(l_0, see(vis(e_0))) \rangle$ and;
2. $\forall u \in I\!N$, if $g_u = \langle e, l \rangle$ and $g_{u+1} = \langle e', l' \rangle$ then

$$\begin{array}{rcll} e' & = & \tau(e, do(l)) & \text{and} \\ l' & = & next(l, e') & \end{array}$$

Let $G_{Env,Ag} \subseteq G$ denote the set of global states that system *Env*, *Ag* could enter during execution.

In order to represent the properties of systems, we assume a set $\Phi = \{p, q, r, \ldots\}$ of primitive propositions. In order to *interpret* these propositions, we use a function $\pi : \Phi \times G_{Ag,Env} \to \{T, F\}$. Thus $\pi(p, g)$ indicates whether proposition $p \in \Phi$ is true (T) or false (F) in state $g \in G$. Note that members of Φ are assumed to express properties of *environment states only*, and *not* the internal properties of agents. We also require that any two different states differ in the valuation of at least one primitive proposition.

We refer to a triple $\langle Env, Ag, \pi \rangle$ as a *model* — our models play the role of interpreted systems in knowledge theory [6, p110]. We use M (with annotations: $M', M_1, \ldots$) to stand for models.

4 A Computationally Grounded Logic

Now that we have the computational model in place, we progressively introduce $\mathcal{VSK}$ logic, which will enable us to represent properties of agents and their environments [31]. $\mathcal{VSK}$ logic is an extension of modal epistemic logic, which in addition to allowing us to represent what is true of an environment and what an agent knows about it, allows us to represent what is visible, or knowable of the environment, then what an agent perceives of the environment. The semantics of $\mathcal{VSK}$ logic are given directly in terms of the computational model presented above, and thus when we write a formula of $\mathcal{VSK}$ logic, it is possible to establish properties of the agent and environment that this formula corresponds to.

I begin by introducing the propositional logic fragment of $\mathcal{L}$, which allows us to represent what is true of the environment. Propositional formulae of $\mathcal{L}$ are built up from Φ using the classical logic connectives "$\wedge$" (and), "$\vee$" (or), "$\neg$" (not), "$\Rightarrow$" (implies), and "$\Leftrightarrow$" (if, and only if), as well as logical constants for truth ("**true**") and falsity ("**false**"). I define the syntax and semantics of the truth constant, disjunction, and negation, and assume the remaining connectives and constants are introduced as abbreviations in the conventional way. Formally, the syntax of the propositional fragment of $\mathcal{L}$ is defined by the following grammar:

$$\langle \mathit{wff} \rangle ::= \textbf{true} \mid \text{any element of } \Phi \mid \neg \langle \mathit{wff} \rangle \mid \langle \mathit{wff} \rangle \vee \langle \mathit{wff} \rangle$$

The semantics are defined via the satisfaction relation "$\models$":

$$\begin{array}{ll} \langle M, g \rangle \models \textbf{true} & \\ \langle M, g \rangle \models p & \text{iff } \pi(p, g) = T \quad (\text{where } p \in \Phi) \\ \langle M, g \rangle \models \neg\varphi & \text{iff not } \langle M, g \rangle \models \varphi \\ \langle M, g \rangle \models \varphi \vee \psi & \text{iff } \langle M, g \rangle \models \varphi \text{ or } \langle M, g \rangle \models \psi \end{array}$$

I will assume the conventional definitions of satisfiability, validity, and validity in a model.

I now enrich $\mathcal{L}$ by the addition of a unary modality "$\mathcal{V}$", which will allow us to represent the information that is instantaneously visible or knowable about an environment state. Thus suppose the formula $\mathcal{V}\varphi$ is true in some state $g \in G$. The intended interpretation of this formula is that the property φ is *knowable* of the environment when it is in state g; in other words, that an agent equipped with suitable sensory apparatus would be able to perceive the information φ. If $\neg\mathcal{V}\varphi$ were true in some state, then *no* agent, no matter how good its sensory apparatus was, would be able to perceive φ.

Note that our concept of visibility is distinct from the everyday notion of visibility as in "object o is visible to the agent". If we were interested in capturing this notion of visibility we could use a first-order logic predicate along the lines of $\mathit{visible}(x, y, o)$ to represent the fact that when an agent is in position (x, y), object o is visible. The arguments to such visibility statements are *terms*, whereas the arguments to the visibility statement $\mathcal{V}\varphi$ is a *proposition*.

In order to give a semantics to the $\mathcal{V}$ operator, I define a binary *visibility accessibility relation* $\sim_\nu \subseteq G_{Ag,Env} \times G_{Ag,Env}$ as follows: $\langle e, l\rangle \sim_\nu \langle e', l'\rangle$ iff $e' \in \nu(e)$. Since ν partitions E, it is easy to see that $\sim_\nu$ is an equivalence relation. The semantic rule for the $\mathcal{V}$ modality is given in terms of the $\sim_\nu$ relation in the standard way for possible worlds semantics: $\langle M, \langle e, l\rangle\rangle \models \mathcal{V}\varphi$ iff $\langle M, \langle e', l'\rangle\rangle \models \varphi$ for all $\langle e', l'\rangle \in G_{Ag,Env}$ such that $\langle e, l\rangle \sim_\nu \langle e', l'\rangle$. As $\sim_\nu$ is an equivalence relation, the $\mathcal{V}$ modality has a logic of S5 [6]. In other words, formula schemas (1)-(5) are valid in $\mathcal{L}$:

$$\mathcal{V}(\varphi \Rightarrow \psi) \Rightarrow ((\mathcal{V}\varphi) \Rightarrow (\mathcal{V}\psi)) \tag{1}$$

$$\mathcal{V}\varphi \Rightarrow \neg\mathcal{V}\neg\varphi \tag{2}$$

$$\mathcal{V}\varphi \Rightarrow \varphi \tag{3}$$

$$\mathcal{V}\varphi \Rightarrow \mathcal{V}(\mathcal{V}\varphi) \tag{4}$$

$$\neg\mathcal{V}\varphi \Rightarrow \mathcal{V}\neg\mathcal{V}\varphi \tag{5}$$

I will omit the (by now standard) proof of this result — see, e.g., [6, pp58-59].

Formula schema (3) captures the first significant interaction between what is true and what is visible. However, we can also consider the converse of this implication:

$$\varphi \Rightarrow \mathcal{V}\varphi \tag{6}$$

This schema says that if φ is true of an environment, then φ is knowable. We can characterise this schema in terms of the environment's visibility function: formula schema (6) is valid in a model iff the visibility function of that model is transparent. Thus in transparent environments, visibility collapses to truth, since $\varphi \Leftrightarrow \mathcal{V}\varphi$ will be valid in such environments. In other words, everything true in a transparent environment is also visible, and *vice versa*. Note that this a *helpful* property of environments — in the terminology of [18], such environments are *accessible*. Unfortunately, most environments do not enjoy this property.

The fact that something is visible in an environment does not mean that an agent actually sees it. What an agent *does* see is determined by its sensors, which in our computational model are represented by the *see* function. I now extend the logic by introducing a unary modal operator "$\mathcal{S}$", which is intended to allow us to represent the information that an agent sees. The intuitive meaning of a formula $\mathcal{S}\varphi$ is thus that the agent perceives the information φ. Note that, as with the $\mathcal{V}$ operator, the argument to $\mathcal{S}$ is a *proposition*, and *not* a term denoting an object.

In order to define the semantics of $\mathcal{S}$, we introduce a *perception accessibility relation* $\sim_s \subseteq G_{Ag,Env} \times G_{Ag,Env}$ as follows: $\langle e, l\rangle \sim_s \langle e', l'\rangle$ iff $\mathit{see}(\nu(e)) = \mathit{see}(\nu(e'))$. That is, $g \sim_s g'$ iff the agent receives the same percept when the system is in state g as it does in state g'. Again, it is straightforward to see that $\sim_s$ is an equivalence relation. Note that, for any of our models, it turns out that $\sim_\nu \subseteq \sim_s$.

The semantic rule for $\mathcal{S}$ is: $\langle M, \langle e, l\rangle\rangle \models \mathcal{S}\varphi$ iff $\langle M, \langle e', l'\rangle\rangle \models \varphi$ for all $\langle e', l'\rangle \in G_{Ag,Env}$ such that $\langle e, l\rangle \sim_s \langle e', l'\rangle$. As $\sim_s$ is an equivalence relation, $\mathcal{S}$ will also validate analogues of the S5 modal axioms.

It is worth asking whether these schemas are appropriate for a logic of perception. If we were attempting to develop a logic of *human* perception, then an S5 logic would *not* be acceptable. Human perception is often faulty, for example, thus rejecting schema $\mathcal{S}\varphi \Rightarrow \varphi$. We would almost certainly reject $\neg\mathcal{S}\varphi \Rightarrow \mathcal{S}\neg\mathcal{S}\varphi$ for similar reasons. However, our interpretation of $\mathcal{S}\varphi$ is that *the percept received by the agent carries the information* φ. Under *this* interpretation, an S5 logic seems appropriate.

I now turn to the relationship between $\mathcal{V}$ and $\mathcal{S}$. Given two unary modal operators, $\Box_1$ and $\Box_2$, the most important interactions between them can be summarised as follows:

$$\Box_1\varphi \;\substack{\Rightarrow \\ \Leftarrow}\; \Box_2\varphi \tag{$*$}$$

I use $(*)$ as the basis of our investigation of the relationship between $\mathcal{V}$ and $\mathcal{S}$. The most important interaction axiom says that if an agent sees φ, then φ must be visible. It turns out that formula schema (7), which characterises this relationship, is valid — this follows from the fact that $\sim_s \subseteq \sim_\nu$.

$$\mathcal{S}\varphi \Rightarrow \mathcal{V}\varphi \tag{7}$$

Turning to the converse direction, the next interaction says that if φ is visible, then φ is seen by the agent — in other words, the agent sees everything visible.

$$\mathcal{V}\varphi \Rightarrow \mathcal{S}\varphi \tag{8}$$

Intuitively, this axiom characterises agents with "perfect" sensory apparatus, i.e., a *see* function that *never loses information*. Formally, we will say a perception function *see* is *perfect* iff it is an injection; otherwise we will say it is *lossy*. Lossy perception functions can map different visibility sets to the same percept, and hence, intuitively, lose information. It turns out that formula schema (8) is valid in a model if the perception function of that model is perfect.

I now extend the language $\mathcal{L}$ by the addition of a unary modal operator $\mathcal{K}$. The intuitive meaning of a formula $\mathcal{K}\varphi$ is that the agent knows φ. In order to give a semantics to $\mathcal{K}$, we introduce a *knowledge accessibility relation* $\sim_k \subseteq G_{Ag,Env} \times G_{Ag,Env}$ in the by-now conventional way [6, p111]: $\langle e, l\rangle \sim_k \langle e', l'\rangle$ iff $l = l'$. As with $\sim_v$ and $\sim_s$, it is easy to see that $\sim_k$ is an equivalence relation. The semantic rule for $\mathcal{K}$ is as expected: $\langle M, \langle e, l\rangle\rangle \models \mathcal{K}\varphi$ iff $\langle M, \langle e', l'\rangle\rangle \models \varphi$ for all $\langle e', l'\rangle \in G_{Ag,Env}$ such that $\langle e, l\rangle \sim_k \langle e', l'\rangle$. Obviously, as with $\mathcal{V}$ and $\mathcal{S}$, the $\mathcal{K}$ modality validates analogues of the S5 modal axioms. Now turn to the relationship between what an agent perceives and what it knows. As with the relationship between $\mathcal{S}$ and $\mathcal{V}$, the main interactions of interest are captured in $(*)$. The first interaction we consider states that when an agent sees something, it knows it.

$$\mathcal{S}\varphi \Rightarrow \mathcal{K}\varphi \tag{9}$$

Intuitively, this property will be true of an agent if its next state function distinguishes between every different percept received. If a next state function has this property, then intuitively, it never loses information from the percepts. We say a next state function is *complete* if it distinguishes between every different percept. Formally, a next state function *next* is *complete* iff $next(l, \rho) = next(l', \rho')$ implies $\rho = \rho'$. Formula schema (9) is valid in a model iff the next state function of that model is complete.

Turning to the converse direction, we might expect the following schema to be valid:

$$\mathcal{K}\varphi \Rightarrow \mathcal{S}\varphi \tag{10}$$

While this schema is satisfiable, it is not valid. To understand what kinds of agents validate this schema, imagine an agent with a next state function that chooses the next state *solely* on the basis of it current state. Let us say that an agent is *local* if it has this property. Formally, an agent's next-state function is local iff $next(l, \rho) = next(l', \rho)$ for all local states $l, l' \in L$, and percepts $\rho \in P$. It is not hard to see that formula schema (10) is valid in a model if the next state function of the agent in this model is local.

4.1 Systems of $\mathcal{VSK}$ Logic

The preceding sections identified the key interactions that may hold between what is true, visible, seen, and known. In this section, we consider *systems* of $\mathcal{VSK}$ logic, by which we mean possible *combinations* of interactions that could hold for any given agent-environment system. To illustrate, consider the class of systems in which: (i) the environment is not transparent; (ii) the agent's perception function is perfect; and (iii) the agent's next state function is neither complete nor local. In this class of models, the formula schemas (3), (7), and (8) are valid. These formula schemas can be understood as characterising a class of agent-environment systems — those in which the environment is not transparent, the agent's perception function is perfect, and the agent's next state function is neither complete nor local. In this way, by systematically considering the possible combinations of $\mathcal{VSK}$ formula schemas, we obtain a classification scheme for agent-environment systems. As the basis of this scheme, we consider only interaction schemas with the following form.

$$\Box_1\varphi \begin{array}{c}\Rightarrow\\ \Leftarrow\end{array} \Box_2\varphi$$

Given the three $\mathcal{VSK}$ modalities there are six such interaction schemas: (6), (3), (8), (7), (9), and (10). This in turn suggests there should be 64 distinct $\mathcal{VSK}$ systems. However, as (3) and (7) are valid in all $\mathcal{VSK}$ systems, there are in fact only 16 distinct systems, summarised in Table 1.

In systems $\mathcal{VSK}$-8 to $\mathcal{VSK}$-15 inclusive, visibility and truth are equivalent, in that everything true is also visible. These systems are characterised by transparent visibility relations. Formally, the schema $\varphi \Leftrightarrow \mathcal{V}\varphi$ is a valid formula in systems $\mathcal{VSK}$-8 to $\mathcal{VSK}$-15. The $\mathcal{V}$ modality is redundant in such systems.

In systems $\mathcal{VSK}$-4 to $\mathcal{VSK}$-7 and $\mathcal{VSK}$-12 to $\mathcal{VSK}$-15, everything visible is seen, and everything seen is visible. Visibility and perception are thus equivalent: the formula schema $\mathcal{V}\varphi \Leftrightarrow \mathcal{S}\varphi$ is valid in such systems. Hence one of the modalities $\mathcal{V}$ or $\mathcal{S}$ is redundant in systems $\mathcal{VSK}$-4 to $\mathcal{VSK}$-7 and $\mathcal{VSK}$-12 to $\mathcal{VSK}$-15. Models for these systems are characterised by agents with perfect perception (*see*) functions.

In systems $\mathcal{VSK}$-3, $\mathcal{VSK}$-7, $\mathcal{VSK}$-11, and $\mathcal{VSK}$-15, knowledge and perception are equivalent: an agent knows everything it sees, and sees everything it knows. In these systems, $\mathcal{S}\varphi \Leftrightarrow \mathcal{K}\varphi$ is valid. Models of such systems are characterised by complete, local next state functions.

In systems $\mathcal{VSK}$-12 to $\mathcal{VSK}$-15, we find that truth, visibility, and perception are equivalent: the schema $\varphi \Leftrightarrow \mathcal{V}\varphi \Leftrightarrow \mathcal{S}\varphi$ is valid. In such systems, the $\mathcal{V}$ and $\mathcal{S}$ modalities are redundant.

An analysis of individual $\mathcal{VSK}$ systems identifies a number of interesting properties, but space limitations prevents such an analysis here. Simply note that in system $\mathcal{VSK}$-15, the formula schema $\varphi \Leftrightarrow \mathcal{V}\varphi \Leftrightarrow \mathcal{S}\varphi \Leftrightarrow \mathcal{K}\varphi$ is valid, and hence all three modalities $\mathcal{V}$, $\mathcal{S}$, and $\mathcal{K}$ are redundant. System $\mathcal{VSK}$-15 thus collapses to propositional logic.

	Formula Schemas					
System	(6)	(3)	(8)	(7)	(9)	(10)
Name	$\varphi \Rightarrow \mathcal{V}\varphi$	$\mathcal{V}\varphi \Rightarrow \varphi$	$\mathcal{V}\varphi \Rightarrow \mathcal{S}\varphi$	$\mathcal{S}\varphi \Rightarrow \mathcal{V}\varphi$	$\mathcal{S}\varphi \Rightarrow \mathcal{K}\varphi$	$\mathcal{K}\varphi \Rightarrow \mathcal{S}\varphi$
$\mathcal{VSK}$-0		×		×		
$\mathcal{VSK}$-1		×		×		×
$\mathcal{VSK}$-2		×		×	×	
$\mathcal{VSK}$-3		×		×	×	×
$\mathcal{VSK}$-4		×	×	×		
$\mathcal{VSK}$-5		×	×	×		×
$\mathcal{VSK}$-6		×	×	×	×	
$\mathcal{VSK}$-7		×	×	×	×	×
$\mathcal{VSK}$-8	×	×		×		
$\mathcal{VSK}$-9	×	×		×		×
$\mathcal{VSK}$-10	×	×		×	×	
$\mathcal{VSK}$-11	×	×		×	×	×
$\mathcal{VSK}$-12	×	×	×	×		
$\mathcal{VSK}$-13	×	×	×	×		×
$\mathcal{VSK}$-14	×	×	×	×	×	
$\mathcal{VSK}$-15	×	×	×	×	×	×

Table 1. The sixteen possible $\mathcal{VSK}$ systems. A cross (×) indicates that the schema is valid in the corresponding system; all systems include (3) and (7).

5 Discussion and Related Work

The notion of computationally grounded theories of agency has been a leitmotif to much of my own research. My PhD thesis was on grounded semantics for multiagent systems [24, 28]; the issues were suggested to me by the work of Seel [20].

Since the mid 1980s, Halpern and colleagues have used modal epistemic logic for reasoning about multi-agent systems [6]. In this work, they demonstrated how *interpreted systems* could be used as models for such logics. Interpreted systems are very close to our agent-environment systems: the key differences are that they only record the *state* of agents within a system, and hence do not represent the percepts received by an agent or distinguish between what is true of an environment and what is visible of that environment. Halpern and colleagues have established a range of significant results relating to such logics, in particular, categorisations of the complexity of various decision problems in epistemic logic, the circumstances under which it is possible for a group of agents to achieve "common knowledge" about some fact, and most recently, the use of such logics for *directly programming* agents. Comparatively little effort has been devoted to characterising "architectural" properties of agents. The only obvious examples are the properties of no learning, perfect recall, and so on [6, pp281–307]. In their "situated automata" paradigm, Kaelbling and Rosenschein directly synthesised agents (in fact, digital circuits) from epistemic specifications of these agents [17]. This synthesis process was only because the semantics of their specification logic were computationally grounded.

Many other formalisms for reasoning about intelligent agents and multi-agent systems have been proposed over the past decade [29]. Following the pioneering work of Moore on the interaction between knowledge and action [13], most of these formalisms have attempted to characterise the "mental state" of agents engaged in various activities. Well-known examples of this work include Cohen-Levesque's theory of intention [4], and the ongoing work of Rao-Georgeff on the belief-desire-intention (BDI) model of agency [14]. The emphasis in this work has been more on axiomatic characterisations of architectural properties; for example, in [15], Rao-Georgeff discuss how various axioms of BDI logic can be seen to intuitively correspond to properties of agent architectures. However, this work is specific to BDI architectures, and in addition, the correspondence is an *intuitive* one: BDI logics are not computationally grounded in the sense described in this paper.

One major open issue in the area of computationally grounded logics of agency is that of giving a semantics to *goals*. The concept of a goal is ubiquitous in AI, and yet goals have no universally accepted semantics. In [28], I proposed a preliminary semantics of goals along the lines of "an agent has a goal of φ if φ is a necessary consequence of the strategy the agent is currently employing"; Singh [23] proposed a similar idea.

To summarise, we have made significant progress in developing logic-based theories of agents and multiagent systems since Hintikka's pioneering work on formal models of knowledge. However, if we are to justifiably claim that the theories and formalism we develop may be used to express the properties of computational multiagent systems, then we need to address the issue of computational grounding.

References

[1] R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984.

[2] K. Binmore. *Fun and Games: A Text on Game Theory*. D. C. Heath and Company: Lexington, MA, 1992.

[3] B. Chellas. *Modal Logic: An Introduction*. Cambridge University Press: Cambridge, England, 1980.

[4] P. R. Cohen and H. J. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42:213–261, 1990.

[5] P. R. Cohen and H. J. Levesque. Rational interaction as the basis for communication. In P. R. Cohen, J. Morgan, and M. E. Pollack, editors, *Intentions in Communication*, pages 221–256. The MIT Press: Cambridge, MA, 1990.

[6] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning About Knowledge*. The MIT Press: Cambridge, MA, 1995.

[7] A. Haddadi. *Communication and Cooperation in Agent Systems (LNAI Volume 1056)*. Springer-Verlag: Berlin, Germany, 1996.

[8] J. Hintikka. *Knowledge and Belief*. Cornell University Press: Ithaca, NY, 1962.

[9] L. P. Kaelbling and S. J. Rosenschein. Action and planning in embedded agents. In P. Maes, editor, *Designing Autonomous Agents*, pages 35–48. The MIT Press: Cambridge, MA, 1990.

[10] S. Kraus, K. Sycara, and A. Evenchik. Reaching agreements through argumentation: a logical model and implementation. *Artificial Intelligence*, 104:1–69, 1998.

[11] H. J. Levesque, P. R. Cohen, and J. H. T. Nunes. On acting together. In *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-90)*, pages 94–99, Boston, MA, 1990.

[12] J.-J. Ch. Meyer, W. van der Hoek, and B. van Linder. A logical approach to the dynamics of commitments. *Artificial Intelligence*, 113:1–40, 1999.

[13] R. C. Moore. A formal theory of knowledge and action. In J. F. Allen, J. Hendler, and A. Tate, editors, *Readings in Planning*, pages 480–519. Morgan Kaufmann Publishers: San Mateo, CA, 1990.

[14] A. S. Rao and M. Georgeff. Decision procedures for BDI logics. *Journal of Logic and Computation*, 8(3):293–344, 1998.

[15] A. S. Rao and M. P. Georgeff. An abstract architecture for rational agents. In C. Rich, W. Swartout, and B. Nebel, editors, *Proceedings of Knowledge Representation and Reasoning (KR&R-92)*, pages 439–449, 1992.

[16] J. S. Rosenschein and G. Zlotkin. *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers*. The MIT Press: Cambridge, MA, 1994.

[17] S. J. Rosenschein and L. P. Kaelbling. A situated view of representation and control. In P. E. Agre and S. J. Rosenschein, editors, *Computational Theories of Interaction and Agency*, pages 515–540. The MIT Press: Cambridge, MA, 1996.

[18] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, 1995.

[19] T. Sandholm. Distributed rational decision making. In G. Weiß, editor, *Multiagent Systems*, pages 201–258. The MIT Press: Cambridge, MA, 1999.

[20] N. Seel. Intentional descriptions of reactive systems. In Y. Demazeau and J.-P. Müller, editors, *Decentralized AI 2 — Proceedings of the Second European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW-90)*, pages 15–34. Elsevier Science Publishers B.V.: Amsterdam, The Netherlands, 1991.

[21] Y. Shoham. Agent-oriented programming. *Artificial Intelligence*, 60(1):51–92, 1993.

[22] M. P. Singh. Towards a formal theory of communication for multi-agent systems. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 69–74, Sydney, Australia, 1991.

[23] M. P. Singh. *Multiagent Systems: A Theoretical Framework for Intentions, Know-How, and Communications (LNAI Volume 799)*. Springer-Verlag: Berlin, Germany, 1994.

[24] M. Wooldridge. *The Logical Modelling of Computational Multi-Agent Systems*. PhD thesis, Department of Computation, UMIST, Manchester, UK, October 1992.

[25] M. Wooldridge. Agent-based software engineering. *IEE Proceedings on Software Engineering*, 144(1):26–37, February 1997.

[26] M. Wooldridge. *Reasoning about Rational Agents*. The MIT Press: Cambridge, MA, 2000.

[27] M. Wooldridge. Semantic issues in the verification of agent communication languages. *Autonomous Agents and Multi-Agent Systems*, 3(1):9–31, 2000.

[28] M. Wooldridge and M. Fisher. A first-order branching time logic of multi-agent systems. In *Proceedings of the Tenth European Conference on Artificial Intelligence (ECAI-92)*, pages 234–238, Vienna, Austria, 1992.

[29] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.

[30] M. Wooldridge and N. R. Jennings. Cooperative problem solving. *Journal of Logic and Computation*, 9(4):563–594, 1999.

[31] Michael Wooldridge and Alessio Lomuscio. Reasoning about visibility, perception and knowledge. In N.R. Jennings and Y. Lespérance, editors, *Intelligent Agents VI — Proceedings of the Sixth International Workshop on Agent Theories, Architectures, and Languages (ATAL-99)*, Lecture Notes in Artificial Intelligence. Springer-Verlag, Berlin, 2000.