In Proceedings of the National Conference on Artificial Intelligence (AAAI), 1991

A Logic of Situated Know-how

Munindar P. Singh*

Dept of Computer Sciences and Center for Cognitive Science
University of Texas University of Texas
Austin, TX 78712-1188 Austin, TX 78712
USA USA

Abstract

Know-how is an important concept in Artificial Intelligence. It has been argued previously that it cannot be successfully reduced to the knowledge of facts. In this paper, I present sound and complete axiomatizations for two non-reductive and intuitively natural formal definitions of the *know-how* of an agent situated in a complex environment. I also present some theorems giving useful properties of *know-how*, and discuss and resolve an interesting paradox (which is described within). This is done using a new operator in the spirit of Dynamic Logic that is introduced herein and whose semantics and proof-theory are given.

1 Introduction

Knowledge and action are the two staples of AI. Traditionally, research in AI has focused on the conception of knowledge corresponding to *know-that* or the knowledge of facts. In earlier work, I have argued that an important notion of knowledge from the point of AI is the one corresponding to *know-how* or the knowledge of skills, and that this cannot be easily reduced to *know-that* [?; ?]. I have also given two formal model-theoretic definitions of *know-how* in an abstract model of action and time—one purely reactive; the other involving abstract strategies. This theory does not require many troublesome assumptions (e.g., that only one event may occur at a time) that are often believed necessary. Thus it applies uniformly to both traditional and recent architectures. This theory has applications in the design and analysis of intelligent systems, and in giving a semantics for communication among different agents in a multiagent system [?]. However, while a formal model-theoretic definition of *know-how* has been given, no axiomatization is available for it—this is something that would facilitate these applications considerably. The goal of this paper is precisely to present a sound and complete axiomatization for the two definitions mentioned above.

*This research was supported by the National Science Foundation (through grant # IRI-8945845 to the Center for Cognitive Science, University of Texas at Austin).

In §2, I briefly outline the motivating arguments of [?]. Next in §3, I present the formal model. In §4, I present the definition of *know-how* for the case of purely reactive agents, and in §5, a sound and complete axiomatization for it. In §6, I define *strategies* and present the definition of *know-how* for complex agents whose behavior is best described by strategies. In §7, I present a sound and complete axiomatization for this notion as well. These definitions involve an extension of dynamic logic [?] that is explained within. Some important theorems about *know-how* are presented and a paradox about *know-how* described and resolved in §8.

2 Know-How

Traditional AI systems consist of a knowledge base of facts or beliefs coupled with an interpreter that reads and writes on it. Recently, this conception has been seen as being quite unsatisfactory [?; ?; ?]. But corresponding to this conception, traditional formal theories of knowledge and action [?; ?; ?] stress the knowledge of facts (or *know-that*), rather than the knowledge of skills (or *know-how*). They assume that *know-how* can be reduced to *know-that*—this assumption too is quite problematic [?]. Moore provides the following definition of *know-how*: an agent knows how to achieve p iff it has a plan such that it knows that it will result in p and that it can execute it [?, p. 347].

This idea embodies many unrealistic assumptions and is objected to in [?] in detail; some problems with it are outlined here. Real-life agents have to act rapidly on the basis of incomplete information, and lack the time and knowledge required to form complete plans before acting. The traditional reduction stresses explicitly represented action descriptions and plans. However, much *know-how* is reactive; e.g., someone may know how to beat someone else at tennis, but not have a plan that will guarantee it, much less one that is explicitly represented and interpreted. Traditional accounts assign the same basic actions to all agents, and allow expertise to vary only because of differences in the knowledge of facts, not because of differences in

skills. Also, they typically consider only single agent plans: thus they cannot account for the *know-how* of a group of agents—organizations know how to do certain things, but may not have any explicit plan [?; ?].

In the theory of [?], which is adopted here, an agent (x) knows how to achieve p, or to "do" p, if it is able to bring about the conditions for p through its actions. The world may change rapidly and unpredictably, but x is able to *force* the situation appropriately. It has a limited knowledge of its rapidly changing environment and too little time to decide on an optimal course of actions, and can succeed only if it has the required physical resources and is appropriately attuned to its environment. While this "situated" theory does not use plans, it can accommodate the richer notion of strategies (see §6). Therefore, it contrasts both with situated theories of *know-that* [?], and informal, and exclusively reactive, accounts of action [?]. While the pure plan-based view is troublesome, it is also the case that real-life agents lack the computational power and perceptual abilities to choose optimal actions in real-time, so the assumption of pure reactivity is not realistic either. Abstract "strategies" (discussed in §6) simplify reactive decision making by letting an agent have partial plans that can be effectively followed in real-time.

3 The Formal Model

The formal model is based on possible worlds. Each possible *world* has a branching *history* of *times*. The actions an agent can do can differ at each time—this allows for learning and forgetting, and changes in the environment. Let $M = \langle F, N\rangle$ be an intensional model, where $F = \langle \mathbf{W}, \mathbf{T}, <, \mathbf{A}, \mathbf{U}\rangle$ is a frame, and $N = \langle [\![\,]\!], \mathbf{B}\rangle$ an interpretation. Here $\mathbf{W}$ is a set of possible worlds; $\mathbf{T}$ is a set of possible times ordered by $<$; $\mathbf{A}$ is the class of agents in different possible worlds; $\mathbf{U}$ is the class of basic actions; as described below, $[\![\,]\!]$ assigns intensions to atomic propositions and actions. $\mathbf{B}$ is the class of functions assigning basic actions to the agents at different worlds and times. Each world $w \in \mathbf{W}$ has exactly one history, constructed from the times in $\mathbf{T}$. Histories are sets of times, partially ordered by $<$. They branch into the future. The times in each history occur only in that history.

A *scenario* at a world and time is any maximal set of times containing the given time, and all times that are in a particular future of it; i.e., a scenario is any single branch of the history of the world that begins at the given time, and contains all times in some linear subrelation of $<$. A *skeletal scenario* is an eternal linear sequence of times in the future of the given time; i.e., SS at w, t is any sequence: $\langle t = t_0, t_1, \ldots\rangle$, where $(\forall i : i \geq 0 \rightarrow t_i < t_{i+1})$ (linearity) and $(\forall i, t' : t' > t_i \rightarrow (\exists j : t' \not> t_j))$ (eternity). Now, a scenario, S, for w, t is the "linear closure" of some skeletal scenario at w, t. Formally, S, relative to some SS, is the minimal set that satisfies the following conditions:

- *Eternity*: $SS \subseteq S$
- *Linear Closure*: $(\forall t'' : t'' \in S \rightarrow (\forall t' : t_0 < t' < t'' \rightarrow t' \in S))$

$\mathbf{S}_{w,t}$ is the class of all scenarios at world w and time t: $(w \neq w' \vee t \neq t') \Rightarrow \mathbf{S}_{w,t} \cap \mathbf{S}_{w',t'} = \emptyset$. $\langle S, t, t'\rangle$ is a subscenario of S from t to t', inclusive.

Basic actions may have different durations relative to one another in different scenarios, even those that begin at the same world and time. The intension of an atomic proposition is the set of worlds and times where it is true; that of an action is, for each agent x, the set of subscenarios in the model in which an instance of it is done (from start to finish) by x; e.g., $\langle S, t, t'\rangle \in [\![a]\!]^x$ means that agent x does action a in the subscenario of S from time t to t'. I assume that $[\![\,]\!]$ respects $\mathbf{B}$; i.e., $a \in \mathbf{B}_{w,t}(x)$. The following coherence conditions on models are imposed: (1) at any time, an action can be done in at most one way on any given scenario; (2) subscenarios are uniquely identified by the times over which they stretch, rather than the scenario used to refer to them; (3) there is always a future time available in the model; and (4) something must be *done* by each agent along each scenario in the model, even if it is a dummy action. Restrictions on $[\![\,]\!]$ can be used to express the limitations of agents, and the ways in which their actions may interfere with those of others; e.g., at most one person enters an elevator at a time—i.e., in the model, if one person takes a step (a basic action) into an elevator, no one else takes a step into it at that time. The habits of agents can be similarly modeled; e.g., x always brakes before turning.

The formal language is CTL* (a propositional branching time logic [?]), augmented with operators $[\,]$, $\langle\!\langle\,\rangle\!\rangle$, K', K and quantification over basic actions. $[\,]$ depends on actions (as defined); $\langle\!\langle\,\rangle\!\rangle$ on trees and strategies (to be defined); K' is a simple version of *know-how*; and K is proper *know-how*. The agent is implicit in K' and K. The semantics is given relative to intensional models: it is standard for CTL*, but is repeated for A and F to make explicit their connection with the other operators, which are also considered below.

4 Reactive Know-how

As will soon become clear, it is useful to define *know-how* relative to a 'tree' of actions. A 'tree' of actions consists of an action (called its 'root'), and a set of subtrees. The idea is that the agent does 'root' initially and then picks out one of the available subtrees to pursue further. An agent, x, knows how to achieve p relative to a tree of actions iff on *all* scenarios where the root of the tree is done, either p occurs or x can choose one of the subtrees of the tree to pursue, and thereby achieve p. The definition requires p to occur on all scenarios where the root is done. The agent gets to "choose" one subtree after the root has been done. This is to allow the choice to be based on how the agent's environment has evolved. However, modulo this

choice, the agent must achieve p by forcing it to occur. This definition is intuitively most natural when applied to models that consist only of "normal" scenarios (see [?] for a discussion). It is important to note that the tree need not be represented by the agent—it just encodes the selection function the agent may be deemed to be using in picking out its actions at each stage (it *could* possibly be implemented as table lookup—a kind of symbolic representation). When a tree is finite in depth, it puts a bound on the number of actions that an agent may perform to achieve something. Since intuitively know how entails the ability to achieve the relevant proposition in finite time, this restriction is imposed here. The definitions do not really depend on the tree being of finite breadth, but that too would be required for finite agents.

Formally, a *tree* is either (1) $\emptyset$, the empty tree or (2) $\langle$root, $\langle$subtrees$\rangle\rangle$, where 'root' is as described above, and 'subtrees' is a non-empty set of trees. 'Tree' refers to a tree of this form. Intuitively, $\langle\!\langle$tree$\rangle\!\rangle p$ is true iff the agent can use 'tree' as a selection function and thereby force p to become true. If we wish, we can impose a restriction that for all trees the empty tree $\emptyset$ is always in 'subtrees' to ensure that the agent does as little work as possible—i.e., so that the agent can stop acting as soon as the right condition comes to hold.

- $M \models_{w,t} \langle\!\langle\emptyset\rangle\!\rangle p$ iff $M \models_{w,t} p$
- $M \models_{w,t} \langle\!\langle$tree$\rangle\!\rangle p$ iff $(\exists S, t' : S \in \mathbf{S}_{w,t} \wedge \langle S, t, t'\rangle \in [\![$root$]\!]) \wedge (\forall S : S \in \mathbf{S}_{w,t} \wedge (\exists t' : \langle S, t, t'\rangle \in [\![$root$]\!]) \rightarrow (\exists t' : \langle S, t, t'\rangle \in [\![$root$]\!] \wedge (\exists$ sub $\in$ tree.subtrees $: M \models_{w,t'} \langle\!\langle$sub$\rangle\!\rangle p)))$

Now we can define *know-how* as follows:

- $M \models_{w,t} \mathsf{K}'p$ iff $(\exists$tree $: M \models_{w,t} \langle\!\langle$tree$\rangle\!\rangle p)$
- $M \models_{w,t} \mathsf{K}p$ iff $M \models_{w,t} (\mathsf{K}'p) \wedge (\exists S : S \in \mathbf{S}_{w,t} \wedge (\forall t' : t' \in S \rightarrow M \not\models_{w,t'} p))$

This definition seems to require that the agent be able to make the right choices at the level of basic actions—it would attribute *know-how* to an agent even if it may not be able to do so, just if it has the right set of basic actions. Thus it can be applied only purely externally on agents that are somewhat rigidly structured, so that they have only those basic actions that they will in fact choose from while acting. But then it de-emphasizes the inherent autonomy of complex intelligent agents. At the same time, it is not acceptable to require that agents explicitly represent and interpret shared plans. In §6, this will motivate us to look at abstract strategies that need not be explicitly represented, but which can characterize the coordinated behavior of intelligent agents.

5 Axioms for Reactive Know-how

Now I present an axiomatization for the definition of K' given above and a proof of its soundness and completeness. It is clear that the definition of K is non-normal [?, p. 114]; e.g., we have $\neg\mathsf{K}$true. In order to take care of this complication, I present the axiomatization in two stages—here I axiomatize K' (which is normal), and then in §8, motivate and consider K.

I now describe the formal language in which the axiomatization will be given. Loosely following standard dynamic logic (DL) [?], I introduce a new modality for actions. For an action a and a formula p, let $[a]p$ denote that on the given scenario, if a is ever done starting now, then p holds when a is completed. Let $\langle a\rangle p$ abbreviate $\neg[a]\neg p$. Let A and E be the path- or scenario-quantifiers of branching temporal logic [?]. A denotes "in *all* scenarios at the present time," and E denotes "in *some* scenario at the present time"—i.e., $\mathsf{E}p \equiv \neg\mathsf{A}\neg p$. Thus $\mathsf{A}[a]p$ denotes that on all scenarios at the present moment, if a is ever done then p holds; i.e., it corresponds to the necessitation operator in DL and $\mathsf{E}\langle a\rangle p$ to the possibility operator in DL. The advantage of defining $[a]p$ over scenarios rather than states (as in DL) is to simplify the connection to times. $p\mathsf{U}q$ denotes "eventually q, and p until q." $\mathsf{F}p$ denotes "eventually p" and abbreviates "$\mathsf{true}\mathsf{U}$p." Formally, we have

- $M \models_{S,t} [a]p$ iff $(\exists t' : \langle S, t, t'\rangle \in [\![a]\!]) \rightarrow (\exists t' : \langle S, t, t'\rangle \in [\![a]\!] \wedge M \models_{S,t'} p)$

  It is easy to see that $([a]p \wedge [a]q) \equiv [a](p \wedge q)$.
- $M \models_{w,t} \mathsf{A}p$ iff $(\forall S : S \in \mathbf{S}_{w,t} \rightarrow M \models_{S,t} p)$
- $M \models_{S,t} p$ iff $M \models_{w,t} p$, if p is not of the form $[a]q$ or $\langle a\rangle q$, and w is the unique world such that $S \in \mathbf{S}_{w,t}$.
- $M \models_{S,t} p\mathsf{U}q$ iff $(\exists t' : t' \leq t \wedge M \models_{S,t'} q \wedge (\forall t'' : t \leq t'' \leq t' \rightarrow M \models_{S,t''} p))$.

Then we have the following axiomatization (except for the underlying operators such as [] and A).

1. $p \rightarrow \mathsf{K}'p$
2. $(\exists a : \mathsf{E}\langle a\rangle\mathsf{true} \wedge \mathsf{A}[a]\mathsf{K}'p) \rightarrow \mathsf{K}'p$

   $\mathsf{E}\langle a\rangle\mathsf{true}$ means that a is a basic action of the agent at the given time.

These axioms, though they refer to only one action, allow any number of them—the agent achieves p trivially when it is true, and knows-how to achieve it whenever there is an action he can do to force it to hold trivially. Axiom 2 can be applied repeatedly.

Theorem 1 The above axiomatization is sound and complete for K'.

Proof.

Construct a model branching time model, M. The indices of M are notated as $\langle w, t\rangle$ and are maximally consistent sets of formulae that obey the usual constraints for $[a]p$, $\mathsf{A}[a]p$, etc. (i.e., they contain all the substitution instances of the theorems of the underlying logic). Furthermore, these sets are closed under the above axioms.

Soundness: For axiom 1 above, soundness is trivial from the definition of $\langle\!\langle\emptyset\rangle\!\rangle p$. For axiom 2, let $(\exists a : \mathsf{E}\langle a\rangle\mathsf{true} \wedge \mathsf{A}[a]\mathsf{K}'p)$ hold at $\langle w, t\rangle$. Then $(\exists S, t' : S \in \mathbf{S}_{w,t} \wedge \langle S, t, t'\rangle \in [\![a]\!]) \wedge (\forall S : S \in \mathbf{S}_{w,t} \wedge (\exists t' : \langle S, t, t'\rangle \in [\![a]\!]) \rightarrow (\exists t' : \langle S, t, t'\rangle \in [\![a]\!] \wedge M \models_{w,t'} \mathsf{K}'p))$. At each $\langle w, t'\rangle$, $(\exists$tree$' : M \models_{w,t'} \langle\!\langle$tree$'\rangle\!\rangle p)$. Define 'tree' as $\langle a, \langle\{$tree$'|$tree$'$ is a tree used to make $\mathsf{K}'p$

true at any of the $\langle w, t' \rangle$ in the above quantification$\}\rangle\rangle$. Thus $M \models_{w,t} \langle\!\langle \text{tree} \rangle\!\rangle p$, or $M \models_{w,t} \mathsf{K}'p$.

Completeness: The proof is by induction on the structure of the formulae. Only the case of the operator K' is described below. Completeness means that $M \models_{w,t} \mathsf{K}'p$ implies $\mathsf{K}'p \in \langle w, t \rangle$. $M \models_{w,t} \mathsf{K}'p$ iff $(\exists \text{tree} : M \models_{w,t} \langle\!\langle \text{tree} \rangle\!\rangle p)$. This proof is also by induction, though on the structure of trees using which different formulae of the form $\mathsf{K}'p$ get satisfied in the model. The base case is the empty tree $\emptyset$. And $M \models_{w,t} \langle\!\langle \emptyset \rangle\!\rangle p$ iff $M \models_{w,t} p$. By the (outer) inductive hypothesis on the structure of the formulae, $p \in \langle w, t \rangle$. By axiom 1 above, $\mathsf{K}'p \in \langle w, t \rangle$ as desired.

For the inductive case, $M \models_{w,t} \langle\!\langle \text{tree} \rangle\!\rangle p$ iff $(\exists S, t' : S \in \mathbf{S}_{w,t} \wedge \langle S, t, t' \rangle \in [\![\text{root}]\!]) \wedge (\forall S : S \in \mathbf{S}_{w,t} \wedge (\exists t' : \langle S, t, t' \rangle \in [\![\text{root}]\!]) \rightarrow (\exists t' : \langle S, t, t' \rangle \in [\![\text{root}]\!] \wedge (\exists \, \text{sub} \in \text{tree.subtrees} : M \models_{w,t'} \langle\!\langle \text{sub} \rangle\!\rangle \mathsf{K}'p)))$. But since 'sub' is a subtree of 'tree,' we can use the inductive hypothesis on trees to show that this is equivalent to $(\exists S, t' : S \in \mathbf{S}_{w,t} \wedge \langle S, t, t' \rangle \in [\![\text{root}]\!]) \wedge (\forall S : S \in \mathbf{S}_{w,t} \wedge (\exists t' : \langle S, t, t' \rangle \in [\![\text{root}]\!]) \rightarrow (\exists t' : \langle S, t, t' \rangle \in [\![\text{root}]\!] \wedge M \models_{w,t'} \mathsf{K}'p))$. But it is easy to see that $(\exists S, t' : S \in \mathbf{S}_{w,t} \wedge \langle S, t, t' \rangle \in [\![\text{root}]\!])$ iff $\mathsf{E}\langle \text{root} \rangle \mathsf{true}$. And (using the definition of $[\,]$) the second conjunct holds iff $\mathsf{A}[\text{root}](\mathsf{K}'p)$. Thus $M \models_{w,t} \langle\!\langle \text{tree} \rangle\!\rangle p$ iff $M \models_{w,t} (\exists \text{root} : \mathsf{E}\langle \text{root} \rangle \mathsf{true} \wedge \mathsf{A}[\text{root}](\mathsf{K}'p))$. But since the axiomatization of the underlying logic is complete, $(\exists \text{root} : \mathsf{E}\langle \text{root} \rangle \mathsf{true} \wedge \mathsf{A}[\text{root}](\mathsf{K}'p)) \in \langle w, t \rangle$. Thus by axiom 2, $\mathsf{K}'p \in \langle w, t \rangle$. Hence we have completeness. □

6 Strategies and Strategic Know-how

Even for situated agents, and especially for complex ones, it is useful to have an abstract description of their behavior at hand. Such descriptions, I call *s-trategies*. Strategies correspond to plans in traditional systems, and to the architectural structure of reactive agents, as instantiated at a given time. In the formalism, they are taken to be of the form of regular programs [?]. A strategy is simply the designer's description of the agent and the way in which it behaves. An agent knows how to achieve p, if it can achieve p whenever it so "intends"—strategies are a simple way of treating intentions. I now let each agent have a strategy that it follows in the current situation. Intuitively, an agent knows how to achieve p relative to a strategy Y, iff it possesses the skills required to follow Y in such a way as to achieve p. Thus *know-how* is partitioned into two components: the "ability" to have satisfactory strategies, and the "ability" to follow them. Strategies abstractly characterize the behavior of agents as they perform their actions in following their strategies. As described below, these actions are derived from the tree (as used in §4) characterizing their selection function for each substrategy.

Let Y be a strategy of agent x; 'current(Y)' the part of Y now up for execution; and 'rest(Y)' the part of Y remaining after 'current(Y)' has been done. I will define strategies, 'current,' 'rest' and the *strategy-relative intension* of a tree (i.e., $[\![\,]\!]_Y$) shortly, but first I formalize *know-how* using the auxiliary definition of *know-how* relative to a strategy. Extend the notation to allow $\langle\!\langle\,\rangle\!\rangle$ to be applied to strategies. $\langle\!\langle Y \rangle\!\rangle p$ means that if the agent can be said to be following strategy Y, it knows how to perform all the substrategies of Y that it may need to perform, and furthermore that it can perform them in such a way as to force the world so as to make p true. Basically, this allows us to have the *know-hows* of an agent to achieve the conditions in different substrategies strung together so that it has the *know-how* to achieve some composite condition. This is especially important from the point of view of designers and analyzers of agents, since they can take advantage of the abstraction provided by strategies to design or understand the *know-how* of an agent in terms of its *know-how* to achieve simpler conditions. Even formally, this latter *know-how* is purely reactive as in §4 (see Theorem 2).

- $M \models_{w,t} \langle\!\langle \text{skip} \rangle\!\rangle p$ iff $M \models_{w,t} p$
- $M \models_{w,t} \langle\!\langle Y \rangle\!\rangle p$ iff $(\exists \text{ tree}: (\exists S, t' : \langle S, t, t' \rangle \in [\![\text{tree}]\!]_{\text{current}(Y)}) \wedge (\forall S : S \in \mathbf{S}_{w,t} \wedge (\exists t' : \langle S, t, t' \rangle \in [\![\text{tree}]\!]_{\text{current}(Y)}) \rightarrow (\exists t' : \langle S, t, t' \rangle \in [\![\text{tree}]\!]_{\text{current}(Y)} \wedge M \models_{w,t'} \langle\!\langle \text{rest}(Y) \rangle\!\rangle p)))$

This definition says that an agent, x knows how to achieve p relative to strategy Y iff there is a tree of actions for it such that it can achieve the 'current' part of its strategy by following that tree, and that on all scenarios where it does so it either achieves p or can continue with the 'rest' of its strategy (and knows how to achieve p relative to that). Now $\mathsf{K}'p$ may be defined as given below. $\mathsf{K}p$ is unchanged relative to $\mathsf{K}'p$.

- $M \models_{w,t} \mathsf{K}'p$ iff $(\exists Y : M \models_{w,t} \langle\!\langle Y \rangle\!\rangle p)$

Define a strategy, Y, recursively as below. These definitions allow us to incorporate abstract specifications of the agent's actions from the designer's point of view, and yet admit the purely reactive aspects of its behavior in case 1 below.

0. skip: the empty strategy
1. do(p): a condition to be achieved
2. $Y_1; Y_2$: a sequence of strategies
3. if p then Y_1 else Y_2: a conditional strategy
4. while p do Y_1: a conditional loop

By definition, 'skip; Y' = Y, for all Y. The 'current' part of a strategy depends on the current situation. For case 0, 'current(Y)' is 'skip'; for case 1, it is 'Y' itself; for case 2, it is 'current(Y_1)'; for case 3, if p holds in the current situation, it is 'current(Y_1),' else 'current(Y_2)'; for case 4, if p holds (in the current situation), it is 'current(Y_1),' else 'skip.' The 'rest' of a strategy is what is left after the current part is performed. For cases 0 and 1 'rest(Y)' is 'skip'; for case 2, it is 'rest(Y_1); Y_2'; for case 3, if p holds, it is 'rest(Y_1),' else 'rest(Y_2)'; for case 4, if p holds, it is 'rest(Y_1); while p do Y_1,' else 'skip.'

Since 'current(Y)' is always of the form 'skip' or 'do(p),' $[\![\text{tree}]\!]_{\text{current}(Y)}$ is invoked (for a given tree) only for cases 0 and 1, and is defined for them below.

The expression $[\![\text{tree}]\!]_{\text{current}(Y)}$ denotes a restricted intension of 'tree' relative to an agent (implicit here) and the substrategy it is achieving by following 'tree'—in the sequel, I refer to it as the *strategy-relative intension* of 'tree.' This expression considers only those subscenarios where the success of the given substrategy is assured, i.e., forced, by the agent—fortuitously successful subscenarios are excluded. Briefly, $[\![\text{tree}]\!]_{\text{current}(Y)}$ is the set of subscenarios corresponding to executions of 'tree' such that they lead to the strategy 'current(Y)' being forced. Here $t, t' \in S$; the world is w.

$\langle S, t, t' \rangle \in [\![\text{tree}]\!]_{\text{current}(Y)}$ iff

0. current(Y) = skip is achieved by the empty tree; i.e., tree = $\emptyset$ and $t = t'$.
1. current(Y) = do(p): 'Tree' follows 'do(p)' iff the agent can achieve p in doing it.

   if tree = $\emptyset$ then $t = t' \wedge M \models_{w,t} p$ else $M \models_{w,t} \langle\!\langle \text{tree} \rangle\!\rangle p \wedge (\exists t'' : t < t'' \leq t' \wedge \langle S, t, t'' \rangle \in [\![\text{root}]\!] \wedge (\exists \text{sub} : \text{sub} \in \text{tree.subtrees} \wedge \langle S, t'', t' \rangle \in [\![\text{sub}]\!]_{\text{current}(Y)}))$

Now for some intuitions about the definition of *know-how* relative to a strategy. The execution of a strategy by an agent is equivalent to its being unraveled into a sequence of substrategies of the form do(p). The agent follows each by doing actions prescribed by some tree. Thus the substrategies serve as abstractions of trees of basic actions. In this way, the definition of *know-how* exhibits a two-layered architecture of agents: the bottom layer determining how substrategies of limited forms are achieved, and the top layer how they are composed to form complex strategies.

7 Axioms for Strategic Know-how

Since strategies are structured, the axiomatization of *know-how* relative to a strategy must involve their structure. This comes into the axiomatization of $\langle\!\langle Y \rangle\!\rangle p$. Many of these axioms mirror those for standard DL modalities, but there are important differences:

1. $\langle\!\langle \text{skip} \rangle\!\rangle p \equiv p$
2. $\langle\!\langle Y_1; Y_2 \rangle\!\rangle p \equiv \langle\!\langle Y_1 \rangle\!\rangle \langle\!\langle Y_2 \rangle\!\rangle p$
3. $\langle\!\langle \text{if } q \text{ then } Y_1 \text{ else } Y_2 \rangle\!\rangle p \equiv (q \rightarrow \langle\!\langle Y_1 \rangle\!\rangle p) \wedge (\neg q \rightarrow \langle\!\langle Y_2 \rangle\!\rangle p)$
4. $\langle\!\langle \text{while } q \text{ do } Y_1 \rangle\!\rangle p \equiv (q \rightarrow \langle\!\langle Y_1 \rangle\!\rangle \langle\!\langle \text{while } q \text{ do } Y_1 \rangle\!\rangle p) \wedge (\neg q \rightarrow p)$
5. $\langle\!\langle \text{do}(q) \rangle\!\rangle p \equiv (q \wedge p) \vee (\neg q \wedge (\exists a : \mathsf{E}\langle a \rangle \mathsf{true} \wedge \mathsf{A}[a] \langle\!\langle \text{do}(q) \rangle\!\rangle p))$

Theorem 2 The above axiomatization is sound and complete for any model M as described in §3, with respect to the definition of K' relative to a strategy.

Proof.

Soundness: The proof is not included here to save space. However, it quite simple for cases 1 through 4; for case 5, it follows the proof in Theorem 1.

Completeness: As in §5, construct a model whose indices are maximally consistent sets of sentences of the language. Completeness is proved here only for formulae of the form $\langle\!\langle Y \rangle\!\rangle p$, and means that $M \models_{w,t} \langle\!\langle Y \rangle\!\rangle p$ entails $\langle\!\langle Y \rangle\!\rangle p \in \langle w, t \rangle$, the corresponding point in the model. Let $\langle\!\langle Y \rangle\!\rangle p$. The proof is by induction of the structure of strategies. $M \models_{w,t} \langle\!\langle \text{skip} \rangle\!\rangle p$ iff $M \models_{w,t} p$. But this is ensured by axiom 1. Similarly, $M \models_{w,t} \langle\!\langle \text{if } q \text{ then } Y_1 \text{ else } Y_2 \rangle\!\rangle p$ iff there exists a tree that follows current(if q then Y_1 else Y_2) and some further properties hold of it. But q implies that current(if q then Y_1 else Y_2) = current(Y_1) and $\neg q$ implies that it equals current(Y_2). Similar conditions hold for the function 'rest.' Therefore, by axiom 3, $M \models_{w,t} \langle\!\langle \text{if } q \text{ then } Y_1 \text{ else } Y_2 \rangle\!\rangle p$ iff (if q then $M \models_{w,t} \langle\!\langle Y_1 \rangle\!\rangle p$ else $M \models_{w,t} \langle\!\langle Y_2 \rangle\!\rangle p$). By induction, since Y_1 and Y_2 are structurally smaller than the above conditional strategy, we have that $\langle\!\langle \text{if } q \text{ then } Y_1 \text{ else } Y_2 \rangle\!\rangle p \in \langle w, t \rangle$. The case for iterative strategies is analogous, since axiom 4 captures the conditions for 'current' and 'rest' of an iterative strategy.

The case of $\langle\!\langle \text{do}(q) \rangle\!\rangle p$ turns out to be quite simple. This is because the right to left direction of axiom 5 is entailed by the following pair, which are (almost) identical to the axioms for reactive *know-how* given in §5. Completeness for this case mirrors the proof of completeness given there and is not repeated here.

- $(q \wedge p) \rightarrow \langle\!\langle \text{do}(q) \rangle\!\rangle p$
- $(\exists a : \mathsf{E}\langle a \rangle \mathsf{true} \wedge \mathsf{A}[a] \langle\!\langle \text{do}(q) \rangle\!\rangle p) \rightarrow \langle\!\langle \text{do}(q) \rangle\!\rangle p$

Surprisingly, the trickiest case turns out to be that of sequencing. When $Y_1 = \text{skip}$, the desired condition for the axiom 2 follows trivially. But in the general case, when $Y_1 \neq \text{skip}$, the satisfaction condition for $\langle\!\langle Y_1; Y_2 \rangle\!\rangle p$ recursively depends on that for $\langle\!\langle \text{rest}(Y_1); Y_2 \rangle\!\rangle p$. However, this strategy does not directly feature in axiom 2. Also, rest(Y_1);Y_2 may not be a structurally smaller strategy than $Y_1; Y_2$ (e.g., if Y_1 is an iterative strategy, rest(Y_1) may be structurally more complex than Y_1). However, we can invoke the fact that here (as in standard DL), iterative strategies are finitary; i.e., they lead to only a finite number of repetitions. Thus we can assume that for any strategy, $Y \neq \text{skip}$, world, w and time t, the execution tree in the model (i.e., as induced by $<$ and restricted to the execution of Y) has a finite "depth," i.e., number of invocations of 'current.' If Y is followed at w, t, then the 'rest(Y)' is followed at those points where 'current(Y)' has just been followed. The depth of 'rest(Y)' = (depth of Y) $-$ 1. The depth of 'skip' is 0. Thus the depth is a metric to do the necessary induction on. The remainder of the proof is quite simple. Thus for all cases in the definition of a strategy, $M \models_{w,t} \langle\!\langle Y \rangle\!\rangle p$ entails $\langle\!\langle Y \rangle\!\rangle p \in \langle w, t \rangle$. □

8 Consequences

Formal definitions are supposed to help clarify our intuitions about the concepts formalized. The above axiomatizations do just that. Some of their consequences are listed and proved below. These consequences are important and useful since they help us better delineate the properties of the concept of *know-how*. While the reactive and strategic definitions K' are significantly different in their implementational import, they share many interesting logical properties.

Theorem 3 $\mathsf{K}'p \wedge \mathsf{AG}(p \rightarrow q) \rightarrow \mathsf{K}'q$

Proof Idea. This follows from the corresponding result for $[a]p$ mentioned in §5. □

Theorem 4 $\mathsf{K}'\mathsf{K}'p \rightarrow \mathsf{K}'p$

Proof Idea. Construct a single tree out of the trees for $\mathsf{K}'\mathsf{K}'p$. □

This seems obvious: if an agent can ensure that it will be able to ensure p, then it can already ensure p. But see the discussion following Theorem 6.

While K' captures a very useful concept, it is sometimes preferable to consider a related concept, which is captured by K. K is meant to exclude cases such as the rising of the sun (assuming it is inevitable)—intuitively, an agent can be said to know how to achieve something only if it is not inevitable anyway. K can be axiomatized simply by adding the following axiom

3. $\mathsf{K}p \equiv (\mathsf{K}'p \wedge \neg \mathsf{AF}p)$

However, this also makes the logic of K non-normal, i.e., not closed under logical consequence. This is because the proposition entailed by the agent's given *know-how* may be something inevitable. Therefore, despite Theorem 3, the corresponding statement for K fails. Indeed, we have

Theorem 5 $\neg \mathsf{K}\mathsf{true}$

Proof. We trivially have $\mathsf{AF}\mathsf{true}$, which by axiom 3 entails $\neg \mathsf{K}\mathsf{true}$. □

Theorem 6 $p \rightarrow \neg \mathsf{K}p$

Proof. Trivially again, since $p \rightarrow \mathsf{AF}p$. □

I.e., if p already holds then the agent cannot be felicitously said to know how to achieve it. By a simple substitution, we obtain $\mathsf{K}p \rightarrow \neg \mathsf{K}\mathsf{K}p$, whose contrapositive is $\mathsf{K}\mathsf{K}p \rightarrow \neg \mathsf{K}p$. This is in direct opposition to Theorem 4 for K', and is surprising, if not counterintuitive. It says that an agent who knows how to know how to achieve p does not know how to achieve p, simply because if it already knows how to achieve p it may not know how to know how to achieve it. This too agrees with our intuitions. The explanation for this paradox is that when we speak of nested *know-how* (and in natural language, we do not do that often), we use two different senses of *know-how*: K for the inner one and K' for the outer one. Thus the correct translation is $\mathsf{K}'\mathsf{K}p$, which entails $\mathsf{K}'p$ as desired. If p describes a condition that persists once it holds (as many p's in natural language examples do) then we also have $\mathsf{K}p$.

9 Conclusions

I presented two sound and complete logics for non-reductive and intuitive definitions of the *know-how* of an intelligent agent situated in a complex environment. This formalization reveals many interesting properties of *know-how* and helps clarify our intuitions. It also simplifies the application of the concept of *know-how* to the design and analysis of situated intelligent agents. Of special technical interest is the operator expressed by $\langle\!| \, |\!\rangle$ that is different from those in standard Dynamic Logic. This operator provides the right formal notion with which to capture the *know-how* of an agent whose behavior is abstractly characterized in terms of strategies. The differences between the reactive and strategic senses of *know-how* are mainly concerned with the complexity of the designs of the agents to whom they may be attributed. The power of the strategic sense arises from the fact that it lets an agent act, and a designer reason about it, using "macro-operators."

References

Agre, Philip and Chapman, David; 1987. Pengi: An implementation of a theory of activity. In *AAAI*. 268–272.

Chellas, Brian F.; 1980. *Modal Logic*. Cambridge University Press, New York, NY.

Emerson, E. A.; 1990. Temporal and modal logic. In Leeuwen, J.van, editor, *Handbook of Theoretical Computer Science*, volume B. North-Holland Publishing Company, Amsterdam, The Netherlands.

Hewitt, Carl; 1988. Organizational knowledge processing. In *Workshop on Distributed Artificial Intelligence*.

Konolige, Kurt G.; 1982. A first-order formalism of knowledge and action for multi-agent planning. In Hayes, J. E.; Mitchie, D.; and Pao, Y., editors, *Machine Intelligence 10*. Ellis Horwood Ltd., Chichester, UK. 41–73.

Kozen, Dexter and Tiurzyn, Jerzy; 1990. Logics of program. In Leeuwen, J.van, editor, *Handbook of Theoretical Computer Science*. North-Holland Publishing Company, Amsterdam, The Netherlands.

Moore, Robert C.; 1984. A formal theory of knowledge and action. In Hobbs, Jerry R. and Moore, Robert C., editors, *Formal Theories of the Commonsense World*. Ablex Publishing Company, Norwood, NJ. 319–358.

Morgenstern, Leora; 1987. A theory of knowledge and planning. In *IJCAI*.

Rosenschein, Stanley J.; 1985. Formal theories of knowledge in AI and robotics. *New Generation Computing* 3(4).

Singh, Munindar P.; 1990. Towards a theory of situated know-how. In *9th European Conference on Artificial Intelligence*.

Singh, Munindar P.; 1991a. Group ability and structure. In Demazeau, Y. and Müller, J.-P., editors, *Decentralized Artificial Intelligence, Volume 2*. Elsevier Science Publishers B.V. / North-Holland, Amsterdam, Holland. 127–145.

Singh, Munindar P.; 1991b. Towards a formal theory of communication for multiagent systems. In *International Joint Conference on Artificial Intelligence (IJCAI)*.